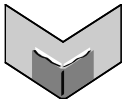
The Journal of Machine Learning Research
Volume 13
Print-Archive Edition

Pages 2503–3768



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 13
Print-Archive Edition

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this volume are articles published electronically in JMLR in 2012.

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

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

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

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

JMLR Editorial Board

Journal of Machine Learning Research

Volume 13, 2012

Journal of Machine Learning Research 13 (2012) 2503-2528 Submitted 8/11; Revised 3/12; Published 9/12

Trading Regret for Efficiency: Online Convex Optimization with Long Term Constraints

Mehrdad Mahdavi MAHDAVIM@CSE.MSU.EDU
Rong Jin RONGJIN@CSE.MSU.EDU
Tianbao Yang YANGTIA1@MSU.EDU
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI, 48824, USA

Editor: Shie Mannor

Abstract

In this paper we propose efficient algorithms for solving constrained online convex optimization problems. Our motivation stems from the observation that most algorithms proposed for online convex optimization require a projection onto the convex set $\mathcal{K}$ from which the decisions are made. While the projection is straightforward for simple shapes (e.g., Euclidean ball), for arbitrary complex sets it is the main computational challenge and may be inefficient in practice. In this paper, we consider an alternative online convex optimization problem. Instead of requiring that decisions belong to $\mathcal{K}$ for all rounds, we only require that the constraints, which define the set $\mathcal{K}$, be satisfied in the long run. By turning the problem into an online convex-concave optimization problem, we propose an efficient algorithm which achieves $O(\sqrt{T})$ regret bound and $O(T^{3/4})$ bound on the violation of constraints. Then, we modify the algorithm in order to guarantee that the constraints are satisfied in the long run. This gain is achieved at the price of getting $O(T^{3/4})$ regret bound. Our second algorithm is based on the mirror prox method (Nemirovski, 2005) to solve variational inequalities which achieves $O(T^{2/3})$ bound for both regret and the violation of constraints when the domain $\mathcal{K}$ can be described by a finite number of linear constraints. Finally, we extend the results to the setting where we only have partial access to the convex set $\mathcal{K}$ and propose a multipoint bandit feedback algorithm with the same bounds in expectation as our first algorithm.

Keywords: online convex optimization, convex-concave optimization, bandit feedback, variational inequality

1. Introduction

Online convex optimization has recently emerged as a primitive framework for designing efficient algorithms for a wide variety of machine learning applications (Cesa-Bianchi and Lugosi, 2006). In general, an online convex optimization problem can be formulated as a repeated game between a learner and an adversary: at each iteration t, the learner first presents a solution $\mathbf{x}_t \in \mathcal{K}$, where $\mathcal{K} \subseteq \mathbb{R}^d$ is a convex domain representing the solution space; it then receives a convex function $f_t(\mathbf{x}) : \mathcal{K} \mapsto \mathbb{R}_+$ and suffers the loss $f_t(\mathbf{x}_t)$ for the submitted solution $\mathbf{x}_t$. The objective of the learner is to generate a sequence of solutions $\mathbf{x}_t \in \mathcal{K}, t = 1, 2, \cdots, T$ that minimizes the regret $\Re_T$ defined as

$$\Re_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{1}$$

Regret measures the difference between the cumulative loss of the learner's strategy and the minimum possible loss had the sequence of loss functions been known in advance and the learner could choose the best fixed action in hindsight. When $\Re_T$ is sub-linear in the number of rounds, that is, $o(T)$, we call the solution Hannan consistent (Cesa-Bianchi and Lugosi, 2006), implying that the learner's average per-round loss approaches the average per-round loss of the best fixed action in hindsight. It is noticeable that the performance bound must hold for any sequence of loss functions, and in particular if the sequence is chosen adversarially.

Many successful algorithms have been developed over the past decade to minimize the regret in the online convex optimization. The problem was initiated in the remarkable work of Zinkevich (2003) which presents an algorithm based on gradient descent with projection that guarantees a regret of $O(\sqrt{T})$ when the set $\mathcal{K}$ is convex and the loss functions are Lipschitz continuous within the domain $\mathcal{K}$. In Hazan et al. (2007) and Shalev-Shwartz and Kakade (2008) algorithms with logarithmic regret bound were proposed for strongly convex loss functions. In particular, the algorithm in Hazan et al. (2007) is based on online Newton step and covers the general class of exp-concave loss functions. Notably, the simple gradient based algorithm also achieves an $O(\log T)$ regret bound for strongly convex loss functions with an appropriately chosen step size. Bartlett et al. (2007) generalizes the results in previous works to the setting where the algorithm can adapt to the curvature of the loss functions without any prior information. A modern view of these algorithms casts the problem as the task of following the regularized leader (Rakhlin, 2009). In Abernethy et al. (2009), using game-theoretic analysis, it has been shown that both $O(\sqrt{T})$ for Lipschitz continuous and $O(\log T)$ for strongly convex loss functions are tight in the minimax sense.

Examining the existing algorithms, most of the techniques usually require a projection step at each iteration in order to get back to the feasible region. For the performance of these online algorithms, the computational cost of the projection step is of crucial importance. To motivate the setting addressed in this paper, let us first examine a popular online learning algorithm for minimizing the regret $\Re_T$ based on the online gradient descent (OGD) method (Zinkevich, 2003). At each iteration t, after receiving the convex function $f_t(\mathbf{x})$, the learner computes the gradient $\nabla f_t(\mathbf{x}_t)$ and updates the solution $\mathbf{x}_t$ by solving the following optimization problem

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{K}}(\mathbf{x}_t - \eta\nabla f_t(\mathbf{x}_t)) = \arg\min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x} - \mathbf{x}_t + \eta\nabla f_t(\mathbf{x}_t)\|^2, \tag{2}$$

where $\Pi_{\mathcal{K}}(\cdot)$ denotes the projection onto $\mathcal{K}$ and $\eta > 0$ is a predefined step size. Despite the simplicity of the OGD algorithm, the computational cost per iteration is crucial for its applicability. For general convex domains, solving the optimization problem in (2) is an offline convex optimization problem by itself and can be computationally expensive. For example, when one envisions a positive semidefinitive cone in applications such as distance metric learning and matrix completion, the full eigen-decomposition of a matrix is required to project the updated solutions back into the cone. Recently several efficient algorithms have been developed for projection onto specific domains, for example, ℓ_1 ball (Duchi et al., 2008; Liu and Ye, 2009); however, when the domain $\mathcal{K}$ is complex, the projection step is a more involved task or computationally burdensome.

To tackle the computational challenge arising from the projection step, we consider an alternative online learning problem. Instead of requiring $\mathbf{x}_t \in \mathcal{K}$, we only require the constraints, which define the convex domain $\mathcal{K}$, to be satisfied in a long run. Then, the online learning problem becomes a task to find a sequence of solutions $\mathbf{x}_t, t \in [T]$ that minimizes the regret defined in (1), under the long term constraints, that is, $\sum_{t=1}^{T} \mathbf{x}_t/T \in \mathcal{K}$. We refer to this problem as **online learning with long term constraints**. In other words, instead of solving the projection problem in (2) on each round, we allow the learner to make decisions at some iterations which do not belong to the set $\mathcal{K}$, but the overall sequence of chosen decisions must obey the constraints at the end by a vanishing convergence rate.

From a different perspective, the proposed online optimization with long term constraints setup is reminiscent of regret minimization with side constraints or constrained regret minimization addressed in Mannor and Tsitsiklis (2006), motivated by applications in wireless communication. In regret minimization with side constraints, beyond minimizing regret, the learner has some side constraints that need to be satisfied on average for all rounds. Unlike our setting, in learning with side constraints, the set $\mathcal{K}$ is controlled by adversary and can vary arbitrarily from trial to trial. It has been shown that if the convex set is affected by both decisions and loss functions, the minimax optimal regret is generally unattainable online (Mannor et al., 2009).

One interesting application of the constrained regret minimization is multi-objective online classification where the learner aims at simultaneously optimizing more than one classification performance criteria. In the simple two objective online classification considered in Bernstein et al. (2010), the goal of the online classifier is to maximize the average true positive classification rate with an additional performance guarantee in terms of the false positive rate. Following the Neyman-Pearson risk, the intuitive approach to tackle this problem is to optimize one criterion (i.e., maximizing the true positive rate) subject to explicit constraint on the other objective (i.e., false positive rate) that needs to be satisfied on average for the sequence of decisions. The constrained regret matching (CRM) algorithm, proposed in Bernstein et al. (2010), efficiently solves this problem by relaxing the objective under mild assumptions on the single-stage constraint. The main idea of the CRM algorithm is to incorporate the penalty, that should be paid by the learner to satisfy the constraint, in the objective (i.e., true positive rate) by subtracting a positive constant at each decision step. It has been shown that the CRM algorithm *asymptotically* satisfies the average constraint (i.e., false positive rate) provided that the relaxation constant is above a certain threshold.

Finally, it is worth mentioning that the proposed setting can be used in certain classes of online learning such as online-to-batch conversion (Cesa-Bianchi et al., 2004), where it is sufficient to guarantee that constraints are satisfied in the long run. More specifically, under the assumption that received examples are i.i.d samples, the solution for batch learning is to average the solutions obtained over all the trials. As a result, if the long term constraint is satisfied, it is guaranteed that the average solution will belong to the domain $\mathcal{K}$.

In this paper, we describe and analyze a general framework for solving online convex optimization with long term constraints. We first show that a direct application of OGD fails to achieve a sub-linear bound on the violation of constraints and an $O(\sqrt{T})$ bound on the regret. Then, by turning the problem into an online convex-concave optimization problem, we propose an efficient algorithm which is an adaption of OGD for online learning with long term constraints. The proposed algorithm achieves the same $O(\sqrt{T})$ regret bound as the general setting and $O(T^{3/4})$ bound for the violation of constraints. We show that by using a simple trick we can turn the proposed method into an algorithm which exactly satisfies the constraints in the long run by achieving $O(T^{3/4})$ re-

gret bound. When the convex domain $\mathcal{K}$ can be described by a finite number of linear constraints, we propose an alternative algorithm based on the mirror prox method (Nemirovski, 2005), which achieves $O(T^{2/3})$ bound for both regret and the violation of constraints. Our framework also handles the cases when we do not have full access to the domain $\mathcal{K}$ except through a limited number of oracle evaluations. In the full-information version, the decision maker can observe the entire convex domain $\mathcal{K}$, whereas in a partial-information (a.k.a bandit setting) the decision maker may only observe the cost of the constraints defining the domain $\mathcal{K}$ at limited points. We show that we can generalize the proposed OGD based algorithm to this setting by only accessing the value oracle for domain $\mathcal{K}$ at two points, which achieves the same bounds in expectation as the case that has a full knowledge about the domain $\mathcal{K}$. In summary, the present work makes the following contributions:

- A general theorem that shows, in online setting, a simple penalty based method attains linear bound $O(T)$ for either the regret or the long term violation of the constraints and fails to achieve sub-linear bound for both regret and the long term violation of the constraints at the same time.
- A convex-concave formulation of online convex optimization with long term constraints, and an efficient algorithm based on OGD that attains a regret bound of $O(T^{1/2})$, and $O(T^{3/4})$ violation of the constraints.
- A modified OGD based algorithm for online convex optimization with long term constraints that has no constraint violation but $O(T^{3/4})$ regret bound.
- An algorithm for online convex optimization with long term constraints based on the mirror prox method that achieves $O(T^{2/3})$ regret and constraint violation.
- A multipoint bandit version of the basic algorithm with $O(T^{1/2})$ regret bound and $O(T^{3/4})$ violation of the constraints in *expectation* by accessing the value oracle for the convex set $\mathcal{K}$ at two points.

The remainder of the paper is structured as follows: In Section 3, we first examine a simple penalty based strategy and show that it fails to attain sub-linear bound for both regret and long term violation of the constraints. Then, we formulate regret minimization as an online convex-concave optimization problem and apply the OGD algorithm to solve it. Our first algorithm allows the constraints to be violated in a controlled way. It is then modified to have the constraints exactly satisfied in the long run. Section 4 presents our second algorithm which is an adaptation of the mirror prox method. Section 5 generalizes the online convex optimization with long term constraints problem to the setting where we only have a partial access to the convex domain $\mathcal{K}$. Section 6 concludes the work with a list of open questions.

2. Notation and Setting

Before proceeding, we define the notations used throughout the paper and state the assumptions made for the analysis of algorithms. Vectors are shown by lower case bold letters, such as $\mathbf{x} \in \mathbb{R}^d$. Matrices are indicated by upper case letters such as A and their pseudoinverse is represented by $A^\dagger$. We use $[m]$ as a shorthand for the set of integers $\{1, 2, \ldots, m\}$. Throughout the paper we denote by $\|\cdot\|$ and $\|\cdot\|_1$ the ℓ_2 (Euclidean) norm and ℓ_1-norm, respectively. We use $\mathbb{E}$ and $\mathbb{E}_t$ to

denote the expectation and conditional expectation with respect to all randomness in early $t-1$ trials, respectively. To facilitate our analysis, we assume that the domain $\mathcal{K}$ can be written as an intersection of a finite number of convex constraints, that is,

$$\mathcal{K} = \{\mathbf{x} \in \mathbb{R}^d : g_i(\mathbf{x}) \leq 0, i \in [m]\},$$

where $g_i(\cdot), i \in [m]$, are Lipschitz continuous functions. Like many other works for online convex optimization such as Flaxman et al. (2005), we assume that $\mathcal{K}$ is a bounded domain, that is, there exist constants $R > 0$ and $r < 1$ such that $\mathcal{K} \subseteq R\mathbb{B}$ and $r\mathbb{B} \subseteq \mathcal{K}$ where $\mathbb{B}$ denotes the unit ℓ_2 ball centered at the origin. For the ease of notation, we use $\mathcal{B} = R\mathbb{B}$.

We focus on the problem of online convex optimization, in which the goal is to achieve a low regret with respect to a fixed decision on a sequence of loss functions. The difference between the setting considered here and the general online convex optimization is that, in our setting, instead of requiring $\mathbf{x}_t \in \mathcal{K}$, or equivalently $g_i(\mathbf{x}_t) \leq 0, i \in [m]$, for all $t \in [T]$, we only require the constraints to be satisfied in the long run, namely $\sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq 0, i \in [m]$. Then, the problem becomes to find a sequence of solutions $\mathbf{x}_t, t \in [T]$ that minimizes the regret defined in (1), under the long term constraints $\sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq 0, i \in [m]$. Formally, we would like to solve the following optimization problem online,

$$\min_{\mathbf{x}_1,\ldots,\mathbf{x}_T \in \mathcal{B}} \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}) \;\; \text{s.t.} \;\; \sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq 0 \, , \, i \in [m]. \tag{3}$$

For simplicity, we will focus on a finite-horizon setting where the number of rounds T is known in advance. This condition can be relaxed under certain conditions, using standard techniques (see, e.g., Cesa-Bianchi and Lugosi, 2006). Note that in (3), (i) the solutions come from the ball $\mathcal{B} \supseteq \mathcal{K}$ instead of $\mathcal{K}$ and (ii) the constraint functions are fixed and are given in advance.

Like most online learning algorithms, we assume that both loss functions and the constraint functions are Lipschitz continuous, that is, there exists constants L_f and L_g such that $|f_t(\mathbf{x}) - f_t(\mathbf{x}')| \leq L_f \|\mathbf{x} - \mathbf{x}'\|$, $\;|g_i(\mathbf{x}) - g_i(\mathbf{x}')| \leq L_g \|\mathbf{x} - \mathbf{x}'\|$ for any $\mathbf{x} \in \mathcal{B}$ and $\mathbf{x}' \in \mathcal{B}$, $i \in [m]$. For simplicity of analysis, we use $G = \max\{L_f, L_g\}$ and

$$\begin{aligned} F &= \max_{t \in [T]} \max_{\mathbf{x}, \mathbf{x}' \in \mathcal{K}} f_t(\mathbf{x}) - f_t(\mathbf{x}') \leq 2L_f R, \\ D &= \max_{i \in [m]} \max_{\mathbf{x} \in \mathcal{B}} g_i(\mathbf{x}) \leq L_g R. \end{aligned}$$

Finally, we define the notion of a Bregman divergence. Let $\phi(\cdot)$ be a strictly convex function defined on a convex set $\mathcal{K}$. The Bregman divergence between $\mathbf{x}$ and $\mathbf{x}'$ is defined as $\mathsf{B}_\phi(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) - \phi(\mathbf{x}') - (\mathbf{x} - \mathbf{x}')^\top \nabla\phi(\mathbf{x}')$ which measures how much the function $\phi(\cdot)$ deviates at $\mathbf{x}$ from it's linear approximation at $\mathbf{x}'$.

3. Online Convex Optimization with Long Term Constraints

In this section we present and analyze our gradient descent based algorithms for online convex optimization problem with long term constraints. We first describe an algorithm which is allowed to violate the constraints and then, by applying a simple trick, we propose a variant of the first algorithm which exactly satisfies the constraints in the long run.

Before we state our formulation and algorithms, let us review a few alternative techniques that do not need explicit projection. A straightforward approach is to introduce an appropriate self-concordant barrier function for the given convex set $\mathcal{K}$ and add it to the objective function such that the barrier diverges at the boundary of the set. Then we can interpret the resulting optimization problem, on the modified objective functions, as an unconstrained minimization problem that can be solved without projection steps. Following the analysis in Abernethy et al. (2012), with an appropriately designed procedure for updating solutions, we could guarantee a regret bound of $O(\sqrt{T})$ without the violation of constraints. A similar idea is used in Abernethy et al. (2008) for online bandit learning and in Narayanan and Rakhlin (2010) for a random walk approach for regret minimization which, in fact, translates the issue of projection into the difficulty of sampling. Even for linear Lipschitz cost functions, the random walk approach requires sampling from a Gaussian distribution with covariance given by the Hessian of the self-concordant barrier of the convex set $\mathcal{K}$ that has the same time complexity as inverting a matrix. The main limitation with these approaches is that they require computing the Hessian matrix of the objective function in order to guarantee that the updated solution stays within the given domain $\mathcal{K}$. This limitation makes it computationally unattractive when dealing with high dimensional data. In addition, except for well known cases, it is often unclear how to efficiently construct a self-concordant barrier function for a general convex domain.

An alternative approach for online convex optimization with long term constraints is to introduce a penalty term in the loss function that penalizes the violation of constraints. More specifically, we can define a new loss function $\hat{f}_t(\cdot)$ as

$$\hat{f}_t(\mathbf{x}) = f_t(\mathbf{x}) + \delta \sum_{i=1}^{m} [g_i(\mathbf{x})]_+, \tag{4}$$

where $[z]_+ = \max(0, 1-z)$ and $\delta > 0$ is a fixed positive constant used to penalize the violation of constraints. We then run the standard OGD algorithm to minimize the modified loss function $\hat{f}_t(\cdot)$. The following theorem shows that this simple strategy fails to achieve sub-linear bound for both regret and the long term violation of constraints at the same time.

Theorem 1 *Given $\delta > 0$, there always exists a sequence of loss functions $\{f_t(\mathbf{x})\}_{t=1}^T$ and a constraint function $g(\mathbf{x})$ such that either $\sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{g(\mathbf{x})\leq 0} \sum_{t=1}^T f_t(\mathbf{x}) = O(T)$ or $\sum_{t=1}^T [g(\mathbf{x}_t)]_+ = O(T)$ holds, where $\{\mathbf{x}_t\}_{t=1}^T$ is the sequence of solutions generated by the OGD algorithm that minimizes the modified loss functions given in (4).*

We defer the proof to Appendix A along with a simple analysis of the OGD when applied to the modified functions in (4). The analysis shows that in order to obtain $O(\sqrt{T})$ regret bound, linear bound on the long term violation of the constraints is unavoidable. The main reason for the failure of using modified loss function in (4) is that the weight constant δ is fixed and independent from the sequence of solutions obtained so far. In the next subsection, we present an online convex-concave formulation for online convex optimization with long term constraints, which explicitly addresses the limitation of (4) by automatically adjusting the weight constant based on the violation of the solutions obtained so far.

As mentioned before, our general strategy is to turn online convex optimization with long term constraints into a convex-concave optimization problem. Instead of generating a sequence of solutions that satisfies the long term constraints, we first consider an online optimization strategy that

allows the violation of constraints on some rounds in a controlled way. We then modify the online optimization strategy to obtain a sequence of solutions that obeys the long term constraints. Although the online convex optimization with long term constraints is clearly easier than the standard online convex optimization problem, it is straightforward to see that optimal regret bound for online optimization with long term constraints should be on the order of $O(\sqrt{T})$, no better than the standard online convex optimization problem.

3.1 An Efficient Algorithm with $O(\sqrt{T})$ Regret Bound and $O(T^{3/4})$ Bound on the Violation of Constraints

The intuition behind our approach stems from the observation that the constrained optimization problem $\min_{\mathbf{x}\in\mathcal{K}}\sum_{t=1}^{T} f_t(\mathbf{x})$ is equivalent to the following convex-concave optimization problem

$$\min_{\mathbf{x}\in\mathcal{B}}\max_{\lambda\in\mathbb{R}^m_+}\sum_{t=1}^{T} f_t(\mathbf{x})+\sum_{i=1}^{m}\lambda_i g_i(\mathbf{x}), \tag{5}$$

where $\lambda=(\lambda_1,\ldots,\lambda_m)^\top$ is the vector of Lagrangian multipliers associated with the constraints $g_i(\cdot), i=1,\ldots,m$ and belongs to the nonnegative orthant $\mathbb{R}^m_+$. To solve the online convex-concave optimization problem, we extend the gradient based approach for variational inequality (Nemirovski, 1994) to (5). To this end, we consider the following regularized convex-concave function as

$$\mathcal{L}_t(\mathbf{x},\lambda)=f_t(\mathbf{x})+\sum_{i=1}^{m}\left\{\lambda_i g_i(\mathbf{x})-\frac{\delta\eta}{2}\lambda_i^2\right\}, \tag{6}$$

where $\delta>0$ is a constant whose value will be decided by the analysis. Note that in (6), we introduce a regularizer $\delta\eta\lambda_i^2/2$ to prevent λ_i from being too large. This is because, when λ_i is large, we may encounter a large gradient for $\mathbf{x}$ because of $\nabla_{\mathbf{x}}\mathcal{L}_t(\mathbf{x},\lambda)\propto\sum_{i=1}^{m}\lambda_i\nabla g_i(\mathbf{x})$, leading to unstable solutions and a poor regret bound. Although we can achieve the same goal by restricting λ_i to a bounded domain, using the quadratic regularizer makes it convenient for our analysis.

Algorithm 1 shows the detailed steps of the proposed algorithm. Unlike standard online convex optimization algorithms that only update $\mathbf{x}$, Algorithm 1 updates both $\mathbf{x}$ and λ. In addition, unlike the modified loss function in (4) where the weights for constraints $\{g_i(\mathbf{x})\le 0\}_{i=1}^m$ are fixed, Algorithm 1 automatically adjusts the weights $\{\lambda_i\}_{i=1}^m$ based on $\{g_i(\mathbf{x})\}_{i=1}^m$, the violation of constraints, as the game proceeds. It is this property that allows Algorithm 1 to achieve sub-linear bound for both regret and the violation of constraints.

To analyze Algorithm 1, we first state the following lemma, the key to the main theorem on the regret bound and the violation of constraints.

Lemma 2 *Let $\mathcal{L}_t(\cdot,\cdot)$ be the function defined in (6) which is convex in its first argument and concave in its second argument. Then for any $(\mathbf{x},\lambda)\in\mathcal{B}\times\mathbb{R}^m_+$ we have*

$$\begin{aligned}\mathcal{L}_t(\mathbf{x}_t,\lambda)-\mathcal{L}_t(\mathbf{x},\lambda_t)\le{}&\frac{1}{2\eta}(\|\mathbf{x}-\mathbf{x}_t\|^2+\|\lambda-\lambda_t\|^2-\|\mathbf{x}-\mathbf{x}_{t+1}\|^2-\|\lambda-\lambda_{t+1}\|^2)\\&+\frac{\eta}{2}(\|\nabla_{\mathbf{x}}\mathcal{L}_t(\mathbf{x}_t,\lambda_t)\|^2+\|\nabla_{\lambda}\mathcal{L}_t(\mathbf{x}_t,\lambda_t)\|^2).\end{aligned}$$

Proof Following the analysis of Zinkevich (2003), convexity of $\mathcal{L}_t(\cdot,\lambda)$ implies that

$$\mathcal{L}_t(\mathbf{x}_t,\lambda_t)-\mathcal{L}_t(\mathbf{x},\lambda_t)\le(\mathbf{x}_t-\mathbf{x})^\top\nabla_{\mathbf{x}}\mathcal{L}_t(\mathbf{x}_t,\lambda_t) \tag{7}$$

Algorithm 1 Gradient based Online Convex Optimization with Long Term Constraints

Input: constraints $g_i(\mathbf{x}) \leq 0, i \in [m]$, step size η, and constant $\delta > 0$
Initialization: $\mathbf{x}_1 = \mathbf{0}$ and $\lambda_1 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ **do**
  Submit solution $\mathbf{x}_t$
  Receive the convex function $f_t(\mathbf{x})$ and experience loss $f_t(\mathbf{x}_t)$
  Compute $\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t) = \nabla f_t(\mathbf{x}_t) + \sum_{i=1}^m \lambda_t^i \nabla g_i(\mathbf{x}_t)$ and $\nabla_{\lambda_i} \mathcal{L}_t(\mathbf{x}_t, \lambda_t) = g_i(\mathbf{x}_t) - \eta\delta\lambda_t^i$
  Update $\mathbf{x}_t$ and λ_t by

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{B}}\left(\mathbf{x}_t - \eta \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\right)$$
$$\lambda_{t+1} = \Pi_{[0,+\infty)^m}\left(\lambda_t + \eta \nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\right)$$

end for

and by concavity of $\mathcal{L}_t(\mathbf{x}, \cdot)$ we have

$$\mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}_t, \lambda_t) \leq (\lambda - \lambda_t)^\top \nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t). \tag{8}$$

Combining the inequalities (7) and (8) results in

$$\mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}, \lambda_t) \leq (\mathbf{x}_t - \mathbf{x})^\top \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t) - (\lambda - \lambda_t)^\top \nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t). \tag{9}$$

Using the update rule for $\mathbf{x}_{t+1}$ in terms of $\mathbf{x}_t$ and expanding, we get

$$\|\mathbf{x} - \mathbf{x}_{t+1}\|^2 \leq \|\mathbf{x} - \mathbf{x}_t\|^2 - 2\eta(\mathbf{x}_t - \mathbf{x})^\top \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t) + \eta^2 \|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2, \tag{10}$$

where the first inequality follows from the nonexpansive property of the projection operation. Expanding the inequality for $\|\lambda - \lambda_{t+1}\|^2$ in terms of λ_t and plugging back into the (9) with (10) establishes the desired inequality. ■

Proposition 3 *Let $\mathbf{x}_t$ and $\lambda_t, t \in [T]$ be the sequence of solutions obtained by Algorithm 1. Then for any $\mathbf{x} \in \mathcal{B}$ and $\lambda \in \mathbb{R}_+^m$, we have*

$$\begin{aligned} &\sum_{t=1}^T \mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}, \lambda_t) \\ &\leq \frac{R^2 + \|\lambda\|^2}{2\eta} + \frac{\eta T}{2}\left((m+1)G^2 + 2mD^2\right) + \frac{\eta}{2}\left((m+1)G^2 + 2m\delta^2\eta^2\right)\sum_{t=1}^T \|\lambda_t\|^2. \end{aligned} \tag{11}$$

Proof We first bound the gradient terms in the right hand side of Lemma 2. Using the inequality $(a_1 + a_2 + \ldots, a_n)^2 \leq n(a_1^2 + a_2^2 + \ldots + a_n^2)$, we have $\|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2 \leq (m+1)G^2\left(1 + \|\lambda_t\|^2\right)$ and $\|\nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2 \leq 2m(D^2 + \delta^2\eta^2\|\lambda_t\|^2)$. In Lemma 2, by adding the inequalities of all iterations, and using the fact $\|\mathbf{x}\| \leq R$ we complete the proof. ■

The following theorem bounds the regret and the violation of the constraints in the long run for Algorithm 1.

Theorem 4 *Define* $a = R\sqrt{(m+1)G^2+2mD^2}$. *Set* $\eta = R^2/[a\sqrt{T}]$. *Assume* T *is large enough such that* $2\sqrt{2}\eta(m+1) \leq 1$. *Choose* δ *such that* $\delta \geq (m+1)G^2+2m\delta^2\eta^2$. *Let* $\mathbf{x}_t, t \in [T]$ *be the sequence of solutions obtained by Algorithm 1. Then for the optimal solution* $\mathbf{x}_* = \min_{\mathbf{x}\in\mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x})$ *we have*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \leq a\sqrt{T} = O(T^{1/2}), \text{ and}$$
$$\sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq \sqrt{2\left(FT + a\sqrt{T}\right)\sqrt{T}\left(\frac{\delta R^2}{a} + \frac{ma}{R^2}\right)} = O(T^{3/4}).$$

Proof We begin by expanding (11) using (6) and rearranging the terms to get

$$\sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x})] + \sum_{i=1}^{m}\left\{\lambda_i \sum_{t=1}^{T} g_i(\mathbf{x}_t) - \sum_{t=1}^{T}\lambda_t^i g_i(\mathbf{x})\right\} - \frac{\delta\eta T}{2}\|\lambda\|^2$$
$$\leq -\frac{\delta\eta}{2}\sum_{t=1}^{T}\|\lambda_t\|^2 + \frac{R^2+\|\lambda\|^2}{2\eta} + \frac{\eta T}{2}\left((m+1)G^2+2mD^2\right)$$
$$+\frac{\eta}{2}\left((m+1)G^2+2m\delta^2\eta^2\right)\sum_{t=1}^{T}\|\lambda_t\|^2.$$

Since $\delta \geq (m+1)G^2+2m\delta^2\eta^2$, we can drop the $\|\lambda_t\|^2$ terms from both sides of the above inequality and obtain

$$\sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x})] + \sum_{i=1}^{m}\left\{\lambda_i\sum_{t=1}^{T} g_i(\mathbf{x}_t) - \left(\frac{\delta\eta T}{2} + \frac{m}{2\eta}\right)\lambda_i^2\right\}$$
$$\leq \sum_{i=1}^{m}\sum_{t=1}^{T}\lambda_t^i g_i(\mathbf{x}) + \frac{R^2}{2\eta} + \frac{\eta T}{2}\left((m+1)G^2+2mD^2)\right).$$

The left hand side of above inequality consists of two terms. The first term basically measures the difference between the cumulative loss of the Algorithm 1 and the optimal solution and the second term includes the constraint functions with corresponding Lagrangian multipliers which will be used to bound the long term violation of the constraints. By taking maximization for λ over the range $(0,+\infty)$, we get

$$\sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x})] + \sum_{i=1}^{m}\left\{\frac{\left[\sum_{t=1}^{T} g_i(\mathbf{x}_t)\right]_+^2}{2(\delta\eta T + m/\eta)} - \sum_{t=1}^{T}\lambda_t^i g_i(\mathbf{x})\right\}$$
$$\leq \frac{R^2}{2\eta} + \frac{\eta T}{2}\left((m+1)G^2+2mD^2)\right).$$

Since $\mathbf{x}_* \in \mathcal{K}$, we have $g_i(\mathbf{x}_*) \leq 0, i \in [m]$, and the resulting inequality becomes

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) + \sum_{i=1}^{m}\frac{\left[\sum_{t=1}^{T} g_i(\mathbf{x}_t)\right]_+^2}{2(\delta\eta T + m/\eta)} \leq \frac{R^2}{2\eta} + \frac{\eta T}{2}\left((m+1)G^2+2mD^2)\right).$$

The statement of the first part of the theorem follows by using the expression for η. The second part is proved by substituting the regret bound by its lower bound as $\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \geq -FT$. ■

Remark 5 *We observe that the introduction of quadratic regularizer $\delta\eta\|\lambda\|^2/2$ allows us to turn the expression $\lambda_i \sum_{t=1}^T g_i(\mathbf{x}_t)$ into $\left[\sum_{t=1}^T g_i(\mathbf{x}_t)\right]_+^2$, leading to the bound for the violation of the constraints. In addition, the quadratic regularizer defined in terms of λ allows us to work with unbounded λ because it cancels the contribution of the $\|\lambda_t\|$ terms from the loss function and the bound on the gradients $\|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x},\lambda)\|$. Note that the constraint for δ mentioned in Theorem 4 is equivalent to*

$$\frac{2}{1/(m+1)+\sqrt{(m+1)^{-2}-8G^2\eta^2}} \leq \delta \leq \frac{1/(m+1)+\sqrt{(m+1)^{-2}-8G^2\eta^2}}{4\eta^2}, \tag{12}$$

from which, when T is large enough (i.e., η is small enough), we can simply set $\delta = 2(m+1)G^2$ that will obey the constraint in (12).

By investigating Lemma 2, it turns out that the boundedness of the gradients is essential to obtain bounds for Algorithm 1 in Theorem 4. Although, at each iteration, λ_t is projected onto the $\mathbb{R}_+^m$, since $\mathcal{K}$ is a compact set and functions $f_t(\mathbf{x})$ and $g_i(\mathbf{x}), i \in [m]$ are convex, the boundedness of the functions implies that the gradients are bounded (Bertsekas et al., 2003, Proposition 4.2.3).

3.2 An Efficient Algorithm with $O(T^{3/4})$ Regret Bound and without Violation of Constraints

In this subsection we generalize Algorithm 1 such that the constrained are satisfied in a long run. To create a sequence of solutions $\{\mathbf{x}_t, t \in [T]\}$ that satisfies the long term constraints $\sum_{t=1}^T g_i(\mathbf{x}_t) \leq 0, i \in [m]$, we make two modifications to Algorithm 1. First, instead of handling all of the m constraints, we consider a single constraint defined as $g(\mathbf{x}) = \max_{i\in[m]} g_i(\mathbf{x})$. Apparently, by achieving zero violation for the constraint $g(\mathbf{x}) \leq 0$, it is guaranteed that all of the constraints $g_i(\cdot), i \in [m]$ are also satisfied in the long term. Furthermore, we change Algorithm 1 by modifying the definition of $\mathcal{L}_t(\cdot,\cdot)$ as

$$\mathcal{L}_t(\mathbf{x},\lambda) = f_t(\mathbf{x}) + \lambda(g(\mathbf{x})+\gamma) - \frac{\eta\delta}{2}\lambda^2, \tag{13}$$

where $\gamma > 0$ will be decided later. This modification is equivalent to considering the constraint $g(\mathbf{x}) \leq -\gamma$, a tighter constraint than $g(\mathbf{x}) \leq 0$. The main idea behind this modification is that by using a tighter constraint in our algorithm, the resulting sequence of solutions will satisfy the long term constraint $\sum_{t=1}^T g(\mathbf{x}_t) \leq 0$, even though the tighter constraint is violated in many trials.

Before proceeding, we state a fact about the Lipschitz continuity of the function $g(\mathbf{x})$ in the following proposition.

Proposition 6 *Assume that functions $g_i(\cdot), i \in [m]$ are Lipschitz continuous with constant G. Then, function $g(\mathbf{x}) = \max_{i\in[m]} g_i(\mathbf{x})$ is Lipschitz continuous with constant G, that is,*

$$|g(\mathbf{x}) - g(\mathbf{x}')| \leq G\|\mathbf{x}-\mathbf{x}'\| \text{ for any } \mathbf{x} \in \mathcal{B} \text{ and } \mathbf{x}' \in \mathcal{B}.$$

Proof First, we rewrite $g(\mathbf{x}) = \max_{i\in[m]} g_i(\mathbf{x})$ as $g(\mathbf{x}) = \max_{\alpha\in\Delta_m} \sum_{i=1}^m \alpha_i g_i(\mathbf{x})$ where Δ_m is the m-simplex, that is, $\Delta_m = \{\alpha \in \mathbb{R}_+^m; \sum_{i=1}^m \alpha_i = 1\}$. Then, we have

$$\begin{aligned}
|g(\mathbf{x}) - g(\mathbf{x}')| &= \left| \max_{\alpha\in\Delta_m} \sum_{i=1}^m \alpha_i g_i(\mathbf{x}) - \max_{\alpha\in\Delta_m} \sum_{i=1}^m \alpha_i g_i(\mathbf{x}') \right| \\
&\leq \max_{\alpha\in\Delta_m} \left| \sum_{i=1}^m \alpha_i g_i(\mathbf{x}) - \sum_{i=1}^m \alpha_i g_i(\mathbf{x}') \right| \\
&\leq \max_{\alpha\in\Delta_m} \sum_{i=1}^m \alpha_i \left| g_i(\mathbf{x}) - g_i(\mathbf{x}') \right| \leq G\|\mathbf{x} - \mathbf{x}'\|,
\end{aligned}$$

where the last inequality follows from the Lipschitz continuity of $g_i(\mathbf{x}), i \in [m]$. ■

To obtain a zero bound on the violation of constraints in the long run, we make the following assumption about the constraint function $g(\mathbf{x})$.

Assumption 1 *Let $\mathcal{K}' \subseteq \mathcal{K}$ be the convex set defined as $\mathcal{K}' = \{\mathbf{x} \in \mathbb{R}^d : g(\mathbf{x}) + \gamma \leq 0\}$ where $\gamma \geq 0$. We assume that the norm of the gradient of the constraint function $g(\mathbf{x})$ is lower bounded at the boundary of $\mathcal{K}'$, that is,*

$$\min_{g(\mathbf{x})+\gamma=0} \|\nabla g(\mathbf{x})\| \geq \sigma.$$

A direct consequence of **Assumption 1** is that by reducing the domain $\mathcal{K}$ to $\mathcal{K}'$, the optimal value of the constrained optimization problem $\min_{\mathbf{x}\in\mathcal{K}} f(\mathbf{x})$ does not change much, as revealed by the following theorem.

Theorem 7 *Let $\mathbf{x}_*$ and $\mathbf{x}_\gamma$ be the optimal solutions to the constrained optimization problems defined as $\min_{g(\mathbf{x})\leq 0} f(\mathbf{x})$ and $\min_{g(\mathbf{x})\leq -\gamma} f(\mathbf{x})$, respectively, where $f(\mathbf{x}) = \sum_{t=1}^T f_t(\mathbf{x})$ and $\gamma \geq 0$. We have*

$$|f(\mathbf{x}_*) - f(\mathbf{x}_\gamma)| \leq \frac{G}{\sigma}\gamma T.$$

Proof We note that the optimization problem $\min_{g(\mathbf{x})\leq -\gamma} f(\mathbf{x}) = \min_{g(\mathbf{x})\leq -\gamma} \sum_{t=1}^T f_t(\mathbf{x})$, can also be written in the minimax form as

$$f(\mathbf{x}_\gamma) = \min_{\mathbf{x}\in\mathcal{B}} \max_{\lambda\in\mathbb{R}_+} \sum_{t=1}^T f_t(\mathbf{x}) + \lambda(g(\mathbf{x}) + \gamma), \tag{14}$$

where we use the fact that $\mathcal{K}' \subseteq \mathcal{K} \subseteq \mathcal{B}$. We denote by $\mathbf{x}_\gamma$ and λ_γ the optimal solutions to (14). We have

$$\begin{aligned}
f(\mathbf{x}_\gamma) &= \min_{\mathbf{x}\in\mathcal{B}} \max_{\lambda\in\mathbb{R}_+} \sum_{t=1}^T f_t(\mathbf{x}) + \lambda(g(\mathbf{x}) + \gamma) \\
&= \min_{\mathbf{x}\in\mathcal{B}} \sum_{t=1}^T f_t(\mathbf{x}) + \lambda_\gamma(g(\mathbf{x}) + \gamma) \\
&\leq \sum_{t=1}^T f_t(\mathbf{x}_*) + \lambda_\gamma(g(\mathbf{x}_*) + \gamma) \leq \sum_{t=1}^T f_t(\mathbf{x}_*) + \lambda_\gamma\gamma,
\end{aligned}$$

where the second equality follows the definition of the $\mathbf{x}_\gamma$ and the last inequality is due to the optimality of $\mathbf{x}_*$, that is, $g(\mathbf{x}_*) \leq 0$.
To bound $|f(\mathbf{x}_\gamma) - f(\mathbf{x}_*)|$, we need to bound λ_γ. Since $\mathbf{x}_\gamma$ is the minimizer of (14), from the optimality condition we have

$$-\sum_{t=1}^{T} \nabla f_t(\mathbf{x}_\gamma) = \lambda_\gamma \nabla g(\mathbf{x}_\gamma). \tag{15}$$

By setting $\mathbf{v} = -\sum_{t=1}^{T} \nabla f_t(\mathbf{x}_\gamma)$, we can simplify (15) as $\lambda_\gamma \nabla g(\mathbf{x}_\gamma) = \mathbf{v}$. From the KKT optimality condition (Boyd and Vandenberghe, 2004), if $g(\mathbf{x}_\gamma) + \gamma < 0$ then we have $\lambda_\gamma = 0$; otherwise according to Assumption 1 we can bound λ_γ by

$$\lambda_\gamma \leq \frac{\|\mathbf{v}\|}{\|\nabla g(\mathbf{x}_\gamma)\|} \leq \frac{GT}{\sigma}.$$

We complete the proof by applying the fact $f(\mathbf{x}_*) \leq f(\mathbf{x}_\gamma) \leq f(\mathbf{x}_*) + \lambda_\gamma \gamma$. ■

As indicated by Theorem 7, when γ is small, we expect the difference between two optimal values $f(\mathbf{x}_*)$ and $f(\mathbf{x}_\gamma)$ to be small. Using the result from Theorem 7, in the following theorem, we show that by running Algorithm 1 on the modified convex-concave functions defined in (13), we are able to obtain an $O(T^{3/4})$ regret bound and zero bound on the violation of constraints in the long run.

Theorem 8 *Set $a = 2R/\sqrt{2G^2 + 3(D^2+b^2)}$, $\eta = R^2/[a\sqrt{T}]$, and $\delta = 4G^2$. Let $\mathbf{x}_t, t \in [T]$ be the sequence of solutions obtained by Algorithm 1 with functions defined in (13) with $\gamma = bT^{-1/4}$ and $b = 2\sqrt{F(\delta R^2 a^{-1} + aR^{-2})}$. Let $\mathbf{x}_*$ be the optimal solution to $\min_{\mathbf{x}\in\mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x})$. With sufficiently large T, that is, $FT \geq a\sqrt{T}$, and under Assumption 1, we have $\mathbf{x}_t, t \in [T]$ satisfy the global constraint $\sum_{t=1}^{T} g(\mathbf{x}_t) \leq 0$ and the regret $\mathfrak{R}_T$ is bounded by*

$$\mathfrak{R}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \leq a\sqrt{T} + \frac{b}{\sigma} GT^{3/4} = O(T^{3/4}).$$

Proof Let $\mathbf{x}_\gamma$ be the optimal solution to $\min_{g(\mathbf{x})\leq -\gamma} \sum_{t=1}^{T} f_t(\mathbf{x})$. Similar to the proof of Theorem 4 when applied to functions in (13) we have

$$\begin{aligned}
&\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}) + \lambda \sum_{t=1}^{T} (g(\mathbf{x}_t) + \gamma) - \left(\sum_{t=1}^{T} \lambda_t\right)(g(\mathbf{x}) + \gamma) - \frac{\delta\eta T}{2}\lambda^2 \\
&\leq \quad -\frac{\delta\eta}{2}\sum_{t=1}^{T}\lambda_t^2 + \frac{R^2 + \lambda^2}{2\eta} + \frac{\eta T}{2}\left(2G^2 + 3(D^2 + \gamma^2)\right) + \frac{\eta}{2}\left(2G^2 + 3\delta^2\eta^2\right)\sum_{t=1}^{T}\lambda_t^2.
\end{aligned}$$

By setting $\delta \geq 2G^2 + 3\delta^2\eta^2$ which is satisfied by $\delta = 4G^2$, we cancel the terms including λ_t from the right hand side of above inequality. By maximizing for λ over the range $(0, +\infty)$ and noting that $\gamma \leq b$, for the optimal solution $\mathbf{x}_\gamma$, we have

$$\sum_{t=1}^{T} \left[f_t(\mathbf{x}_t) - f_t(\mathbf{x}_\gamma)\right] + \frac{\left[\sum_{t=1}^{T} g(\mathbf{x}_t) + \gamma T\right]_+^2}{2(\delta\eta T + 1/\eta)} \leq \frac{R^2}{2\eta} + \frac{\eta T}{2}\left(2G^2 + 3(D^2 + b^2)\right),$$

which, by optimizing for η and applying the lower bound for the regret as $\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_\gamma) \geq -FT$, yields the following inequalities

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_\gamma) \leq a\sqrt{T} \tag{16}$$

and

$$\sum_{t=1}^{T} g(\mathbf{x}_t) \leq \sqrt{2\left(FT + a\sqrt{T}\right)\sqrt{T}\left(\frac{\delta R^2}{a} + \frac{a}{R^2}\right)} - \gamma T, \tag{17}$$

for the regret and the violation of the constraint, respectively. Combining (16) with the result of Theorem 7 results in $\sum_{t=1}^{T} f_t(\mathbf{x}_\gamma) \leq \sum_{t=1}^{T} f_t(\mathbf{x}_*) + a\sqrt{T} + (G/\sigma)\gamma T$. By choosing $\gamma = bT^{-1/4}$ we attain the desired regret bound as

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \leq a\sqrt{T} + \frac{bG}{\sigma}T^{3/4} = O(T^{3/4}).$$

To obtain the bound on the violation of constraints, we note that in (17), when T is sufficiently large, that is, $FT \geq a\sqrt{T}$, we have $\sum_{t=1}^{T} g(\mathbf{x}_t) \leq 2\sqrt{F(\delta R^2 a^{-1} + aR^{-2})}T^{3/4} - bT^{3/4}$. Choosing $b = 2\sqrt{F(\delta R^2 a^{-1} + aR^{-2})}T^{3/4}$ guarantees the zero bound on the violation of constraints as claimed. ■

4. A Mirror Prox Based Approach

The bound for the violation of constraints for Algorithm 1 is unsatisfactory since it is significantly worse than $O(\sqrt{T})$. In this section, we pursue a different approach that is based on the mirror prox method in Nemirovski (2005) to improve the bound for the violation of constraints. The basic idea is that solving (5) can be reduced to the problem of approximating a saddle point $(\mathbf{x}, \lambda) \in \mathcal{B} \times [0, \infty)^m$ by solving the associated variational inequality.

We first define an auxiliary function $\mathcal{F}(\mathbf{x}, \lambda)$ as

$$\mathcal{F}(\mathbf{x}, \lambda) = \sum_{i=1}^{m} \left\{ \lambda_i g_i(\mathbf{x}) - \frac{\delta\eta}{2}\lambda_i^2 \right\}.$$

In order to successfully apply the mirror prox method, we follow the fact that any convex domain can be written as an intersection of linear constraints, and make the following assumption:

Assumption 2 *We assume that $g_i(\mathbf{x}), i \in [m]$ are linear, that is, $\mathcal{K} = \{\mathbf{x} \in \mathbb{R}^d : g_i(\mathbf{x}) = \mathbf{x}^\top \mathbf{a}_i - b_i \leq 0, i \in [m]\}$ where $\mathbf{a}_i \in \mathbb{R}^d$ is a normalized vector with $\|\mathbf{a}_i\| = 1$ and $b_i \in \mathbb{R}$.*

The following proposition shows that under Assumptions 2, the function $\mathcal{F}(\mathbf{x}, \lambda)$ has Lipschitz continuous gradient, a basis for the application of the mirror prox method.

Proposition 9 *Under Assumption 2, $\mathcal{F}(\mathbf{x}, \lambda)$ has Lipschitz continuous gradient, that is,*

$$\left\|\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{x}, \lambda) - \nabla_{\mathbf{x}'} \mathcal{F}(\mathbf{x}', \lambda')\right\|^2 + \left\|\nabla_{\lambda} \mathcal{F}(\mathbf{x}, \lambda) - \nabla_{\lambda'} \mathcal{F}(\mathbf{x}', \lambda')\right\|^2 \leq 2(m + \delta^2\eta^2)(\|\mathbf{x} - \mathbf{x}'\|^2 + \|\lambda - \lambda'\|^2).$$

Algorithm 2 Prox Method with Long Term Constraints

Input: constraints $g_i(\mathbf{x}) \leq 0, i \in [m]$, step size η, and constant δ
Initialization: $\mathbf{z}_1 = \mathbf{0}$ and $\mu_1 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ **do**
  Compute the solution for $\mathbf{x}_t$ and λ_t as

$$\mathbf{x}_t = \Pi_{\mathcal{B}}\left(\mathbf{z}_t - \eta \nabla_{\mathbf{x}} \mathcal{F}(\mathbf{z}_t, \mu_t)\right)$$
$$\lambda_t = \Pi_{[0,+\infty)^m}\left(\mu_t + \eta \nabla_{\lambda} \mathcal{F}(\mathbf{z}_t, \mu_t)\right)$$

  Submit solution $\mathbf{x}_t$
  Receive the convex function $f_t(\mathbf{x})$ and experience loss $f_t(\mathbf{x}_t)$
  Compute $\mathcal{L}_t(\mathbf{x}, \lambda) = f_t(\mathbf{x}) + \mathcal{F}(\mathbf{x}, \lambda) = f_t(\mathbf{x}) + \sum_{i=1}^m \left\{\lambda_i g_i(\mathbf{x}) - \frac{\delta\eta}{2}\lambda_i^2\right\}$
  Update $\mathbf{z}_t$ and μ_t by

$$\mathbf{z}_{t+1} = \Pi_{\mathcal{B}}\left(\mathbf{z}_t - \eta \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\right)$$
$$\mu_{t+1} = \Pi_{[0,+\infty)^m}\left(\mu_t + \eta \nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\right)$$

end for

Proof

$$\begin{aligned}
& \left\|\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{x}, \lambda) - \nabla_{\mathbf{x}'} \mathcal{F}(\mathbf{x}', \lambda')\right\|^2 + \left\|\nabla_{\lambda} \mathcal{F}(\mathbf{x}, \lambda) - \nabla_{\lambda'} \mathcal{F}(\mathbf{x}', \lambda')\right\|^2 \\
= & \left\|\sum_{i=1}^m (\lambda_i - \lambda_i')\mathbf{a}_i\right\|^2 + \left\|\sum_{i=1}^m \mathbf{a}_i^\top (\mathbf{x} - \mathbf{x}') + \delta\eta \sum_{i=1}^m (\lambda_i' - \lambda_i)\right\|^2 \\
\leq & \|A^\top(\lambda - \lambda')\|^2 + 2\|A(\mathbf{x} - \mathbf{x}')\|^2 + 2\delta^2\eta^2\|\lambda - \lambda'\|^2 \\
\leq & 2\sigma_{\max}^2(A)\|\mathbf{x} - \mathbf{x}'\|^2 + (\sigma_{\max}^2(A) + 2\delta^2\eta^2)\|\lambda - \lambda'\|^2.
\end{aligned}$$

Since

$$\sigma_{\max}(A) = \sqrt{\lambda_{\max}(AA^\top)} \leq \sqrt{\mathrm{Tr}(AA^\top)} \leq \sqrt{m},$$

we have $\sigma_{\max}^2(A) \leq m$, leading to the desired result. ■

Algorithm 2 shows the detailed steps of the mirror prox based algorithm for online convex optimization with long term constraints defined in (5). Compared to Algorithm 1, there are two key features of Algorithm 2. First, it introduces auxiliary variables $\mathbf{z}_t$ and μ_t besides the variables $\mathbf{x}_t$ and λ_t. At each iteration t, it first computes the solutions $\mathbf{x}_t$ and λ_t based on the auxiliary variables $\mathbf{z}_t$ and μ_t; it then updates the auxiliary variables based on the gradients computed from $\mathbf{x}_t$ and λ_t. Second, two different functions are used for updating $(\mathbf{x}_t, \lambda_t)$ and $(\mathbf{z}_t, \mu_t)$: function $\mathcal{F}(\mathbf{x}, \lambda)$ is used for computing the solutions $\mathbf{x}_t$ and λ_t, while function $\mathcal{L}_t(\mathbf{x}, \lambda)$ is used for updating the auxiliary variables $\mathbf{z}_t$ and μ_t.

Our analysis is based on the Lemma 3.1 from Nemirovski (2005) which is restated here for completeness.

Lemma 10 *Let* $\mathsf{B}(\mathbf{x}, \mathbf{x}')$ *be a Bregman distance function that has modulus* α *with respect to a norm* $\|\cdot\|$*, that is,* $\mathsf{B}(\mathbf{x}, \mathbf{x}') \geq \alpha\|\mathbf{x} - \mathbf{x}'\|^2/2$*. Given* $\mathbf{u} \in \mathcal{B}$*,* $\mathbf{a}$*, and* $\mathbf{b}$*, we set*

$$\mathbf{w} = \arg\min_{\mathbf{x} \in \mathcal{B}} \mathbf{a}^\top(\mathbf{x} - \mathbf{u}) + \mathsf{B}(\mathbf{x}, \mathbf{u}), \ \mathbf{u}_+ = \arg\min_{\mathbf{x} \in \mathcal{B}} \mathbf{b}^\top(\mathbf{x} - \mathbf{u}) + \mathsf{B}(\mathbf{x}, \mathbf{u}).$$

Then for any $\mathbf{x} \in \mathcal{B}$ *and* $\eta > 0$*, we have*

$$\eta \mathbf{b}^\top (\mathbf{w} - \mathbf{x}) \leq \mathsf{B}(\mathbf{x}, \mathbf{u}) - \mathsf{B}(\mathbf{x}, \mathbf{u}_+) + \frac{\eta^2}{2\alpha} \|\mathbf{a} - \mathbf{b}\|_*^2 - \frac{\alpha}{2} \left[\|\mathbf{w} - \mathbf{u}\|^2 + \|\mathbf{w} - \mathbf{u}_+\|^2 \right].$$

We equip $\mathcal{B} \times [0, +\infty)^m$ with the norm $\|\cdot\|$ defined as

$$\|(\mathbf{z}, \mu)\|^2 = \frac{\|\mathbf{z}\|^2 + \|\mu\|^2}{2},$$

where $\|\cdot\|^2$ is the Euclidean norm defined separately for each domain. It is immediately seen that the Bregman distance function defined as

$$\mathsf{B}(\mathbf{z}_t, \mu_t, \mathbf{z}_{t+1}, \mu_{t+1}) = \frac{1}{2} \|\mathbf{z}_t - \mathbf{z}_{t+1}\|^2 + \frac{1}{2} \|\mu_t - \mu_{t+1}\|^2$$

is $\alpha = 1$ modules with respect to the norm $\|\cdot\|$.

To analyze the mirror prox algorithm, we begin with a simple lemma which is the direct application of Lemma 10 when applied to the updating rules of Algorithm 3.

Lemma 11 *If* $\eta(m + \delta^2 \eta^2) \leq \frac{1}{4}$ *holds, we have*

$$\begin{aligned} &\mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}, \lambda_t) \\ &\leq \frac{\|\mathbf{x} - \mathbf{z}_t\|^2 - \|\mathbf{x} - \mathbf{z}_{t+1}\|^2}{2\eta} + \frac{\|\lambda - \mu_t\|^2 - \|\lambda - \mu_{t+1}\|^2}{2\eta} + \eta \|\nabla f_t(\mathbf{x}_t)\|^2. \end{aligned}$$

Proof To apply Lemma 10, we define $\mathbf{u}$, $\mathbf{w}$, $\mathbf{u}_+$, $\mathbf{a}$ and $\mathbf{b}$ as follows

$$\begin{aligned} &\mathbf{u} = (\mathbf{z}_t, \mu_t), \mathbf{u}_+ = (\mathbf{z}_{t+1}, \mu_{t+1}), \mathbf{w} = (\mathbf{x}_t, \lambda_t), \\ &\mathbf{a} = (\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{z}_t, \mu_t), -\nabla_\lambda \mathcal{F}(\mathbf{z}_t, \mu_t)), \mathbf{b} = (\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t), -\nabla_\lambda \mathcal{L}_t(\mathbf{x}_t, \lambda_t)). \end{aligned}$$

Using Lemmas 2 and 10, we have

$$\begin{aligned} &\mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}, \lambda_t) - \frac{\|\mathbf{x} - \mathbf{z}_t\|^2 - \|\mathbf{x} - \mathbf{z}_{t+1}\|^2}{2\eta} - \frac{\|\lambda - \mu_t\|^2 - \|\lambda - \mu_{t+1}\|^2}{2\eta} \\ &\leq \underbrace{\frac{\eta}{2} \left\{ \|\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{z}_t, \mu_t) - \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2 + \|\nabla_\lambda \mathcal{F}(\mathbf{z}_t, \mu_t) - \nabla_\lambda \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2 \right\}}_{\mathbf{I}} \\ &\quad - \underbrace{\frac{1}{2} \left\{ \|\mathbf{z}_t - \mathbf{x}_t\|^2 + \|\mu_t - \lambda_t\|^2 \right\}}_{\mathbf{II}}. \end{aligned}$$

By expanding the gradient terms and applying the inequality $(a+b)^2 \leq 2(a^2 + b^2)$, we upper bound (**I**) as:

$$\begin{aligned} (\mathbf{I}) &= \frac{\eta}{2} \{ 2\|\nabla f_t(\mathbf{x}_t)\|^2 + 2\|\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{z}_t, \mu_t) - \nabla_{\mathbf{x}} \mathcal{F}(\mathbf{x}_t, \lambda_t)\|^2 + \|\nabla_\lambda \mathcal{F}(\mathbf{z}_t, \mu_t) - \nabla_\lambda \mathcal{F}(\mathbf{x}_t, \lambda_t)\|^2 \} \\ &\leq \eta \|\nabla f_t(\mathbf{x}_t)\|^2 + \eta \left\{ \|\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{z}_t, \mu_t) - \nabla_{\mathbf{x}} \mathcal{F}(\mathbf{x}_t, \lambda_t)\|^2 + \|\nabla_\lambda \mathcal{F}(\mathbf{x}_t, \lambda_t) - \nabla_\lambda \mathcal{F}(\mathbf{x}_t, \lambda_t)\|^2 \right\} \\ &\leq \eta \|\nabla f_t(\mathbf{x}_t)\|^2 + 2\eta(m + \delta^2 \eta^2) \left\{ \|\mathbf{z}_t - \mathbf{x}_t\|^2 + \|\mu_t - \lambda_t\|^2 \right\}, \end{aligned} \tag{18}$$

where the last inequality follows from Proposition 9. Combining (**II**) with (18) results in

$$\begin{aligned}\mathcal{L}_t(\mathbf{x}_t,\lambda)-\mathcal{L}_t(\mathbf{x},\lambda_t)-\frac{\|\mathbf{x}-\mathbf{z}_t\|^2-\|\mathbf{x}-\mathbf{z}_{t+1}\|^2}{2\eta}-\frac{\|\lambda-\mu_t\|^2-\|\lambda-\mu_{t+1}\|^2}{2\eta}\\ \leq \quad \eta\|\nabla f_t(\mathbf{x}_t)\|_2^2+\left(2\eta(m+\delta^2\eta^2)-\frac{1}{2}\right)\left\{\|\mathbf{z}_t-\mathbf{x}_t\|^2+\|\mu_t-\lambda_t\|_2^2\right\}.\end{aligned}$$

We complete the proof by rearranging the terms and setting $\eta(m+\delta^2\eta^2)\leq\frac{1}{4}$. ■

Theorem 12 *Set $\eta=T^{-1/3}$ and $\delta=T^{-2/3}$. Let $\mathbf{x}_t, t\in[T]$ be the sequence of solutions obtained by Algorithm 2. Then for $T\geq 164(m+1)^3$ we have*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t)-f_t(\mathbf{x}_*)\leq O(T^{2/3}) \text{ and } \sum_{t=1}^{T} g_i(\mathbf{x}_t)\leq O(T^{2/3}).$$

Proof Similar to the proof of Theorem 4, by summing the bound in Lemma 11 for all rounds $t=1,\cdots,T$, and taking maximization for λ we have the following inequality for any $\mathbf{x}_*\in\mathcal{K}$,

$$\sum_{t=1}^{T}[f_t(\mathbf{x}_t)-f_t(\mathbf{x}_*)]+\sum_{i=1}^{m}\frac{\left[\sum_{t=1}^{T} g_i(\mathbf{x}_t)\right]_+^2}{2(\delta\eta T+m/\eta)}\leq\frac{R^2}{2\eta}+\frac{\eta T}{2}G^2.$$

By setting $\delta=\frac{1}{\eta T}$ and using the fact that $\sum_{t=1}^{T} f_t(\mathbf{x}_t)-f_t(\mathbf{x}_*)\geq -FT$ we have:

$$\sum_{t=1}^{T}[f_t(\mathbf{x}_t)-f_t(\mathbf{x})]\leq\frac{R^2}{2\eta}+\frac{\eta T}{2}G^2$$

and

$$\sum_{t=1}^{T} g_i(\mathbf{x}_t)\leq\sqrt{(1+\frac{m}{\eta})\left(\frac{R^2}{\eta}+\eta TG^2+FT\right)}.$$

Substituting the stated value for η, we get the desired bounds as mentioned in the theorem. Note that the condition $\eta(m+\delta^2\eta^2)\leq\frac{1}{4}$ in Lemma 11 is satisfied for the stated values of η and δ as long as $T\geq 164(m+1)^3$. ■

Using the same trick as Theorem 8, by introducing appropriate γ, we will be able to establish the solutions that exactly satisfy the constraints in the long run with an $O(T^{2/3})$ regret bound as shown in the following corollary. In the case when all the constraints are linear, that is, $g_i(\mathbf{x})=\mathbf{a}_i^\top\mathbf{x}\leq b_i, i\in[m]$, Assumption 1 is simplified into the following condition,

$$\min_{\alpha\in\Delta_m}\left\|\sum_{i=1}^{m}\alpha_i\mathbf{a}_i\right\|\geq\sigma, \tag{19}$$

where Δ_m is a m dimensional simplex, that is, $\Delta_m=\{\alpha\in\mathbb{R}_+^m:\sum_{i=1}^{m}\alpha_i=1\}$. This is because $g(\mathbf{x})=\max_{\alpha\in\Delta_m}\sum_{i=1}^{m}\alpha_i g_i(\mathbf{x})$ and as a result, the (sub)gradient of $g(\mathbf{x})$ can always be written as $\partial g(\mathbf{x})=\sum_{i=1}^{m}\alpha_i\nabla g_i(\mathbf{x})=\sum_{i=1}^{m}\alpha_i\mathbf{a}_i$ where $\alpha\in\Delta_m$. As an illustrative example, consider the case when the norm vectors $\mathbf{a}_i, i\in[m]$ are linearly independent. In this case the condition mentioned in (19) obviously holds which indicates that the assumption does not limit the applicability of the proposed algorithm.

Corollary 13 *Let $\eta = \delta = T^{-1/3}$. Let $\mathbf{x}_t, t \in [T]$ be the sequence of solutions obtained by Algorithm 2 with $\gamma = bT^{-1/3}$ and $b = 2\sqrt{F}$. With sufficiently large T, that is, $FT \geq R^2T^{1/3} + G^2T^{2/3}$, under Assumptions 2 and condition in (19), we have $\mathbf{x}_t, t \in [T]$ satisfy the global constraints $\sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq 0, i \in [m]$ and the regret $\mathfrak{R}_T$ is bounded by*

$$\mathfrak{R}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \leq \frac{R^2}{2}T^{1/3} + \left(\frac{G^2}{2} + \frac{2G\sqrt{F}}{\sigma}\right)T^{2/3} = O(T^{2/3}).$$

The proof is similar to that of Theorem 8 and we defer it to Appendix B. As indicated by Corollary 13, for any convex domain defined by a finite number of halfspaces, that is, Polyhedral set, one can easily replace the projection onto the Polyhedral set with the ball containing the Polyhedral at the price of satisfying the constraints in the long run and achieving $O(T^{2/3})$ regret bound.

5. Online Convex Optimization with Long Term Constraints under Bandit Feedback for Domain

We now turn to extending the gradient based convex-concave optimization algorithm discussed in Section 3 to the setting where the learner only receives partial feedback for constraints. More specifically, the exact definition of the domain $\mathcal{K}$ is not exposed to the learner, only that the solution is within a ball $\mathcal{B}$. Instead, after receiving a solution $\mathbf{x}_t$, the oracle will present the learner with the convex loss function $f_t(\mathbf{x})$ and the maximum violation of the constraints for $\mathbf{x}_t$, that is, $g(\mathbf{x}_t) = \max_{i\in[m]} g_i(\mathbf{x}_t)$. We remind that the function $g(\mathbf{x})$ defined in this way is Lipschitz continuous with constant G as proved in Proposition 6. In this setting, the convex-concave function defined in (6) becomes as

$$\mathcal{L}_t(\mathbf{x}, \lambda) = f_t(\mathbf{x}) + \lambda g(\mathbf{x}) - (\delta\eta/2)\lambda^2.$$

The mentioned setting is closely tied to the bandit online convex optimization. In the bandit setting, in contrast to the full information setting, only the cost of the chosen decision (i.e., the incurred loss $f_t(\mathbf{x}_t)$) is revealed to the algorithm, not the function itself. There is a rich body of literature that deals with the bandit online convex optimization. In the seminal papers of Flaxman et al. (2005) and Awerbuch and Kleinberg (2004) it has been shown that one could design algorithms with $O(T^{3/4})$ regret bound even in the bandit setting where only evaluations of the loss functions are revealed at a single point. If we specialize to the online bandit optimization of linear loss functions, Dani et al. (2007) proposed an inefficient algorithm with $O(\sqrt{T})$ regret bound and Abernethy et al. (2008) obtained $O(\sqrt{T \log T})$ bound by an efficient algorithm if the convex set admits an efficiently computable self-concordant barrier. For general convex loss functions, Agarwal et al. (2010) proposed optimal algorithms in a new bandit setting, in which multiple points can be queried for the cost values. By using multiple evaluations, they showed that the modified online gradient descent algorithm can achieve $O(\sqrt{T})$ regret bound in expectation.

Algorithm 3 gives a complete description of the proposed algorithm under the bandit setting, which is a slight modification of Algorithm 1. Algorithm 3 accesses the constraint function $g(\mathbf{x})$ at two points. To facilitate the analysis, we define

$$\widehat{\mathcal{L}}_t(\mathbf{x}, \lambda) = f_t(\mathbf{x}) + \lambda \hat{g}(\mathbf{x}) - \frac{\eta\delta}{2}\lambda^2,$$

Algorithm 3 Multipoint Bandit Online Convex Optimization with Long Term Constraints

Input: constraint $g(\mathbf{x})$, step size η, constant $\delta > 0$, exploration parameter $\zeta > 0$, and shrinkage coefficient ξ
Initialization: $\mathbf{x}_1 = \mathbf{0}$ and $\lambda_1 = 0$
for $t = 1, 2, \ldots, T$ **do**
  Submit solution $\mathbf{x}_t$
  Select unit vector $\mathbf{u}_t$ uniformly at random
  Query $g(\mathbf{x})$ at points $\mathbf{x}_t + \zeta\mathbf{u}_t$ and $\mathbf{x}_t - \zeta\mathbf{u}_t$ and incur average of them as violation of constraints
  Compute $\tilde{g}_{\mathbf{x},t} = \nabla f_t(\mathbf{x}_t) + \lambda_t \left[\frac{d}{2\zeta}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) - g(\mathbf{x}_t - \zeta\mathbf{u}_t))\mathbf{u}_t\right]$
  Compute $\tilde{g}_{\lambda,t} = \frac{1}{2}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) + g(\mathbf{x}_t - \zeta\mathbf{u}_t)) - \eta\delta\lambda_t$
  Receive the convex function $f_t(\mathbf{x})$ and experience loss $f_t(\mathbf{x}_t)$
  Update $\mathbf{x}_t$ and λ_t by

$$\mathbf{x}_{t+1} = \Pi_{(1-\xi)\mathcal{B}}(\mathbf{x}_t - \eta\tilde{g}_{\mathbf{x},t})$$
$$\lambda_{t+1} = \Pi_{[0,+\infty)}(\lambda_t + \eta\tilde{g}_{\lambda,t})$$

end for

where $\hat{g}(\mathbf{x})$ is the smoothed version of $g(\mathbf{x})$ defined as $\hat{g}(\mathbf{x}) = \mathbb{E}_{\mathbf{v}\in\mathbb{S}}[\frac{d}{\zeta}g(\mathbf{x}+\zeta\mathbf{v})\mathbf{v}]$ at point $\mathbf{x}_t$ where $\mathbb{S}$ denotes the unit sphere centered at the origin. Note that $\hat{g}(\mathbf{x})$ is Lipschitz continuous with the same constant G, and it is always differentiable even though $g(\mathbf{x})$ is not in our case.

Since we do not have access to the function $\hat{g}(\cdot)$ to compute $\nabla_{\mathbf{x}}\widehat{\mathcal{L}}(\mathbf{x},\lambda)$, we need a way to estimate its gradient at point $\mathbf{x}_t$. Our gradient estimation closely follows the idea in Agarwal et al. (2010) by querying $g(\mathbf{x})$ function at two points. The main advantage of using two points to estimate the gradient with respect to one point gradient estimation used in Flaxman et al. (2005) is that the former has a bounded norm which is independent of ζ and leads to improved regret bounds.

The gradient estimators for $\nabla_{\mathbf{x}}\widehat{\mathcal{L}}(\mathbf{x}_t,\lambda_t) = \nabla f(\mathbf{x}_t) + \lambda_t\nabla\hat{g}(\mathbf{x}_t)$ and $\nabla_{\lambda}\widehat{\mathcal{L}}(\mathbf{x}_t,\lambda_t) = \hat{g}(\mathbf{x}_t) - \delta\eta\lambda_t$ in Algorithm 3 are computed by evaluating the $g(\mathbf{x})$ function at two random points around $\mathbf{x}_t$ as

$$\tilde{g}_{\mathbf{x},t} = \nabla f_t(\mathbf{x}_t) + \lambda_t\left[\frac{d}{2\zeta}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) - g(\mathbf{x}_t - \zeta\mathbf{u}_t))\mathbf{u}_t\right]$$

and

$$\tilde{g}_{\lambda,t} = \frac{1}{2}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) + g(\mathbf{x}_t - \zeta\mathbf{u}_t)) - \eta\delta\lambda_t,$$

where $\mathbf{u}_t$ is chosen uniformly at random from the surface of the unit sphere. Using Stock's theorem, Flaxman et al. (2005) showed that $\frac{1}{2\zeta}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) - g(\mathbf{x}_t - \zeta\mathbf{u}_t))\mathbf{u}_t$ is a conditionally unbiased estimate of the gradient of $\hat{g}(\mathbf{x})$ at point $\mathbf{x}_t$. To make sure that randomized points around $\mathbf{x}_t$ live inside the convex domain $\mathcal{B}$, we need to stay away from the boundary of the set such that the ball of radius ζ around $\mathbf{x}_t$ is contained in $\mathcal{B}$. In particular, in Flaxman et al. (2005) it has been shown that for any $\mathbf{x} \in (1-\xi)\mathcal{B}$ and any unit vector $\mathbf{u}$ it holds that $(\mathbf{x} + \zeta\mathbf{u}) \in \mathcal{B}$ as soon as $\zeta \in [0, \xi r]$.

In order to facilitate the analysis of the Algorithm 3, we define the convex-concave function $\mathcal{H}_t(\cdot,\cdot)$ as

$$\mathcal{H}_t(\mathbf{x},\lambda) = \widehat{\mathcal{L}}_t(\mathbf{x},\lambda) + \left(\tilde{g}_{\mathbf{x},t} - \nabla_{\mathbf{x}}\widehat{\mathcal{L}}(\mathbf{x}_t,\lambda_t)\right)\mathbf{x} + \left(\tilde{g}_{\lambda,t} - \nabla_{\lambda}\widehat{\mathcal{L}}(\mathbf{x}_t,\lambda_t)\right)\lambda. \tag{20}$$

It is easy to check that $\nabla_{\mathbf{x}}\mathcal{H}(\mathbf{x}_t,\lambda_t)=\tilde{g}_{\mathbf{x},t}$ and $\nabla_{\lambda}\mathcal{H}(\mathbf{x}_t,\lambda_t)=\tilde{g}_{\lambda,t}$. By defining functions $\mathcal{H}_t(\mathbf{x},\lambda)$, Algorithm 3 reduces to Algorithm 1 by doing gradient descent on functions $\mathcal{H}_t(\mathbf{x},\lambda)$ except the projection is made onto the set $(1-\xi)\mathcal{B}$ instead of $\mathcal{B}$.

We begin our analysis by reproducing Proposition 3 for functions $\mathcal{H}_t(\cdot,\cdot)$.

Lemma 14 *If the Algorithm 1 is performed over convex set $\mathcal{K}$ with functions $\mathcal{H}_t$ defined in (20), then for any $\mathbf{x}\in\mathcal{K}$ we have*

$$\sum_{t=1}^{T}\mathcal{H}_t(\mathbf{x}_t,\lambda)-\mathcal{H}_t(\mathbf{x},\lambda_t)\leq\frac{R^2+\|\lambda\|_2^2}{2\eta}+\eta(D^2+G^2)T+\eta(d^2G^2+\eta^2\delta^2)\sum_{t=1}^{T}\lambda_t^2.$$

Proof We have $\nabla_{\mathbf{x}}\mathcal{H}_t(\mathbf{x}_t,\lambda_t)=\tilde{g}_{\mathbf{x},t}$ and $\nabla_{\lambda}\mathcal{H}_t(\mathbf{x}_t,\lambda_t)=\tilde{g}_{\lambda,t}$. It is straightforward to show that $\frac{1}{2\zeta}(g(\mathbf{x}_t+\zeta\mathbf{u}_t)-g(\mathbf{x}_t-\zeta\mathbf{u}_t))\mathbf{u}_t$ has norm bounded by Gd (Agarwal et al., 2010). So, the norm of gradients are bounded as $\|\tilde{g}_{\mathbf{x},t}\|_2^2\leq 2(G^2+d^2G^2\lambda_t^2)$ and $\|\tilde{g}_{\lambda,t}\|_2^2\leq 2(D^2+\eta^2\delta^2\lambda_t^2)$. Using Lemma 2, by adding for all rounds we get the desired inequality. ■

The following theorem gives the regret bound and the expected violation of the constraints in the long run for Algorithm 3.

Theorem 15 *Let $c=\sqrt{D^2+G^2}(\sqrt{2}R+\frac{\sqrt{2}D}{\delta R})+(\frac{D}{r}+1)\frac{GD}{r}$. Set $\eta=R/\sqrt{2(D^2+G^2)T}$. Choose δ such that $\delta\geq 2(d^2G^2+\eta^2\delta^2)$. Let $\zeta=\frac{\delta}{T}$ and $\xi=\frac{\zeta}{r}$. Let $\mathbf{x}_t, t\in[T]$ be the sequence of solutions obtained by Algorithm 3. We then have*

$$\sum_{t=1}^{T}f_t(\mathbf{x}_t)-f_t(\mathbf{x})\leq\frac{GD}{r}+c\sqrt{T}=O(T^{1/2}),\ \textit{and}$$

$$\mathbb{E}\Big[\sum_{t=1}^{T}g(\mathbf{x}_t)\Big]\leq G\delta+\sqrt{\Big(\frac{\delta R^2+2(D^2+G^2)}{R\sqrt{D^2+G^2}}\Big)(\frac{GD}{r}+c\sqrt{T}+FT)\sqrt{T}}=O(T^{3/4}).$$

Proof Using Lemma 2 for the functions $\widehat{\mathcal{L}}_t(\cdot,\cdot)$ and $\mathcal{H}_t(\cdot,\cdot)$ we have

$$\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda)-\widehat{\mathcal{L}}_t(\mathbf{x},\lambda_t)\leq(\mathbf{x}_t-\mathbf{x})^{\top}\nabla_{\mathbf{x}}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t)-(\lambda-\lambda_t)^{\top}\nabla_{\lambda}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t),$$

and also

$$\mathcal{H}_t(\mathbf{x}_t,\lambda)-\mathcal{H}_t(\mathbf{x},\lambda_t)\leq(\mathbf{x}_t-\mathbf{x})^{\top}\tilde{g}_{\mathbf{x},t}-(\lambda-\lambda_t)^{\top}\tilde{g}_{\lambda,t}.$$

Subtracting the preceding inequalities, taking expectation, and summing for all t from 1 to T we get

$$\begin{aligned}&\mathbb{E}\left[\sum_{t=1}^{T}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda)-\widehat{\mathcal{L}}_t(\mathbf{x},\lambda_t)\right]\\&=\mathbb{E}\left[\sum_{t=1}^{T}\mathcal{H}_t(\mathbf{x}_t,\lambda)-\mathcal{H}_t(\mathbf{x},\lambda_t)\right]\\&\quad+\mathbb{E}\left[\sum_{t=1}^{T}(\mathbf{x}_t-\mathbf{x})^{\top}(\nabla_{\mathbf{x}}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t)-\mathbb{E}_t[\tilde{g}_{\mathbf{x}_t,t}])+(\lambda_t-\lambda)^{\top}(\nabla_{\lambda}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t)-\mathbb{E}_t[\tilde{g}_{\lambda_t,t}])\right].\end{aligned}\tag{21}$$

Next we provide an upper bound on the difference between the gradients of two functions. First, $\mathbb{E}_t[\tilde{g}_{\mathbf{x},t}] = \nabla_{\mathbf{x}}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t)$, so $\tilde{g}_{\mathbf{x},t}$ is an unbiased estimator of $\nabla_{\mathbf{x}}\widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t)$. Considering the update rule for λ_{t+1} we have $|\lambda_{t+1}| \le (1-\eta^2\delta)|\lambda_t| + \eta D$ which implies that $|\lambda_t| \le \frac{D}{\delta\eta}$ for all t. So we obtain

$$\begin{aligned}
&(\lambda_t - \lambda)^\top (\nabla_\lambda \widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t) - \mathbb{E}_t[\tilde{g}_{\lambda_t,t}]) \\
&\le |\lambda_t - \lambda| \mathbb{E}_t\Big[\|\nabla_\lambda \widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda_t) - \tilde{g}_{\lambda_t,t}\|_2\Big] \\
&\le \frac{D}{\delta\eta}\left|\frac{1}{2}(g(\mathbf{x}_t + \zeta\mathbf{u}_t) + g(\mathbf{x}_t - \zeta\mathbf{u}_t)) - \hat{g}(\mathbf{x}_t)\right| \le \frac{DG}{\delta\eta}\zeta\|\mathbf{u}_t\| \le \frac{DG}{\delta\eta}\zeta, \qquad (22)
\end{aligned}$$

where the last inequality follows from Lipschitz property of the functions $g(\mathbf{x})$ and $\hat{g}(\mathbf{x})$ with the same constant G. Combining the inequalities (21) and (22) and using Lemma 14, we have

$$\mathbb{E}\Big[\sum_{t=1}^T \widehat{\mathcal{L}}_t(\mathbf{x}_t,\lambda) - \widehat{\mathcal{L}}_t(\mathbf{x},\lambda_t)\Big] \le \frac{R^2+\lambda^2}{2\eta} + \eta(D^2+G^2)T + \eta(d^2G^2+\eta^2\delta^2)\sum_{t=1}^T \lambda_t^2 + \frac{DG\zeta}{\delta\eta}T.$$

By expanding the right hand side of above inequality, we obtain

$$\begin{aligned}
&\sum_{t=1}^T [f_t(\mathbf{x}_t) - f_t((1-\xi)\mathbf{x})] + \lambda\mathbb{E}\Big[\sum_{t=1}^T \hat{g}(\mathbf{x}_t)\Big] - \mathbb{E}\Big[\hat{g}((1-\xi)\mathbf{x})\Big]\sum_{t=1}^T \lambda_t - \frac{\eta\delta T}{2}\lambda^2 + \frac{\eta\delta}{2}\sum_{t=1}^T \lambda_t^2 \\
&\le \frac{R^2+\lambda^2}{2\eta} + \eta(D^2+G^2)T + \eta(d^2G^2+\eta^2\delta^2)\sum_{t=1}^T \lambda_t^2 + \frac{DG\zeta}{\delta\eta}T.
\end{aligned}$$

By choosing $\delta \ge 2(d^2G^2+\eta^2\delta^2)$ we cancel λ_t^2 terms from both sides and have

$$\begin{aligned}
&\sum_{t=1}^T [f_t(\mathbf{x}_t) - f_t((1-\xi)\mathbf{x})] + \lambda\mathbb{E}\Big[\sum_{t=1}^T \hat{g}(\mathbf{x}_t)\Big] - \mathbb{E}\Big[\hat{g}((1-\xi)\mathbf{x})\Big]\sum_{t=1}^T \lambda_t - \frac{\eta\delta T}{2}\lambda^2 \\
&\le \frac{R^2+\lambda^2}{2\eta} + \eta(D^2+G^2)T + \frac{DG\zeta}{\delta\eta}T. \qquad (23)
\end{aligned}$$

By convexity and Lipschitz property of $f_t(\mathbf{x})$ and $g(\mathbf{x})$ we have

$$f_t((1-\xi)\mathbf{x}) \le (1-\xi)f_t(\mathbf{x}) + \xi f_t(0) \le f_t(\mathbf{x}) + DG\xi, \qquad (24)$$

$$g(\mathbf{x}) \le \hat{g}(\mathbf{x}) + G\zeta \text{ , and } \hat{g}((1-\xi)\mathbf{x}) \le g((1-\xi)\mathbf{x}) + G\zeta \le g(\mathbf{x}) + G\zeta + DG\xi. \qquad (25)$$

Plugging (24) and (25) back into (23), for any optimal solution $\mathbf{x}_* \in \mathcal{K}$ we get

$$\begin{aligned}
&\sum_{t=1}^T [f_t(\mathbf{x}_t) - f_t(\mathbf{x})] + \lambda\mathbb{E}\Big[\sum_{t=1}^T g(\mathbf{x}_t)\Big] - \frac{\eta\delta T}{2}\lambda^2 - \lambda G\zeta T \\
&\le \frac{R^2+\lambda^2}{2\eta} + \eta(D^2+G^2)T + \frac{DG\zeta}{\delta\eta}T + DG\xi T + (DG\xi + G\zeta)\sum_{t=1}^T \lambda_t. \qquad (26)
\end{aligned}$$

Considering the fact that $\lambda_t \le \frac{D}{\delta\eta}$ we have $\sum_{t=1}^T \lambda_t \le \frac{DT}{\delta\eta}$. Plugging back into the (26) and rearranging the terms we have

$$\begin{aligned}
&\sum_{t=1}^T [f_t(\mathbf{x}_t) - f_t(\mathbf{x})] + \lambda\mathbb{E}\Big[\sum_{t=1}^T g(\mathbf{x}_t)\Big] - \frac{\eta\delta T}{2}\lambda^2 - \lambda G\zeta T - \frac{\lambda^2}{2\eta} \\
&\le \frac{R^2}{2\eta} + \eta(D^2+G^2)T + \frac{DG\zeta}{\delta\eta}T + DG\xi T + (DG\xi + G\zeta)\frac{DT}{\delta\eta}.
\end{aligned}$$

By setting $\xi = \frac{\zeta}{r}$ and $\zeta = \frac{1}{T}$ we get

$$\sum_{t=1}^{T} [f_t(\mathbf{x}_t) - f_t(\mathbf{x})] \leq \frac{R^2}{2\eta} + \eta(D^2+G^2)T + \frac{DG\zeta}{\delta\eta}T + \frac{\zeta DGT}{r} + (\frac{D}{r}+1)\zeta\frac{DGT}{\delta\eta},$$

which gives the mentioned regret bound by optimizing for η. Maximizing for λ over the range $(0,+\infty)$ and using $\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \geq -FT$, yields the following inequality for the violation of constraints

$$\frac{\left[\mathbb{E}\left[\sum_{t=1}^{T} g(\mathbf{x}_t)\right] - G\zeta T\right]_+^2}{4(\delta\eta T/2 + 1/2\eta)} \leq \frac{DG}{r} + c\sqrt{T} + FT.$$

Plugging in the stated values of parameters completes the proof. Note that $\delta = 4d^2G^2$ obeys the condition specified in the theorem. ■

6. Conclusion

In this study we have addressed the problem of online convex optimization with constraints, where we only need the constraints to be satisfied in the long run. In addition to the regret bound which is the main tool in analyzing the performance of general online convex optimization algorithms, we defined the bound on the violation of constraints in the long term which measures the cumulative violation of the solutions from the constraints for all rounds. Our setting is applied to solving online convex optimization without projecting the solutions onto the complex convex domain at each iteration, which may be computationally inefficient for complex domains. Our strategy is to turn the problem into an online convex-concave optimization problem and apply online gradient descent algorithm to solve it. We have proposed efficient algorithms in three different settings; the violation of constraints is allowed, the constraints need to be exactly satisfied, and finally we do not have access to the target convex domain except it is bounded by a ball. Moreover, for domains determined by linear constraints, we used the mirror prox method, a simple gradient based algorithm for variational inequalities, and obtained an $O(T^{2/3})$ bound for both regret and the violation of the constraints.

Our work leaves open a number of interesting directions for future work. In particular it would be interesting to see if it is possible to improve the bounds obtained in this paper, i.e., getting an $O(\sqrt{T})$ bound on the regret and better bound than $O(T^{3/4})$ on the violation of constraints for general convex domains. Proving optimal lower bounds for the proposed setting also remains as an open question. Also, it would be interesting to consider strongly convex loss or constraint functions. Finally, relaxing the assumption we made to exactly satisfy the constraints in the long run is an interesting problem to be investigated.

Acknowledgments

The authors would like to thank the Action Editor and three anonymous reviewers for their constructive comments and helpful suggestions on the original version of this paper. This work was sup-

ported in part by National Science Foundation (IIS-0643494) and Office of Navy Research (Award N000141210431).

Appendix A. Proof of Theorem 1

We first show that when $\delta < 1$, there exists a loss function and a constraint function such that the violation of constraint is linear in T. To see this, we set $f_t(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}, t \in [T]$ and $g(\mathbf{x}) = 1 - \mathbf{w}^\top \mathbf{x}$. Assume we start with an infeasible solution, that is, $g(\mathbf{x}_1) > 0$ or $\mathbf{x}_1^\top \mathbf{w} < 1$. Given the solution $\mathbf{x}_t$ obtained at tth trial, using the standard gradient descent approach, we have $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta(1-\delta)\mathbf{w}$. Hence, if $\mathbf{x}_t^\top \mathbf{w} < 1$, since we have $\mathbf{x}_{t+1}^\top \mathbf{w} < \mathbf{x}_t^\top \mathbf{w} < 1$, if we start with an infeasible solution, all the solutions obtained over the trails will violate the constraint $g(\mathbf{x}) \leq 0$, leading to a linear number of violation of constraints. Based on this analysis, we assume $\delta > 1$ in the analysis below.

Given a strongly convex loss function $f(\mathbf{x})$ with modulus γ, we consider a constrained optimization problem given by

$$\min_{g(\mathbf{x}) \leq 0} f(\mathbf{x}),$$

which is equivalent to the following unconstrained optimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) + \lambda [g(\mathbf{x})]_+,$$

where $\lambda \geq 0$ is the Lagrangian multiplier. Since we can always scale $f(\mathbf{x})$ to make $\lambda \leq 1/2$, it is safe to assume $\lambda \leq 1/2 < \delta$. Let $\mathbf{x}_*$ and $\mathbf{x}_a$ be the optimal solutions to the constrained optimization problems $\arg\min_{g(\mathbf{x}) \leq 0} f(\mathbf{x})$ and $\arg\min_{\mathbf{x}} f(\mathbf{x}) + \delta [g(\mathbf{x})]_+$, respectively. We choose $f(\mathbf{x})$ such that $\|\nabla f(\mathbf{x}_*)\| > 0$, which leads to $\mathbf{x}_a \neq \mathbf{x}_*$. This holds because according to the first order optimality condition, we have

$$\nabla f(\mathbf{x}_*) = -\lambda \nabla g(\mathbf{x}_*), \ \nabla f(\mathbf{x}_a) = -\delta \nabla g(\mathbf{x}_*),$$

and therefore $\nabla f(\mathbf{x}_*) \neq \nabla f(\mathbf{x}_a)$ when $\lambda < \delta$. Define $\Delta = f(\mathbf{x}_a) - f(\mathbf{x}_*)$. Since $\Delta \geq \gamma \|\mathbf{x}_a - \mathbf{x}_*\|^2/2$ due to the strong convexity of $f(\mathbf{x})$, we have $\Delta > 0$.

Let $\{\mathbf{x}_t\}_{t=1}^T$ be the sequence of solutions generated by the OGD algorithm that minimizes the modified loss function $f(\mathbf{x}) + \delta [g(\mathbf{x})]_+$. We have

$$\begin{aligned}
& \sum_{t=1}^T f(\mathbf{x}_t) + \delta [g(\mathbf{x}_t)]_+ \geq T \min_{\mathbf{x}} f(\mathbf{x}) + \delta [g(\mathbf{x})]_+ \\
= \ & T(f(\mathbf{x}_a) + \delta [g(\mathbf{x}_a)]_+) \geq T(f(\mathbf{x}_a) + \lambda [g(\mathbf{x}_a)]_+) \\
= \ & T(f(\mathbf{x}_*) + \lambda [g(\mathbf{x}_*)]_+) + T(f(\mathbf{x}_a) + \lambda [g(\mathbf{x}_a)]_+ - f(\mathbf{x}_*) - \lambda [g(\mathbf{x}_*)]) \\
\geq \ & T \min_{g(\mathbf{x}) \leq 0} f(\mathbf{x}) + T\Delta.
\end{aligned}$$

As a result, we have

$$\sum_{t=1}^T f(\mathbf{x}_t) + \delta [g(\mathbf{x}_t)]_+ - \min_{g(\mathbf{x}) \leq 0} f(\mathbf{x}) = O(T),$$

implying that either the regret $\sum_{t=1}^T f(\mathbf{x}_t) - T f(\mathbf{x}_*)$ or the violation of the constraints $\sum_{t=1}^T [g(\mathbf{x})]_+$ is linear in T.

To better understand the performance of penalty based approach, here we analyze the performance of the OGD in solving the online optimization problem in (3). The algorithm is analyzed using the following lemma from Zinkevich (2003).

Lemma 16 *Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ be the sequence of solutions obtained by applying OGD on the sequence of bounded convex functions $f_1, f_2, \ldots, f_T$. Then, for any solution $\mathbf{x}_* \in \mathcal{K}$ we have*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}_*) \leq \frac{R^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\nabla f_t(\mathbf{x}_t)\|^2.$$

We apply OGD to functions $\hat{f}_t(\mathbf{x})$, $t \in [T]$ defined in (4), that is, instead of updating the solution based on the gradient of $f_t(\mathbf{x})$, we update the solution by the gradient of $\hat{f}_t(\mathbf{x})$. Using Lemma 16, by expanding the functions $\hat{f}_t(\mathbf{x})$ based on (4) and considering the fact that $\sum_{i=1}^{m} [g_i(\mathbf{x}_*)]_+^2 = 0$, we get

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}_*) + \frac{\delta}{2} \sum_{t=1}^{T} \sum_{i=1}^{m} [g_i(\mathbf{x})]_+^2 \leq \frac{R^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\nabla \hat{f}_t(\mathbf{x}_t)\|^2. \tag{27}$$

From the definition of $\hat{f}_t(\mathbf{x})$, the norm of the gradient $\nabla \hat{f}_t(\mathbf{x}_t)$ is bounded as follows

$$\|\nabla \hat{f}_t(\mathbf{x})\|^2 = \|\nabla f_t(\mathbf{x}) + \delta \sum_{i=1}^{m} [g_i(\mathbf{x})]_+ \nabla g_i(\mathbf{x})\|^2 \leq 2G^2(1 + m\delta^2 D^2), \tag{28}$$

where the inequality holds because $(a_1 + a_2)^2 \leq 2(a_1^2 + a_2^2)$. By substituting (28) into the (27) we have:

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}_*) + \frac{\delta}{2} \sum_{t=1}^{T} \sum_{i=1}^{m} [g_i(\mathbf{x}_t)]_+^2 \leq \frac{R^2}{2\eta} + \eta G^2(1 + m\delta^2 D^2)T. \tag{29}$$

Since $[\cdot]_+^2$ is a convex function, from Jensen's inequality and following the fact that $\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_*) \geq -FT$, we have:

$$\frac{\delta}{2T} \sum_{i=1}^{m} \left[\sum_{t=1}^{T} g_i(\mathbf{x}_t) \right]_+^2 \leq \frac{\delta}{2} \sum_{i=1}^{m} \sum_{t=1}^{T} [g_i(\mathbf{x}_t)]_+^2 \leq \frac{R^2}{2\eta} + \eta G^2(1 + m\delta^2 D^2)T + FT.$$

By minimizing the right hand side of (29) with respect to η, we get the regret bound as

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}_*) \leq RG \sqrt{2(1 + m\delta^2 D^2)T} = O(\delta \sqrt{T}) \tag{30}$$

and the bound for the violation of constraints as

$$\sum_{t=1}^{T} g_i(\mathbf{x}_t) \leq \sqrt{\left(\frac{R^2}{2\eta} + \eta G^2(1 + m\delta^2 D^2)T + FT \right) \frac{2T}{\delta}} = O(T^{1/4}\delta^{1/2} + T\delta^{-1/2}). \tag{31}$$

Examining the bounds obtained in (30) and (31), it turns out that in order to recover $O(\sqrt{T})$ regret bound, we need to set δ to be a constant, leading to $O(T)$ bound for the violation of constraints in the long run, which is not satisfactory at all.

Appendix B. Proof of Corollary 13

Let $\mathbf{x}_\gamma$ be the optimal solution to $\min_{g(\mathbf{x})\leq -\gamma}\sum_{t=1}^{T} f_t(\mathbf{x})$. Similar to the proof of Theorem 12, we have

$$\sum_{t=1}^{T}\big[f_t(\mathbf{x}_t)-f_t(\mathbf{x}_\gamma)\big]+\frac{\big[\sum_{t=1}^{T} g(\mathbf{x}_t)+\gamma T\big]_+^2}{2(\delta\eta T+1/\eta)}\leq \frac{R^2}{2\eta}+\frac{\eta T}{2}G^2.$$

Using the stated values for the parameters $\eta=\delta=T^{-1/3}$, and applying the fact that $\sum_{t=1}^{T} f_t(\mathbf{x}_t)-f_t(\mathbf{x}_\gamma)\geq -FT$ we obtain,

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t)-f_t(\mathbf{x}_\gamma)\leq \frac{R^2}{2}T^{1/3}+\frac{G^2}{2}T^{2/3} \tag{32}$$

and

$$\left[\sum_{t=1}^{T} g(\mathbf{x}_t)+\gamma T\right]_+^2\leq 2\big(R^2T^{1/3}+G^2T^{2/3}+FT\big)T^{1/3}. \tag{33}$$

From Theorem 7, we have the bound

$$\sum_{t=1}^{T} f_t(\mathbf{x}_\gamma)\leq \sum_{t=1}^{T} f_t(\mathbf{x}_*)+\frac{G}{\sigma}\gamma T. \tag{34}$$

Combining inequalities (32) and (34) with substituting the stated value of $\gamma=bT^{-1/3}$ yields the regret bound as desired. To obtain the bound for the violation of the constraints, from (33) we have

$$\sum_{t=1}^{T} g(\mathbf{x}_t)\leq \sqrt{2\big(R^2T^{1/3}+G^2T^{2/3}+FT\big)T^{1/3}}-bT^{2/3}.$$

For sufficiently large values of T, that is, $FT\geq R^2T^{1/3}+G^2T^{2/3}$ we can simplify above inequality as $\sum_{t=1}^{T} g(\mathbf{x}_t)\leq 2\sqrt{F}T^{2/3}-bT^{2/3}$. By setting $b=2\sqrt{F}$ the zero bound on the violation of constraints is guaranteed.

References

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *COLT*, pages 263–274, 2008.

Jacob Abernethy, Alekh Agarwal, Peter L. Bartlett, and Alexander Rakhlin. A stochastic view of optimal regret through minimax duality. In *COLT*, 2009.

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Interior-point methods for full-information and bandit online learning. *IEEE Transactions on Information Theory*, 58(7):4164–4175, 2012.

Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT*, pages 28–40, 2010.

Baruch Awerbuch and Robert D. Kleinberg. Adaptive routing with end-to-end feedback: distributed learning and geometric approaches. In *STOC*, pages 45–53, 2004.

Peter L. Bartlett, Elad Hazan, and Alexander Rakhlin. Adaptive online gradient descent. In *NIPS*, pages 257–269, 2007.

Andrey Bernstein, Shie Mannor, and Nahum Shimkin. Online classification with specificity constraints. In *NIPS*, pages 190–198, 2010.

Dimitri P. Bertsekas, Angelia Nedic, and Asuman E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

Varsha Dani, Thomas P. Hayes, and Sham Kakade. The price of bandit information for online optimization. In *NIPS*, 2007.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l_1-ball for learning in high dimensions. In *ICML*, pages 272–279, 2008.

Abraham Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA*, pages 385–394, 2005.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

Jun Liu and Jieping Ye. Efficient euclidean projections in linear time. In *ICML*, pages 83–90, 2009.

Shie Mannor and John N. Tsitsiklis. Online learning with constraints. In *COLT*, pages 529–543, 2006.

Shie Mannor, John N. Tsitsiklis, and Jia Yuan Yu. Online learning with sample path constraints. *Journal of Machine Learning Research*, 10:569–590, 2009.

Hariharan Narayanan and Alexander Rakhlin. Random walk approach to regret minimization. In *NIPS*, pages 1777–1785, 2010.

Arkadi Nemirovski. Efficient methods in convex programming. Lecture Notes, Available at http://www2.isye.gatech.edu/ nemirovs, 1994.

Arkadi Nemirovski. Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. on Optimization*, 15(1):229–251, 2005.

Alexander Rakhlin. Lecture notes on online learning. Lecture Notes, Available at http://www-stat.wharton.upenn.edu/ rakhlin/papers, 2009.

Shai Shalev-Shwartz and Sham M. Kakade. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS*, pages 1457–1464, 2008.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.

Journal of Machine Learning Research 13 (2012) 2529-2565 Submitted 9/11; Revised 4/12; Published 9/12

Robust Kernel Density Estimation

JooSeuk Kim STANNUM@UMICH.EDU
Clayton D. Scott* CLAYSCOT@UMICH.EDU
Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122 USA

Editor: Kenji Fukumizu

Abstract

We propose a method for nonparametric density estimation that exhibits robustness to contamination of the training sample. This method achieves robustness by combining a traditional kernel density estimator (KDE) with ideas from classical M-estimation. We interpret the KDE based on a positive semi-definite kernel as a sample mean in the associated reproducing kernel Hilbert space. Since the sample mean is sensitive to outliers, we estimate it robustly via M-estimation, yielding a robust kernel density estimator (RKDE).

An RKDE can be computed efficiently via a kernelized iteratively re-weighted least squares (IRWLS) algorithm. Necessary and sufficient conditions are given for kernelized IRWLS to converge to the global minimizer of the M-estimator objective function. The robustness of the RKDE is demonstrated with a representer theorem, the influence function, and experimental results for density estimation and anomaly detection.

Keywords: outlier, reproducing kernel Hilbert space, kernel trick, influence function, M-estimation

1. Introduction

The kernel density estimator (KDE) is a well-known nonparametric estimator of univariate or multivariate densities, and numerous articles have been written on its properties, applications, and extensions (Silverman, 1986; Scott, 1992). However, relatively little work has been done to understand or improve the KDE in situations where the training sample is contaminated. This paper addresses a method of nonparametric density estimation that generalizes the KDE, and exhibits robustness to contamination of the training sample.[1]

Consider training data following a contamination model

$$\mathbf{X}_1, \ldots, \mathbf{X}_n \overset{iid}{\sim} (1-p) f_0 + p f_1,$$

where f_0 is the "nominal" density to be estimated, f_1 is the density of the contaminating distribution, and $p < \frac{1}{2}$ is the proportion of contamination. Labels are not available, so that the problem is unsupervised. The objective is to estimate f_0 while making no parametric assumptions about the nominal or contaminating distributions.

∗. Also in the Department of Statistics.

1. Shorter versions of this work previously appeared at the International Conference on Acoustics, Speech, and Signal Processing (Kim and Scott, 2008) and the International Conference on Machine Learning (Kim and Scott, 2011).

Clearly f_0 cannot be recovered if there are *no* assumptions on f_0, f_1 and p. Instead, we will focus on a set of nonparametric conditions that are reasonable in many practical applications. In particular, we will assume that, relative to the nominal data, the contaminated data are

(a) *outlying*: the densities f_0 and f_1 have relatively little overlap

(b) *diffuse*: f_1 is not too spatially concentrated relative to f_0

(c) *not abundant*: a minority of the data come from f_1

Although we will not be stating these conditions more precisely, they capture the intuition behind the quantitative results presented below.

As a motivating application, consider anomaly detection in a computer network. Imagine that several multi-dimensional measurements $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are collected. For example, each $\mathbf{X}_i$ may record the volume of traffic along certain links in the network, at a certain instant in time (Chhabra et al., 2008). If each measurement is collected when the network is in a nominal state, these data could be used to construct an anomaly detector by first estimating the density f_0 of nominal measurements, and then thresholding that estimate at some level to obtain decision regions. Unfortunately, it is often difficult to know that the data are free of anomalies, because assigning labels (nominal vs. anomalous) can be a tedious, labor intensive task. Hence, it is necessary to estimate the nominal density (or a level set thereof) from contaminated data. Furthermore, the distributions of both nominal and anomalous measurements are potentially complex, and it is therefore desirable to avoid parametric models.

The proposed method achieves robustness by combining a traditional kernel density estimator with ideas from M-estimation (Huber, 1964; Hampel, 1974). The KDE based on a translation invariant, positive semi-definite (PSD) kernel is interpreted as a sample mean in the reproducing kernel Hilbert space (RKHS) associated with the kernel. Since the sample mean is sensitive to outliers, we estimate it robustly via M-estimation, yielding a robust kernel density estimator (RKDE). We describe a kernelized iteratively re-weighted least squares (KIRWLS) algorithm to efficiently compute the RKDE, and provide necessary and sufficient conditions for the convergence of KIRWLS to the RKDE.

We also offer three arguments to support the claim that the RKDE robustly estimates the nominal density and its level sets. First, we characterize the RKDE by a representer theorem. This theorem shows that the RKDE is a weighted KDE, and the weights are smaller for more outlying data points. Second, we study the influence function of the RKDE, and show through an exact formula and numerical results that the RKDE is less sensitive to contamination by outliers than the KDE. Third, we conduct experiments on several benchmark data sets that demonstrate the improved performance of the RKDE, relative to competing methods, at both density estimation and anomaly detection.

One motivation for this work is that the traditional kernel density estimator is well-known to be sensitive to outliers. Even without contamination, the standard KDE tends to overestimate the density in regions where the true density is low. This has motivated several authors to consider variable kernel density estimators (VKDEs), which employ a data-dependent bandwidth at each data point (Breiman et al., 1977; Abramson, 1982; Terrell and Scott, 1992). This bandwidth is adapted to be larger where the data are less dense, with the aim of decreasing the aforementioned bias. Such methods have been applied in outlier detection and computer vision applications (Comaniciu et al., 2001; Latecki et al., 2007), and are one possible approach to robust nonparametric density estimation. We compare against these methods in our experimental study.

Density estimation with positive semi-definite kernels has been studied by several authors. Vapnik and Mukherjee (2000) optimize a criterion based on the empirical cumulative distribution function over the class of weighted KDEs based on a PSD kernel. Shawe-Taylor and Dolia (2007) provide a refined theoretical treatment of this approach. Song et al. (2008) adopt a different criterion based on Hilbert space embeddings of probability distributions. Our approach is somewhat similar in that we attempt to match the mean of the empirical distribution in the RKHS, but our criterion is different. These methods were also not designed with contaminated data in mind.

We show that the standard kernel density estimator can be viewed as the solution to a certain least squares problem in the RKHS. The use of quadratic criteria in density estimation has also been previously developed. The aforementioned work of Song et al. optimizes the norm-squared in Hilbert space, whereas Kim (1995), Girolami and He (2003), Kim and Scott (2010) and Mahapatruni and Gray (2011) adopt the integrated squared error. Once again, these methods are not designed for contaminated data.

Previous work combining robust estimation and kernel methods has focused primarily on supervised learning problems. M-estimation applied to kernel regression has been studied by various authors (Christmann and Steinwart, 2007; Debruyne et al., 2008a,b; Zhu et al., 2008; Wibowo, 2009; Brabanter et al., 2009). Robust surrogate losses for kernel-based classifiers have also been studied (Xu et al., 2006). In unsupervised learning, a robust way of doing kernel principal component analysis, called spherical KPCA, has been proposed, which applies PCA to feature vectors projected onto a unit sphere around the spatial median in a kernel feature space (Debruyne et al., 2010). The kernelized spatial depth was also proposed to estimate depth contours nonparametrically (Chen et al., 2009). To our knowledge, the RKDE is the first application of M-estimation ideas in kernel density estimation.

In Section 2 we propose robust kernel density estimation. In Section 3 we present a representer theorem for the RKDE. In Section 4 we describe the KIRWLS algorithm and its convergence. The influence function is developed in Section 5, Section 6 describes a straightforward extension to non-reproducing kernels, and experimental results are reported in Section 7. Conclusions are offered in Section 8. Section 9 contains proofs of theorems. Matlab code implementing our algorithm is available at `www.eecs.umich.edu/~cscott`.

2. Robust Kernel Density Estimation

Let $\mathbf{X}_1,\ldots,\mathbf{X}_n \in \mathbb{R}^d$ be a random sample from a distribution F with a density f. The kernel density estimate of f, also called the Parzen window estimate, is a nonparametric estimate given by

$$\widehat{f}_{KDE}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} k_\sigma(\mathbf{x},\mathbf{X}_i)$$

where k_σ is a kernel function with bandwidth σ. To ensure that $\widehat{f}_{KDE}(\mathbf{x})$ is a density, we assume the kernel function satisfies $k_\sigma(\cdot,\cdot) \geq 0$ and $\int k_\sigma(\mathbf{x},\cdot)\,d\mathbf{x} = 1$. We will also assume that $k_\sigma(\mathbf{x},\mathbf{x}')$ is *translation invariant*, in that $k_\sigma(\mathbf{x}-\mathbf{z},\mathbf{x}'-\mathbf{z}) = k_\sigma(\mathbf{x},\mathbf{x}')$ for all $\mathbf{x},\mathbf{x}'$, and $\mathbf{z}$.

In addition, we require that k_σ be *positive semi-definite*, which means that the matrix $(k_\sigma(x_i,x_j))_{1\leq i,j\leq m}$ is positive semi-definite for all positive integers m and all $x_1,\ldots,x_m \in \mathbb{R}^d$. Well-known examples of kernels satisfying all of the above properties are the Gaussian kernel

$$k_\sigma(\mathbf{x},\mathbf{x}') = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^d \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right), \tag{1}$$

the multivariate Student kernel

$$k_\sigma(\mathbf{x},\mathbf{x}') = \left(\frac{1}{\sqrt{\pi}\sigma}\right)^d \cdot \frac{\Gamma\big((\nu+d)/2\big)}{\Gamma(\nu/2)} \cdot \left(1+\frac{1}{\nu}\cdot\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}\right)^{-\frac{\nu+d}{2}},$$

and the Laplacian kernel

$$k_\sigma(\mathbf{x},\mathbf{x}') = \frac{c_d}{\sigma^d}\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|}{\sigma}\right)$$

where c_d is a constant depending on the dimension d that ensures $\int k_\sigma(\mathbf{x},\cdot)\,d\mathbf{x} = 1$ (Scovel et al., 2010).

Every PSD kernel k_σ is associated with a unique Hilbert space of functions called its reproducing kernel Hilbert space (RKHS) which we will denote $\mathcal{H}$, and k_σ is called the reproducing kernel of $\mathcal{H}$. For every $\mathbf{x}$, $\Phi(\mathbf{x}) \triangleq k_\sigma(\cdot,\mathbf{x})$ is an element of $\mathcal{H}$, and therefore so too is the KDE. See Steinwart and Christmann (2008) for a thorough treatment of PSD kernels and RKHSs. For our purposes, the critical property of $\mathcal{H}$ is the so-called *reproducing property*. It states that for all $g \in \mathcal{H}$ and all $\mathbf{x} \in \mathbb{R}^d$, $g(\mathbf{x}) = \langle \Phi(\mathbf{x}), g\rangle_{\mathcal{H}}$. As a special case, taking $g = k_\sigma(\cdot,\mathbf{x}')$, we obtain

$$k_\sigma(\mathbf{x},\mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}')\rangle_{\mathcal{H}}$$

for all $\mathbf{x},\mathbf{x}' \in \mathbb{R}^d$. We also note that, by translation invariance, the functions $\Phi(\mathbf{x})$ have constant norm in $\mathcal{H}$ because

$$\|\Phi(\mathbf{x})\|^2_{\mathcal{H}} = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x})\rangle_{\mathcal{H}} = k_\sigma(\mathbf{x},\mathbf{x}) = k_\sigma(\mathbf{0},\mathbf{0}).$$

We will denote $\tau = \|\Phi(\mathbf{x})\|_{\mathcal{H}}$.

From this point of view, the KDE can be expressed as

$$\begin{aligned}\widehat{f}_{KDE}(\cdot) &= \frac{1}{n}\sum_{i=1}^n k_\sigma(\cdot,\mathbf{X}_i)\\ &= \frac{1}{n}\sum_{i=1}^n \Phi(\mathbf{X}_i),\end{aligned}$$

the sample mean of the $\Phi(\mathbf{X}_i)$'s in $\mathcal{H}$. Equivalently, $\widehat{f}_{KDE} \in \mathcal{H}$ is the solution of

$$\min_{g\in\mathcal{H}} \sum_{i=1}^n \|\Phi(\mathbf{X}_i) - g\|^2_{\mathcal{H}}.$$

Being the solution of a least squares problem, the KDE is sensitive to the presence of outliers among the $\Phi(\mathbf{X}_i)$'s. To reduce the effect of outliers, we propose to use M-estimation (Huber, 1964) to find a robust sample mean of the $\Phi(\mathbf{X}_i)$'s. For a robust loss function $\rho(x)$ on $x \geq 0$, the robust kernel density estimate is defined as

$$\widehat{f}_{RKDE} = \arg\min_{g\in\mathcal{H}} \sum_{i=1}^n \rho\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}\big). \tag{2}$$

Well-known examples of robust loss functions are Huber's or Hampel's ρ. Unlike the quadratic loss, these loss functions have the property that $\psi \triangleq \rho'$ is bounded. Huber's ρ and ψ are given by

$$\rho(x) = \begin{cases} x^2/2 & , 0 \leq x \leq a \\ ax - a^2/2 & , a < x \end{cases}$$

$$\psi(x) = \begin{cases} x & , 0 \le x \le a \\ a & , a < x, \end{cases} \tag{3}$$

and Hampel's ρ and ψ are

$$\rho(x) = \begin{cases} x^2/2 & , 0 \le x < a \\ ax - a^2/2 & , a \le x < b \\ a(x-c)^2/2(b-c) + a(b+c-a)/2 & , b \le x < c \\ a(b+c-a)/2 & , c \le x \end{cases}$$

$$\psi(x) = \begin{cases} x & , 0 \le x < a \\ a & , a \le x < b \\ a \cdot (c-x)/(c-b) & , b \le x < c \\ 0 & , c \le x. \end{cases} \tag{4}$$

The functions $\rho(x), \psi(x)$, and $\psi(x)/x$ are plotted in Figure 1, for the quadratic, Huber, and Hampel losses. Note that while $\psi(x)/x$ is constant for the quadratic loss, for Huber's or Hampel's loss, this function is decreasing in x. This is a desirable property for a robust loss function, which will be explained later in detail. While our examples and experiments employ Huber's and Hampel's losses, many other losses can be employed.

We will argue below that $\widehat{f}_{RKDE}$ is a valid density, having the form $\sum_{i=1}^n w_i k_\sigma(\cdot, \mathbf{X}_i)$ with weights w_i that are nonnegative and sum to one. To illustrate the estimator, Figure 2 (a) shows a contour plot of a Gaussian mixture distribution on $\mathbb{R}^2$. Figure 2 (b) depicts a contour plot of a KDE based on a training sample of size 200 from the Gaussian mixture. As we can see in Figure 2 (c) and (d), when 20 contaminating data points are added, the KDE is significantly altered in low density regions, while the RKDE is much less affected.

We remark that the definition of the RKDE does not require that k_σ be a reproducing kernel, only that the functions $\Phi(\mathbf{x}) = k_\sigma(\cdot, \mathbf{x})$ belong to a Hilbert space. Indeed, almost all of the results presented below hold in this more general setting. However, we restrict our attention to reproducing kernels for two reasons. First, with reproducing kernels, inner products in $\mathcal{H}$ can be easily computed via the kernel, leading to efficient implementation. Second, the reproducing property allows us to interpret the representer theorem and influence function to infer robustness of the RKDE. With non-reproducing kernels, these interpretations are less clear. The extension to non-RKHS Hilbert spaces is discussed in Section 6, with specific examples.

Throughout this paper, we define $\varphi(x) \triangleq \psi(x)/x$ and consider the following assumptions on ρ, ψ, and φ:

(A1) ρ is non-decreasing, $\rho(0) = 0$, and $\rho(x)/x \to 0$ as $x \to 0$

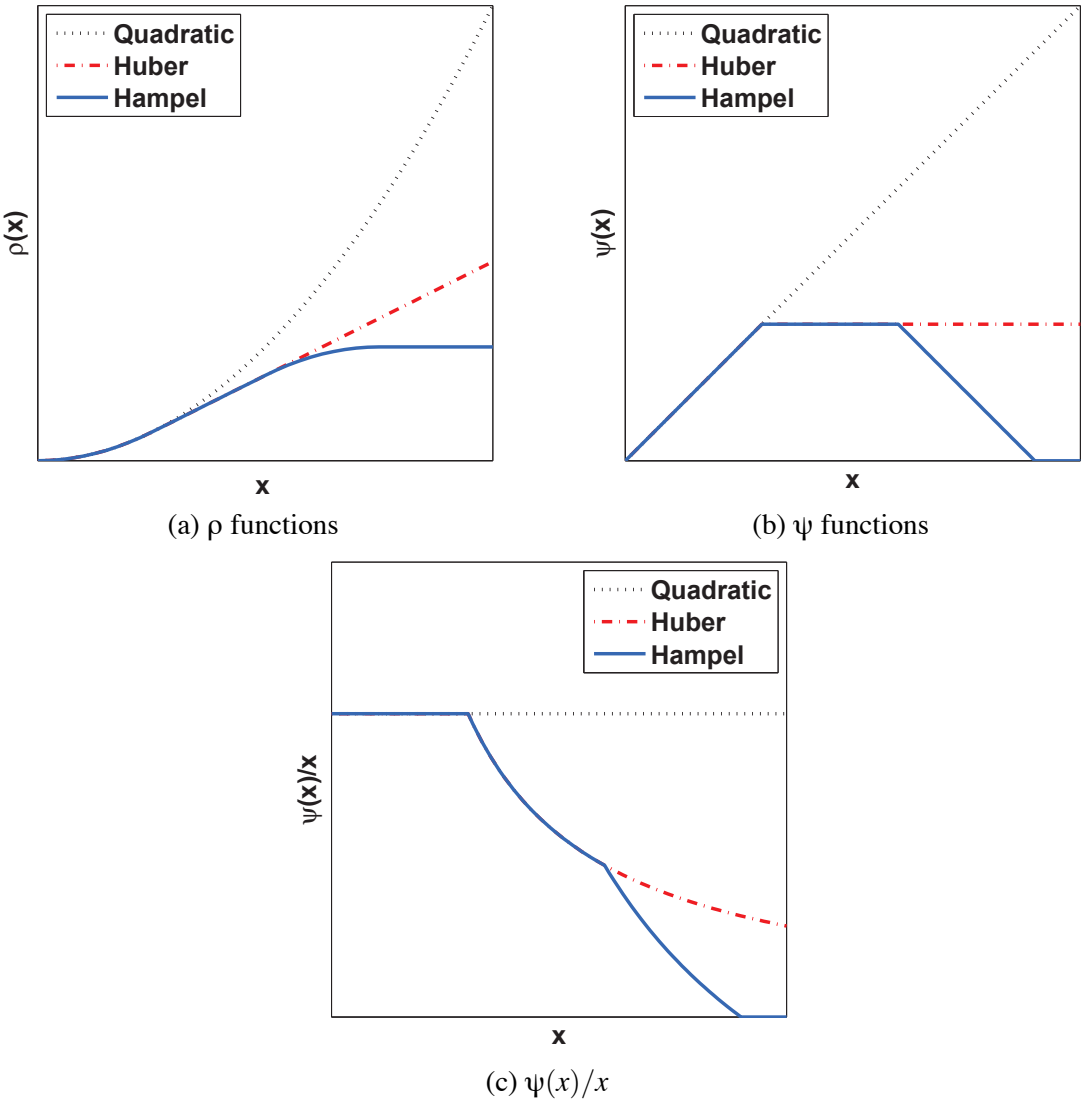
(A2) $\varphi(0) \triangleq \lim_{x \to 0} \frac{\psi(x)}{x}$ exists and is finite

(A3) ψ and φ are continuous

(A4) ψ and φ are bounded

(A5) φ is Lipschitz continuous

which hold for Huber's and Hampel's losses, as well as several others.



(a) ρ functions

(b) ψ functions

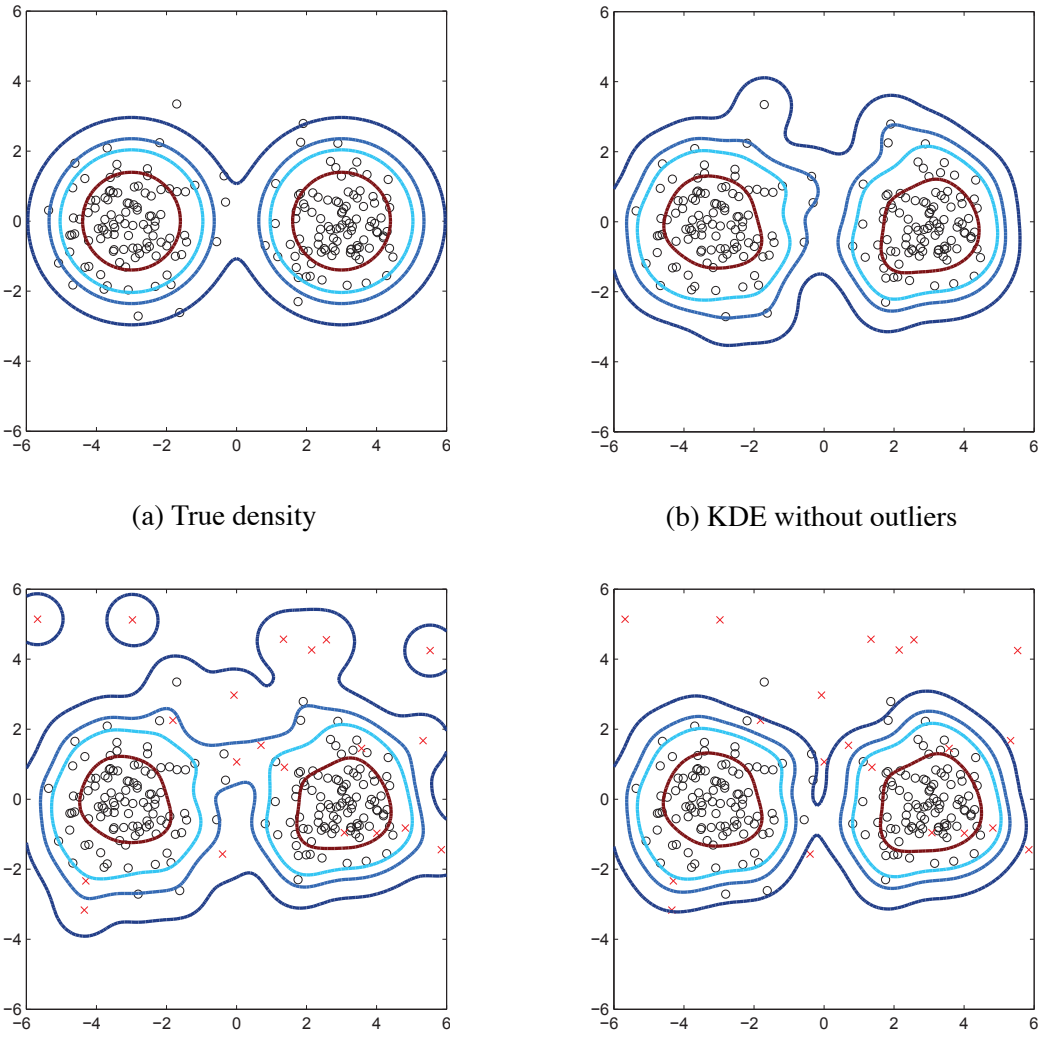
(c) $\psi(x)/x$

Figure 1: The comparison between three different $\rho(x)$, $\psi(x)$, and $\psi(x)/x$: quadratic, Huber's, and Hampel's.

3. Representer Theorem

In this section, we will describe how $\widehat{f}_{RKDE}$ can be expressed as a weighted combination of the $k_\sigma(\cdot, \mathbf{X}_i)$'s. A formula for the weights explains how a robust sample mean in $\mathcal{H}$ translates to a robust nonparametric density estimate. We also present necessary and sufficient conditions for a function to be an RKDE. From (2), $\widehat{f}_{RKDE} = \arg\min_{g \in \mathcal{H}} J(g)$, where

$$J(g) = \frac{1}{n} \sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}). \tag{5}$$



(a) True density

(b) KDE without outliers

(c) KDE with outliers

(d) RKDE with outliers

Figure 2: Contours of a nominal density (a) and kernel density estimates (b-d) along with data samples from the nominal density (o) and contaminating density (x). 200 points are from the nominal distribution and 20 contaminating points are from a uniform distribution.

First, let us find necessary conditions for g to be a minimizer of J. Since the space over which we are optimizing J is a Hilbert space, the necessary conditions are characterized through Gateaux differentials of J. Given a vector space $\mathcal{X}$ and a function $T : \mathcal{X} \to \mathbb{R}$, the Gateaux differential of T at $x \in \mathcal{X}$ with incremental $h \in \mathcal{X}$ is defined as

$$\delta T(x;h) = \lim_{\alpha \to 0} \frac{T(x+\alpha h) - T(x)}{\alpha}.$$

If $\delta T(x_0;h)$ is defined for all $h \in \mathcal{X}$, a necessary condition for T to have a minimum at x_0 is that $\delta T(x_0;h) = 0$ for all $h \in \mathcal{X}$ (Luenberger, 1997). From this optimality principle, we can establish the following lemma.

Lemma 1 *Suppose assumptions (A1) and (A2) are satisfied. Then the Gateaux differential of J at $g \in \mathcal{H}$ with incremental $h \in \mathcal{H}$ is*

$$\delta J(g;h) = -\big\langle V(g), h \big\rangle_{\mathcal{H}}$$

where $V : \mathcal{H} \to \mathcal{H}$ is given by

$$V(g) = \frac{1}{n}\sum_{i=1}^{n} \varphi(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}) \cdot \big(\Phi(\mathbf{X}_i) - g\big).$$

A necessary condition for $g = \widehat{f}_{RKDE}$ is $V(g) = \mathbf{0}$.

Lemma 1 is used to establish the following representer theorem, so named because $\widehat{f}_{RKDE}$ can be represented as a weighted combination of kernels centered at the data points. Similar results are known for supervised kernel methods (Schölkopf et al., 2001).

Theorem 2 *Suppose assumptions (A1) and (A2) are satisfied. Then,*

$$\widehat{f}_{RKDE}(\mathbf{x}) = \sum_{i=1}^{n} w_i k_\sigma(\mathbf{x}, \mathbf{X}_i) \tag{6}$$

where $w_i \geq 0$, $\sum_{i=1}^n w_i = 1$. Furthermore,

$$w_i \propto \varphi(\|\Phi(\mathbf{X}_i) - \widehat{f}_{RKDE}\|_{\mathcal{H}}). \tag{7}$$

It follows that $\widehat{f}_{RKDE}$ is a density. The representer theorem also gives the following interpretation of the RKDE. If φ is decreasing, as is the case for a robust loss, then w_i will be small when $\|\Phi(\mathbf{X}_i) - \widehat{f}_{RKDE}\|_{\mathcal{H}}$ is large. Now for any $g \in \mathcal{H}$,

$$\begin{aligned}
\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}^2 &= \langle \Phi(\mathbf{X}_i) - g, \Phi(\mathbf{X}_i) - g \rangle_{\mathcal{H}} \\
&= \|\Phi(\mathbf{X}_i)\|_{\mathcal{H}}^2 - 2\langle \Phi(\mathbf{X}_i), g \rangle_{\mathcal{H}} + \|g\|_{\mathcal{H}}^2 \\
&= \tau^2 - 2g(\mathbf{X}_i) + \|g\|_{\mathcal{H}}^2,
\end{aligned}$$

where the last step follows from the reproducing property. Taking $g = \widehat{f}_{RKDE}$, we see that w_i is small when $\widehat{f}_{RKDE}(\mathbf{X}_i)$ is small. Therefore, the RKDE is robust in the sense that it down-weights outlying points.

Theorem 2 provides a necessary condition for $\widehat{f}_{RKDE}$ to be the minimizer of (5). With an additional assumption on J, this condition is also sufficient.

Theorem 3 *Suppose that assumptions (A1) and (A2) are satisfied, and J is strictly convex. Then (6), (7), and $\sum_{i=1}^n w_i = 1$ are sufficient for $\widehat{f}_{RKDE}$ to be the minimizer of (5).*

Since the previous result assumes J is strictly convex, we give some simple conditions that imply this property.

Lemma 4 *J is strictly convex provided either of the following conditions is satisfied:*

(i) ρ *is strictly convex and non-decreasing.*

(ii) ρ *is convex, strictly increasing,* $n \geq 3$*, and* $K = (k_\sigma(\mathbf{X}_i, \mathbf{X}_j))_{i,j=1}^n$ *is positive definite.*

The second condition implies that J can be strictly convex even for the Huber loss, which is convex but not strictly convex.

4. KIRWLS Algorithm and Its Convergence

In general, (2) does not have a closed form solution and $\widehat{f}_{RKDE}$ has to be found by an iterative algorithm. Fortunately, the iteratively re-weighted least squares (IRWLS) algorithm used in classical M-estimation (Huber, 1964) can be extended to a RKHS using the *kernel trick*. The kernelized iteratively re-weighted least squares (KIRWLS) algorithm starts with initial $w_i^{(0)} \in \mathbb{R}$, $i = 1, \ldots, n$ such that $w_i^{(0)} \geq 0$ and $\sum_{i=1}^n w_i^{(0)} = 1$, and generates a sequence $\{f^{(k)}\}$ by iterating on the following procedure:

$$f^{(k)} = \sum_{i=1}^n w_i^{(k-1)} \Phi(\mathbf{X}_i),$$
$$w_i^{(k)} = \frac{\varphi(\|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}})}{\sum_{j=1}^n \varphi(\|\Phi(\mathbf{X}_j) - f^{(k)}\|_{\mathcal{H}})}.$$

Intuitively, this procedure is seeking a fixed point of Equations (6) and (7). The computation of $\|\Phi(\mathbf{X}_j) - f^{(k)}\|_{\mathcal{H}}$ can be done by observing

$$\begin{aligned}\|\Phi(\mathbf{X}_j) - f^{(k)}\|_{\mathcal{H}}^2 &= \left\langle \Phi(\mathbf{X}_j) - f^{(k)}, \Phi(\mathbf{X}_j) - f^{(k)} \right\rangle_{\mathcal{H}} \\ &= \left\langle \Phi(\mathbf{X}_j), \Phi(\mathbf{X}_j) \right\rangle_{\mathcal{H}} - 2\left\langle \Phi(\mathbf{X}_j), f^{(k)} \right\rangle_{\mathcal{H}} + \left\langle f^{(k)}, f^{(k)} \right\rangle_{\mathcal{H}}.\end{aligned}$$

Since $f^{(k)} = \sum_{i=1}^n w_i^{(k-1)} \Phi(\mathbf{X}_i)$, we have

$$\begin{aligned}\left\langle \Phi(\mathbf{X}_j), \Phi(\mathbf{X}_j) \right\rangle_{\mathcal{H}} &= k_\sigma(\mathbf{X}_j, \mathbf{X}_j) \\ \left\langle \Phi(\mathbf{X}_j), f^{(k)} \right\rangle_{\mathcal{H}} &= \sum_{i=1}^n w_i^{(k-1)} k_\sigma(\mathbf{X}_j, \mathbf{X}_i) \\ \left\langle f^{(k)}, f^{(k)} \right\rangle_{\mathcal{H}} &= \sum_{i=1}^n \sum_{l=1}^n w_i^{(k-1)} w_l^{(k-1)} k_\sigma(\mathbf{X}_i, \mathbf{X}_l).\end{aligned}$$

Recalling that $\Phi(\mathbf{x}) = k_\sigma(\cdot, \mathbf{x})$, after the kth iteration

$$f^{(k)}(\mathbf{x}) = \sum_{i=1}^n w_i^{(k-1)} k_\sigma(\mathbf{x}, \mathbf{X}_i).$$

Therefore, KIRWLS produces a sequence of weighted KDEs. The computational complexity is $O(n^2)$ per iteration. In our experience, the number of iterations needed is typically well below 100. Initialization and termination are discussed in the experimental study below.

KIRWLS can also be viewed as a kind of optimization transfer/majorize-minimize algorithm (Lange et al., 2000; Jacobson and Fessler, 2007) with a quadratic surrogate for ρ. This perspective is used in our analysis in Section 9.4, where $f^{(k)}$ is seen to be the solution of a weighted least squares problem in $\mathcal{H}$.

The next theorem characterizes the convergence of KIRWLS in terms of $\{J(f^{(k)})\}_{k=1}^{\infty}$ and $\{f^{(k)}\}_{k=1}^{\infty}$.

Theorem 5 *Suppose assumptions (A1) - (A3) are satisfied, and $\varphi(x)$ is nonincreasing. Let*

$$\mathcal{S} = \left\{ g \in \mathcal{H} \,\middle|\, V(g) = \mathbf{0} \right\}$$

and $\{f^{(k)}\}_{k=1}^{\infty}$ be the sequence produced by the KIRWLS algorithm. Then, $J(f^{(k)})$ monotonically decreases at every iteration and converges. Also, $\mathcal{S} \neq \emptyset$ and

$$\|f^{(k)} - \mathcal{S}\|_{\mathcal{H}} \triangleq \inf_{g \in \mathcal{S}} \|f^{(k)} - g\|_{\mathcal{H}} \to 0$$

as $k \to \infty$.

In words, as the number of iterations grows, $f^{(k)}$ becomes arbitrarily close to the set of stationary points of J, points $g \in \mathcal{H}$ satisfying $\delta J(g;h) = 0 \quad \forall h \in \mathcal{H}$.

Corollary 6 *Suppose that the assumptions in Theorem 5 hold and J is strictly convex. Then $\{f^{(k)}\}_{k=1}^{\infty}$ converges to $\widehat{f}_{RKDE}$ in the $\mathcal{H}$-norm and the supremum norm.*

Proof Convergence in the $\mathcal{H}$-norm follows from the previous result because under strict convexity of J, $|\mathcal{S}| = 1$. Convergence in the supremum norm now follows from the reproducing property and Cauchy-Schwarz because, for any $\mathbf{x}$,

$$\begin{aligned} |f^{(k)}(\mathbf{x}) - \widehat{f}_{RKDE}(\mathbf{x})| &= |\langle \Phi(\mathbf{x}), f^{(k)} - \widehat{f}_{RKDE} \rangle_{\mathcal{H}}| \\ &\leq \tau \|f^{(k)} - \widehat{f}_{RKDE}\|_{\mathcal{H}}. \end{aligned}$$

■

5. Influence Function for Robust KDE

To quantify the robustness of the RKDE, we study the influence function. First, we recall the traditional influence function from robust statistics. Let $T(F)$ be an estimator of a scalar parameter based on a distribution F. As a measure of robustness of T, the influence function was proposed by Hampel (1974). The influence function (IF) for T at F is defined as

$$IF(x';T,F) = \lim_{s \to 0} \frac{T((1-s)F + s\delta_{x'}) - T(F)}{s},$$

where $\delta_{x'}$ represents a discrete distribution that assigns probability 1 to the point x'. Basically, $IF(x';T,F)$ represents how $T(F)$ changes when the distribution F is contaminated with infinitesimal probability mass at x'. One robustness measure of T is whether the corresponding IF is bounded or not.

For example, the maximum likelihood estimator for the unknown mean of a Gaussian distribution is the sample mean $T(F)$,

$$T(F) = E_F[X] = \int x dF(x). \tag{8}$$

The influence function for $T(F)$ in (8) is

$$\begin{aligned} IF(x';T,F) &= \lim_{s\to 0} \frac{T((1-s)F + s\delta_{x'}) - T(F)}{s} \\ &= x' - E_F[X]. \end{aligned}$$

Since $|IF(x';T,F)|$ increases without bound as x' goes to $\pm\infty$, the estimator is considered to be not robust.

Now, consider a similar concept for a function estimate. Since the estimate is a function, not a scalar, we should be able to express the change of the function value at every $\mathbf{x}$.

Definition 7 (IF for function estimate) *Let $T(\mathbf{x};F)$ be a function estimate based on F, evaluated at $\mathbf{x}$. We define the influence function for $T(\mathbf{x};F)$ as*

$$IF(\mathbf{x},\mathbf{x}';T,F) = \lim_{s\to 0} \frac{T(\mathbf{x};F_s) - T(\mathbf{x};F)}{s}$$

where $F_s = (1-s)F + s\delta_{\mathbf{x}'}$.

$IF(\mathbf{x},\mathbf{x}';T,F)$ represents the change of the estimated function T at $\mathbf{x}$ when we add infinitesimal probability mass at $\mathbf{x}'$ to F. For example, the standard KDE is

$$\begin{aligned} T(\mathbf{x};F) = \widehat{f}_{KDE}(\mathbf{x};F) &= \int k_\sigma(\mathbf{x},\mathbf{y}) dF(\mathbf{y}) \\ &= E_F[k_\sigma(\mathbf{x},\mathbf{X})] \end{aligned}$$

where $\mathbf{X} \sim F$. In this case, the influence function is

$$\begin{aligned} IF(\mathbf{x},\mathbf{x}';\widehat{f}_{KDE},F) &= \lim_{s\to 0} \frac{\widehat{f}_{KDE}(\mathbf{x};F_s) - \widehat{f}_{KDE}(\mathbf{x};F)}{s} \\ &= \lim_{s\to 0} \frac{E_{F_s}[k_\sigma(\mathbf{x},\mathbf{X})] - E_F[k_\sigma(\mathbf{x},\mathbf{X})]}{s} \\ &= \lim_{s\to 0} \frac{-sE_F[k_\sigma(\mathbf{x},\mathbf{X})] + sE_{\delta_{\mathbf{x}'}}[k_\sigma(\mathbf{x},\mathbf{X})]}{s} \\ &= -E_F[k_\sigma(\mathbf{x},\mathbf{X})] + E_{\delta_{\mathbf{x}'}}[k_\sigma(\mathbf{x},\mathbf{X})] \\ &= -E_F[k_\sigma(\mathbf{x},\mathbf{X})] + k_\sigma(\mathbf{x},\mathbf{x}'). \end{aligned} \tag{9}$$

With the empirical distribution $F_n = \frac{1}{n}\sum_{i=1}^n \delta_{\mathbf{X}_i}$,

$$IF(\mathbf{x},\mathbf{x}';\widehat{f}_{KDE},F_n) = -\frac{1}{n}\sum_{i=1}^n k_\sigma(\mathbf{x},\mathbf{X}_i) + k_\sigma(\mathbf{x},\mathbf{x}'). \tag{10}$$

To investigate the influence function of the RKDE, we generalize its definition to a general distribution μ, writing $\widehat{f}_{RKDE}(\cdot;\mu) = f_\mu$ where

$$f_\mu = \operatorname*{arg\,min}_{g\in\mathcal{H}} \int \rho(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}})\, d\mu(\mathbf{x}).$$

For the robust KDE, $T(\mathbf{x},F) = \widehat{f}_{RKDE}(\mathbf{x};F) = \langle \Phi(\mathbf{x}), f_F \rangle_{\mathcal{H}}$, we have the following characterization of the influence function. Let $q(x) = x\psi'(x) - \psi(x)$.

Theorem 8 *Suppose assumptions (A1)-(A5) are satisfied. In addition, assume that $f_{F_s} \to f_F$ as $s \to 0$. If $\dot{f}_F \triangleq \lim_{s\to 0} \frac{f_{F_s} - f_F}{s}$ exists, then*

$$IF(\mathbf{x},\mathbf{x}';\widehat{f}_{RKDE},F) = \dot{f}_F(\mathbf{x})$$

where $\dot{f}_F \in \mathcal{H}$ satisfies

$$\begin{aligned}
&\left(\int \varphi(\|\Phi(\mathbf{x}) - f_F\|_{\mathcal{H}})dF\right)\cdot \dot{f}_F \\
&+ \quad \int\left(\frac{\left\langle \dot{f}_F, \Phi(\mathbf{x}) - f_F\right\rangle_{\mathcal{H}}}{\|\Phi(\mathbf{x}) - f_F\|^3_{\mathcal{H}}}\cdot q(\|\Phi(\mathbf{x}) - f_F\|_{\mathcal{H}})\cdot\left(\Phi(\mathbf{x}) - f_F\right)\right)dF(\mathbf{x}) \\
&= \quad (\Phi(\mathbf{x}') - f_F)\cdot\varphi(\|\Phi(\mathbf{x}') - f_F\|_{\mathcal{H}}). \qquad (11)
\end{aligned}$$

Unfortunately, for Huber or Hampel's ρ, there is no closed form solution for $\dot{f}_F$ of (11). However, if we work with F_n instead of F, we can find $\dot{f}_{F_n}$ explicitly. Let

$$\mathbf{1} = [1,\ldots,1]^T,$$
$$\mathbf{k}' = [k_\sigma(\mathbf{x}',\mathbf{X}_1),\ldots,k_\sigma(\mathbf{x}',\mathbf{X}_n)]^T,$$

I_n be the $n\times n$ identity matrix, $K \triangleq (k_\sigma(\mathbf{X}_i,\mathbf{X}_j))_{i=1,j=1}^n$ be the kernel matrix, Q be a diagonal matrix with $Q_{ii} = q(\|\Phi(\mathbf{X}_i) - f_{F_n}\|_{\mathcal{H}})/\|\Phi(\mathbf{X}_i) - f_{F_n}\|^3_{\mathcal{H}}$,

$$\gamma = \sum_{i=1}^n \varphi(\|\Phi(\mathbf{X}_i) - f_{F_n}\|_{\mathcal{H}}),$$

and

$$\mathbf{w} = [w_1,\ldots,w_n]^T,$$

where $\mathbf{w}$ gives the RKDE weights as in (6).

Theorem 9 *Suppose assumptions (A1)-(A5) are satisfied. In addition, assume that*

- *$f_{F_{n,s}} \to f_{F_n}$ as $s \to 0$ (satisfied when J is strictly convex)*
- *the extended kernel matrix K' based on $\{\mathbf{X}_i\}_{i=1}^n \bigcup \{\mathbf{x}'\}$ is positive definite.*

Then,

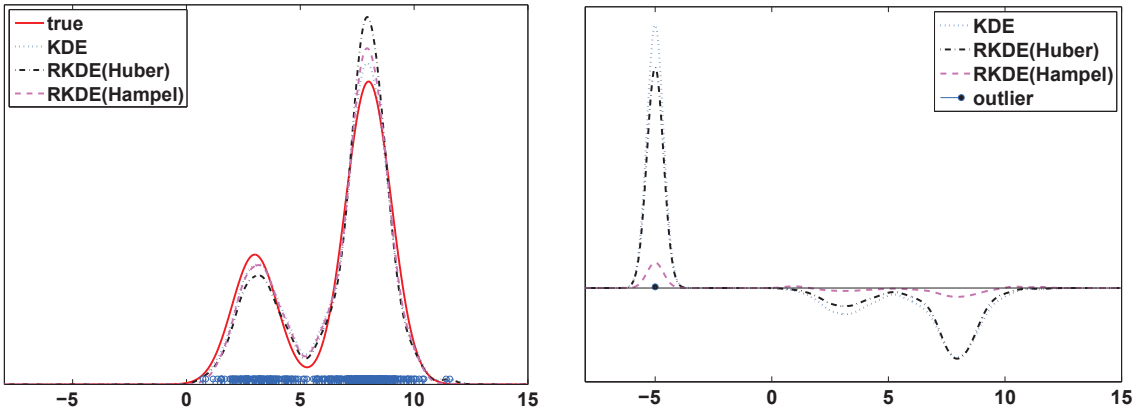
$$IF(\mathbf{x},\mathbf{x}';\widehat{f}_{RKDE},F_n) = \sum_{i=1}^n \alpha_i k_\sigma(\mathbf{x},\mathbf{X}_i) + \alpha' k_\sigma(\mathbf{x},\mathbf{x}')$$

where

$$\alpha' = n\cdot\varphi(\|\Phi(\mathbf{x}') - f_{F_n}\|_{\mathcal{H}})/\gamma$$

and $\alpha = [\alpha_1,\ldots,\alpha_n]^T$ is the solution of the following system of linear equations:

$$\begin{aligned}
&\left\{\gamma I_n + (I_n - \mathbf{1}\cdot\mathbf{w}^T)^T Q(I_n - \mathbf{1}\cdot\mathbf{w}^T)K\right\}\alpha \\
&= \quad -n\varphi(\|\Phi(\mathbf{x}') - f_{F_n}\|_{\mathcal{H}})\mathbf{w} - \alpha'(I_n - \mathbf{1}\cdot\mathbf{w}^T)^T Q\cdot(I_n - \mathbf{1}\cdot\mathbf{w}^T)\cdot\mathbf{k}'.
\end{aligned}$$



(a) (b)

Figure 3: (a) true density and density estimates. (b) IF as a function of $\mathbf{x}$ when $\mathbf{x}' = -5$

Note that α' captures the amount by which the density estimator changes near $\mathbf{x}'$ in response to contamination at $\mathbf{x}'$. Now α' is given by

$$\alpha' = \frac{\varphi(\|\Phi(\mathbf{x}') - f_{F_n}\|_{\mathcal{H}})}{\frac{1}{n}\sum_{i=1}^{n}\varphi(\|\Phi(\mathbf{X}_i) - f_{F_n}\|_{\mathcal{H}})}.$$

For a standard KDE, we have $\varphi \equiv 1$ and $\alpha' = 1$, in agreement with (10). For robust ρ, $\varphi(\|\Phi(\mathbf{x}') - f_{F_n}\|_{\mathcal{H}})$ can be viewed as a measure of "inlyingness", with more inlying points having larger values. This follows from the discussion just after Theorem 2, which leverages the reproducing property. If the contaminating point $\mathbf{x}'$ is less inlying than the average $\mathbf{X}_i$, then $\alpha' < 1$. Thus, the RKDE is less sensitive to outlying points than the KDE.

As mentioned above, in classical robust statistics, the robustness of an estimator can be inferred from the boundedness of the corresponding influence function. However, the influence functions for density estimators are bounded even if $\|\mathbf{x}'\| \to \infty$. Therefore, when we compare the robustness of density estimates, we compare how close the influence functions are to the zero function.

Simulation results are shown in Figure 3 for a synthetic univariate distribution. Figure 3 (a) shows the density of the distribution, and three estimates. Figure 3 (b) shows the corresponding influence functions. As we can see in (b), for a point $\mathbf{x}'$ in the tails of F, the influence functions for the robust KDEs are overall smaller, in absolute value, than those of the standard KDE (especially with Hampel's loss). Additional numerical results are given in Section 7.2.

Finally, it is interesting to note that for any density estimator $\widehat{f}$,

$$\int IF(\mathbf{x}, \mathbf{x}'; \widehat{f}, F)\, d\mathbf{x} = \lim_{s \to 0} \frac{\int \widehat{f}(\mathbf{x}; F_s)\, d\mathbf{x} - \int \widehat{f}(\mathbf{x}; F)\, d\mathbf{x}}{s} = 0.$$

Thus $\alpha' = -\sum_{i=1}^{n} \alpha_i$ for a robust KDE. This suggests that since $\widehat{f}_{RKDE}$ has a smaller increase at $\mathbf{x}'$ (compared to the KDE), it will also have a smaller decrease (in absolute value) near the training data. Therefore, the norm of $IF(\mathbf{x}, \mathbf{x}'; \widehat{f}_{RKDE}, F_n)$, viewed as a function of $\mathbf{x}$, should be smaller when $\mathbf{x}'$ is an outlier. We confirm this in our experiments in Section 7.

6. Generalization to Other Hilbert Spaces

So far, we have focused our attention on PSD kernels and viewed the KDE as an element of the RKHS associated with the kernel. However, the RKDE can be defined in a more general setting. In particular, it is only necessary that the functions $\Phi(\mathbf{x}) = k_\sigma(\cdot, \mathbf{x})$ belong to a Hilbert space $\mathcal{H}$. Then one can still obtain all the previous results, that is, Lemmas 1 and 4, Theorems 2, 3, 5, 8, and 9, and Corollary 6 (except convergence in the supremum norm). (For Theorems 8 and 9 it is necessary to additionally assume that $\|\Phi(\mathbf{x})\|_\mathcal{H}$ is bounded independent of $\mathbf{x}$.) The only necessary change is that inner products of the form $\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_\mathcal{H}$ can no longer be computed as $k_\sigma(\mathbf{x}, \mathbf{x}')$. Thus, K in Lemma 4 *(ii)*, $\mathbf{k}', K, K'$ in Theorem 9, and various terms in the KIRWLS algorithm should now be computed with the inner product on $\mathcal{H}$.

It is also interesting to note that this generalization gives a representer theorem for non-RKHS Hilbert spaces. This contrasts with standard approaches to supervised learning that minimize an empirical risk plus regularization term. In those cases, a representer theorem may be more difficult to obtain when the function class is not an RKHS.

There are some examples of non-RKHS Hilbert spaces where the inner product can still be computed efficiently. For example, consider $\mathcal{H} = L^2(\mathbb{R}^d)$ and denote

$$\begin{aligned}\tilde{k}_\sigma(\mathbf{x}, \mathbf{x}') &= \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{L^2(\mathbb{R}^d)} \\ &= \int k_\sigma(\mathbf{z}, \mathbf{x}) k_\sigma(\mathbf{z}, \mathbf{x}') d\mathbf{z}.\end{aligned}$$

For the multivariate Gaussian kernel, $\tilde{k}_\sigma = k_{\sqrt{2}\sigma}$. For the multivariate Cauchy kernel (the multivariate Student kernel with $\nu = 1$; see Section 2), it holds that $\tilde{k}_\sigma(\mathbf{x}, \mathbf{x}') = k_{2\sigma}(\mathbf{x}, \mathbf{x}')$ (Berry et al., 1996). For the multivariate Laplacian product kernel,

$$k_\sigma(\mathbf{x}, \mathbf{x}') = \frac{1}{(2\sigma)^d} \exp\left\{ -\frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\sigma} \right\},$$

it is true that

$$\tilde{k}_\sigma(\mathbf{x}, \mathbf{x}') = \frac{1}{(4\sigma)^d} \prod_{l=1}^{d} \left(1 + \frac{|\mathbf{x}_l - \mathbf{x}'_l|}{\sigma} \right) \exp\left\{ -\frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\sigma} \right\}.$$

For kernels without a closed-form expression for $\tilde{k}_\sigma$, the inner product may still be calculated numerically. For radially symmetric kernels this entails a simple one-dimensional table, as $\tilde{k}_\sigma(\mathbf{x}, \mathbf{x}')$ depends only on $\|\mathbf{x} - \mathbf{x}'\|$, and these values may be tabulated in advance.

As we noted previously, however, we rely on the reproducing property to deduce robustness of the RKDE from the representer theorem and the influence function. For non-RKHS Hilbert spaces, these arguments are less clear. We have not experimentally investigated non-reproducing kernels, and so cannot comment on the robustness of RKDEs based on such kernels in practice.

7. Experiments

The experimental setup is described in 7.1, and results are presented in 7.2.

7.1 Experimental Setup

Data, methods, and evaluation are now discussed.

7.1.1 DATA

We conduct experiments on 15 benchmark data sets (Banana, B. Cancer, Diabetes, F. Solar, German, Heart, Image, Ringnorm, Splice, Thyroid, Twonorm, Waveform, Pima Indian, Iris, MNIST), which were originally used in the task of classification. The data sets are available online: see http://www.fml.tuebingen.mpg.de/Members/ for the first 12 data sets and the UCI machine learning repository for the last 3 data sets. There are 100 randomly permuted partitions of each data set into "training" and "test" sets (20 for Image, Splice, and MNIST).

Given $\mathbf{X}_1,\ldots,\mathbf{X}_n \sim f = (1-p)\cdot f_0 + p\cdot f_1$, our goal is to estimate f_0, or the level sets of f_0. For each data set with two classes, we take one class as the nominal data from f_0 and the other class as contamination from f_1. For Iris, there are 3 classes and we take one class as nominal data and the other two as contamination. For MNIST, we choose to use digit 0 as nominal and digit 1 as contamination. For MNIST, the original dimension 784 is reduced to 8 via kernel PCA using a Gaussian kernel with bandwidth 30. For each data set, the training sample consists of n_0 nominal data and n_1 contaminating points, where $n_1 = \varepsilon \cdot n_0$ for $\varepsilon = 0, 0.05, 0.10, 0.15, 0.20, 0.25$ and 0.30. Note that each ε corresponds to an anomaly proportion p such that $p = \frac{\varepsilon}{1+\varepsilon}$. n_0 is always taken to be the full amount of training data for the nominal class.

7.1.2 METHODS

In our experiments, we compare three density estimators: the standard kernel density estimator (KDE), variable kernel density estimator (VKDE), and robust kernel density estimator (RKDE) with Hampel's loss. For all methods, the Gaussian kernel in (1) is used as the kernel function k_σ and the kernel bandwidth σ is set as the median distance of a training point $\mathbf{X}_i$ to its nearest neighbor.

The VKDE has a variable bandwidth for each data point,

$$\widehat{f}_{VKDE}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} k_{\sigma_i}(\mathbf{x},\mathbf{X}_i),$$

and the bandwidth σ_i is set as

$$\sigma_i = \sigma \cdot \left(\frac{\eta}{\widehat{f}_{KDE}(\mathbf{X}_i)}\right)^{1/2}$$

where η is the mean of $\{\widehat{f}_{KDE}(\mathbf{X}_i)\}_{i=1}^n$ (Abramson, 1982; Comaniciu et al., 2001). There is another implementation of the VKDE where σ_i is based on the distance to its k-th nearest neighbor (Breiman et al., 1977). However, this version did not perform as well and is therefore omitted.

For the RKDE, the parameters a, b, and c in (4) are set as follows. First, we compute $\widehat{f}_{med}$, which is the RKDE obtained with $\rho = |\cdot|$, and set $d_i = \|\Phi(\mathbf{X}_i) - \widehat{f}_{med}\|_{\mathcal{H}}$. Then, a is set to be the median of $\{d_i\}$, b the 75th percentile of $\{d_i\}$, and c the 85th percentile of $\{d_i\}$. After finding these parameters, we initialize $w_i^{(0)}$ such that $f^{(1)} = \widehat{f}_{med}$ and terminate KIRWLS when

$$\frac{|J(f^{(k+1)}) - J(f^{(k)})|}{J(f^{(k)})} < 10^{-8}.$$

7.1.3 EVALUATION

We evaluate the performance of the three density estimators in three different settings. First, we use the influence function to study sensitivity to outliers. Second and third, we compare the methods

at the tasks of density estimation and anomaly detection, respectively. In each case, an appropriate performance measure is adopted. These are explained in detail in Section 7.2. To compare a pair of methods across multiple data sets, we adopt the Wilcoxon signed-rank test (Wilcoxon, 1945). Given a performance measure, and given a pair of methods and ε, we compute the difference h_i between the performance of two density estimators on the ith data set. The data sets are ranked 1 through 15 according to their absolute values $|h_i|$, with the largest $|h_i|$ corresponding to the rank of 15. Let R_1 be the sum of ranks over these data sets where method 1 beats method 2, and let R_2 be the sum of the ranks for the other data sets. The signed-rank test statistic $T \triangleq \min(R_1, R_2)$ and the corresponding p-value are used to test whether the performances of the two methods are significantly different. For example, the critical value of T for the signed rank test is 25 at a significance level of 0.05. Thus, if $T \leq 25$, the two methods are significantly different at the given significance level, and the larger of R_1 and R_2 determines the method with better performance.

7.2 Experimental Results

We begin by studying influence functions.

7.2.1 SENSITIVITY USING INFLUENCE FUNCTION

As the first measure of robustness, we compare the influence functions for KDEs and RKDEs, given in (10) and Theorem 9, respectively. To our knowledge, there is no formula for the influence function of VKDEs, and therefore VKDEs are excluded in the comparison. We examine $\alpha(\mathbf{x}') = IF(\mathbf{x}', \mathbf{x}'; T, F_n)$ and

$$\beta(\mathbf{x}') = \left(\int \left(IF(\mathbf{x}, \mathbf{x}'; T, F_n) \right)^2 d\mathbf{x} \right)^{1/2}.$$

In words, $\alpha(\mathbf{x}')$ reflects the change of the density estimate value at an added point $\mathbf{x}'$ and $\beta(\mathbf{x}')$ is an overall impact of $\mathbf{x}'$ on the density estimate over $\mathbb{R}^d$.

In this experiment, ε is equal to 0, that is, the density estimators are learned from a pure nominal sample. Then, we take contaminating points from the test sample, each of which serves as an $\mathbf{x}'$. This gives us multiple $\alpha(\mathbf{x}')$'s and $\beta(\mathbf{x}')$'s. The performance measures are the medians of $\{\alpha(\mathbf{x}')\}$ and $\{\beta(\mathbf{x}')\}$ (smaller means better performance). The results using signed rank statistics are shown in Table 1. The results clearly states that for all data sets, RKDEs are less affected by outliers than KDEs.

7.2.2 KULLBACK-LEIBLER (KL) DIVERGENCE

Second, we present the Kullback-Leibler (KL) divergence between density estimates $\widehat{f}$ and f_0,

$$D_{KL}(\widehat{f} \,||\, f_0) = \int \widehat{f}(\mathbf{x}) \log \frac{\widehat{f}(\mathbf{x})}{f_0(\mathbf{x})} d\mathbf{x}.$$

This KL divergence is large whenever $\widehat{f}$ estimates f_0 to have mass where it does not. For contamination characterized by properties **(a)**, **(b)**, and **(c)** in the Introduction, we expect this performance measure to capture the robustness of a density estimator.

The computation of D_{KL} is done as follows. Since we do not know the nominal f_0, it is estimated as $\widetilde{f}_0$, a KDE based on a separate nominal sample, obtained from the test data for each benchmark

method 1	method 2		$\alpha(\mathbf{x}')$	$\beta(\mathbf{x}')$
RKDE	KDE	R_1	120	120
		R_2	0	0
		T	0	0
		p-value	0.00	0.00

Table 1: The signed-rank statistics and p-values of the Wilcoxon signed-rank test using the medians of $\{\alpha(\mathbf{x}')\}$ and $\{\beta(\mathbf{x}')\}$ as a performance measure. If R_1 is larger than R_2, method 1 is better than method 2.

data set. Then, the integral is approximated by the sample mean, that is,

$$D_{KL}(\widehat{f}\,||\,f_0) \approx \frac{1}{n'}\sum_{i=1}^{n'} \log \frac{\widehat{f}(\mathbf{x}'_i)}{\widetilde{f_0}(\mathbf{x}'_i)}$$

where $\{\mathbf{x}'_i\}_{i=1}^{n'}$ is an i.i.d sample from the estimated density $\widehat{f}$ with $n' = 2n = 2(n_0+n_1)$. Note that the estimated KL divergence can have an infinite value when $\widetilde{f_0}(\mathbf{y}) = 0$ (to machine precision) and $\widehat{f}(\mathbf{y}) > 0$ for some $\mathbf{y} \in \mathbb{R}^d$. The averaged KL divergence over the permutations are used as the performance measure (smaller means better performance). In Table 2, the rank of the three methods are shown for each data set and ε.

Table 3 summarizes the results using the Wilcoxon signed-rank test. When comparing RKDEs and KDEs, the results show that KDEs have smaller KL divergence than RKDEs with $\varepsilon = 0$. As ε increases, however, RKDEs estimate f_0 more accurately than KDEs. The results also demonstrate that VKDEs are the worst in the sense of KL divergence. Note that VKDEs place a total mass of $1/n$ at all $\mathbf{X}_i$, whereas the RKDE will place a mass $w_i < 1/n$ at outlying points.

Since KL divergence is not symmetric, we also compute KL divergence between f_0 and $\widehat{f}$,

$$\begin{aligned} D_{KL}(f_0\,||\,\widehat{f}) &= \int f_0(\mathbf{x}) \log \frac{f_0(\mathbf{x})}{\widehat{f}(\mathbf{x})} d\mathbf{x} \\ &= \int f_0(\mathbf{x}) \log f_0(\mathbf{x}) d\mathbf{x} - \int f_0(\mathbf{x}) \log \widehat{f}(\mathbf{x}) d\mathbf{x} \end{aligned}$$

This KL divergence is large whenever $\widehat{f}$ estimates f_0 not to have mass where it does.

Since f_0 is the same when comparing different estimate $\widehat{f}$, we only have to compare the second term, which is approximated as

$$-\int f_0(\mathbf{x}) \log \widehat{f}(\mathbf{x}) d\mathbf{x} \approx -\frac{1}{n''}\sum_{i=1}^{n''} \log \widehat{f}(\mathbf{x}''_i),$$

where $\{\mathbf{x}''_i\}_{i=1}^{n''}$ is a separate nominal sample, obtained from the test data. Table 4 and 5 show that with this KL divergence as performance measure, VKDE performs best for almost all data sets and ε. When comparing RKDE and KDE based on Wilcoxon signed-rank test, T has a large value for every ε which makes it inconclusive whether one method is better than the other.

Data Set	ε						
	0.00	0.05	0.10	0.15	0.20	0.25	0.30
Banana	(3,1,2)	(3,1,2)	(3,1,2)	(3,1,2)	(3,1,2)	(3,1,2)	(3,1,2)
B. Cancer	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Diabetes	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(1,2,3)	(1,2,3)
F. Solar	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)
German	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Heart	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Image	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Ringnorm	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Splice	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Thyroid	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Twonorm	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Waveform	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)
Pima Indian	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(1,2,3)
Iris	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
MNIST	(3,1,2)	(2,1,3)	(2,1,3)	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)

Table 2: The ranks of (RKDE, KDE, VKDE) using $D_{KL}(\widehat{f}\,||\,f_0)$ as a performance measure. For example, (2, 1, 3) means that KDE performs best, RKDE next, and VKDE worst.

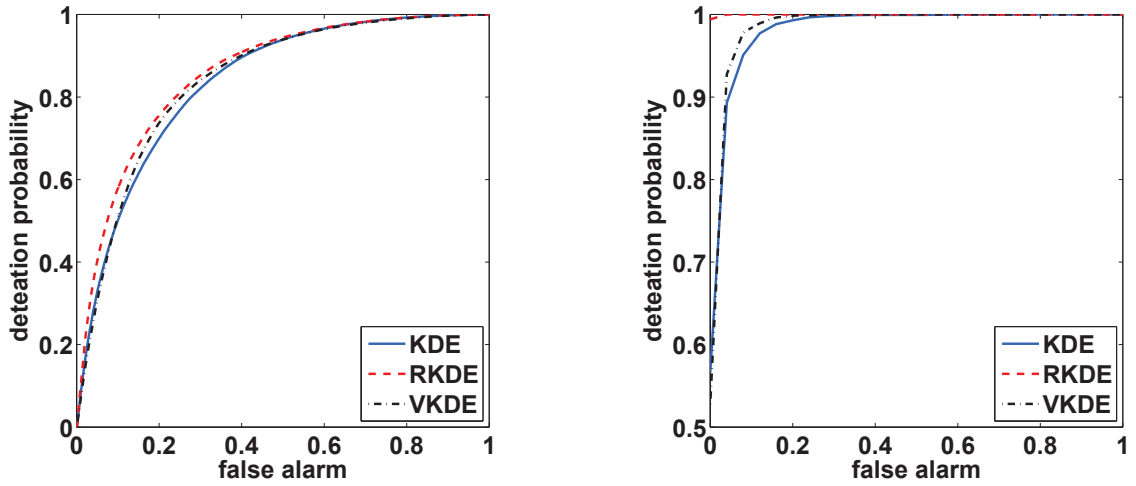
7.2.3 ANOMALY DETECTION

In this experiment, we apply the density estimators in anomaly detection problems. If we had a pure sample from f_0, we would estimate f_0 and use $\{\mathbf{x} : \widehat{f_0}(\mathbf{x}) > \lambda\}$ as a detector. For each λ, we could get a false negative and false positive probability using test data. By varying λ, we would then obtain a receiver operating characteristic (ROC) and area under the curve (AUC). However, since we have a contaminated sample, we have to estimate f_0 robustly. Robustness can be checked by comparing the AUC of the anomaly detectors, where the density estimates are based on the contaminated training data (higher AUC means better performance).

Examples of the ROCs are shown in Figure 4. The RKDE provides better detection probabilities, especially at low false alarm rates. This results in higher AUC. For each pair of methods and each ε, R_1, R_2, T and p-values are shown in Table 7. The results indicate that RKDEs are significantly better than KDEs when $\varepsilon \geq 0.20$ with significance level 0.05. RKDEs are also better than VKDEs when $\varepsilon \geq 0.15$ but the difference is not significant. We also note that we have also evaluated the kernelized spatial depth (KSD) (Chen et al., 2009) in this setting. While this method does not yield a density estimate, it does aim to estimate density contours robustly. We found that the KSD performs worse in terms of AUC that either the RKDE or KDE, so those results are omitted (Kim and Scott, 2011).

method 1	method 2		ε						
			0.00	0.05	0.10	0.15	0.20	0.25	0.30
RKDE	KDE	R_1	26	67	78	83	94	101	103
		R_2	94	53	42	37	26	19	17
		T	26	53	42	37	26	19	17
		p-value	0.06	0.72	0.33	0.21	0.06	0.02	0.01
RKDE	VKDE	R_1	104	117	117	117	117	119	119
		R_2	16	3	3	3	3	1	1
		T	16	3	3	3	3	1	1
		p-value	0.01	0.00	0.00	0.00	0.00	0.00	0.00
VKDE	KDE	R_1	0	0	0	0	0	0	0
		R_2	120	120	120	120	120	120	120
		T	0	0	0	0	0	0	0
		p-value	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 3: The signed-rank statistics and p-values of the Wilcoxon signed-rank test using $D_{KL}(\widehat{f}\,||\,f_0)$ as a performance measure. If R_1 is larger than R_2, method 1 is better than method 2.



(a) Banana, $\varepsilon = 0.2$ (b) Iris, $\varepsilon = 0.1$

Figure 4: Examples of ROCs.

8. Conclusions

When kernel density estimators employ a smoothing kernel that is also a PSD kernel, they may be viewed as M-estimators in the RKHS associated with the kernel. While the traditional KDE corresponds to the quadratic loss, the RKDE employs a robust loss to achieve robustness to contamination of the training sample. The RKDE is a weighted kernel density estimate, where smaller weights are given to more outlying data points. These weights can be computed efficiently using

Data Set	ε						
	0.00	0.05	0.10	0.15	0.20	0.25	0.30
Banana	(3,1,2)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)
B. Cancer	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)
Diabetes	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)
F. Solar	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
German	(3,2,1)	(3,2,1)	(3,2,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Heart	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Image	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)
Ringnorm	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Splice	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Thyroid	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Twonorm	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Waveform	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(2,3,1)	(2,3,1)	(2,3,1)
Pima Indian	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)
Iris	(3,2,1)	(3,2,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
MNIST	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)

Table 4: The ranks of (RKDE, KDE, VKDE) using $D_{KL}(f_0 \,||\, \widehat{f})$ as a performance measure. For example, $(2, 1, 3)$ means that KDE performs best, RKDE next, and VKDE worst.

a kernelized iteratively re-weighted least squares algorithm. The decreased sensitivity of RKDEs to contamination is further attested by the influence function, as well as experiments on anomaly detection and density estimation problems.

Robust kernel density estimators are nonparametric, making no parametric assumptions on the data generating distributions. However, their success is still contingent on certain conditions being satisfied. Obviously, the percentage of contaminating data must be less than 50%; our experiments examine contamination up to around 25%. In addition, the contaminating distribution must be outlying with respect to the nominal distribution. Furthermore, the anomalous component should not be too concentrated, otherwise it may look like a mode of the nominal component. Such assumptions seem necessary given the unsupervised nature of the problem, and are implicit in our interpretation of the representer theorem and influence functions.

Although our focus has been on density estimation, in many applications the ultimate goal is not to estimate a density, but rather to estimate decision regions. Our methodology is immediately applicable to such situations, as evidenced by our experiments on anomaly detection. It is only necessary that the kernel be PSD here; the assumption that the kernel be nonnegative and integrate to one can clearly be dropped. This allows for the use of more general kernels, such as polynomial kernels, or kernels on non-Euclidean domains such as strings and trees. The learning problem here could be described as one-class classification with contaminated data.

In future work it would be interesting to investigate asymptotics, the bias-variance trade-off, and the efficiency-robustness trade-off of robust kernel density estimators, as well as the impact of different losses and kernels.

method 1	method 2		ε						
			0.00	0.05	0.10	0.15	0.20	0.25	0.30
RKDE	KDE	R_1	47	52	55	58	64	65	68
		R_2	73	68	65	62	56	55	52
		T	47	52	55	58	56	55	52
		p-value	0.49	0.68	0.80	0.93	0.85	0.80	0.68
RKDE	VKDE	R_1	0	0	0	0	0	0	0
		R_2	120	120	120	120	120	120	120
		T	0	0	0	0	0	0	0
		p-value	0.00	0.00	0.00	0.00	0.00	0.00	0.00
VKDE	KDE	R_1	119	120	120	120	120	120	120
		R_2	1	0	0	0	0	0	0
		T	1	0	0	0	0	0	0
		p-value	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 5: The signed-rank statistics and p-values of the Wilcoxon signed-rank test using $D_{KL}(f_0 \,||\, \widehat{f})$ divergence as a performance measure. If R_1 is larger than R_2, method 1 is better than method 2.

9. Proofs

We begin with three lemmas and proofs. The first lemma will be used in the proofs of Lemma 11 and Theorem 9, the second one in the proof of Lemma 4, and the third one in the proof of Theorem 5.

Lemma 10 *Let $\mathbf{z}_1, \ldots, \mathbf{z}_m$ be distinct points in $\mathbb{R}^d$. If $K = (k(\mathbf{z}_i, \mathbf{z}_j))_{i,j=1}^n$ is positive definite, then $\Phi(\mathbf{z}_i) = k(\cdot, \mathbf{z}_i)$'s are linearly independent.*

Proof $\sum_{i=1}^m \alpha_i \Phi(\mathbf{z}_i) = 0$ implies

$$\begin{aligned} 0 &= \left\| \sum_{i=1}^m \alpha_i \Phi(\mathbf{z}_i) \right\|_{\mathcal{H}}^2 \\ &= \left\langle \sum_{i=1}^m \alpha_i \Phi(\mathbf{z}_i), \sum_{j=1}^m \alpha_j \Phi(\mathbf{z}_j) \right\rangle_{\mathcal{H}} \\ &= \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j k(\mathbf{z}_i, \mathbf{z}_j) \end{aligned}$$

and from positive definiteness of K, $\alpha_1 = \cdots = \alpha_m = 0$. ■

Lemma 11 *Let $\mathcal{H}$ be a RKHS associated with a kernel k, and $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ be distinct points in $\mathbb{R}^d$. Assume that $K = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^3$ is positive definite. For any $g, h \in \mathcal{H}$ with $g \neq h$, $\Phi(\mathbf{x}_i) - g$ and $\Phi(\mathbf{x}_i) - h$ are linearly independent for some $i \in \{1,2,3\}$.*

Data Set	ε						
	0.00	0.05	0.10	0.15	0.20	0.25	0.30
Banana	(3,2,1)	(3,2,1)	(3,2,1)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)
B. Cancer	(2,1,3)	(2,1,3)	(2,1,3)	(1,3,2)	(1,3,2)	(1,3,2)	(2,3,1)
Diabetes	(3,1,2)	(3,2,1)	(2,3,1)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)
F. Solar	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(3,1,2)
German	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)
Heart	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Image	(3,1,2)	(3,1,2)	(3,1,2)	(2,3,1)	(2,3,1)	(1,3,2)	(1,3,2)
Ringnorm	(2,1,3)	(2,1,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)	(1,2,3)
Splice	(1,2,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)	(2,1,3)
Thyroid	(3,1,2)	(3,2,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)	(2,3,1)
Twonorm	(3,2,1)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)
Waveform	(2,3,1)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)
Pima Indian	(3,1,2)	(3,2,1)	(2,3,1)	(2,3,1)	(2,3,1)	(1,3,2)	(1,3,2)
Iris	(3,1,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)	(1,3,2)
MNIST	(3,1,2)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)	(3,2,1)

Table 6: The ranks of (RKDE, KDE, VKDE) using $D_{KL}(f_0 \,||\, \widehat{f})$ as a performance measure. For example, (2, 1, 3) means that KDE performs best, RKDE next, and VKDE worst.

method 1	method 2		ε						
			0.00	0.05	0.10	0.15	0.20	0.25	0.30
RKDE	KDE	R_1	26	46	67	90	95	96	99
		R_2	94	74	53	30	25	24	21
		T	26	46	53	30	25	24	21
		p-value	0.06	0.45	0.72	0.09	0.05	0.04	0.03
RKDE	VKDE	R_1	33	49	58	75	80	90	86
		R_2	87	71	62	45	40	30	34
		T	33	49	58	45	40	30	34
		p-value	0.14	0.56	0.93	0.42	0.28	0.09	0.15
VKDE	KDE	R_1	38	70	79	91	95	96	99
		R_2	82	50	41	29	25	24	21
		T	38	50	41	29	25	24	21
		p-value	0.23	0.60	0.30	0.08	0.05	0.04	0.03

Table 7: The signed-rank statistics of the Wilcoxon signed-rank test using AUC as a performance measure. If R_1 is larger than R_2, method 1 is better than method 2.

Proof We will prove the lemma by contradiction. Suppose $\Phi(\mathbf{x}_i) - g$ and $\Phi(\mathbf{x}_i) - h$ are linearly dependent for all $i = 1,2,3$. Then, there exists $(\alpha_i, \beta_i) \neq (0,0)$ for $i = 1,2,3$ such that

$$\alpha_1(\Phi(\mathbf{x}_1)-g)+\beta_1(\Phi(\mathbf{x}_1)-h)=\mathbf{0} \tag{12}$$
$$\alpha_2(\Phi(\mathbf{x}_2)-g)+\beta_2(\Phi(\mathbf{x}_2)-h)=\mathbf{0} \tag{13}$$
$$\alpha_3(\Phi(\mathbf{x}_3)-g)+\beta_3(\Phi(\mathbf{x}_3)-h)=\mathbf{0}. \tag{14}$$

Note that $\alpha_i+\beta_i\neq 0$ since $g\neq h$.

First consider the case $\alpha_2=0$. This gives $h=\Phi(\mathbf{x}_2)$, and $\alpha_1\neq 0$ and $\alpha_3\neq 0$. Then, (12) and (13) simplify to

$$g=\frac{\alpha_1+\beta_1}{\alpha_1}\Phi(\mathbf{x}_1)-\frac{\beta_1}{\alpha_1}\Phi(\mathbf{x}_2),$$
$$g=\frac{\alpha_3+\beta_3}{\alpha_3}\Phi(\mathbf{x}_3)-\frac{\beta_3}{\alpha_3}\Phi(\mathbf{x}_2),$$

respectively. This is contradiction because $\Phi(\mathbf{x}_1)$, $\Phi(\mathbf{x}_2)$, and $\Phi(\mathbf{x}_3)$ are linearly independent by Lemma 10 and

$$\frac{\alpha_1+\beta_1}{\alpha_1}\Phi(\mathbf{x}_1)+\left(\frac{\beta_3}{\alpha_3}-\frac{\beta_1}{\alpha_1}\right)\Phi(\mathbf{x}_2)-\frac{\alpha_3+\beta_3}{\alpha_3}\Phi(\mathbf{x}_3)=\mathbf{0}$$

where $(\alpha_1+\beta_1)/\alpha_1\neq 0$.

Now consider the case where $\alpha_2\neq 0$. Subtracting (13) multiplied by α_1 from (12) multiplied by α_2 gives

$$(\alpha_1\beta_2-\alpha_2\beta_1)h=-\alpha_2(\alpha_1+\beta_1)\Phi(\mathbf{x}_1)+\alpha_1(\alpha_2+\beta_2)\Phi(\mathbf{x}_2).$$

In the above equation $\alpha_1\beta_2-\alpha_2\beta_1\neq 0$ because this implies $\alpha_2(\alpha_1+\beta_1)=0$ and $\alpha_1(\alpha_2+\beta_2)=0$, which, in turn, implies $\alpha_2=0$. Therefore, h can be expressed as $h=\lambda_1\Phi(\mathbf{x}_1)+\lambda_2\Phi(\mathbf{x}_2)$ where

$$\lambda_1=-\frac{\alpha_2(\alpha_1+\beta_1)}{\alpha_1\beta_2-\alpha_2\beta_1},\quad \lambda_2=\frac{\alpha_1(\alpha_2+\beta_2)}{\alpha_1\beta_2-\alpha_2\beta_1}.$$

Similarly, from (13) and (14), $h=\lambda_3\Phi(\mathbf{x}_2)+\lambda_4\Phi(\mathbf{x}_3)$ where

$$\lambda_3=-\frac{\alpha_3(\alpha_2+\beta_2)}{\alpha_2\beta_3-\alpha_3\beta_2},\quad \lambda_4=\frac{\alpha_2(\alpha_3+\beta_3)}{\alpha_2\beta_3-\alpha_3\beta_2}.$$

Therefore, we have $h=\lambda_1\Phi(\mathbf{x}_1)+\lambda_2\Phi(\mathbf{x}_2)=\lambda_3\Phi(\mathbf{x}_2)+\lambda_4\Phi(\mathbf{x}_3)$. Again, from the linear independence of $\Phi(\mathbf{x}_1)$, $\Phi(\mathbf{x}_2)$, and $\Phi(\mathbf{x}_3)$, we have $\lambda_1=0$, $\lambda_2=\lambda_3$, $\lambda_4=0$. However, $\lambda_1=0$ leads to $\alpha_2=0$.

Therefore $\Phi(\mathbf{x}_i)-g$ and $\Phi(\mathbf{x}_i)-h$ are linearly independent for some $i\in\{1,2,3\}$. ■

Lemma 12 *Given $\mathbf{X}_1,\ldots,\mathbf{X}_n$, let $\mathcal{D}_n\subset\mathcal{H}$ be defined as*

$$\mathcal{D}_n=\left\{g\,\middle|\,g=\sum_{i=1}^n w_i\cdot\Phi(\mathbf{X}_i),\quad w_i\geq 0,\quad \sum_{i=1}^n w_i=1\right\}$$

Then $\mathcal{D}_n$ is compact.

Proof Define

$$A = \left\{ (w_1, \ldots, w_n) \in \mathbb{R}^n \middle| \, w_i \geq 0, \quad \sum_{i=1}^{n} w_i = 1 \right\},$$

and a mapping W

$$W : (w_1, \ldots, w_n) \in A \to \sum_{i=1}^{n} w_i \cdot \Phi(\mathbf{X}_i) \in \mathcal{H}.$$

Note that A is compact, W is continuous, and $\mathcal{D}_n$ is the image of A under W. Since the continuous image of a compact space is also compact (Munkres, 2000), $\mathcal{D}_n$ is compact. ■

9.1 Proof of Lemma 1

We begin by calculating the Gateaux differential of J. We consider the two cases: $\Phi(\mathbf{x}) - (g + \alpha h) = \mathbf{0}$ and $\Phi(\mathbf{x}) - (g + \alpha h) \neq \mathbf{0}$.

For $\Phi(\mathbf{x}) - (g + \alpha h) \neq \mathbf{0}$,

$$\begin{aligned}
&\frac{\partial}{\partial \alpha} \rho\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big) \\
&= \psi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big) \cdot \frac{\partial}{\partial \alpha} \|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}} \\
&= \psi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big) \cdot \frac{\partial}{\partial \alpha} \sqrt{\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}^2} \\
&= \psi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big) \cdot \frac{\frac{\partial}{\partial \alpha} \|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}^2}{2\sqrt{\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}^2}} \\
&= \frac{\psi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big)}{2\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}} \cdot \frac{\partial}{\partial \alpha} \left(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}^2 - 2\big\langle \Phi(\mathbf{x}) - g, \alpha h \big\rangle_{\mathcal{H}} + \alpha^2 \|h\|_{\mathcal{H}}^2 \right) \\
&= \frac{\psi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big)}{\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}} \cdot \left(-\big\langle \Phi(\mathbf{x}) - g, h \big\rangle_{\mathcal{H}} + \alpha \|h\|_{\mathcal{H}}^2 \right) \\
&= \varphi\big(\|\Phi(\mathbf{x}) - (g + \alpha h)\|_{\mathcal{H}}\big) \cdot \big(-\big\langle \Phi(\mathbf{x}) - (g + \alpha h), h \big\rangle_{\mathcal{H}} \big). \qquad (15)
\end{aligned}$$

For $\Phi(\mathbf{x}) - (g+\alpha h) = \mathbf{0}$,

$$\begin{aligned}
&\frac{\partial}{\partial\alpha}\rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \\
&= \lim_{\delta\to 0} \frac{\rho\big(\|\Phi(\mathbf{x}) - (g+(\alpha+\delta)h)\|_{\mathcal{H}}\big) - \rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big)}{\delta} \\
&= \lim_{\delta\to 0} \frac{\rho\big(\|\delta h\|_{\mathcal{H}}\big) - \rho\big(0\big)}{\delta} \\
&= \lim_{\delta\to 0} \frac{\rho\big(\delta\|h\|_{\mathcal{H}}\big)}{\delta} \\
&= \begin{cases} \lim_{\delta\to 0} \frac{\rho(0)}{\delta}, & h = \mathbf{0} \\ \lim_{\delta\to 0} \frac{\rho(\delta\|h\|_{\mathcal{H}})}{\delta\|h\|_{\mathcal{H}}} \cdot \|h\|_{\mathcal{H}}, & h \neq \mathbf{0} \end{cases} \\
&= 0 \\
&= \varphi\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \cdot \big(-\big\langle \Phi(\mathbf{x}) - (g+\alpha h), h\big\rangle_{\mathcal{H}}\big) \qquad (16)
\end{aligned}$$

where the second to the last equality comes from (A1) and the last equality comes from the facts that $\Phi(\mathbf{x}) - (g+\alpha h) = \mathbf{0}$ and $\varphi(0)$ is well-defined by (A2).

From (15) and (16), we can conclude that for any $g, h \in \mathcal{H}$, and $\mathbf{x} \in \mathbb{R}^d$,

$$\begin{aligned}
&\frac{\partial}{\partial\alpha}\rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \\
&\quad = \varphi\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \cdot \big(-\big\langle \Phi(\mathbf{x}) - (g+\alpha h), h\big\rangle_{\mathcal{H}}\big) \qquad (17)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\delta J(g;h) &= \frac{\partial}{\partial\alpha} J(g+\alpha h)\big|_{\alpha=0} \\
&= \frac{\partial}{\partial\alpha}\left(\frac{1}{n}\sum_{i=1}^{n}\rho\big(\|\Phi(\mathbf{X}_i) - (g+\alpha h)\|_{\mathcal{H}}\big)\right)\Bigg|_{\alpha=0} \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\alpha}\rho\big(\|\Phi(\mathbf{X}_i) - (g+\alpha h)\|_{\mathcal{H}}\big)\Big|_{\alpha=0} \\
&= \frac{1}{n}\sum_{i=1}^{n}\varphi\big(\|\Phi(\mathbf{X}_i) - (g+\alpha h)\|_{\mathcal{H}}\big) \cdot \big(-\big\langle \Phi(\mathbf{X}_i) - (g+\alpha h), h\big\rangle_{\mathcal{H}}\big)\Big|_{\alpha=0} \\
&= -\frac{1}{n}\sum_{i=1}^{n}\varphi\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}\big) \cdot \big\langle \Phi(\mathbf{X}_i) - g, h\big\rangle_{\mathcal{H}} \\
&= -\left\langle \frac{1}{n}\sum_{i=1}^{n}\varphi\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}\big) \cdot \big(\Phi(\mathbf{X}_i) - g\big), h\right\rangle_{\mathcal{H}} \\
&= -\big\langle V(g), h\big\rangle_{\mathcal{H}}.
\end{aligned}$$

The necessary condition for g to be a minimizer of J, that is, $g = \widehat{f}_{RKDE}$, is that $\delta J(g;h) = 0, \quad \forall h \in \mathcal{H}$, which leads to $V(g) = \mathbf{0}$.

9.2 Proof of Theorem 2

From Lemma 1, $V(\widehat{f}_{RKDE}) = \mathbf{0}$, that is,

$$\frac{1}{n}\sum_{i=1}^{n} \varphi(\|\Phi(\mathbf{X}_i) - \widehat{f}_{RKDE}\|_{\mathcal{H}}) \cdot (\Phi(\mathbf{X}_i) - \widehat{f}_{RKDE}) = \mathbf{0}.$$

Solving for $\widehat{f}_{RKDE}$, we have $\widehat{f}_{RKDE} = \sum_{i=1}^{n} w_i \Phi(\mathbf{X}_i)$ where

$$w_i = \left(\sum_{j=1}^{n} \varphi(\|\Phi(\mathbf{X}_j) - \widehat{f}_{RKDE}\|_{\mathcal{H}}) \right)^{-1} \cdot \varphi(\|\Phi(\mathbf{X}_i) - \widehat{f}_{RKDE}\|_{\mathcal{H}}).$$

Since ρ is non-decreasing, $w_i \geq 0$. Clearly $\sum_{i=1}^{n} w_i = 1$

9.3 Proof of Lemma 4

J is strictly convex on $\mathcal{H}$ if for any $0 < \lambda < 1$, and $g, h \in \mathcal{H}$ with $g \neq h$

$$J(\lambda g + (1-\lambda)h) < \lambda J(g) + (1-\lambda)J(h).$$

Note that

$$\begin{aligned}
J(\lambda g + (1-\lambda)h) &= \frac{1}{n}\sum_{i=1}^{n} \rho\big(\|\Phi(\mathbf{X}_i) - \lambda g - (1-\lambda)h\|_{\mathcal{H}}\big) \\
&= \frac{1}{n}\sum_{i=1}^{n} \rho\big(\|\lambda(\Phi(\mathbf{X}_i) - g) + (1-\lambda)(\Phi(\mathbf{X}_i) - h)\|_{\mathcal{H}}\big) \\
&\leq \frac{1}{n}\sum_{i=1}^{n} \rho\big(\lambda\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}} + (1-\lambda)\|\Phi(\mathbf{X}_i) - h\|_{\mathcal{H}}\big) \\
&\leq \frac{1}{n}\sum_{i=1}^{n} \lambda\rho\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}\big) + (1-\lambda)\rho\big(\|\Phi(\mathbf{X}_i) - h\|_{\mathcal{H}}\big) \\
&= \lambda J(g) + (1-\lambda)J(h).
\end{aligned}$$

The first inequality comes from the fact that ρ is non-decreasing and

$$\|\lambda(\Phi(\mathbf{X}_i) - g) + (1-\lambda)(\Phi(\mathbf{X}_i) - h)\|_{\mathcal{H}} \leq \lambda\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}} + (1-\lambda)\|\Phi(\mathbf{X}_i) - h\|_{\mathcal{H}},$$

and the second inequality comes from the convexity of ρ.

Under condition *(i)*, ρ is strictly convex and thus the second inequality is strict, implying J is strictly convex. Under condition *(ii)*, we will show that the first inequality is strict using proof by contradiction. Suppose the first inequality holds with equality. Since ρ is strictly increasing, this can happen only if

$$\|\lambda(\Phi(\mathbf{X}_i) - g) + (1-\lambda)(\Phi(\mathbf{X}_i) - h)\|_{\mathcal{H}} = \lambda\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}} + (1-\lambda)\|\Phi(\mathbf{X}_i) - h\|_{\mathcal{H}},$$

for $i = 1, \ldots, n$. Equivalently, it can happen only if $(\Phi(\mathbf{X}_i) - g)$ and $(\Phi(\mathbf{X}_j) - h)$ are linearly dependent for all $i = 1, \ldots, n$. However, from $n \geq 3$ and positive definiteness of K, there exist three distinct $\mathbf{X}_i$'s, say $\mathbf{Z}_1$, $\mathbf{Z}_2$, and $\mathbf{Z}_3$ with positive definite $K' = (k_\sigma(\mathbf{Z}_i, \mathbf{Z}_j))_{i,j=1}^{3}$. By Lemma 11, it must be the case that for some $i \in \{1,2,3\}$, $(\Phi(\mathbf{Z}_i) - g)$ and $(\Phi(\mathbf{Z}_i) - h)$ are linearly independent. Therefore, the inequality is strict, and thus J is strictly convex.

9.4 Proof of Theorem 5

First, we will prove the monotone decreasing property of $J(f^{(k)})$. Given $r \in \mathbb{R}$, define

$$u(x;r) = \rho(r) - \frac{1}{2} r\psi(r) + \frac{1}{2}\varphi(r)x^2.$$

If φ is nonincreasing, then u is a surrogate function of ρ, having the following property (Huber, 1981):

$$u(r;r) = \rho(r) \tag{18}$$
$$u(x;r) \geq \rho(x), \quad \forall x. \tag{19}$$

Define

$$Q(g;f^{(k)}) = \frac{1}{n}\sum_{i=1}^{n} u\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}, \|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}\big).$$

Note that since ψ and φ are continuous, $Q(\cdot;\cdot)$ is continuous in both arguments.

From (18) and (19), we have

$$\begin{aligned} Q(f^{(k)};f^{(k)}) &= \frac{1}{n}\sum_{i=1}^{n} u\big(\|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}, \|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}\big) \\ &= \frac{1}{n}\sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}) \\ &= J(f^{(k)}) \end{aligned} \tag{20}$$

and

$$\begin{aligned} Q(g;f^{(k)}) &= \frac{1}{n}\sum_{i=1}^{n} u\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}, \|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}\big) \\ &\geq \frac{1}{n}\sum_{i=1}^{n} \rho\big(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}\big) \\ &= J(g), \quad \forall g \in \mathcal{H} \end{aligned} \tag{21}$$

The next iterate $f^{(k+1)}$ is the minimizer of $Q(g;f^{(k)})$ since

$$\begin{aligned} f^{(k+1)} &= \sum_{i=1}^{n} w_i^{(k)} \Phi(\mathbf{X}_i) \\ &= \sum_{i=1}^{n} \frac{\varphi(\|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}})}{\sum_{j=1}^{n} \varphi(\|\Phi(\mathbf{X}_j) - f^{(k)}\|_{\mathcal{H}})} \Phi(\mathbf{X}_i) \\ &= \arg\min_{g \in \mathcal{H}} \sum_{i=1}^{n} \varphi(\|\Phi(\mathbf{X}_i) - f^{(k)}\|_{\mathcal{H}}) \cdot \|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}^2 \\ &= \arg\min_{g \in \mathcal{H}} Q(g;f^{(k)}) \end{aligned} \tag{22}$$

From (20), (21), and (22),

$$J(f^{(k)}) = Q(f^{(k)};f^{(k)}) \geq Q(f^{(k+1)};f^{(k)}) \geq J(f^{(k+1)})$$

and thus $J(f^{(k)})$ monotonically decreases at every iteration. Since $\{J(f^{(k)})\}_{k=1}^{\infty}$ is bounded below by 0, it converges.

Next, we will prove that every limit point f^* of $\{f^{(k)}\}_{k=1}^{\infty}$ belongs to $\mathcal{S}$. Since the sequence $\{f^{(k)}\}_{k=1}^{\infty}$ lies in the compact set $\mathcal{D}_n$ (see Theorem 2 and Lemma 12), it has a convergent subsequence $\{f^{(k_l)}\}_{l=1}^{\infty}$. Let f^* be the limit of $\{f^{(k_l)}\}_{l=1}^{\infty}$. Again, from (20), (21), and (22),

$$\begin{aligned} Q(f^{(k_l+1)};f^{(k_l+1)}) &= J(f^{(k_l+1)}) \\ &\leq J(f^{(k_l+1)}) \\ &\leq Q(f^{(k_l+1)};f^{(k_l)}) \\ &\leq Q(g;f^{(k_l)}) \quad ,\forall g \in \mathcal{H}, \end{aligned}$$

where the first inequality comes from the monotone decreasing property of $J(f^{(k)})$. By taking the limit on the both side of the above inequality, we have

$$Q(f^*;f^*) \leq Q(g;f^*) \quad ,\forall g \in \mathcal{H}.$$

Therefore,

$$\begin{aligned} f^* &= \arg\min_{g\in\mathcal{H}} Q(g;f^*) \\ &= \sum_{i=1}^{n} \frac{\varphi(\|\mathbf{\Phi}(\mathbf{X}_i)-f^*\|_{\mathcal{H}})}{\sum_{j=1}^{n}\varphi(\|\mathbf{\Phi}(\mathbf{X}_j)-f^*\|_{\mathcal{H}})}\mathbf{\Phi}(\mathbf{X}_i) \end{aligned}$$

and thus

$$\sum_{i=1}^{n} \varphi(\|\mathbf{\Phi}(\mathbf{X}_i)-f^*\|_{\mathcal{H}})\cdot(\mathbf{\Phi}(\mathbf{X}_i)-f^*) = \mathbf{0}.$$

This implies $f^* \in \mathcal{S}$.

Now we will prove $\|f^{(k)}-\mathcal{S}\|_{\mathcal{H}} \to 0$ by contradiction. Suppose $\inf_{g\in\mathcal{S}}\|f^{(k)}-g\|_{\mathcal{H}} \not\to 0$. Then, there exists $\varepsilon > 0$ such that $\forall K \in \mathbb{N}, \exists k > K$ with $\inf_{g\in\mathcal{S}}\|f^{(k)}-g\|_{\mathcal{H}} \geq \varepsilon$. Thus, we can construct an increasing sequence of indices $\{k_l\}_{l=1}^{\infty}$ such that $\inf_{g\in\mathcal{S}}\|f^{(k_l)}-g\|_{\mathcal{H}} \geq \varepsilon$ for all $l = 1,2,\ldots$. Since $\{f^{(k_l)}\}_{l=1}^{\infty}$ lies in the compact set $\mathcal{D}_n$, it has a subsequence converging to some $f^\dagger$, and we can choose j such that $\|f^{(k_j)}-f^\dagger\|_{\mathcal{H}} < \varepsilon/2$. Since $f^\dagger$ is also a limit point of $\{f^{(k)}\}_{k=1}^{\infty}$, $f^\dagger \in \mathcal{S}$. This is a contradiction because

$$\varepsilon \leq \inf_{g\in\mathcal{S}}\|f^{(k_j)}-g\|_{\mathcal{H}} \leq \|f^{(k_j)}-f^\dagger\|_{\mathcal{H}} \leq \varepsilon/2.$$

9.5 Proof of Theorem 8

Since the RKDE is given as $\widehat{f}_{RKDE}(\mathbf{x};F) = f_F(\mathbf{x})$, the influence function for the RKDE is

$$\begin{aligned} IF(\mathbf{x},\mathbf{x}';\widehat{f}_{RKDE},F) &= \lim_{s\to 0}\frac{\widehat{f}_{RKDE}(\mathbf{x};F_s)-\widehat{f}_{RKDE}(\mathbf{x};F)}{s} \\ &= \lim_{s\to 0}\frac{f_{F_s}(\mathbf{x})-f_F(\mathbf{x})}{s} \end{aligned}$$

and thus we need to find $\dot{f}_F \triangleq \lim_{s\to 0} \frac{f_{F_s} - f_F}{s}$.

As we generalize the definition of RKDE from $\widehat{f}_{RKDE}$ to f_F, the necessary condition $V(\widehat{f}_{RKDE})$ also generalizes. However, a few things must be taken care of since we are dealing with integral instead of summation. Suppose ψ and φ are bounded by B' and B'', respectively. Given a probability measure μ, define

$$J_\mu(g) = \int \rho(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}})\, d\mu(\mathbf{x}). \tag{23}$$

From (17),

$$\begin{aligned}
\delta J_\mu(g;h) &= \frac{\partial}{\partial\alpha} J_\mu(g+\alpha h)\big|_{\alpha=0} \\
&= \frac{\partial}{\partial\alpha} \int \rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big)\, d\mu(\mathbf{x})\bigg|_{\alpha=0} \\
&= \int \frac{\partial}{\partial\alpha} \rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big)\, d\mu(\mathbf{x})\bigg|_{\alpha=0} \\
&= \int \varphi\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \cdot \big(-\big\langle \Phi(\mathbf{x}) - (g+\alpha h), h\big\rangle_{\mathcal{H}}\big)\, d\mu(\mathbf{x})\bigg|_{\alpha=0} \\
&= -\int \varphi\big(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}\big) \cdot \big\langle \Phi(\mathbf{x}) - g, h\big\rangle_{\mathcal{H}} d\mu(\mathbf{x}) \\
&= -\int \Big\langle \varphi\big(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}\big) \cdot \big(\Phi(\mathbf{x}) - g\big), h \Big\rangle_{\mathcal{H}} d\mu(\mathbf{x}).
\end{aligned}$$

The exchange of differential and integral is valid (Lang, 1993) since for any fixed $g, h \in \mathcal{H}$, and $\alpha \in (-1,1)$

$$\begin{aligned}
&\left| \frac{\partial}{\partial\alpha} \rho\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|_{\mathcal{H}}\big) \right| \\
&\quad = \varphi\big(\|\Phi(\mathbf{x}) - (g+\alpha h)\|\big) \cdot \big|-\big\langle \Phi(\mathbf{x}) - (g+\alpha h), h\big\rangle_{\mathcal{H}}\big| \\
&\quad \le B'' \cdot \|\Phi(\mathbf{x}) - (g+\alpha h)\| \cdot \|h\|_{\mathcal{H}} \\
&\quad \le B'' \cdot \big(\|\Phi(\mathbf{x})\|_{\mathcal{H}} + \|g\|_{\mathcal{H}} + \|h\|_{\mathcal{H}}\big) \cdot \|h\|_{\mathcal{H}} \\
&\quad \le B'' \cdot \big(\tau + \|g\|_{\mathcal{H}} + \|h\|_{\mathcal{H}}\big) \cdot \|h\|_{\mathcal{H}} < \infty.
\end{aligned}$$

Since $\varphi(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}) \cdot \big(\Phi(\mathbf{x}) - g\big)$ is strongly integrable, that is,

$$\int \big\|\varphi\big(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}\big) \cdot \big(\Phi(\mathbf{x}) - g\big)\big\|_{\mathcal{H}} d\mu(\mathbf{x}) \le B' < \infty,$$

its Bochner-integral (Berlinet and Thomas-Agnan, 2004)

$$V_\mu(g) \triangleq \int \varphi(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}) \cdot (\Phi(\mathbf{x}) - g)\, d\mu(\mathbf{x})$$

is well-defined. Therefore, we have

$$\begin{aligned}
\delta J_\mu(g;h) &= -\left\langle \int \varphi\big(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}}\big) \cdot \big(\Phi(\mathbf{x}) - g\big)\, d\mu(\mathbf{x}), h \right\rangle_{\mathcal{H}} \\
&= -\big\langle V_\mu(g), h\big\rangle_{\mathcal{H}}.
\end{aligned}$$

and $V_\mu(f_\mu) = \mathbf{0}$.

From the above condition for f_{F_s}, we have

$$\begin{aligned}\mathbf{0} &= V_{F_s}(f_{F_s}) \\ &= (1-s)\cdot V_F(f_{F_s}) + sV_{\delta_{\mathbf{x}'}}(f_{F_s}), \quad \forall s \in [0,1)\end{aligned}$$

Therefore,

$$\begin{aligned}\mathbf{0} &= \lim_{s\to 0}(1-s)\cdot V_F(f_{F_s}) + \lim_{s\to 0} s\cdot V_{\delta_{\mathbf{x}'}}(f_{F_s}) \\ &= \lim_{s\to 0} V_F(f_{F_s}).\end{aligned}$$

Then,

$$\begin{aligned}\mathbf{0} &= \lim_{s\to 0}\frac{1}{s}\Big(V_{F_s}(f_{F_s}) - V_F(f_F)\Big) \\ &= \lim_{s\to 0}\frac{1}{s}\Big((1-s)V_F(f_{F_s}) + sV_{\delta_{\mathbf{x}'}}(f_{F_s}) - V_F(f_F)\Big) \\ &= \lim_{s\to 0}\frac{1}{s}\Big(V_F(f_{F_s}) - V_F(f_F)\Big) - \lim_{s\to 0} V_F(f_{F_s}) + \lim_{s\to 0} V_{\delta_{\mathbf{x}'}}(f_{F_s}) \\ &= \lim_{s\to 0}\frac{1}{s}\Big(V_F(f_{F_s}) - V_F(f_F)\Big) + \lim_{s\to 0} V_{\delta_{\mathbf{x}'}}(f_{F_s}) \\ &= \lim_{s\to 0}\frac{1}{s}\Big(V_F(f_{F_s}) - V_F(f_F)\Big) + \lim_{s\to 0}\varphi(\|\Phi(\mathbf{x}') - f_{F_s}\|)\cdot(\Phi(\mathbf{x}') - f_{F_s}) \\ &= \lim_{s\to 0}\frac{1}{s}\Big(V_F(f_{F_s}) - V_F(f_F)\Big) + \varphi(\|\Phi(\mathbf{x}') - f_F\|)\cdot(\Phi(\mathbf{x}') - f_F). \qquad (24)\end{aligned}$$

where the last equality comes from the facts that $f_{F_s} \to f_F$ and continuity of φ.

Let U denote the mapping $\mu \mapsto f_\mu$. Then,

$$\begin{aligned}\dot{f}_F &\triangleq \lim_{s\to 0}\frac{f_{F_s} - f_F}{s} \\ &= \lim_{s\to 0}\frac{U(F_s) - U(F)}{s} \\ &= \lim_{s\to 0}\frac{U\big((1-s)F + s\delta_{\mathbf{x}'}\big) - U(F)}{s} \\ &= \lim_{s\to 0}\frac{U\big(F + s(\delta_{\mathbf{x}'} - F)\big) - U(F)}{s} \\ &= \delta U(F; \delta_{\mathbf{x}'} - F) \qquad (25)\end{aligned}$$

where $\delta U(P;Q)$ is the Gateaux differential of U at P with increment Q. The first term in (24) is

$$\begin{aligned}
&\lim_{s\to 0}\frac{1}{s}\Big(V_F\big(f_{F_s}\big)-V_F\big(f_F\big)\Big)\\
&= \lim_{s\to 0}\frac{1}{s}\Big(V_F\big(U(F_s)\big)-V_F\big(U(F)\big)\Big)\\
&= \lim_{s\to 0}\frac{1}{s}\Big((V_F\circ U)\big(F_s\big)-(V_F\circ U)(F)\Big)\\
&= \lim_{s\to 0}\frac{1}{s}\Big((V_F\circ U)\big(F+s(\delta_{\mathbf{x}'}-F)\big)-(V_F\circ U)(F)\Big)\\
&= \delta(V_F\circ U)(F;\delta_{\mathbf{x}'}-F)\\
&= \delta V_F\big(U(F);\delta U(F;\delta_{\mathbf{x}'}-F)\big)\\
&= \delta V_F\big(f_F;\dot{f}_F\big) \qquad (26)
\end{aligned}$$

where we apply the chain rule of Gateaux differential, $\delta(G\circ H)(u;x)=\delta G(H(u);\delta H(u;x))$, in the second to the last equality. Although $\dot{f}_F$ is technically not a Gateaux differential since the space of probability distributions is not a vector space, the chain rule still applies.

Thus, we only need to find the Gateaux differential of V_F. For $g,h\in\mathcal{H}$

$$\begin{aligned}
\delta V_F(g;h) &= \lim_{s\to 0}\frac{1}{s}\Big(V_F(g+s\cdot h)-V_F(g)\Big)\\
&= \lim_{s\to 0}\frac{1}{s}\Big(\int\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})\cdot(\Phi(\mathbf{x})-g-s\cdot h)dF(\mathbf{x})\\
&\qquad -\int\varphi(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})\cdot(\Phi(\mathbf{x})-g)dF(\mathbf{x})\Big)\\
&= \lim_{s\to 0}\frac{1}{s}\int\Big(\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})-\varphi(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})\Big)\cdot(\Phi(\mathbf{x})-g)dF(\mathbf{x})\\
&\quad -\lim_{s\to 0}\frac{1}{s}\int\Big(\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})\cdot s\cdot h\Big)dF(\mathbf{x})\\
&= \int\lim_{s\to 0}\frac{1}{s}\Big(\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})-\varphi(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})\Big)\cdot(\Phi(\mathbf{x})-g)dF(\mathbf{x})\\
&\quad -h\cdot\int\lim_{s\to 0}\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})\,dF(\mathbf{x})\\
&= -\int\Big(\frac{\psi'(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})\cdot\|\Phi(\mathbf{x})-g\|_{\mathcal{H}}-\psi(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})}{\|\Phi(\mathbf{x})-g\|^2_{\mathcal{H}}}\cdot\frac{\langle h,\Phi(\mathbf{x})-g\rangle_{\mathcal{H}}}{\|\Phi(\mathbf{x})-g\|_{\mathcal{H}}}\Big)\\
&\qquad\cdot\big(\Phi(\mathbf{x})-g\big)\,dF(\mathbf{x})\\
&\quad -h\cdot\int\varphi(\|\Phi(\mathbf{x})-g\|_{\mathcal{H}})\,dF(\mathbf{x}) \qquad (27)
\end{aligned}$$

where in the last equality, we use the fact

$$\frac{\partial}{\partial s}\varphi(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})=\varphi'(\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}})\cdot\frac{\langle\Phi(\mathbf{x})-g-s\cdot h,h\rangle_{\mathcal{H}}}{\|\Phi(\mathbf{x})-g-s\cdot h\|_{\mathcal{H}}}$$

and

$$\varphi'(x) = \frac{d}{dx}\frac{\psi(x)}{x} = \frac{\psi'(x)x - \psi(x)}{x^2}.$$

The exchange of limit and integral is valid due to the dominated convergence theorem since under the assumption that φ is bounded and Lipschitz continuous with Lipschitz constant L,

$$\left|\varphi(\|\Phi(\mathbf{x}) - g - s\cdot h\|)\right| < \infty, \quad \forall \mathbf{x}$$

and

$$\begin{aligned}
&\left\|\frac{1}{s}\left(\varphi(\|\Phi(\mathbf{x}) - g - s\cdot h\|_{\mathcal{H}}) - \varphi(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}})\right)\cdot\left(\Phi(\mathbf{x}) - g\right)\right\|_{\mathcal{H}} \\
&= \frac{1}{s}\left|\varphi(\|\Phi(\mathbf{x}) - g - s\cdot h\|_{\mathcal{H}}) - \varphi(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}})\right|\cdot\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}} \\
&\leq \frac{1}{s}L\cdot\|s\cdot h\|_{\mathcal{H}}\cdot\left(\|\Phi(\mathbf{x})\|_{\mathcal{H}} + \|g\|_{\mathcal{H}}\right) \\
&\leq L\cdot\|h\|_{\mathcal{H}}\cdot\left(\|\Phi(\mathbf{x})\|_{\mathcal{H}} + \|g\|_{\mathcal{H}}\right) < \infty, \quad \forall \mathbf{x}.
\end{aligned}$$

By combining (24), (25), (26), and (27), we have

$$\begin{aligned}
&\left(\int \varphi(\|\Phi(\mathbf{x}) - f_F\|)dF\right)\cdot\dot{f}_F \\
&+ \int\left(\frac{\left\langle\dot{f}_F, \Phi(\mathbf{x}) - f_F\right\rangle_{\mathcal{H}}}{\|\Phi(\mathbf{x}) - f_F\|^3}\cdot q(\|\Phi(\mathbf{x}) - f_F\|)\cdot\left(\Phi(\mathbf{x}) - f_F\right)\right)dF(\mathbf{x}) \\
&= (\Phi(\mathbf{x}') - f_F)\cdot\varphi(\|\Phi(\mathbf{x}') - f_F\|)
\end{aligned}$$

where $q(x) = x\psi'(x) - \psi(x)$.

9.6 Proof of Theorem 9

With F_n instead of F, (11) becomes

$$\begin{aligned}
&\left(\frac{1}{n}\sum_{i=1}^{n}\varphi(\|\Phi(\mathbf{X}_i) - f_{F_n}\|)\right)\cdot\dot{f}_{F_n} \\
&+ \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\left\langle\dot{f}_{F_n}, \Phi(\mathbf{X}_i) - f_{F_n}\right\rangle_{\mathcal{H}}}{\|\Phi(\mathbf{X}_i) - f_{F_n}\|^3}\cdot q(\|\Phi(\mathbf{X}_i) - f_{F_n}\|)\cdot\left(\Phi(\mathbf{X}_i) - f_{F_n}\right)\right) \\
&= (\Phi(\mathbf{x}') - f_{F_n})\cdot\varphi(\|\Phi(\mathbf{x}') - f_{F_n}\|). \qquad (28)
\end{aligned}$$

Let $r_i = \|\Phi(\mathbf{X}_i) - f_{F_n}\|$, $r' = \|\Phi(\mathbf{x}') - f_{F_n}\|$, $\gamma = \sum_{i=1}^{n}\varphi(r_i)$ and

$$d_i = \left\langle\dot{f}_{F_n}, \Phi(\mathbf{X}_i) - f_{F_n}\right\rangle_{\mathcal{H}}\cdot\frac{q(r_i)}{r_i^3}.$$

Then, (28) simplifies to

$$\gamma\cdot\dot{f}_{F_n} + \sum_{i=1}^{n} d_i\cdot\left(\Phi(\mathbf{X}_i) - f_{F_n}\right) = n\cdot(\Phi(\mathbf{x}') - f_{F_n})\cdot\varphi(r')$$

Since $f_{F_n} = \sum_{i=1}^n w_i \Phi(\mathbf{X}_i)$, we can see that $\dot{f}_{F_n}$ has a form of $\sum_{i=1}^n \alpha_i \Phi(\mathbf{X}_i) + \alpha' \Phi(\mathbf{x}')$. By substituting this, we have

$$\begin{aligned} & \gamma \sum_{j=1}^n \alpha_j \Phi(\mathbf{X}_j) + \gamma \cdot \alpha' \Phi(\mathbf{x}') + \sum_{i=1}^n d_i \left(\Phi(\mathbf{X}_i) - \sum_{k=1}^n w_k \Phi(\mathbf{X}_k) \right) \\ = \quad & n \cdot \left(\Phi(\mathbf{x}') - \sum_{k=1}^n w_k \Phi(\mathbf{X}_k) \right) \cdot \varphi(r'). \end{aligned}$$

Since K' is positive definite, $\Phi(\mathbf{X}_i)$'s and $\Phi(\mathbf{x}')$ are linearly independent (see Lemma 10). Therefore, by comparing the coefficients of the $\Phi(\mathbf{X}_j)$'s and $\Phi(\mathbf{x}')$ in both sides, we have

$$\gamma \cdot \alpha_j + d_j - w_j \cdot \left(\sum_{i=1}^n d_i \right) = -w_j \frac{\psi(r')}{r'} \cdot n \tag{29}$$

$$\gamma \alpha' = n \cdot \varphi(r'). \tag{30}$$

From (30), $\alpha' = n\varphi(r')/\gamma$. Let $q_i = q(r_i)/r_i^3$ and $\Phi(\mathbf{X}_i) - f_{F_n} = \sum_{k=1}^n w_{k,i} \Phi(\mathbf{X}_k)$ where

$$w_{k,i} = \begin{cases} -w_k & , \quad k \neq i \\ 1 - w_k & , \quad k = i. \end{cases}$$

Then,

$$\begin{aligned} d_i &= \frac{q(r_i)}{r_i^3} \left\langle \dot{f}_{F_n}, \Phi(\mathbf{X}_i) - f_{F_n} \right\rangle_{\mathcal{H}} \\ &= q_i \left\langle \sum_{j=1}^n \alpha_j \Phi(\mathbf{X}_j) + \alpha' \Phi(\mathbf{x}'), \sum_{k=1}^n w_{k,i} \Phi(\mathbf{X}_k) \right\rangle_{\mathcal{H}} \\ &= q_i \left(\sum_{j=1}^n \sum_{k=1}^n \alpha_j w_{k,i} k_\sigma(\mathbf{X}_j, \mathbf{X}_k) + \alpha' \sum_{k=1}^n w_{k,i} k_\sigma(\mathbf{x}', \mathbf{X}_k) \right) \\ &= q_i (\mathbf{e}_i - \mathbf{w})^T K \alpha + q_i \alpha' \cdot (\mathbf{e}_i - \mathbf{w})^T \mathbf{k}' \\ &= q_i (\mathbf{e}_i - \mathbf{w})^T \left(K\alpha + \alpha' \mathbf{k}' \right) \end{aligned}$$

where $K := (k_\sigma(\mathbf{X}_i, \mathbf{X}_j))_{i,j=1}^n$ is a kernel matrix, $\mathbf{e}_i$ denotes the ith standard basis vector, and $\mathbf{k}' = [k_\sigma(\mathbf{x}', \mathbf{X}_1, \ldots, k_\sigma(\mathbf{x}', \mathbf{X}_n)]^T$. By letting $Q = diag([q_1, \ldots, q_n])$,

$$\mathbf{d} = Q \cdot (I_n - \mathbf{1}\mathbf{w}^T)(K\alpha + \alpha' \cdot \mathbf{k}').$$

Thus, (29) can be expressed in matrix-vector form,

$$\begin{aligned} \gamma\alpha + Q \cdot (I_n - \mathbf{1} \cdot \mathbf{w}^T)(K\alpha + \alpha' \cdot \mathbf{k}') - \mathbf{w} \cdot \left(\mathbf{1}^T Q \cdot (I_n - \mathbf{1} \cdot \mathbf{w}^T)(K\alpha + \alpha' \cdot \mathbf{k}') \right) \\ = -n \cdot \mathbf{w} \varphi(r'). \end{aligned}$$

Thus, α can be found solving the following linear system of equations,

$$\begin{aligned} & \left\{ \gamma I_n + (I_n - \mathbf{1} \cdot \mathbf{w}^T)^T Q \cdot (I_n - \mathbf{1} \cdot \mathbf{w}^T) \cdot K \right\} \alpha \\ = \quad & -n \cdot \varphi(r') \mathbf{w} - \alpha' (I_n - \mathbf{1} \cdot \mathbf{w}^T)^T Q \cdot (I_n - \mathbf{1} \cdot \mathbf{w}^T) \mathbf{k}'. \end{aligned}$$

Therefore,

$$\begin{aligned} IF(\mathbf{x},\mathbf{x}';\widehat{f}_{RKDE},F_n) &= \left\langle \Phi(\mathbf{x}), \dot{f}_{F_n} \right\rangle_{\mathcal{H}} \\ &= \left\langle \Phi(\mathbf{x}), \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{X}_i) + \alpha' \Phi(\mathbf{x}') \right\rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{n} \alpha_i k_\sigma(\mathbf{x},\mathbf{X}_i) + \alpha' k_\sigma(\mathbf{x},\mathbf{x}'). \end{aligned}$$

The condition $\lim_{s\to 0} f_{F_{n,s}} = f_{F_n}$ is implied by the strict convexity of J. Given $\mathbf{X}_1,\ldots,\mathbf{X}_n$ and $\mathbf{x}'$, define $\mathcal{D}_{n+1}$ as in Lemma 12. From Theorem 2, $f_{F_n,s}$ and f_{F_n} are in $\mathcal{D}_{n+1}$. With the definition in (23),

$$\begin{aligned} J_{F_{n,s}}(g) &= \int \rho(\|\Phi(\mathbf{x}) - g\|_{\mathcal{H}})\, dF_{n,s}(\mathbf{x}) \\ &= \frac{(1-s)}{n} \sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}) + s \cdot \rho(\|\Phi(\mathbf{x}') - g\|_{\mathcal{H}}). \end{aligned}$$

Note that $J_{F_{n,s}}$ uniformly converges to J on $\mathcal{D}_{n+1}$, that is, $\sup_{g\in\mathcal{D}_{n+1}} |J_{F_{n,s}}(g) - J(g)| \to 0$ as $s \to 0$, since for any $g \in \mathcal{D}_{n+1}$

$$\begin{aligned} &\left|J_{F_{n,s}}(g) - J(g)\right| \\ &= \left| \frac{(1-s)}{n} \sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}) + s \cdot \rho(\|\Phi(\mathbf{x}') - g\|_{\mathcal{H}}) - \frac{1}{n} \sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}) \right| \\ &= \frac{s}{n} \sum_{i=1}^{n} \rho(\|\Phi(\mathbf{X}_i) - g\|_{\mathcal{H}}) + s \cdot \rho(\|\Phi(\mathbf{x}') - g\|_{\mathcal{H}}) \\ &\leq \frac{s}{n} \sum_{i=1}^{n} \rho(2\tau) + s \cdot \rho(2\tau) \\ &= 2s \cdot \rho(2\tau) \end{aligned}$$

where in the inequality we use the fact that ρ is nondecreasing and

$$\begin{aligned} \|\Phi(\mathbf{x}) - g\|_{\mathcal{H}} &\leq \|\Phi(\mathbf{x})\| + \|g\|_{\mathcal{H}} \\ &\leq 2\tau. \end{aligned}$$

since $g \in \mathcal{D}_{n+1}$, and by the triangle inequality.

Now, let $\varepsilon > 0$ and $B_\varepsilon(f_{F_n}) \subset \mathcal{H}$ be the open ball centered at f_{F_n} with radius ε. Since $\mathcal{D}_{n+1}^{\varepsilon} \triangleq \mathcal{D}_{n+1} \setminus B_\varepsilon(f_{F_n})$ is also compact, $\inf_{g\in\mathcal{D}_{n+1}^{\varepsilon}} J(g)$ is attained by some $g^* \in \mathcal{D}_{n+1}^{\varepsilon}$ by the extreme value theorem (Adams and Franzosa, 2008). Since f_{F_n} is unique, $M_\varepsilon = J(g^*) - J(f_{F_n}) > 0$. For sufficiently small s, $\sup_{g\in\mathcal{D}_{n+1}} |J_{F_{n,s}}(g) - J(g)| < M_\varepsilon/2$ and thus

$$J(g) - \frac{M_\varepsilon}{2} < J_{F_{n,s}}(g) < J(g) + \frac{M_\varepsilon}{2}, \quad \forall g \in \mathcal{D}_{n+1}.$$

Therefore,

$$\begin{aligned}\inf_{g\in\mathcal{D}^{\varepsilon}_{n+1}} J_{F_{n,s}}(g) &> \inf_{g\in\mathcal{D}^{\varepsilon}_{n+1}} J(g) - \frac{M_\varepsilon}{2}\\ &= J(g^*) - \frac{M_\varepsilon}{2}\\ &= J(f_{F_n}) + M_\varepsilon - \frac{M_\varepsilon}{2}\\ &= J(f_{F_n}) + \frac{M_\varepsilon}{2}\\ &> J_{F_{n,s}}(f_{F_n}).\end{aligned}$$

Since the minimum of $J_{F_{n,s}}$ is not attained on $\mathcal{D}^{\varepsilon}_{n+1}$, $f_{F_{n,s}} \in B_\varepsilon(f_{F_n})$. Since ε is arbitrary, $\lim_{s\to 0} f_{F_{n,s}} = f_{F_n}$.

Acknowledgments

This work was supported in part by NSF Award No. 0830490 and 1047871. The authors thank an anonymous reviewer for pointing out the generalization to non-reproducing kernels.

References

I. S. Abramson. On bandwidth variation in kernel estimates-a square root law. *The Annals of Statistics*, 10(4):1217–1223, 1982.

C. Adams and R. Franzosa. *Introduction to Topology Pure and Applied*. Pearson Prentice Hall, New Jersey, 2008.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces In Probability And Statistics*. Kluwer Academic Publishers, Norwell, 2004.

D. Berry, K. Chaloner, and J. Geweke. *Bayesian Analysis in Statistics and Econometrics: Essays in Honor of Arnold Zellner*. Wiley, New York, 1996.

K. D. Brabanter, K. Pelckmans, J. D. Brabanter, M. Debruyne, J.A.K. Suykens, M. Hubert, and B. D. Moor. Robustness of kernel based regression: A comparison of iterative weighting schemes. *Proceedings of the 19th International Conference on Artificial Neural Networks (ICANN)*, pages 100–110, 2009.

L. Breiman, W. Meisel, and E. Purcell. Variable kernel estimates of multivariate densities. *Technometrics*, 19(2):135–144, 1977.

Y. Chen, X. Dang, H. Peng, and H. Bart. Outlier detection with the kernelized spatial depth function. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):288–305, 2009.

P. Chhabra, C. Scott, E. D. Kolaczyk, and M. Crovella. Distributed spatial anomaly detection. *Proc. IEEE Conference on Computer Communications (INFOCOM)*, pages 1705–1713, 2008.

A. Christmann and I. Steinwart. Consistency and robustness of kernel based regression in convex risk minimization. *Bernoulli*, 13(3):799–819, 2007.

D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data-driven scale selection. *IEEE International Conference on Computer Vision*, 1:438–445, 2001.

M. Debruyne, A. Christmann, M. Hubert, and J.A.K. Suykens. Robustness and stability of reweighted kernel based regression. *Technical Report 06-09, Department of Mathematics, K.U.Leuven, Leuven, Belgium*, 2008a.

M. Debruyne, M. Hubert, and J.A.K. Suykens. Model selection in kernel based regression using the influence function. *Journal of Machine Learning Research*, 9:2377–2400, 2008b.

M. Debruyne, M. Hubert, and J. V. Horebeek. Detecting influential observations in kernel PCA. *Computational Statistics & Data Analysis*, 54:3007–3019, 2010.

M. Girolami and C. He. Probability density estimation from optimally condensed data samples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1253–1264, OCT 2003.

F. R. Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69:383–393, 1974.

P. Huber. Robust estimation of a location parameter. *Ann. Math. Statist*, 35:45, 1964.

P. Huber. *Robust Statistics*. Wiley, New York, 1981.

M. W. Jacobson and J. A. Fessler. An expanded theoretical treatment of iteration-dependent majorize-minimize algorithms. *IEEE Transactions on Image Processing*, 16(10):2411–2422, October 2007.

D. Kim. *Least Squares Mixture Decomposition Estimation*. Doctoral dissertation, Dept. of Statistics, Virginia Polytechnic Inst. and State Univ., 1995.

J. Kim and C. Scott. Robust kernel density estimation. *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3381–3384, 2008.

J. Kim and C. Scott. L_2 kernel classification. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(10):1822–1831, 2010.

J. Kim and C. Scott. On the robustness of kernel density M-estimators. *to be published, Proceedings of the Twenty-Eighth International Conference on Machine Learning (ICML)*, 2011.

S. Lang. *Real and Functional Analysis*. Spinger, New York, 1993.

K. Lange, D. R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. *J. Computational and Graphical Stat.*, 9(1):1–20, March 2000.

L. J. Latecki, A. Lazarevic, and D. Pokrajac. Outlier detection with kernel density functions. In *Proceedings of the 5th Int. Conf. on Machine Learning and Data Mining in Pattern Recognition*, pages 61–75, Berlin, Heidelberg, 2007. Springer-Verlag.

D. G. Luenberger. *Optimization by Vector Space Methods*. Wiley-Interscience, New York, 1997.

R. S. G. Mahapatruni and A. Gray. CAKE: Convex adaptive kernel density estimation. In D. Dunson G. Gordon and M. Dud, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS) 2011*, volume 15, pages 498–506. JMLR: W&CP, 2011.

J. R. Munkres. *Topology*. Prentice Hall, 2000.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. *Proc. Annu. Conf. Comput. Learning Theory*, pages 416–426, 2001.

D. W. Scott. *Multivariate Density Estimation*. Wiley, New York, 1992.

C. Scovel, D. Hush, I. Steinwart, and J. Theiler. Radial kernels and their reproducing kernel Hilbert spaces. *Journal of Complexity*, 26:641–660, 2010.

J. Shawe-Taylor and A. N. Dolia. A framework for probability density estimation. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics,*, pages 468–475., 2007.

B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CR, New York, 1986.

L. Song, X. Zhang, A. J. Smola, A. Gretton, and B. Schölkopf. Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th Int. Conf. on Machine Learning*, ICML '08, pages 992–999, New York, NY, USA, 2008. ACM.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, New York, 2008.

G. R. Terrell and D. W. Scott. Variable kernel density estimation. *The Annals of Statistics*, 20(3): 1236–1265, 1992.

V. N. Vapnik and S. Mukherjee. Support vector method for multivariate density estimation. In *Advances in Neural Information Processing Systems*, pages 659–665. MIT Press, 2000.

A. Wibowo. Robust kernel ridge regression based on M-estimation. *Computational Mathematics and Modeling*, 20(4), 2009.

F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

L. Xu, K. Crammer, and D. Schuurmans. Robust support vector machine training via convex outlier ablation. *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, 2006.

J. Zhu, S. Hoi, and M. R.-T. Lyu. Robust regularized kernel regression. *IEEE Transaction on Systems, Man, and Cybernetics. Part B: Cybernetics,*, 38(6):1639–1644, December 2008.

Journal of Machine Learning Research 13 (2012) 2567-2588 Submitted 4/12; Revised 7/12; Published 9/12

Nonparametric Guidance of Autoencoder Representations using Label Information

Jasper Snoek JASPER@CS.TORONTO.EDU
Department of Computer Science
University of Toronto
Toronto, ON, Canada

Ryan P. Adams RPA@SEAS.HARVARD.EDU
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA, USA

Hugo Larochelle HUGO.LAROCHELLE@USHERBROOKE.EDU
Department of Computer Science
University of Sherbrooke
Sherbrooke, QC, Canada

Editor: Neil Lawrence

Abstract

While unsupervised learning has long been useful for density modeling, exploratory data analysis and visualization, it has become increasingly important for discovering features that will later be used for discriminative tasks. Discriminative algorithms often work best with highly-informative features; remarkably, such features can often be learned without the labels. One particularly effective way to perform such unsupervised learning has been to use autoencoder neural networks, which find latent representations that are constrained but nevertheless informative for reconstruction. However, *pure* unsupervised learning with autoencoders can find representations that may or may not be useful for the ultimate discriminative task. It is a continuing challenge to guide the training of an autoencoder so that it finds features which will be useful for predicting labels. Similarly, we often have *a priori* information regarding what statistical variation will be irrelevant to the ultimate discriminative task, and we would like to be able to use this for guidance as well. Although a typical strategy would be to include a parametric discriminative model as part of the autoencoder training, here we propose a nonparametric approach that uses a Gaussian process to guide the representation. By using a nonparametric model, we can ensure that a useful discriminative function exists for a given set of features, without explicitly instantiating it. We demonstrate the superiority of this guidance mechanism on four data sets, including a real-world application to rehabilitation research. We also show how our proposed approach can learn to explicitly ignore statistically significant covariate information that is label-irrelevant, by evaluating on the small NORB image recognition problem in which pose and lighting labels are available.

Keywords: autoencoder, gaussian process, gaussian process latent variable model, representation learning, unsupervised learning

1. Introduction

One of the central tasks of machine learning is the inference of latent representations. Most often these can be interpreted as representing aggregate features that explain various properties of the data.

In probabilistic models, such latent representations typically take the form of unobserved random variables. Often this latent representation is of direct interest and may reflect, for example, cluster identities. It may also be useful as a way to explain statistical variation as part of a density model. In this work, we are interested in the discovery of latent features which can be later used as alternate representations of data for discriminative tasks. That is, we wish to find ways to extract statistical structure that will make it as easy as possible for a classifier or regressor to produce accurate labels.

We are particularly interested in methods for learning latent representations that result in fast feature extraction for out-of-sample data. We can think of these as devices that have been trained to perform rapid approximate inference of hidden values associated with data. Neural networks have proven to be an effective way to perform such processing, and autoencoder neural networks, specifically, have been used to find representatons for a variety of downstream machine learning tasks, for example, image classification (Vincent et al., 2008), speech recognition (Deng et al., 2010), and Bayesian nonparametric modeling (Adams et al., 2010).

The critical insight of the autoencoder neural network is the idea of using a constrained (typically either sparse or low-dimensional) representation within a feedforward neural network. The training objective induces the network to learn to reconstruct its input at its output. The constrained central representation at the bottleneck forces the network to find a compact way to explain the statistical variation in the data. While this often leads to representations that are useful for discriminative tasks, it does require that the salient variations in the data distribution be relevant for the eventual labeling. This assumption does not necessarily always hold; often irrelevant factors can dominate the input distribution and make it poorly-suited for discrimination (Larochelle et al., 2007). In previous work to address this issue, Bengio et al. (2007) introduced weak supervision into the autoencoder training objective by adding label-specific output units in addition to the reconstruction. This approach was also followed by Ranzato and Szummer (2008) for learning document representations.

The difficulty of this approach is that it complicates the task of learning the autoencoder representation. The objective now is to learn not only a hidden representation that is good for reconstruction, but also one that is immediately good for discrimination under the simplified choice of model, for example, logistic regression. This is undesirable because it potentially prevents us from discovering informative representations for the more sophisticated nonlinear classifiers that we might wish to use later. We are forced to solve two problems at once, and the result of one of them (the classifier) will be immediately thrown away.

Here we propose a different take on the issue of introducing supervised guidance into autoencoder representations. We consider Gaussian process priors on the discriminative function that maps the latent codes into labels. The result of this choice is a Gaussian process latent variable model (GPLVM) (Lawrence, 2005) for the labels. This not only allows us to flexibly represent a wide class of classifiers, but also prevents us from having to commit to a particular function at training time. We are then able to combine the efficient parametric feed-forward aspects of the autoencoder with a flexible Bayesian nonparametric model for the labels. This also leads to an interesting interpretation of the back-constrained GPLVM itself as a limiting case of an autoencoder in which the decoder has been marginalized out. In Section 4, we empirically examine our proposed approach on four data sets, including a real-world rehabilitation problem. We also examine a data set that highlights the value of our approach, in which we cannot only use guidance from desired labels, but also introduce guidance away from irrelevant representations.

2. Unsupervised Learning of Latent Representations

The nonparametrically-guided autoencoder presented in this paper is motivated largely by the relationship between two different approaches to latent variable modeling. In this section, we review these two approaches, the GPLVM and autoencoder neural network, and examine precisely how they are related.

2.1 Autoencoder Neural Networks

The autoencoder (Cottrell et al., 1987) is a neural network architecture that is designed to create a latent representation that is informative of the input data. Through training the model to reproduce the input data at its output, a latent embedding must arise within the hidden layer of the model. Its computations can intuitively be separated into two parts:

- An encoder, which maps the input into a latent (often lower-dimensional) representation.
- A decoder, which reconstructs the input through a map from the latent representation.

We will denote the latent space by $\mathcal{X}$ and the visible (data) space by $\mathcal{Y}$ and assume they are real valued with dimensionality J and K respectively, that is, $\mathcal{X}=\mathbb{R}^J$ and $\mathcal{Y}=\mathbb{R}^K$. The encoder, then, is defined as a function $g(y;\phi):\mathcal{Y}\to\mathcal{X}$ and the decoder as $f(x;\psi):\mathcal{X}\to\mathcal{Y}$. Given N data examples $\mathcal{D}=\{y^{(n)}\}_{n=1}^N, y^{(n)}\in\mathcal{Y}$, we jointly optimize the parameters of the encoder ϕ and decoder ψ over the least-squares reconstruction cost:

$$\phi^\star,\psi^\star=\arg\min_{\phi,\psi}\sum_{n=1}^{N}\sum_{k=1}^{K}(y_k^{(n)}-f_k(g(y^{(n)};\phi);\psi))^2, \tag{1}$$

where $f_k(\cdot)$ is the kth output dimension of $f(\cdot)$. It is easy to demonstrate that this model is equivalent to principal components analysis when f and g are linear projections. However, nonlinear basis functions allow for a more powerful nonlinear mapping. In our empirical analysis we use sigmoidal

$$g(y;\phi)=(1+\exp(-y_+^T\phi))^{-1}$$

and noisy rectified linear

$$g(y;\phi)=\max\{0,y_+^T\phi+\varepsilon\},\qquad\qquad \varepsilon\sim\mathcal{N}(0,1)$$

basis functions for the encoder where y_+ denotes y with a 1 appended to account for a bias term. The noisy rectified linear units or NReLU (Nair and Hinton, 2010)) exhibit the property that they are more *equivariant* to the scaling of the inputs (the non-noisy version being perfectly equivariant when the bias term is fixed to 0). This is a useful property for image data, for example, as (in contrast to sigmoidal basis functions) global lighting changes will cause uniform changes in the activations across hidden units.

Recently, autoencoders have regained popularity as they have been shown to be an effective module for "greedy pre-training" of deep neural networks (Bengio et al., 2007). *Denoising* autoencoders (Vincent et al., 2008) are of particular interest, as they are robust to the trivial "identity" solutions that can arise when trying to learn overcomplete representations. Overcomplete representations, which are of higher dimensionality than the input, are considered to be ideal for discriminative tasks. However, these are difficult to learn because a trivial minimum of the autoencoder

reconstruction objective is reached when the autoencoder learns the identity transformation. The denoising autoencoder forces the model to learn more interesting structure from the data by providing as input a corrupted training example, while evaluating reconstruction on the noiseless original. The objective of Equation (1) then becomes

$$\phi^\star, \psi^\star = \arg\min_{\phi,\psi} \sum_{n=1}^{N} \sum_{k=1}^{K} (y_k^{(n)} - f_k(g(\widetilde{y}^{(n)};\phi);\psi))^2,$$

where $\widetilde{y}^{(n)}$ is the corrupted version of $y^{(n)}$. Thus, in order to infer missing components of the input or fix the corruptions, the model must extract a richer latent representation.

2.2 Gaussian Process Latent Variable Models

While the denoising autoencoder learns a latent representation that is distributed over the hidden units of the model, an alternative strategy is to consider that the data intrinsically lie on a lower-dimensional latent *manifold* that reflects their statistical structure. Such a manifold is difficult to define *a priori*, however, and thus the problem is often framed as learning the latent embedding under an assumed smooth functional mapping between the visible and latent spaces. Unfortunately, a major challenge arising from this strategy is the simultaneous optimization of the latent embedding and the functional parameterization. The Gaussian process latent variable model (Lawrence, 2005) addresses this challenge under a Bayesian probabilistic formulation. Using a Gaussian process prior, the GPLVM marginalizes over the infinite possible mappings from the latent to visible spaces and optimizes the latent embedding over a distribution of mappings. The GPLVM results in a powerful nonparametric model that analytically integrates over the infinite number of functional parameterizations from the latent to the visible space.

Similar to the autoencoder, linear kernels in the GPLVM recover principal components analysis. Under a nonlinear basis, however, the GPLVM can represent an arbitrarily complex continuous mapping, depending on the functions supported by the Gaussian process prior. Although GPLVMs were initially introduced for the visualization of high dimensional data, they have been used to obtain state-of-the-art results for a number of tasks, including modeling human motion (Wang et al., 2008), classification (Urtasun and Darrell, 2007) and collaborative filtering (Lawrence and Urtasun, 2009).

The GPLVM assumes that the N data examples $\mathcal{D} = \{y^{(n)}\}_{n=1}^N$ are the image of a homologous set $\{x^{(n)}\}_{n=1}^N$ arising from a vector-valued "decoder" function $f(x) : \mathcal{X} \to \mathcal{Y}$. Analogously to the squared-loss of the previous section, the GPLVM assumes that the observed data have been corrupted by zero-mean Gaussian noise: $y^{(n)} = f(x^{(n)}) + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_K)$. The innovation of the GPLVM is to place a Gaussian process prior on the function $f(x)$ and then optimize the latent representation $\{x^{(n)}\}_{n=1}^N$, while marginalizing out the unknown $f(x)$.

2.2.1 GAUSSIAN PROCESS PRIORS

Rather than requiring a specific finite basis, the Gaussian process provides a distribution over random functions of a particular family, the properties of which are specified via a positive definite covariance function. Typically, Gaussian processes are defined in terms of a distribution over scalar functions and in keeping with the convention for the GPLVM, we shall assume that K independent GPs are used to construct the vector-valued function $f(x)$. We denote each of these functions as $f_k(x) : \mathcal{X} \to \mathbb{R}$. The GP requires a covariance kernel function, which we denote as

$C(x,x') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The defining characteristic of the GP is that for any finite set of N data in $\mathcal{X}$ there is a corresponding N-dimensional Gaussian distribution over the function values, which in the GPLVM we take to be the components of $\mathcal{Y}$. The $N \times N$ covariance matrix of this distribution is the matrix arising from the application of the covariance kernel to the N points in $\mathcal{X}$. We denote any additional parameters governing the behavior of the covariance function by θ.

Under the component-wise independence assumptions of the GPLVM, the Gaussian process prior allows one to analytically integrate out the K latent scalar functions from $\mathcal{X}$ to $\mathcal{Y}$. Allowing for each of the K Gaussian processes to have unique hyperparameter θ_k, we write the marginal likelihood, that is, the probability of the observed data given the hyperparameters and the latent representation, as

$$p(\{y^{(n)}\}_{n=1}^N \,|\, \{x^{(n)}\}_{n=1}^N, \{\theta_k\}_{k=1}^K, \sigma^2) = \prod_{k=1}^K \mathcal{N}(y_k^{(\cdot)} \,|\, 0, \Sigma_{\theta_k} + \sigma^2 \mathbb{I}_N),$$

where $y_k^{(\cdot)}$ refers to the vector $[y_k^{(1)}, \ldots, y_k^{(N)}]$ and where Σ_{θ_k} is the matrix arising from $\{x_n\}_{n=1}^N$ and θ_k. In the basic GPLVM, the optimal x_n are found by maximizing this marginal likelihood.

2.2.2 COVARIANCE FUNCTIONS

Here we will briefly describe the covariance functions used in this work. For a more thorough treatment, we direct the reader to Rasmussen and Williams (2006, Chapter 4).

A common choice of covariance function for the GP is the automatic relevance determination (ARD) *exponentiated quadratic* (also known as *squared exponential*) kernel

$$K_{\mathsf{EQ}}(\mathrm{x}, \mathrm{x}') = \exp\left\{-\frac{1}{2} r^2(\mathrm{x}, \mathrm{x}')\right\}, \qquad r^2(\mathrm{x}, \mathrm{x}') = (\mathrm{x} - \mathrm{x}')^{\mathsf{T}} \Psi (\mathrm{x} - \mathrm{x}')$$

where the covariance between outputs of the GP depends on the distance between corresponding inputs. Here Ψ is a symmetric positive definite matrix that defines the metric (Vivarelli and Williams, 1999). Typically, Ψ is a diagonal matrix with $\Psi_{d,d} = 1/\ell_d^2$, where the *length-scale* parameters, ℓ_d, scale the contribution of each dimension of the input independently. In the case of the GPLVM, these parameters are made redundant as the inputs themselves are learned. Thus, in this work we assume these kernel hyperparameters are set to a fixed value.

The exponentiated quadratic construction is not appropriate for all functions. Consider a function that is periodic in the inputs. The covariance between outputs should then depend not on the Euclidian distance between inputs but rather on their phase. A solution is to warp the inputs to capture this property and then apply the exponentiated quadratic in this warped space. To model a periodic function, MacKay (1998) suggests applying the exponentiated quadratic covariance to the output of an embedding function $u(x)$, where for a single dimensional input x, $u(x) = [\sin(x), \cos(x)]$ expands from $\mathbb{R}$ to $\mathbb{R}^2$. The resulting *periodic* covariance becomes

$$K_{\mathsf{PER}}(x, x') = \exp\left\{-\frac{2\sin^2 \frac{x - x'}{2}}{\ell^2}\right\}.$$

The exponentiated quadratic covariance can be shown (MacKay, 1998) to be the similarity between inputs after they are projected into a feature space by an infinite number of centered radial basis functions. Williams (1998) derived a kernel that similarly, under a specific activation function,

reflects a feature projection by a neural network in the limit of infinite units. This results in the *neural network* covariance

$$K_{\mathrm{NN}}(\mathrm{x},\mathrm{x}') = \frac{2}{\pi}\sin^{-1}\left(\frac{2\tilde{\mathrm{x}}^T\Psi\tilde{\mathrm{x}}'}{\sqrt{(1+2\tilde{\mathrm{x}}^T\Psi\tilde{\mathrm{x}}')(1+2\tilde{\mathrm{x}}^T\Psi\tilde{\mathrm{x}}')}}\right), \tag{2}$$

where $\tilde{\mathrm{x}}$ is x with a 1 prepended. An important distinction from the exponentiated quadratic is that the neural net covariance is non-stationary. Unlike the exponentiated quadratic, the neural network covariance is not invariant to translation. We use the neural network covariance primarily to draw a theoretical connection between GPs and autoencoders. However, the non-stationary properties of this covariance in the context of the GPLVM, which can allow the GPLVM to capture more complex structure, warrant further investigation.

2.2.3 THE BACK-CONSTRAINED GPLVM

Although the GPLVM constrains the mapping from the latent space to the data to be smooth, it does not enforce smoothness in the inverse mapping. This can be an undesirable property, as data that are intrinsically close in observed space need not be close in the latent representation. Not only does this introduce arbitrary gaps in the latent manifold, but it also complicates the encoding of novel data points into the latent space as there is no direct mapping. The latent representations of out-of-sample data must thus be optimized, conditioned on the latent embedding of the training examples. Lawrence and Quiñonero-Candela (2006) reformulated the GPLVM to address these issues, with the constraint that the hidden representation be the result of a smooth map from the observed space. They proposed multilayer perceptrons and radial-basis-function networks as possible implementations of this smooth mapping. We will denote this "encoder" function, parameterized by ϕ, as $g(y;\phi):\mathcal{Y}\rightarrow\mathcal{X}$. The marginal likelihood objective of this *back-constrained* GPLVM can now be formulated as finding the optimal ϕ under:

$$\phi^\star = \arg\min_{\phi}\sum_{k=1}^{K}\ln|\Sigma_{\theta_k,\phi}+\sigma^2\mathbb{I}_N| + y_k^{(\cdot)^\mathsf{T}}(\Sigma_{\theta_k,\phi}+\sigma^2\mathbb{I}_N)^{-1}y_k^{(\cdot)}, \tag{3}$$

where the kth covariance matrix $\Sigma_{\theta_k,\phi}$ now depends not only on the kernel hyperparameters θ_k, but also on the parameters of $g(y;\phi)$, that is,

$$[\Sigma_{\theta_k,\phi}]_{n,n'} = C(g(y^{(n)};\phi), g(y^{(n')};\phi);\theta_k). \tag{4}$$

Lawrence and Quiñonero-Candela (2006) motivate the back-constrained GPLVM partially through the NeuroScale algorithm of Lowe and Tipping (1997). The NeuroScale algorithm is a radial basis function network that creates a one-way mapping from data to a latent space using a heuristic loss that attempts to preserve pairwise distances between data cases. Thus, the back-constrained GPLVM can be viewed as a combination of NeuroScale and the GPLVM where the pairwise distance loss is removed and rather the loss is backpropagated from the GPLVM.

2.3 GPLVM as an Infinite Autoencoder

The relationship between Gaussian processes and artificial neural networks was established by Neal (1996), who showed that the prior over functions implied by many parametric neural networks becomes a GP in the limit of an infinite number of hidden units. Williams (1998) subsequently

derived a GP covariance function corresponding to such an infinite neural network (Equation 2) with a specific activation function.

An interesting and overlooked consequence of this relationship is that it establishes a connection between autoencoders and the back-constrained Gaussian process latent variable model. A GPLVM with the covariance function of Williams (1998), although it does not impose a density over the data, is similar to a density network (MacKay, 1994) with an infinite number of hidden units in the single hidden layer. We can transform this density network into a semiparametric autoencoder by applying a neural network as the backconstraint network of the GPLVM. The encoder of the resulting model is a parametric neural network and the decoder a Gaussian process.

We can alternatively derive this model starting from an autoencoder. With a least-squares reconstruction cost and a linear decoder, one can integrate out the weights of the decoder assuming a zero-mean Gaussian prior over the weights. This results in a Gaussian process for the decoder and learning thus corresponds to the minimization of Equation (3) with a linear kernel for Equation (4). Incorporating any non-degenerate positive definite kernel, which corresponds to a decoder of infinite size, also recovers the general back-constrained GPLVM algorithm.

This infinite autoencoder exhibits some attractive properties. After training, the decoder network of an autoencoder is generally superfluous. Learning a parametric form for this decoder is thus a nuisance that complicates the objective. The infinite decoder network, as realized by the GP, obviates the need to learn a parameterization and instead marginalizes over all possible decoders. The parametric encoder offers a rapid encoding and persists as the training data changes, permitting, for example, stochastic gradient descent. A disadvantage, however, is that the decoder naturally inherits the computational costs of the GP by memorizing the data. Thus, for very high dimensional data, a standard autoencoder may be more desirable.

3. Supervised Guidance of Latent Representations

Unsupervised learning has proven to be effective for learning latent representations that excel in discriminative tasks. However, when the salient statistics of the data are only weakly informative about a desired discriminative task, it can be useful to incorporate label information into unsupervised learning. Bengio et al. (2007) demonstrated, for example, that while a purely supervised signal can lead to overfitting, mild supervised guidance can be beneficial when initializing a discriminative deep neural network. Therefore, Bengio et al. (2007) proposed a hybrid approach under which the unsupervised model's latent representation also be trained to predict the label information, by adding a parametric mapping $c(x;\Lambda) : \mathcal{X} \to \mathcal{Z}$ from the latent space $\mathcal{X}$ to the labels $\mathcal{Z}$ and backpropagating error gradients from the output. Bengio et al. (2007) used a linear logistic regression classifier for this parametric mapping. This "partial supervision" thus encourages the model to encode statistics within the latent representation that are useful for a specific (but learned) parameterization of such a linear mapping. Ranzato and Szummer (2008) adopted a similar strategy to learn compact representations of documents.

There are disadvantages to this approach. The assumption of a specific parametric form for the mapping $c(x;\Lambda)$ restricts the supervised guidance to classifiers within that family of mappings. Also, the learned representation is committed to one particular setting of the parameters Λ. Consider the learning dynamics of gradient descent optimization for this strategy. At every iteration t of descent (with current state $\phi_t, \psi_t, \Lambda_t$), the gradient from supervised guidance encourages the latent representation (currently parametrized by ϕ_t, ψ_t) to become more predictive of the labels under the

current label map $c(x;\Lambda_t)$. Such behavior discourages moves in ϕ,ψ space that make the latent representation more predictive under some other label map $c(x;\Lambda^\star)$ where $\Lambda^\star$ is potentially distant from Λ_t. Hence, while the problem would seem to be alleviated by the fact that Λ is learned jointly, this constant pressure towards representations that are immediately useful increases the difficulty of learning the unsupervised component.

3.1 Nonparametrically Guided Autoencoder

Instead of specifying a particular discriminative regressor for the supervised guidance and jointly optimizing for its parameters and those of an autoencoder, it seems more desirable to enforce only that a mapping to the labels exists while optimizing for the latent representation. That is, rather than learning a latent representation that is tied to a specific parameterized mapping to the labels, we would instead prefer to find a latent representation that is consistent with an entire class of mappings. One way to arrive at such a guidance mechanism is to marginalize out the parameters Λ of a label map $c(x;\Lambda)$ under a distribution that permits a wide family of functions. We have seen previously that this can be done for reconstructions of the input space with a decoder $f(x;\psi)$. We follow the same reasoning and do this instead for $c(x;\Lambda)$. Integrating out the parameters of the label map yields a back-constrained GPLVM acting on the label space $\mathcal{Z}$, where the back constraints are determined by the input space $\mathcal{Y}$. The positive definite kernel specifying the Gaussian process then determines the properties of the distribution over mappings from the latent representation to the labels. The result is a hybrid of the autoencoder and back-constrained GPLVM, where the encoder is shared across models. For notation, we will refer to this approach to guided latent representation as a *nonparametrically guided autoencoder*, or NPGA.

Let the label space $\mathcal{Z}$ be an M-dimensional real space,[1] that is, $\mathcal{Z}=\mathbb{R}^M$, and the nth training example has a label vector $z^{(n)}\in\mathcal{Z}$. The covariance function that relates label vectors in the NPGA is

$$[\Sigma_{\theta_m,\phi,\Gamma}]_{n,n'} = C(\Gamma\cdot g(y^{(n)};\phi),\Gamma\cdot g(y^{(n')};\phi)\,;\theta_m),$$

where $\Gamma\in\mathbb{R}^{H\times J}$ is an H-dimensional linear projection of the encoder output. For $H\ll J$, this projection improves efficiency and reduces overfitting. Learning in the NPGA is then formulated as finding the optimal ϕ,ψ,Γ under the combined objective:

$$\phi^\star,\psi^\star,\Gamma^\star = \arg\min_{\phi,\psi,\Gamma}(1-\alpha)L_{\text{auto}}(\phi,\psi)+\alpha L_{\text{GP}}(\phi,\Gamma)$$

where $\alpha\in[0,1]$ linearly blends the two objectives

$$L_{\text{auto}}(\phi,\psi) = \frac{1}{K}\sum_{n=1}^{N}\sum_{k=1}^{K}(y_k^{(n)}-f_k(g(y^{(n)};\phi);\psi))^2,$$

$$L_{\text{GP}}(\phi,\Gamma) = \frac{1}{M}\sum_{m=1}^{M}\left[\ln|\Sigma_{\theta_m,\phi,\Gamma}+\sigma^2\mathbb{I}_N| + z_m^{(\cdot)\mathsf{T}}(\Sigma_{\theta_m,\phi,\Gamma}+\sigma^2\mathbb{I}_N)^{-1}z_m^{(\cdot)}\right].$$

We use a linear decoder for $f(x;\psi)$, and the encoder $g(y;\phi)$ is a linear transformation followed by a fixed element-wise nonlinearity. As is common for autoencoders and to reduce the number of free parameters in the model, the encoder and decoder weights are tied. As proposed in the

1. For discrete labels, we use a "one-hot" encoding.

denoising autoencoder variant of Vincent et al. (2008), we always add noise to the encoder inputs in cost $L_{\text{auto}}(\phi, \psi)$, keeping the noise fixed during each iteration of learning. That is, we update the denoising autoencoder noise every three iterations of conjugate gradient descent optimization. For the larger data sets, we divide the training data into mini-batches of 350 training cases and perform three iterations of conjugate gradient descent per mini-batch. The optimization proceeds sequentially over the batches such that model parameters are updated after each mini-batch.

3.2 Related Models

An example of the hybridization of an unsupervised connectionist model and Gaussian processes has been explored in previous work. Salakhutdinov and Hinton (2008) used restricted Boltzmann machines (RBMs) to initialize a multilayer neural network mapping into the covariance kernel of a Gaussian process regressor or classifier. They then adjusted the mapping through backpropagating gradients from the Gaussian process through the neural network. In contrast to the NPGA, this model did not use a Gaussian process in the initial learning of the latent representation and relies on a Gaussian process for inference at test time. Unfortunately, this poses significant practical issues for large data sets such as NORB or CIFAR-10, as the computational complexity of GP inference is cubic in the number of data examples. Note also that when the salient variations of the data are not relevant to a given discriminative task, the initial RBM training will not encourage the encoding of the discriminative information in the latent representation. The NPGA circumvents these issues by applying a GP to small mini-batches during the learning of the latent representation and uses the GP to learn a representation that is better even for a linear discriminative model.

Previous work has merged parametric unsupervised learning and nonparametric supervised learning. Salakhutdinov and Hinton (2007) combined autoencoder training with neighborhood component analysis (Goldberger et al., 2004), which encouraged the model to encode similar latent representations for inputs belonging to the same class. Hadsell et al. (2006) employ a similar objective in a fully supervised setting to preserve distances in label space in a latent representation. They used this method to visualize the different latent embeddings that can arise from using additional labels on the NORB data set. Note that within the NPGA, the backconstrained-GPLVM performs an analogous role. In Equation 3, the first term, the log determinant of the kernel, regularizes the latent space. Since the determinant is minimized when the covariance between all pairs is maximized, it pulls all examples together in the latent space. The second term, however, pushes examples that are distant in label space apart in the latent space. For example, when a one-hot coding is used, the labels act as indicator variables reflecting same-class pairs in the concentration matrix. This pushes apart examples that are of different class and pulls together examples of the same class. Thus, the GPLVM enforces that examples close in label space will be closer in the latent representation than examples that are distant in label space.

There are several important differences, however, between the aforementioned approaches and the NPGA. First, the NPGA can be intuitively interpreted as using a marginalization over mappings to labels. Second, the NPGA naturally accommodates continuous labels and enables the use of any covariance function within the wide library from the Gaussian process literature. Incorporating periodic labels, for example, is straightforward through using a periodic covariance. Encoding such periodic signals in a parametric neural network and blending this with unsupervised learning can be challenging (Zemel et al., 1995). Similarly to a subset of the aforementioned work, the NPGA exhibits the property that it not only enables the learning of latent representations that encode

information that is relevant for discrimination but as we show in Section 4.3, it can *ignore* salient information that is not relevant to the discriminative representation.

Although it was originally developed as a model for unsupervised dimensionality reduction, a number of approaches have explored the addition of auxiliary signals within the GPLVM. The Discriminative GPLVM (Urtasun and Darrell, 2007), for example, added a discriminant analysis based prior that enforces inter-class separability in the latent space. The DGPLVM is, however, restricted to discrete labels, requires that the latent dimensionality be smaller than the number of classes and uses a GP mapping to the data, which is computationally prohibitive for high dimensional data. A GPLVM formulation in which multiple GPLVMs mapping to different signals share a single latent space, the shared GPLVM (SGPLVM), was introduced by Shon et al. (2005). Wang et al. (2007) showed that using product kernels within the context of the GPLVM results in a generalisation of multilinear models and allows one to separate the encoding of various signals in the latent representation. As discussed above, the reliance on a Gaussian process mapping to the data prohibits the application of these approaches to large and high dimensional data sets. Our model overcomes these limitations through using a natural parametric form of the GPLVM, the autoencoder, to map to the data.

4. Empirical Analyses

We now present experiments with NPGA on four different classification data sets. In all experiments, the discriminative value of the learned representation is evaluated by training a linear (logistic) classifier, a standard practice for evaluating latent representations.

4.1 Oil Flow Data

We begin our emprical analysis by exploring the benefits of using the NPGA on a multi-phase oil flow classification problem (Bishop and James, 1993). The data are twelve-dimensional, real-valued gamma densitometry measurements from a simulation of multi-phase oil flow. The relatively small sample size of these data—1,000 training and 1,000 test examples—makes this problem useful for exploring different models and training procedures. We use these data primarily to explore two questions:

- To what extent does the nonparametric guidance of an unsupervised parametric autoencoder improve the learned feature representation with respect to the classification objective?
- What additional benefit is gained through using nonparametric guidance over simply incorporating a parametric mapping to the labels?

In order to address these concerns, we linearly blend our nonparametric guidance cost $L_{\text{GP}}(\phi,\Gamma)$ with the one Bengio et al. (2007) proposed, referred to as $L_{\text{LR}}(\phi,\Lambda)$:

$$L(\phi,\psi,\Lambda,\Gamma;\alpha,\beta) = (1-\alpha)L_{\text{auto}}(\phi,\psi) + \alpha((1-\beta)L_{\text{LR}}(\phi,\Lambda) + \beta L_{\text{GP}}(\phi,\Gamma)), \tag{5}$$

where $\beta \in [0,1]$ and Λ are the parameters of a multi-class logistic regression mapping to the labels.

Thus, α allows us to adjust the relative contribution of the unsupervised guidance while β weighs the relative contributions of the parametric and nonparametric supervised guidance.

To assess the benefit of the nonparametric guidance, we perform a grid search over the range of settings for α and β at intervals of 0.1. For each of these intervals, a model was trained for 100

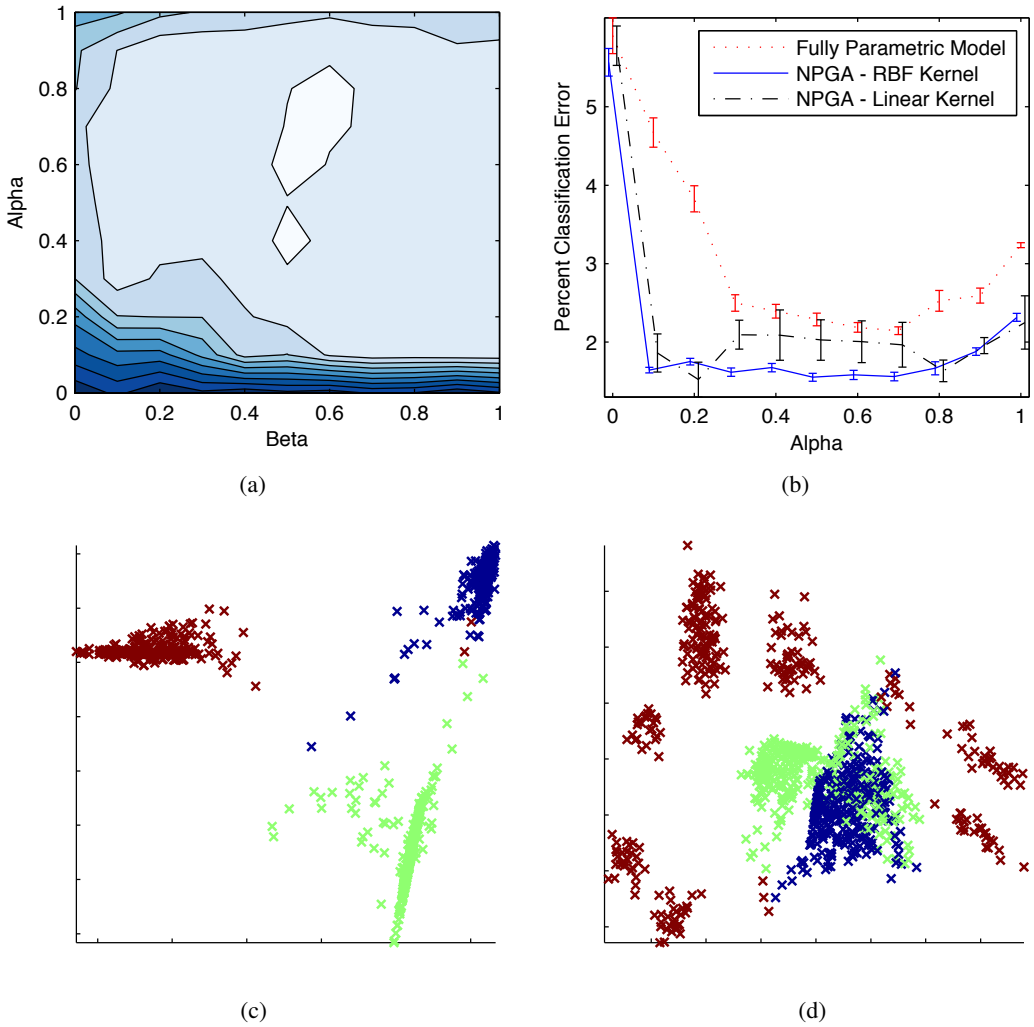


Figure 1: We explore the benefit of the NPGA on the oil data through adjusting the relative contributions of the autoencoder, logistic regressor and GP costs in the hybrid objective by modifying α and β. (a) Classification error on the test set on a linear scale from 6% (dark) to 1% (light) (b) Cross-sections of (a) at $\beta = 0$ (a fully parametric model) and $\beta = 1$ (NPGA). (c & d) Latent projections of the 1000 test cases within the two dimensional latent space of the GP, Γ, for a NPGA ($\alpha = 0.5$) and a back-constrained GPLVM.

iterations of conjugate gradient descent and classification performance was assessed by applying logistic regression on the hidden units of the encoder. 250 NRenLU units were used in the encoder, and zero-mean Gaussian noise with a standard deviation of 0.05 was added to the inputs of the denoising autoencoder cost. The GP label mapping used an RBF covariance with $H = 2$. To make the problem more challenging, a subset of 100 training samples was used. Each experiment was repeated over 20 different random initializations.

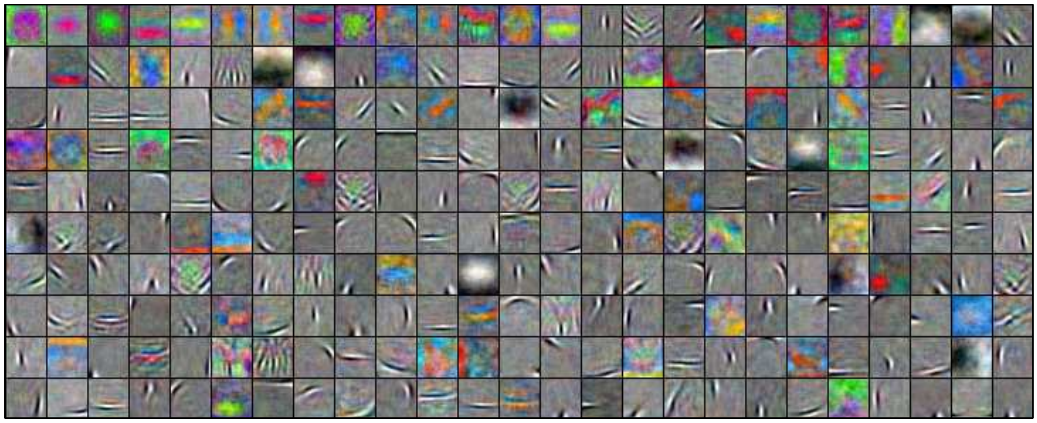


Figure 2: A sample of filters learned on the CIFAR 10 data set. This model achieved a test accuracy of 65.71%. The filters are sorted by norm.

The results of this analysis are visualized in Figure 1. Figure 1b demonstrates that, even when compared with direct optimization under the discriminative family that will be used at test time (logistic regression), performance improves by integrating out the label map. However, in Figure 1a we can see that some parametric guidance can be beneficial, presumably because it is from the same discriminative family as the final classifier. A visualisation of the latent representation learned by an NPGA and a standard back-constrained GPLVM is provided in Figures 1c and 1d. The former clearly embeds much more class-relevant structure than the latter.

We observe also that using a GP with a linear covariance function within the NPGA outperforms the parametric guidance (see Fig. 1b). While the performance of the model does depend on the choice of kernel, this helps to confirm that the benefit of our approach is achieved mainly through integrating out the label mapping, rather than having a more powerful nonlinear mapping to the labels. Another interesting result is that the results of the linear covariance NPGA are significantly noisier than the RBF mapping. Presumably, this is due to the long-range global support of the linear covariance causing noisier batch updates.

4.2 CIFAR 10 Image Data

We also apply the NPGA to a much larger data set that has been widely studied in the connectionist learning literature. The CIFAR 10 object classification data set[2] is a labeled subset of the *80 million tiny images* data (Torralba et al., 2008) with a training set of 50,000 32×32 color images and a test set of an additional 10,000 images. The data are labeled into ten classes. As GPs scale poorly on large data sets we consider it pertinent to explore the following:

> Are the benefits of nonparametric guidance still observed in a larger scale classification problem, when mini-batch training is used?

To answer this question, we evaluate the use of nonparametric guidance on three different combinations of preprocessing, architecture and convolution. For each experiment, an autoencoder is

2. CIFAR data set at `http://www.cs.utoronto.ca/~kriz/cifar.html`.

Experiment	α	Accuracy
1. Full Images	0.0	46.91%
	0.1	56.75%
	0.5	52.11%
	1.0	45.45%
2. 28x28 Patches	0.0	63.20%
	0.8	65.71%
3. Convolutional	0.0	73.52%
	0.1	75.82%
Sparse Autoencoder (Coates et al., 2011)		73.4%
Sparse RBM (Coates et al., 2011)		72.4%
K-means (Hard) (Coates et al., 2011)		68.6%
K-means (Triangle) (Coates et al., 2011)		77.9%

Table 1: Results on CIFAR 10 for various training strategies, varying the nonparametric guidance α. Recently published convolutional results are shown for comparison.

compared to a NPGA by modifying α. Experiments[3] were performed following three different strategies:

1. Full images: A one-layer autoencoder with 2400 NReLU units was trained on the raw data (which was reduced from $32\times32\times3 = 3072$ to 400 dimensions using PCA). A GP mapping to the labels operated on a $H = 25$ dimensional space.

2. 28×28 patches: An autoencoder with 1500 logistic hidden units was trained on $28\times28\times3$ patches subsampled from the full images, then reduced to 400 dimensions using PCA. All models were fine tuned using backpropagation with softmax outputs and predictions were made by taking the expectation over all patches (i.e., to classify an image, we consider all 28×28 patches obtained from that image and then average the label distributions over all patches). A $H=25$ dimensional latent space was used for the GP.

3. Convolutional: Following Coates et al. (2011), 6×6 patches were subsampled and each patch was normalized for lighting and contrast. This resulted in a $36\times3 = 108$ dimensional feature vector as input to the autoencoder. For classification, features were computed densely over all 6×6 patches. The images were divided into 4×4 blocks and features were pooled through summing the feature activations in each block. 1600 NReLU units were used in the autoencoder but the GP was applied to only 400 of them. The GP used a $H=10$ dimensional space.

3. When PCA preprocessing was used for autoencoder training, the inputs were corrupted with zero-mean Gaussian noise with standard deviation 0.05. Otherwise, raw pixels were corrupted by deleting (i.e., set to zero) 10% of the pixels. Autoencoder training then corresponds to reconstructing the original input. Each model used a neural net (MLP) covariance with fixed hyperparameters.

After training, a logistic regression classifier was applied to the features resulting from the hidden layer of each autoencoder to evaluate their quality with respect to the classification objective. The results, presented in Table 1, show that supervised guidance helps in all three strategies. The use of different architectures, methodologies and hidden unit activations demonstrates that the nonparametric guidance can be beneficial for a wide variety of formulations. Although, we do not achieve state of the art results on this data set, these results demonstrate that nonparametric guidance is beneficial for a wide variety of model architechtures. We note that the optimal amount of guidance differs for each experiment and setting α too high can often be detrimental to performance. This is to be expected, however, as the amount of discriminative information available in the data differs for each experiment. The small patches in the convolutional strategy, for example, likely encode very weak discriminative information. Figure 2 visualises the encoder weights learned by the NPGA on the CIFAR data.

4.3 Small NORB Image Data

In the following empirical analysis, the use of the NPGA is explored on the small NORB data (LeCun et al., 2004). The data are stereo image pairs of fifty toys belonging to five generic categories. Each toy was imaged under six lighting conditions, nine elevations and eighteen azimuths. The 108×108 images were subsampled to half their size to yield a $48\times48\times2$ dimensional input vector per example. The objects were divided evenly into test and training sets yielding 24,300 examples each. The objective is to classify to which object category each of the test examples belongs.

This is an interesting problem as the variations in the data due to the different imaging conditions are the salient ones and will be the strongest signal learned by the autoencoder. This is nuisance structure that will influence the latent embedding in undesirable ways. For example, neighbors in the latent space may reflect lighting conditions in observed space rather than objects of the same class. Certainly, the squared pixel difference objective of the autoencoder will be affected more by significant lighting changes than object categories. Fortunately, the variations due to the imaging conditions are known *a priori*. In addition to an object category label, there are two real-valued vectors (elevation and azimuth) and one discrete vector (lighting type) associated with each example. In our empirical analysis we examine the following:

> As the autoencoder attempts to coalesce the various sources of structure into its hidden layer, can the NPGA guide the learning in such a way as to separate the class-invariant transformations of the data from the class-relevant information?

An NPGA was constructed with Gaussian processes mapping to each of the four label types to address this question. In order to *separate* the latent embedding of the salient information related to each label, the GPs were applied to disjoint subsets of the hidden units of the autoencoder. The autoencoder's 2400 NReLU units were partitioned such that half were used to encode structure relevant for classification and the other half were evenly divided to encode the remaining three labels. Thus a GP mapping from a four dimensional latent space, $H=4$, to class labels was applied to 1200 hidden units. GPs, with $H=2$, mapping to the three auxiliary labels were applied each to 400 hidden units. As the lighting labels are discrete, we used a one-hot coding, similarly to the class labels. The elevation labels are continuous, so the GP was mapped directly to the labels. Finally, because the azimuth is a periodic signal, a periodic kernel was used for the azimuth GP. This highlights a major advantage of our approach, as the broad library of GP covariance functions facilitate a flexibility to the mapping that would be challenging with a parametric model.

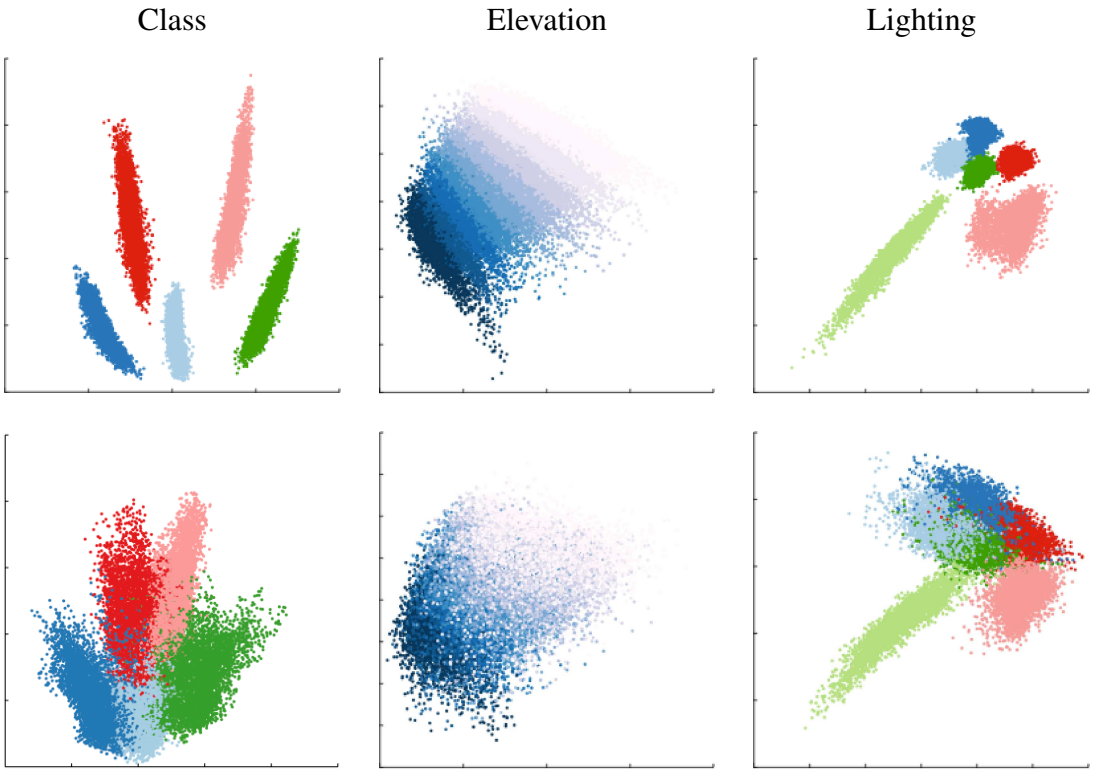


Figure 3: Visualisations of the NORB training (top) and test (bottom) data latent space representations in the NPGA, corresponding to class (first column), elevation (second column), and lighting (third column). The visualizations are in the GP latent space, Γ, of a model with $H = 2$ for each GP. Colors correspond to the respective labels.

To validate this configuration, we empirically compared it to a standard autoencoder (i.e., $\alpha = 0$), an autoencoder with parametric logistic regression guidance and an NPGA with a single GP applied to all hidden units mapping to the class labels. For comparison, we also provide results obtained by a back-constrained GPLVM and SGPLVM.[4] For all models, a validation set of 4300 training cases was withheld for parameter selection and early stopping. Neural net covariances were used for each GP except the one applied to azimuth, which used a periodic RBF kernel. GP hyperparameters were held fixed as their influence on the objective would confound the analysis of the role of α. For denoising autoencoder training, the raw pixels were corrupted by setting 20% of pixels to zero in the inputs. Each image was lighting- and contrast-normalized and the error on the test set was evaluated using logistic regression on the hidden units of each model. A visualisation of the structure learned by the GPs is shown in Figure 3. Results of the empirical comparison are presented in Table 2.

4. The GPLVM and SGPLVM were applied to a 96 dimensional PCA of the data for computional tractability, used a neural net covariance mapping to the data, and otherwise used the same back-constraints, kernel configuration, and mini-batch training as the NPGA. The SGPLVM consisted of a GPLVM with a latent space that is shared by multiple GPLVM mappings to the data and each of the labels

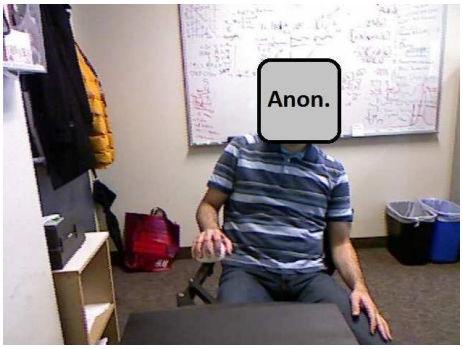
Model	Accuracy
Autoencoder + 4(Log)reg ($\alpha = 0.5$)	85.97%
GPLVM	88.44%
SGPLVM (4 GPs)	89.02%
NPGA (4 GPs Lin – $\alpha = 0.5$)	92.09%
Autoencoder	92.75%
Autoencoder + Logreg ($\alpha = 0.5$)	92.91%
NPGA (1 GP NN – $\alpha = 0.5$)	93.03%
NPGA (1 GP Lin – $\alpha = 0.5$)	93.12%
NPGA (4 GPs Mix – $\alpha = 0.5$)	94.28%
K-Nearest Neighbors (LeCun et al., 2004)	83.4%
Gaussian SVM (Salakhutdinov and Larochelle, 2010)	88.4%
3 Layer DBN (Salakhutdinov and Larochelle, 2010)	91.69%
DBM: MF-FULL (Salakhutdinov and Larochelle, 2010)	92.77%
Third Order RBM (Nair and Hinton, 2009)	93.5%

Table 2: Experimental results on the small NORB data test set. Relevant published results are shown for comparison. NN, Lin and Mix indicate neural network, linear and a combination of neural network and periodic covariances respectively. Logreg indicates that a parametric logistic regression mapping to labels is blended with the autoencoder.

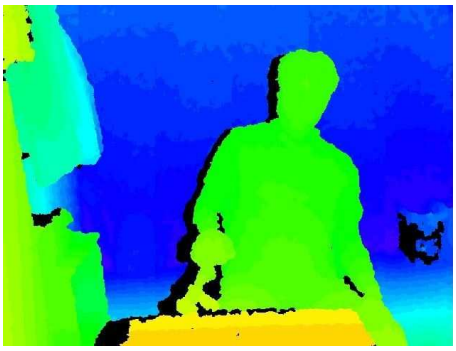
The NPGA model with four nonlinear kernel GPs significantly outperforms all other models, with an accuracy of 94.28%. This is to our knowledge the best (non-convolutional) result for a shallow model on this data set. The model indeed appears to separate the irrelevant transformations of the data from the structure relevant to the classification objective. In fact, a logistic regression classifier applied to only the 1200 hidden units on which the class GP was applied achieves a test error rate of 94.02%. This implies that the half of the latent representation that encodes the information to which the model should be invariant can be discarded with virtually no discriminative penalty. Given the significant difference in accuracy between this formulation and the other models, it appears to be very important to separate the encoding of different sources of variation within the autoencoder hidden layer.

The NPGA with four linear covariance GPs performed more poorly than the NPGA with a single linear covariance GP to class labels (92.09% compared to 93.03%). This interesting observation highlights the importance of using an appropriate mapping to each label. For example, it is unlikely that a linear covariance would be able to appropriately capture the structure of the periodic azimuth signal. An autoencoder with parametric guidance to all four labels, mimicking the configuration of the NPGA, achieved the poorest performance of the models tested, with 86% accuracy. This model incorporated two logistic and two Gaussian outputs applied to separate partitions of the hidden units. These results demonstrate the advantage of the GP formulation for supervised guidance, which gives the flexibility of choosing an appropriate kernel for different label mappings (e.g., a periodic kernel for the rotation label).

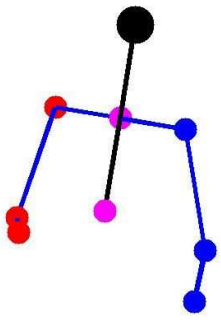
(a) The Robot



(b) Using the Robot



(c) Depth Image



(d) Skeletal Joints

Figure 4: The rehabilitation robot setup and sample data captured by the sensor.

4.4 Rehabilitation Data

As a final analysis, we demonstrate the utility of the NPGA on a real-world problem from the domain of assistive technology for rehabilitation. Rehabilitation patients benefit from performing repetitive rehabilitation exercises as frequently as possible but are limited due to a shortage of rehabilitation therapists. Thus, Kan et al. (2011), Huq et al. (2011), Lu et al. (2011) and Taati et al. (2012) developed a system to automate the role of a therapist guiding rehabilitation patients through repetitive upper limb rehabilitation exercises. The system allows users to perform upper limb reaching exercises using a robotic arm (see Figures 4a, 4b) while tailoring the amount of resistance to match the user's ability level. Such a system can alleviate the burden on therapists and significantly expedite rehabilitation for patients.

Critical to the effectiveness of the system is its ability to discriminate between various types of incorrect posture and prompt the user accordingly. The current system (Taati et al., 2012) uses a Microsoft Kinect sensor to observe a patient performing upper limb reaching exercises and records their posture as a temporal sequence of seven estimated upper body skeletal joint angles (see Figures 4c, 4d for an example depth image and corresponding pose skeleton captured by the system). A classifier is then employed to discriminate between five different classes of posture, consisting of good posture and four common forms of improper posture resulting from compensation due to limited agility. Taati et al. (2012) obtained a data set of seven users each performing each class of action at least once, creating a total of 35 sequences (23,782 frames). They compare the use of

Model	Accuracy
SVM^{Multiclass} Taati et al. (2012)	80.0%
Hidden Markov SVM Taati et al. (2012)	85.9%
ℓ_2-Regularized Logistic Regression	86.1%
NPGA ($\alpha = 0.8147, \beta = 0.3227, H = 3$, 242 hidden units)	91.7%

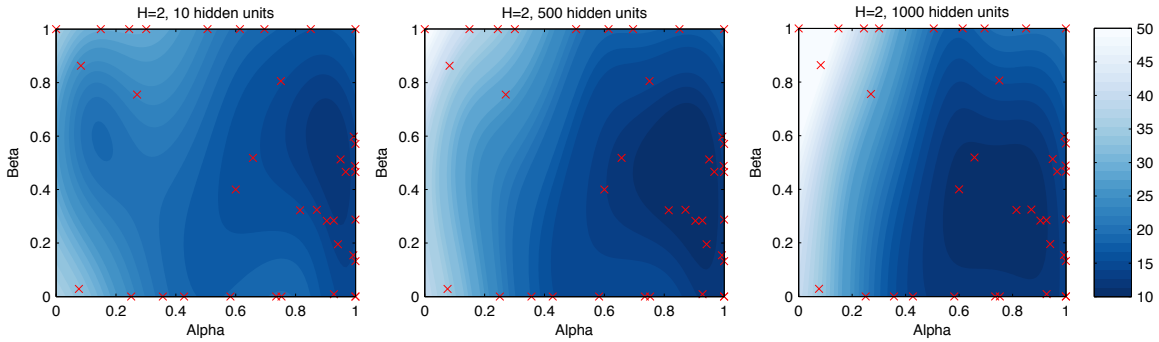
Table 3: Experimental results on the rehabilitation data. Per-frame classification accuracies are provided for different classifiers on the test set. Bayesian optimization was performed on a validation set to select hyperparameters for the ℓ_2-regularized logistic regression and the best performing NPGA algorithm.

a multiclass support vector machine and a hidden Markov support vector machine in a leave-one-subject-out test setting to distinguish these classes and report best per-frame classification accuracy rates of 80.0% and 85.9% respectively.

In our analysis of this problem we use a NPGA to encode a latent embedding of postures that facilitates better discrimination between different posture types. The same formulation as presented in Section 4.1 is applied here. We interpolate between a standard autoencoder ($\alpha = 0$), a classification neural net ($\alpha = 1, \beta = 1$), and a nonparametrically guided autoencoder by linear blending of their objectives according to Equation 5. Rectified linear units were used in the autoencoder. As in Taati et al. (2012), the input to the model is the seven skeletal joint angles, that is, $\mathcal{Y} = \mathbb{R}^7$, and the label space $\mathcal{Z}$ is over the five classes of posture.

In this setting, rather than perform a grid search for parameter selection as in Section 4.1, we optimize validation set error over the hyperparameters of the model using Bayesian optimization (Mockus et al., 1978). Bayesian optimization is a methodology for globally optimizing noisy, black-box functions based on the principles of Bayesian statistics. Particularly, given a prior over functions and a limited number of observations, Bayesian optimization explicitly models uncertainty over functional outputs and uses this to determine where to search for the optimum. For a more in-depth overview of Bayesian optimization see Brochu et al. (2010). We use the Gaussian process expected improvement algorithm with a Matern-$\frac{5}{2}$ covariance, as described in Snoek et al. (2012), to search over $\alpha \in [0,1]$, $\beta \in [0,1]$, $10-1000$ hidden units in the autoencoder and the GP latent dimensionality $H \in \{1..10\}$. The best validation set error observed by the algorithm, on the twelfth of thirty-seven iterations, was at $\alpha = 0.8147$, $\beta = 0.3227$, $H = 3$ and 242 hidden units. These settings correspond to a per-frame classification error rate of 91.70%, which is significantly higher than that reported by Taati et al. (2012). Results obtained using various models are presented in Table 3.

The relatively low number of experiments required by Bayesian optimization to find a state-of-the-art result implies that the validation error is a well behaved function of the various hyperparameters. The relationship between the model hyperparameters and validation error is challenging to visualize, but it is important to assess their relative effect on the performance of the model. Thus, in Figure 5 we explore how the relationship between validation error and the amount of nonparametric guidance α, and parametric guidance β is expected to change as the number of autoencoder hidden units is varied. That is, we show the expected value of the validation error for unobserved points under a Gaussian process regression. Similarly to the results observed in Section 4.1, it seems clear that the best region in hyperparameter space is a combination of all three objectives, the parametric



(a) (b) (c)

Figure 5: The posterior mean learned by Bayesian optimization over the validation set classification error (in percent) for α and β with H fixed at 2 and three different settings of autoencoder hidden units: (a) 10, (b) 500, and (c) 1000. This shows how the relationship between validation error and the amount of nonparametric guidance, α, and parametric guidance, β, is expected to change as the number of autoencoder hidden units is increased. The red x's indicate points that were explored by the Bayesian optimization routine.

guidance, nonparametric guidance and unsupervised learning. This reinforces the theory that although incorporating a parametric logistic regressor to the labels more directly reflects the ultimate goal of the model, it is more prone to overfit the training data than the GP. Also, as we increase the number of hidden units in the autoencoder, the amount of guidance required appears to decrease. As the capacity of the autoencoder is increased, it is likely that the autoencoder encodes increasingly subtle statistical structure in the data. When there are fewer hidden units, this structure is not encoded unless the autoencoder objective is augmented to reflect a preference for it. Interestingly, the validation error for $H = 2$ was significantly better than $H = 1$ but it did not appear to change significantly for $2 \geq H \leq 10$.

An additional interesting result is that, on this problem, the classification performance is worse when the nearest neighbors algorithm is used on the learned representation of the NPGA for discrimination. With the best performing NPGA reported above, a nearest neighbors classifier applied to the hidden units of the autoencoder achieved an accuracy of 85.05%. Adjusting β in this case also did not improve accuracy. This likely reflects the fact that the autoencoder must still encode information that is useful for reconstruction but not discrimination.

In this example, the resulting classfier must operate in real time to be useful for the rehabilitation task. The final product of our system is a simple softmax neural network, which is directly applicable to this problem. It is unlikely that a Gaussian process based classifier would be feasible in this context.

5. Conclusion

In this paper we present an interesting theoretical link between the autoencoder neural network and the back-constrained Gaussian process latent variable model. A particular formulation of the back-constrained GPLVM can be interpreted as an autoencoder in which the decoder has an in-

finite number of hidden units. This formulation exhibits some attractive properties as it allows one to learn the encoder half of the autoencoder while marginalizing over decoders. We examine the use of this model to guide the latent representation of an autoencoder to encode auxiliary label information without instantiating a parametric mapping to the labels. The resulting nonparametric *guidance* encourages the autoencoder to encode a latent representation that captures salient structure within the input data that is harmonious with the labels. Conceptually, this approach enforces simply that a smooth mapping exists from the latent representation to the labels rather than choosing or learning a specific parameterization. The approach is empirically validated on four data sets, demonstrating that the nonparametrically guided autoencoder encourages latent representations that are better with respect to a discriminative task. Code to run the NPGA is available at `http://hips.seas.harvard.edu/files/npga.tar.gz`. We demonstrate on the NORB data that this model can also be used to *discourage* latent representations that capture statistical structure that is known to be irrelevant through guiding the autoencoder to separate multiple sources of variation. This achieves state-of-the-art performance for a shallow non-convolutional model on NORB. Finally, in Section 4.4, we show that the hyperparameters introduced in this formulation can be optimized automatically and efficiently using Bayesian optimization. With these automatically selected hyperparameters the model achieves state of the art performance on a real-world applied problem in rehabilitation research.

References

Ryan P. Adams, Zoubin Ghahramani, and Michael I. Jordan. Tree-structured stick breaking for hierarchical data. In *Advances in Neural Information Processing Systems*, 2010.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*, 2007.

Christopher M. Bishop and G. D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, 1993.

Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *pre-print*, 2010. arXiv:1012.2599.

Adam Coates, Honglak Lee, and Andrew Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Conference on Artificial Intelligence and Statistics*, 2011.

Garrison W. Cottrell, Paul Munro, and David Zipser. Learning internal representations from gray-scale images: An example of extensional programming. In *Conference of the Cognitive Science Society*, 1987.

Li Deng, Mike Seltzer, Dong Yu, Alex Acero, Abdel-Rahman Mohamed, and Geoffrey E. Hinton. Binary coding of speech spectrograms using a deep autoencoder. In *Interspeech*, 2010.

Jacob Goldberger, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, 2004.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

Rajibul Huq, Patricia Kan, Robby Goetschalckx, Debbie Hébert, Jesse Hoey, and Alex Mihailidis. A decision-theoretic approach in the design of an adaptive upper-limb stroke rehabilitation robot. In *International Conference of Rehabilitation Robotics (ICORR)*, 2011.

Patricia Kan, Rajibul Huq, Jesse Hoey, Robby Goestschalckx, and Alex Mihailidis. The development of an adaptive upper-limb stroke rehabilitation robotic system. *Neuroengineering and Rehabilitation*, 2011.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *International Conference on Machine Learning*, 2007.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

Neil D. Lawrence and Joaquin Quiñonero-Candela. Local distance preservation in the GP-LVM through back constraints. In *International Conference on Machine Learning*, 2006.

Neil D. Lawrence and Raquel Urtasun. Non-linear matrix factorization with Gaussian processes. In *International Conference on Machine Learning*, 2009.

Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

David Lowe and Michael E. Tipping. Neuroscale: Novel topographic feature extraction using RBF networks. In *Advances in Neural Information Processing Systems*, 1997.

Elaine Lu, Rosalie Wang, Rajibul Huq, Don Gardner, Paul Karam, Karl Zabjek, Debbie Hébert, Jennifer Boger, and Alex Mihailidis. Development of a robotic device for upper limb stroke rehabilitation: A user-centered design approach. *Journal of Behavioral Robotics*, 2(4):176–184, 2011.

David J. C. MacKay. Introduction to Gaussian processes. *Neural Networks and Machine Learning*, 1998.

David J.C. MacKay. Bayesian neural networks and density networks. In *Nuclear Instruments and Methods in Physics Research, A*, pages 73–80, 1994.

Jonas Mockus, Vytautas Tiešis, and Antanas Žilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

Vinod Nair and Geoffrey E. Hinton. 3D object recognition with deep belief nets. In *Advances in Neural Information Processing Systems*, 2009.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning*, 2010.

Radford Neal. Bayesian learning for neural networks. *Lecture Notes in Statistics*, 118, 1996.

Marc'Aurelio Ranzato and Martin Szummer. Semi-supervised learning of compact document representations with deep networks. In *International Conference on Machine Learning*, 2008.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

Ruslan Salakhutdinov and Geoffrey Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Conference on Artificial Intelligence and Statistics*, 2007.

Ruslan Salakhutdinov and Geoffrey Hinton. Using deep belief nets to learn covariance kernels for Gaussian processes. In *Advances in Neural Information Processing Systems*, 2008.

Ruslan Salakhutdinov and Hugo Larochelle. Efficient learning of deep Boltzmann machines. In *Conference on Artificial Intelligence and Statistics*, 2010.

Aaron P. Shon, Keith Grochow, Aaron Hertzmann, and Rajesh P. N. Rao. Learning shared latent structure for image synthesis and robotic imitation. In *Advances in Neural Information Processing Systems*, 2005.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2012.

Babak Taati, Rosalie Wang, Rajibul Huq, Jasper Snoek, and Alex Mihailidis. Vision-based posture assessment to detect and categorize compensation during robotic rehabilitation therapy. In *International Conference on Biomedical Robotics and Biomechatronics*, 2012.

Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

Raquel Urtasun and Trevor Darrell. Discriminative Gaussian process latent variable model for classification. In *International Conference on Machine Learning*, 2007.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, 2008.

Francesco Vivarelli and Christopher K. I. Williams. Discovering hidden features with Gaussian process regression. In *Advances in Neural Information Processing Systems*, 1999.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Multifactor Gaussian process models for style-content separation. In *International Conference on Machine Learning*, 2007.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

Christopher K. I. Williams. Computation with infinite neural networks. *Neural Computation*, 10 (5):1203–1216, 1998.

Richard S. Zemel, Christopher K. I. Williams, and Michael C. Mozer. Lending direction to neural networks. *Neural Networks*, 8:503–512, 1995.

Journal of Machine Learning Research 13 (2012) 2589-2615 Submitted 11/11; Revised 5/12; Published 9/12

Finding Recurrent Patterns from Continuous Sign Language Sentences for Automated Extraction of Signs

Sunita Nayak SNAYAK@TAAZ.COM
Taaz Inc.
4250 Executive Square, Suite 420
La Jolla, CA 92037 USA

Kester Duncan KKDUNCAN@CSE.USF.EDU
Sudeep Sarkar SARKAR@CSE.USF.EDU
Department of Computer Science & Engineering
University of South Florida
Tampa, FL 33620, USA

Barbara Loeding BARBARA@USF.EDU
Department of Special Education
University of South Florida
Lakeland, FL 33803, USA

Editor: Isabelle Guyon

Abstract

We present a probabilistic framework to automatically learn models of recurring signs from multiple sign language video sequences containing the vocabulary of interest. We extract the parts of the signs that are present in most occurrences of the sign in context and are robust to the variations produced by adjacent signs. Each sentence video is first transformed into a multidimensional time series representation, capturing the motion and shape aspects of the sign. Skin color blobs are extracted from frames of color video sequences, and a probabilistic relational distribution is formed for each frame using the contour and edge pixels from the skin blobs. Each sentence is represented as a trajectory in a low dimensional space called the space of relational distributions. Given these time series trajectories, we extract signemes from multiple sentences concurrently using iterated conditional modes (ICM). We show results by learning single signs from a collection of sentences with one common pervading sign, multiple signs from a collection of sentences with more than one common sign, and single signs from a mixed collection of sentences. The extracted signemes demonstrate that our approach is robust to some extent to the variations produced within a sign due to different contexts. We also show results whereby these learned sign models are used for spotting signs in test sequences.

Keywords: pattern extraction, sign language recognition, signeme extraction, sign modeling, iterated conditional modes

1. Introduction

Sign language research in the computer vision community has primarily focused on improving recognition rates of signs either by improving the motion representation and similarity measures (Yang et al., 2002; Al-Jarrah and Halawani, 2001; Athitsos et al., 2004; Cui and Weng, 2000; Wang et al., 2007; Bauer and Hienz, 2000) or by adding linguistic clues during the recognition process

(Bowden et al., 2004; Derpanis et al., 2004). Ong and Ranganath (2005) presented a review of the automated sign language research and also highlighted one important issue in continuous sign language recognition. While signing a sentence, there exists transitions of the hands between two consecutive signs that do not belong to either sign. This is called movement epenthesis (Liddell and Johnson, 1989). This needs to be dealt with first before dealing with any other phonological issues in sign language (Ong and Ranganath, 2005). Most of the existing work in sign language assumes that the training signs are already available and often signs used in the training set are the isolated signs with the boundaries chopped off, or manually selected frames from continuous sentences. The ability to recognize isolated signs does not guarantee the recognition of signs in continuous sentences. Unlike isolated signs, a sign in a continuous sentence is strongly affected by its context in the sentence. Figure 1 shows two sentences 'I BUY TICKET WHERE?' and 'YOU CAN BUY THIS FOR HER' with a common sign 'BUY' between them. The frames representing the sign 'BUY' and the neighboring signs are marked. The unmarked frames between the signs indicate the frames corresponding to movement epenthesis. It can be observed that the same sign 'BUY' is preceded and succeeded by movement epenthesis that depends on the end and start of the preceding and succeeding sign respectively. The movement epenthesis also affects how the sign is signed. This effect makes the automated extraction, modeling and recognition of signs from continuous sentences more difficult when compared to just plain gestures, isolated signs, or finger spelling.

In this paper, we address the problem of automatically extracting the part of a sign that is most common in all occurrences of the sign, and hence expected to be robust with respect to the variation of adjacent signs. These common parts can be used for spotting or recognition of signs in continuous sign language sentences. They can also be used by sign language experts for teaching or studying variations between instances of signs in continuous sign language sentences, or in automated sign language tutoring systems. Furthermore, they can be used even in the process of translating sign language videos directly to spoken words.

In a related work inspired by the success of the use of phonemes in speech recognition, the authors sought to extract common parts in different instances of a sign and thus arrive at a phoneme-analogue for signs (Bauer and Kraiss, 2002). But unlike speech, sign language does not have a completely defined set of phonemes. Hence, we consider extracting commonalities at the sentence and sub-sentence level.

A different but a closely related problem is the extraction of common subsequences, also called motifs, from very long multiple gene sequences in biology (Bailey and Elkan, 1995; Lawrence et al., 1993; Pevzner and Sze, 2000; Rigoutsos and Floratos, 1998). Lawrence et al. (1993) used a Gibbs sampling approach based on discrete matches or mismatches of subsequences that were strings of symbols of gene sequences. Bailey and Elkan (1995) used expectation maximization to find common subsequences in univariate biopolymer sequences. In biology, researchers deal with univariate discrete sequences, and hence their algorithms are not always directly applicable to other multivariate continuous domains in time series like speech or sign language. Some researchers tried to symbolize a continuous time series into discrete sequences and used existing algorithms from bioinformatics. For example, Chiu et al. (2003) symbolized the time series into a sequence of symbols using local approximations and used random projections to extract common subsequences in noisy data. Tanaka et al. (2005) extended their work by performing principal component analysis on the multivariate time series data and projected them onto a single dimension and symbolized the data into discrete sequences. However, it is not always possible to get all the important information in

(a) Continuous Sentence 'I BUY TICKET WHERE?'

(b) Continuous Sentence 'YOU CAN BUY THIS FOR HER'

Figure 1: Movement epenthesis in sign language sentences. Frames corresponding to the common sign 'BUY' are marked in red. Signs adjacent to BUY are marked in magenta. Frames between marked frames represent movement epenthesis that is, the transition between signs. Note that the sign itself is also affected by having different signs preceding or following it.

the first principal component alone. Further extending his work, Duchne et al. (2007) find recurrent patterns from multivariate discrete data using time series random projections.

Due to the inherent continuous nature of many time series data like gesture and speech, new methods were developed that do not require approximating the data to a sequence of discrete symbols. Denton (2005) used a continuous random-walk noise model to cluster similar substrings. Nayak et al. (2005) and Minnen et al. (2007) use continuous multivariate sequences and dynamic time warping to find distances between the substrings. Oates (2002); Nayak et al. (2005) and Nayak et al. (2009a) are among the few works in finding recurrent patterns that address non-uniform sampling of time series. The recurrent pattern extraction approach proposed in this paper is based

on multivariate continuous time series, uses dynamic time warping to find distances between substrings, and handles length variations of common patterns.

Following the success of Hidden Markov Models (HMMs) in speech recognition, they were used by sign language researchers (Vogler and Metaxas, 1999; Starner and Pentland, 1997; Bowden et al., 2004; Bauer and Hienz, 2000; Starner et al., 1998) for representing and recognizing signs. However, HMMs require a large number of training data and unlike speech, data from native signers is not as easily available as speech data. Hence, non-HMM-based approaches have been used (Farhadi et al., 2007; Nayak et al., 2009a; Yang et al., 2010; Buehler et al., 2009; Nayak et al., 2009b; Oszust and Wysocki, 2010; Han et al., 2009). In this paper, we use a continuous trajectory representation of signs in a multidimensional space and use dynamic time warping to match subsequences. The relative configuration of the two hands and face in each frame is represented by a relational distribution (Vega and Sarkar, 2003; Nayak et al., 2005), which in itself is a probability density function. The motion dynamics of the signer is captured as changes in the relational distributions. It also allows us to interpolate motion, if required, for data sets with lower frame capture rates. It should also be noted that, unlike many of the previous works in sign language that perform tracking of the hands using 3D magnetic trackers or color gloves (Fang et al., 2004; Vogler and Metaxas, 2001; Wang et al., 2002; Ma et al., 2000; Cooper and Bowden, 2009), our representation does not require tracking and relies on skin segmentation.

We present a Bayesian framework to extract the common subsequences or signemes from all the given sentences simultaneously. Figure 2 depicts the overview of our approach. With this framework, we can extract the first most common sign, the second most common sign, the third most common sign and so on. We represent each sentence as a trajectory in a multi-dimensional space that implicitly captures the shape and motion in the video. Skin color blobs are extracted from frames of color video, and a relational distribution is formed for each frame using the edge pixels in the skin blobs. Each sentence is then represented as a trajectory in a low dimensional space called the space of relational distributions, which is arrived at by performing principal component analysis (PCA) on the relational distributions. There are other alternatives to PCA that are possible and discussed in Nayak et al. (2009b). The other choices do not change the nature of the signeme finding approach, they only affect the quality of the features. The starting locations $(a_1, ...a_n)$ and widths $(w_1, ...w_n)$ of the candidate signemes in all the n sentences are together represented by a parameter vector. The starting locations are initialized with random starting locations, based on uniform random sampling from each sentence, and the initial width values are randomly selected from a given range of values. The parameter vector is updated sequentially by sampling the starting point and width of the possible signeme in each sentence from a joint conditional distribution that is based on the locations and widths of the target possible signeme in all other sentences. The process is iterated till the parameter values converge to a stable solution. Monte Carlo approaches like Gibbs sampling (Robert and Casella, 2004; Gilks et al., 1998; Casella and George, 1992), which is a special case of the Metropolis-Hastings algorithm (Chib and Greenberg, 1995) can be used for global optimization while updating the parameter vector by performing importance sampling on the conditional probability distribution. However, this has a high burn-in period.

In this paper, we adopt a greedy approach based on the use of iterated conditional modes (ICM) (Besag, 1986). ICM converges much faster than a Gibbs sampler, but is known to be largely dependent on the initialization. We overcome this limitation by performing ICM a number of times equal to the average length of the n sentences, with different initializations. The most frequently occurring solution from all the ICM runs is considered as the final solution.

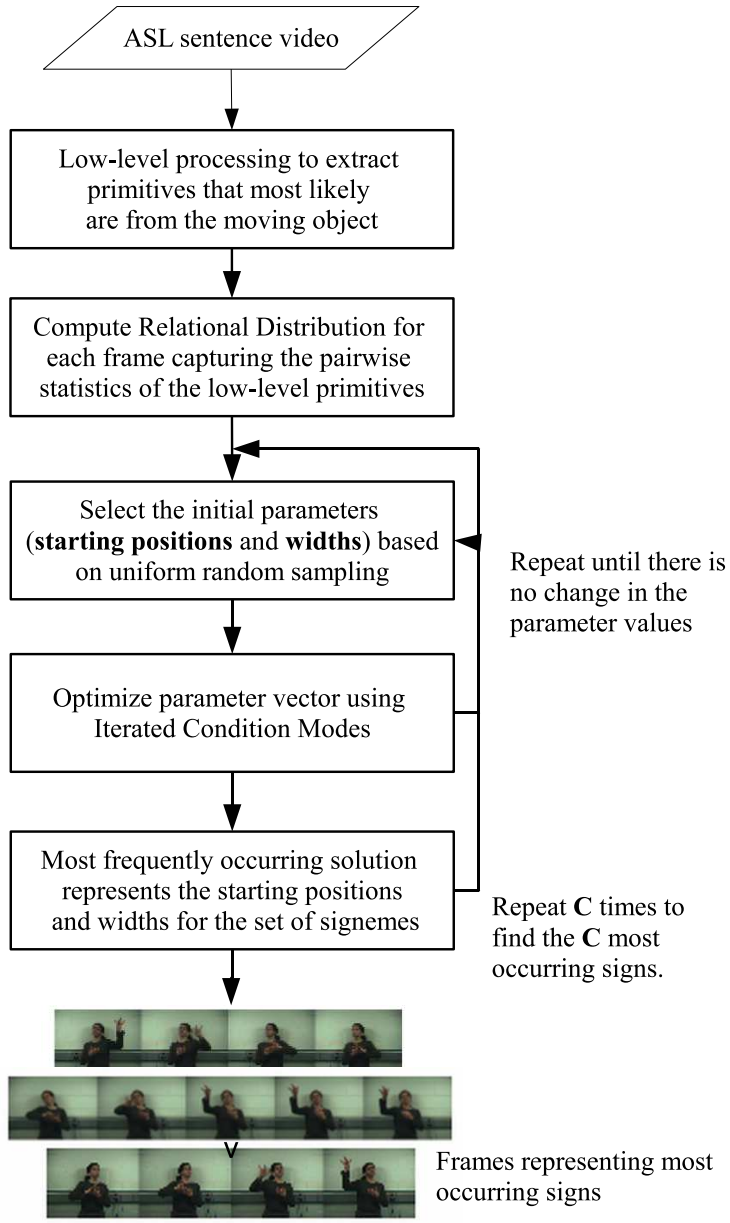


Figure 2: Overview of our approach. Each of the n sentences is represented as a sequence in the Space of Relational Distributions, and common patterns are extracted using iterated conditional modes (ICM). The parameter set $\{a_1, w_1, ...a_n, w_n\}$ is initialized using uniform random sampling and the conditional density corresponding to each sentence is updated in a sequential manner.

The work in this paper builds on the work of Nayak et al. (2009a) and is different in multiple respects. We propose a system that is generalized to extract more than one common sign from a collection of sentences (first most common sign, second most common sign and so on), whereas

in the previous work, only single signs were extracted. We also extract single signs from a mixed collection of sentences where there are more than one common sign in context. In addition to this, we present a more in-depth exposition of the underlying theory.

The contributions of this paper can be summarized as follows: (i) we present an unsupervised approach to automatically extract parts of signs that are robust to the variation of adjacent signs simultaneously from multiple sign language sentences, (ii) our approach does not consider all possible parameter combinations, instead samples each of them in a sequential manner until convergence, which saves a lot of computation, (iii) we show results on extracting signs from plain color videos of continuous sign language sentences without using any color gloves or magnetic trackers, and (iv) we show results whereby the learned signs are used for spotting signs in test sequences.

We organize the paper as follows. Section 2 presents a short review of relational distributions. In Section 3, we present the definition of signeme and then formulate the problem of finding signemes from a given set of sequences in a probabilistic framework. We describe how we solve it using iterated conditional modes. It is then followed by a description of our experiments and results in Section 4. Finally, Section 5 concludes the paper and discusses possible future work.

2. Relational Distributions

We use relational distributions to capture the global and relative configuration of the hands and the face in an image. Motion is then captured as the changes in the relational distributions. They were originally introduced by Vega and Sarkar (2003) for human gait recognition. They have also been used before for representing sign language sentences without the use of color gloves or magnetic trackers (Nayak et al., 2005, 2009b). We briefly review them here in this section.

How do we capture the global configuration of the object? We start with low-level primitives that are most likely to come from the articulated object. The exact nature of the low-level primitives can vary. Some common choices include edges, salient points, Gabor filter outputs and so on. We use edges in this work. We start from some level of segmentation of the object from the scene. These processes are fairly standard and have been used widely in gesture and sign recognition. They may involve color-based segmentation, skin-color segmentation, or background subtraction. In this work, we perform skin-color segmentation using histogram-based Bayesian classification (Phung et al., 2005). We use the contours of the skin blobs and Canny edges within the blobs as our low-level image primitives. The global configuration is captured by considering the relationships between these primitives.

We use the distance between two primitives in the vertical and horizontal directions (dx, dy) as relational attributes. Let vector $\mathbf{u} = \{dx, dy\}$ represent the vector of relational attributes. The joint probability function $P(\mathbf{u})$ then describes the distribution of primitives within an image and captures the shape of the pattern in the image. This probability is called a *relational distribution*. It captures the global configuration of the low-level primitives. Figure 3(c) illustrates how motion is captured using relational distributions. It shows the top view of the distributions. The region near to center represents points closer to each other, for example, the edge points within the face or within the hand, while farther from center represents the farther away points, for example, the relationship between edge points of a hand and the face. Notice the change in the relational distribution as the signer moves one of her hands. To be able to discriminate symmetrically opposite motion, we maintain the signs (or directions) of the horizontal and vertical distances between the edge pixels in each ordered pair. This leads to representing the probability distribution in a four quadrant system.

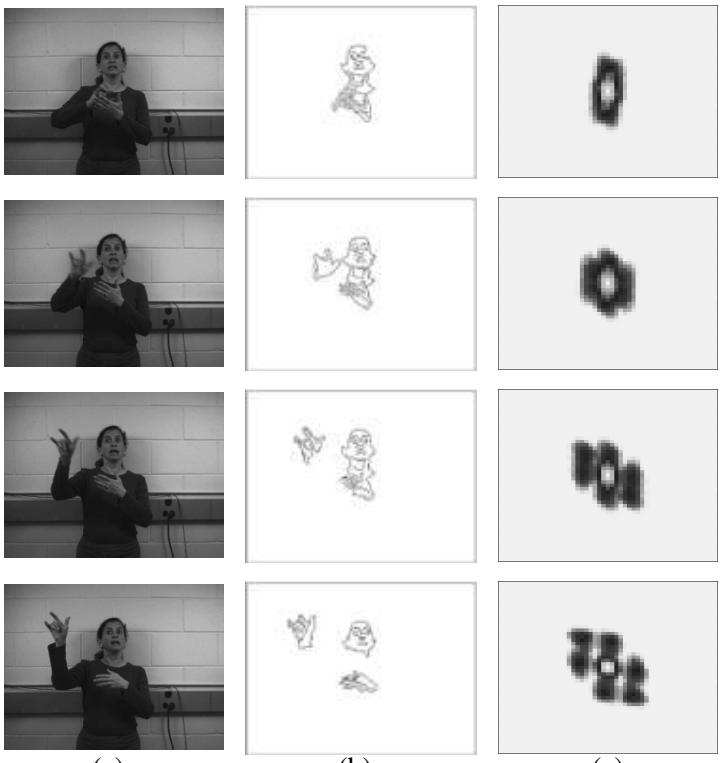


Figure 3: Variations in relational distributions with motion. (a) Motion sequence. (b) Edge pixels from the skin color blobs. (c) Relational distributions constructed from the low level features (edge pixels) of the images in the motion sequence. The horizontal axis of the relational distribution represents the horizontal distance between the edge pixels and its vertical axis represents the vertical distance between edge pixels.

Given that these relational distributions exhibit complicated shapes that are difficult to be modeled readily using a combination of simple shaped distributions such as Gaussian mixtures, we adopt non-parametric histogram-based representation. For better discrimination of the probabilities, we do not add counts to the center of the histogram which represents the distance of the edge pixels from itself or very close adjacent pixels. Each bin then counts the pairs of edge pixels between which the horizontal and vertical distances each lie in some fixed range that depends on the location of the bin in the histogram.

In our experiments, we found that an empirically-determined fixed histogram size of 51×51 was sufficient. The above range is then defined using linear mapping between the image size and the histogram size, for example, image size along the horizontal direction corresponds to half the histogram size in the horizontal direction. One could use histogram bin size optimization techniques for optimizing the histograms, but we do not address them in this paper. We then reduce the dimensionality of the relational distributions by performing PCA on the set of relational distributions from all the input sentences and retain the number of dimensions required to keep a certain percentage

of energy, typically 95%. The new subspace arrived at is called the space of relational distributions (SoRD). Each video sequence is thus represented as a sequence of points in the SoRD space.

Note that the choice of the relational distribution is not a central requirement for the signeme learning process discussed in this paper. We use relational distributions to enable us to work with pure video data, without the use of markers or colored gloves. If magnetic markers or colored gloves are available then one could use their attributes to construct a different feature space and consider trajectories in them. One advantage of our representation is that the face and head locations are implicitly taken into account in addition to the hands. In short, the first step of the process is to construct a time series representation in an appropriate feature space.

3. Problem Formulation

Sign language sentences are series of signs. Figure 4 illustrates the traces of the first vs. second dimension in the feature space, of three sentences S_1, S_2 and S_3 with only one common sign, R, among them. The signeme represents the portion of the sign that is most similar across the sentences. Table 3 defines the notations that will be used in this paper. We formulate the signeme extraction

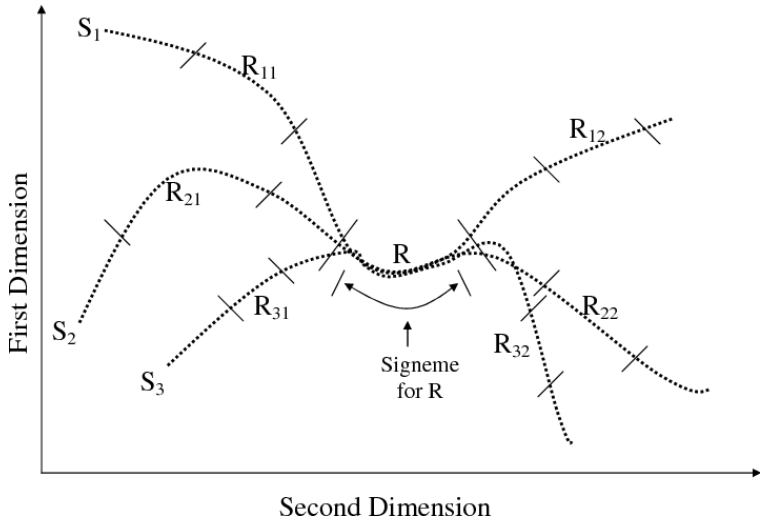


Figure 4: Concept of signemes. First vs. second dimensions of sentences S_1 with signs R_{11}, R, R_{12} in order, S_2 with signs R_{21}, R, R_{22} and S_3 with signs R_{31}, R, R_{32}. The common sign is R. The portion of R that is most similar across sentences is the signeme representative of R.

problem as finding the most recurring patterns among a set of n sentences $\{\vec{S}_1, \cdots, \vec{S}_n\}$, that have at least one common sign present in all the sentences. The commonality concept underlying the definition of a signeme can be cast in terms of distances. Let $\vec{s}_{a_i}^{w_i}$ represent a substring from the sequence $\vec{S}_i$ consisting of the points with indices $a_i, \cdots a_i + w_i - 1$, and $d(\vec{x}, \vec{y})$ denote the distance between two substrings $\vec{x}$ and $\vec{y}$ based on dynamic time warping. We define the set of signemes to be the set of substrings denoted by $\{\vec{s}_{a_1}^{w_1}, \cdots, \vec{s}_{a_n}^{w_n}\}$ that is most similar among all possible substrings from the given set of sentences. In the generalized case where C most common signs are sought, the

$\{\vec{S}_1, \cdots, \vec{S}_n\}$	Set of n sentences with at least one common sign present in all the sentences. The index within a sentence could represent time or arc length in configuration shape space
L_i	Length of sentence S_i
$\vec{s}_{a_j}^{w_j}$	Subsequence of sentence S_j starting from index a_j to $a_j + w_j - 1$. We may sometimes use $\vec{s}_{j,a}^{w}$ to make explicit the j-th index if it is not represented along with any other superscript or subscript of this term.
A, B	Possible choices of width for signemes of a sign include all integers from A to B. The values of A and B are decided based on the dynamics involved in the sign.
θ	Set of parameters $\{a_1, w_1, \cdots, a_n, w_n\}$ defining a set of substrings of the given sentences
$\theta_{(a_i)}$	Set of all parameters *excluding* the parameter a_i. We have similar interpretations for $\theta_{(w_i)}$ or $\theta_{(i)}$.
$d(\vec{x}, \vec{y})$	Distance between the subsequences $\vec{x}$ and $\vec{y}$ based on a mapping found using dynamic time warping (DTW). This distance has to be calculated carefully so that it is not biased towards finding short subsequences only.

Table 1: Notations

set of signemes are defined as $\{\vec{s}_{a_{11}}^{w_{11}}, \vec{s}_{a_{12}}^{w_{11}}, \cdots, \vec{s}_{a_{nC}}^{w_{nC}}\}$. In theory, C can extend to the number of words in the shortest sentence.

Let $\theta = \{a_1, w_1, \cdots a_n, w_n\}$ denote the parameter set representing a set of substrings, at least one from each of the n sentences, and θ_m denote the parameter set representing the target set of signemes in the n sentences. We find θ_m using the probabilistic framework of Equation 1.

$$\theta_m = \arg\max_{\theta} p(\theta) \tag{1}$$

Note that $p(\theta)$ is a probability over the space of all possible substrings. We define this probability to be a function of the inter-substring distances in Equation 2:

$$p(\theta) = \frac{g(\theta)}{\sum_{\theta} g(\theta)}. \tag{2}$$

The term $g(\theta)$ is defined in Equation 3 as follows:

$$g(\theta) = \exp\left(-\beta \sum_{i=1}^{n} \sum_{j=1}^{n} d(\vec{s}_{a_i}^{w_i}, \vec{s}_{a_j}^{w_j})\right) \tag{3}$$

with β being a positive constant.

Note that $g(\theta)$ varies inversely with the summation of the pair-wise distances of all the subsequences given by θ. Also note that $p(\theta)$ is hard to compute or even sample from because it is computationally expensive to compute the denominator in Equation 2, as it involves the summation over all possible parameter combinations. β acts as a scale parameter, which controls the slopes of the peaks in the probability space. It can also be looked upon as the smoothing parameter. If probability sampling algorithms like Gibbs sampling (Casella and George, 1992) are used in later steps, then the rate of convergence would be determined by this parameter.

Let θ_i represent the parameters from the i^{th} sentence, that is, $\{a_i, w_i\}$ and $\theta_{(i)}$ represent the rest of the parameters, $\{a_1, w_1 \cdots a_{i-1}, w_{i-1}, a_{i+1}, w_{i+1} \cdots a_n, w_n\}$. To make sampling easier, we construct a *conditional* density function of the parameters from each sentence, that is, θ_i, *given* the values of

the rest of the parameters, that is, $\theta_{(i)}$. In other words, we construct a probability density function of the possible starting points and widths in each sentence, given the estimated starting points and widths of the common pattern in all other sentences, that is, $f(\theta_i|\theta_{(i)})$. Of course, this conditional density function has to be *derived* from the joint density function specified in Eq. 2. This is outlined in Equation 4 as follows:

$$f(\theta_i|\theta_{(i)}) \quad = \quad \frac{p(\theta)}{p(\theta_{(i)})} = \frac{p(\theta)}{\sum_{\theta_i} p(\theta)} = \frac{g(\theta)}{\sum_{\theta_i} g(\theta)}. \tag{4}$$

Since the normalization to arrive at this conditional density function involves summation over one parameter, it is now easier to compute and sample from. The specific form for this conditional density function using the dynamic time warping (DTW) distances as described in Equation 5 is

$$f(\theta_i|\theta_{(i)}) = \frac{\exp\left(-\beta \sum_{k=1}^{n} d(\vec{s}_{a_i}^{w_i}, \vec{s}_{a_k}^{w_k}))\right)}{\sum_{\theta_i} \exp\left(-\beta \sum_{k=1}^{n} d(\vec{s}_{a_i}^{w_i}, \vec{s}_{a_k}^{w_k}))\right)}. \tag{5}$$

Note that the distance terms that do not involve a_i and w_i, that is, do not involve the i-th sentence appear both in the numerator and the denominator and so cancel out. For notational convenience, this is sometimes represented using conditional g functions described below in Equation 6 as:

$$f(\theta_i|\theta_{(i)}) = \frac{g(\theta_i|\theta_{(i)})}{\sum_{\theta_i} g(\theta_i|\theta_{(i)})}, \tag{6}$$

where $g(\theta_i|\theta_{(i)}) = \exp\left(-\beta \sum_{k=1}^{n} d(\vec{s}_{a_i}^{w_i}, \vec{s}_{a_k}^{w_k}))\right)$.

3.1 Distance Measure

The distance function d in the above equations needs to be chosen carefully such that it is not biased towards the shorter subsequences. Here, we briefly describe how we compute the distance between two substrings using dynamic time warping. Let l_1 and l_2 represent the length of the two substrings and $e(i, j)$ represent the Euclidean distance between the i^{th} data point from the first substring and the j^{th} data point from the second substring. Let D represent the score matrix of size $(l_1+1) \times (l_2+1)$. The 0^{th} row and 0^{th} column of D are initialized to infinity, except $D(0,0)$, which is initialized to 0. The rest of the score matrix, D, is completed using the following recursion of Equation 7:

$$D(i, j) = e(i, j) + \min\{D(i-1, j), D(i-1, j-1), D(i, j-1)\}, \tag{7}$$

where $1 \leq i \leq l_1$ and $1 \leq j \leq l_2$. The optimal warp path is then traced back from $D(l_1, l_2)$ to $D(0,0)$. The distance measure between the two substrings is then given by $D(l_1, l_2)$ normalized by the length of the optimal warping path.

3.2 Parameter Estimation

In order to extract the common signs from a given set of sign language sentences, we need to compute θ_i for each of the sentences sequentially. Gibbs sampling (Casella and George, 1992) is a Markov Chain Monte Carlo approach (Gilks et al., 1998) that allows us to sample the conditional probability density $f(\theta_i|\theta_{(i)})$ for all the sequences sequentially and then iterate the whole process

until convergence. Gibbs sampling results in a global optimum, but its convergence is very slow. The burn-in period is typically thousands of iterations. Therefore, we perform the optimization using iterated conditional modes (ICM), first proposed by Besag (1986). ICM has much faster convergence, but it is also known to be heavily dependent on the initialization. We address this limitation by running the optimization multiple times with different initializations and choosing the most frequently occurring solution as the final solution.

Algorithm 1: Iterated Conditional Modes($\{a_1^0, w_1^0, \cdots, a_n^0, w_n^0\}$)

comment: Choose $\{a_1, w_1, \cdots, a_n, w_n\}$ that maximizes the distribution $p(a_1, w_1, \cdots, a_n, w_n)$

comment: Initialization:

$\theta_0 \leftarrow \{a_1^0, w_1^0, \cdots, a_n^0, w_n^0\}$

repeat

- **for** $i \leftarrow 0$ **to** n
  - **do**
    - **comment:** Jointly sample a_i, w_i. L_i is the length of sequence S_i
    - **for** $w_i \leftarrow A$ **to** B
      - **do**
        - **for** $a_i \leftarrow 0$ **to** $L_i - w_i + 1$
          - **do** $g(a_i, w_i|\theta_{(a_i,w_i)}) \leftarrow \exp\left(-\beta \sum_{k=1}^{n} d(\vec{s}_{a_i}^{w_i}, \vec{s}_{a_k}^{w_k})\right)$
    - **comment:** Normalize
    - **for** $w_i \leftarrow A$ **to** B
      - **do**
        - **for** $a_i \leftarrow 0$ **to** $L_i - w_i + 1$
          - **do** $f(a_i, w_i|\theta_{(a_i,w_i)}) \leftarrow \frac{g(a_i,w_i|\theta_{(a_i,w_i)})}{\sum_{a_i,w_i} g(a_i,w_i|\theta_{(a_i,w_i)})}$
    - $a_i, w_i \leftarrow$ ARG MAX $(f(a_i, w_i|\theta_{(a_i,w_i)}))$

until CHANGE IN PARAMETERS$(\{a_1, w_1, \cdots, a_n, w_n\}) == 0$

Algorithm 1 outlines the process of ICM to extract the common patterns or signemes from a set of sentences with a given initial parameter vector. We aim to select the set of parameters that maximizes the probability $p(\theta)$ or $p(a_1, w_1, \cdots, a_n, w_n)$. We do that by estimating each of the parameters $a_1, w_1, \cdots a_n, w_n$ in a sequential manner. Since we expect the starting location and width of a subsequence representing the common sign to be strongly correlated, we estimate a_i and w_i jointly. First we compute $g(\theta_i|\theta_{(i)})$ that is, $g(a_i, w_i|\theta_{(a_i,w_i)})$ from which we compute the conditional density functions $f(\theta_i|\theta_{(i)})$ that is, $f(a_i, w_i|\theta_{(a_i,w_i)})$. Note that it involves a summation over a_i and w_i only, which involves much less computation than that required for computing $p(\theta)$ which involves a summation over $a_1, w_1, \cdots a_n, w_n$. The values for a_i and w_i are updated with those that maximize the conditional density $f(\theta_i|\theta_{(i)})$. The process is carried out sequentially for $i = 1$ to n, and then repeated iteratively till the values of the parameter vector $\{a_1, w_1, a_2, w_2, \cdots a_n, w_n\}$ do not change any more.

Figure 5 depicts the sampling process for a single iteration, r. Note the conditional and sequential nature of sampling from various sentences within the single iteration. In Figure 6, we show an example of how the conditional probability $f(\theta_{a_i,w_i}|\theta_{(a_i,w_i)})$ changes for the first seven sentences from a given set of fourteen video sentences containing a common sign 'DEPART'. The vertical axis

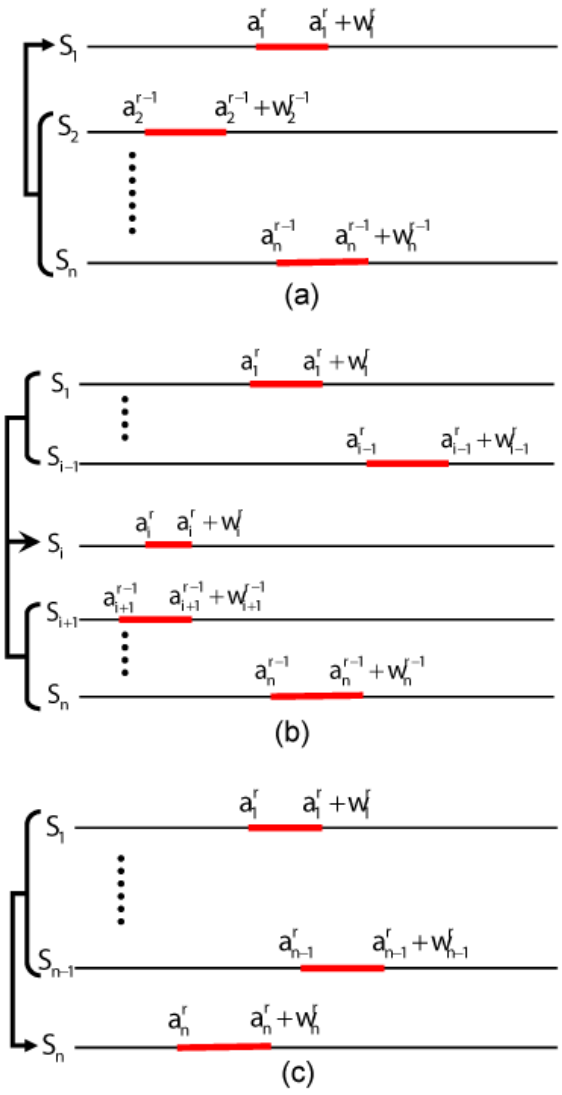


Figure 5: Sequential update of the parameter values using ICM. (a), (b) and (c) respectively show the parameter updates in the first sentence, the i^{th} and the n^{th} sentences. In the r^{th} iteration, the parameters of the common sign in i^{th} sentence is computed based on the parameter values of the previous $(i-1)$ sentences obtained in the same iteration, and those of the $(i+1)^{th}$ to n^{th} sentences obtained in the previous, that is, the $(r-1)^{th}$ iteration.

in the probabilities represents the starting locations and the horizontal axis represents the possible widths. The brighter regions represent a higher probability value. Note that the probabilities are spread out in the first iteration for each sentence and it slowly converges to a fixed starting location for each of them. They remain more spread out across the horizontal (width) axis because we vary

the width only in a small range of A to B for each sign, that is decided based on the amount of motion present in the sign.

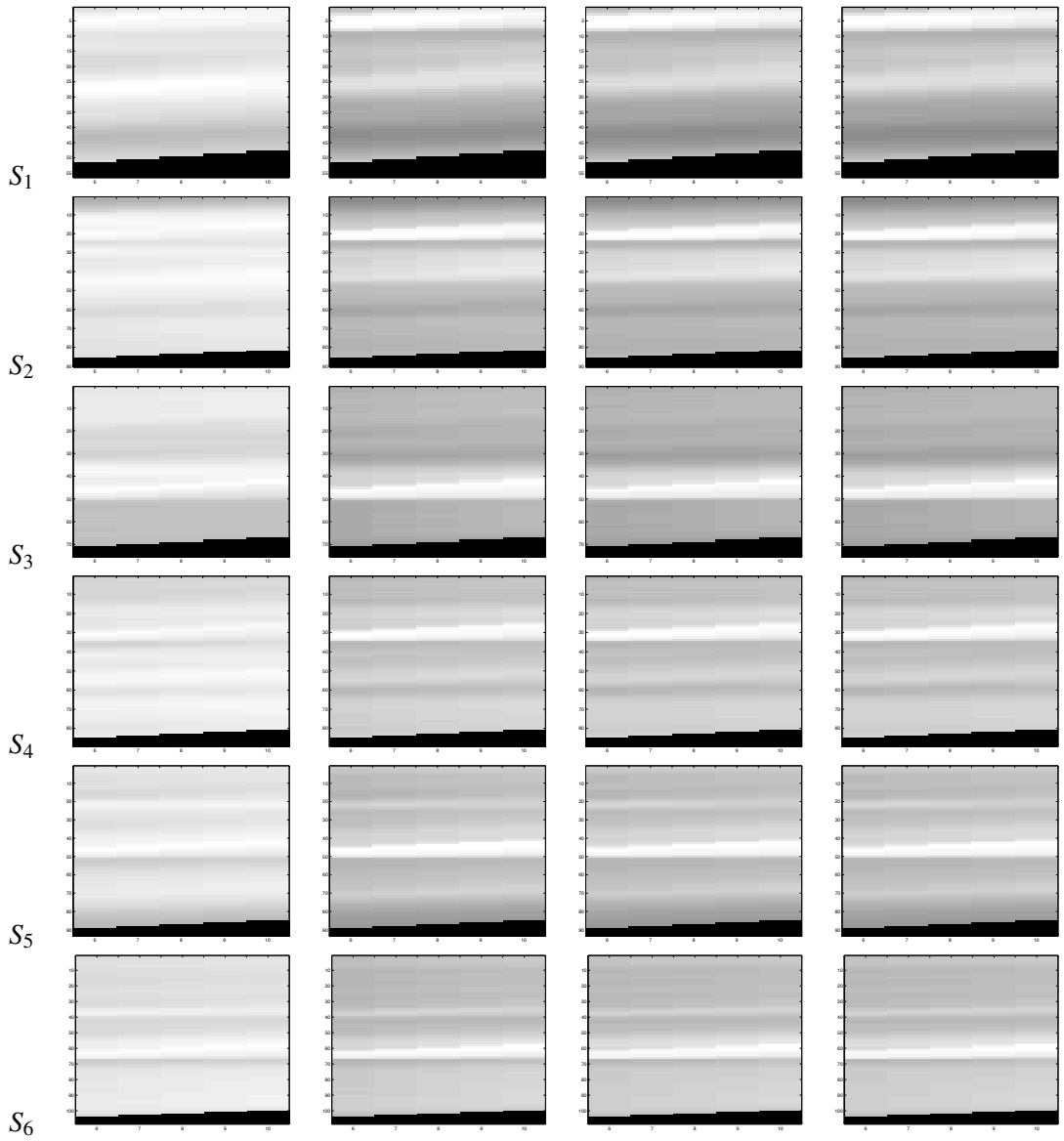


Figure 6: Convergence of the conditional probability density $f(\theta_i|\theta_{(i)})$ for sentences $S_1 ... S_6$ from a given set of sentences $S_1 ... S_{14}$. The brighter regions represent a higher probability value. The vertical axis in the probabilities represents the starting locations and the horizontal axis represents the possible widths. Note that the probabilities are spread out in the first iteration and it slowly converges to a particular starting location. They are still spread across the horizontal (width) axis because we vary the width only in a small range that is decided based on the amount of motion present in the sign.

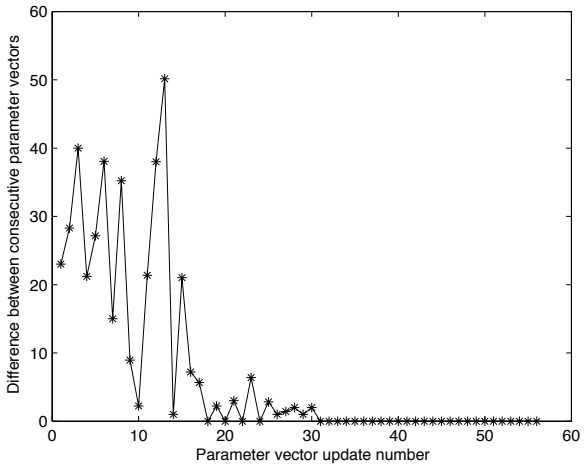


Figure 7: Convergence of values of the parameter set. The above plot shows the norm of the difference between two consecutive parameter vectors representing the set of starting points and widths of the common subsequence in the given set of sequences. It shows the typical convergence with a given initialization vector. ICM is repeated with multiple initializations and the most frequently occurring solution is considered as the final solution.

Figure 7 plots the typical convergence of the parameter values in a single ICM run. It plots the norm of difference between consecutive parameter vectors versus the parameter vector update count, which is incremented each time a parameter is sampled or selected from the probability distribution $f(\theta_i|\theta_{(i)})$. It shows that ICM converges in less than $56/14 = 4$ iterations. This, in turn, also indicates the local nature of the optimization achieved with ICM. The initialization is very important in this case. In the next subsection, we describe how we address this problem.

3.3 Sampling Starting Points For ICM

In order to address the local convergence nature of ICM, we adopt a uniform random sampling-based approach. We start by randomly assigning values to the parameter vector θ. The width w_i^0 is obtained by sampling a width value based on uniform random distribution from the set of all possible widths in a given range $[A,B]$. The value for a_i^0 is obtained by sampling a starting point based on uniform random distribution from the set of all possible starting points in the i^{th} sequence, that is, from the set $\{1 \cdots (L_i - w_i^0 + 1)\}$.

Different initial parameter vectors are obtained by independently sampling the sentences multiple times. ICM is run using each initial parameter vector generated and the most common solution is considered as the final solution. The uniform sampling of the frames in the sentences for selecting the starting locations ensures the whole parameter space is covered uniformly. The number of times we sample the initial parameter vector and run the ICM algorithm decides how densely we cover the whole parameter space. We run it the number of times equal to the average number of frames in each sentence from the given set of sentences for extracting the sign. One could choose to run a multiple of the average number of times as well, but we found the average number to be

sufficient to show the stability of the solution in our experiments. Algorithm 2 presents the process as a pseudocode.

Algorithm 2: Extract Signemes($L_1, \cdots L_n, A, B$)

comment: Generate multiple initialization vectors and call ICM with each of them.

$N = \text{MEAN}(L_1, L_2, \cdots, L_n)$
for $j \leftarrow 1$ **to** N
do $\begin{cases} \textbf{for } i \leftarrow 1 \textbf{ to } n \\ \quad \textbf{do } \begin{cases} w_i^0 = \text{UNIFORM}(A \cdots B) \\ a_i^0 = \text{UNIFORM}(1 \cdots L_i - w_i^0 + 1) \end{cases} \\ \{a_1^j, w_1^j, \cdots, a_n^j, w_n^j\} = \text{ITERATED CONDITIONAL MODES}(a_1^0, w_1^0, \\ \cdots, a_n^0, w_n^0) \end{cases}$
for $i \leftarrow 1$ **to** n
do $\begin{cases} \textbf{comment:} \text{ Assign most frequently occurring value as the final value for each parameter.} \\ w_i = \text{MODE}(w_i^j) \\ a_i = \text{MODE}(a_i^j) \end{cases}$

For extracting the sign 'DEPART' from 14 sentences, we had 89 frames per sentence on an average. Hence we ran 89 different ICM runs for extracting the common subsequence representing 'DEPART'. Figure 8 shows the plots of histograms of start and end location of the sign in each of the 14 sentences from the 89 runs. It should be noted that in most of the sentences, more than 50% of the total number of runs result in the same solution.

4. Experiments And Results

In this section, we present visual and quantitative results of our approach for extracting signemes from video sequences representing sentences from American Sign Language. We first describe the data set used then present the results of the automatic common pattern extraction.

4.1 Data Set

Our data set consists of 155 American Sign Language (ASL) video sequences organized into 12 groups (collections) based on the vocabulary (word that pervades the sentences of the group). For instance, the 'DEPART' group is comprised of all the sentences containing the word 'DEPART', the 'PASSPORT' group is comprised of all the sentences containing the word 'PASSPORT' and so on. The breakdown of these 'pure' groups and the number of sentences (sequences) in each are as follows.

- DEPART - 14 sentences
- BAGGAGE - 14 sentences
- CANT - 14 sentences

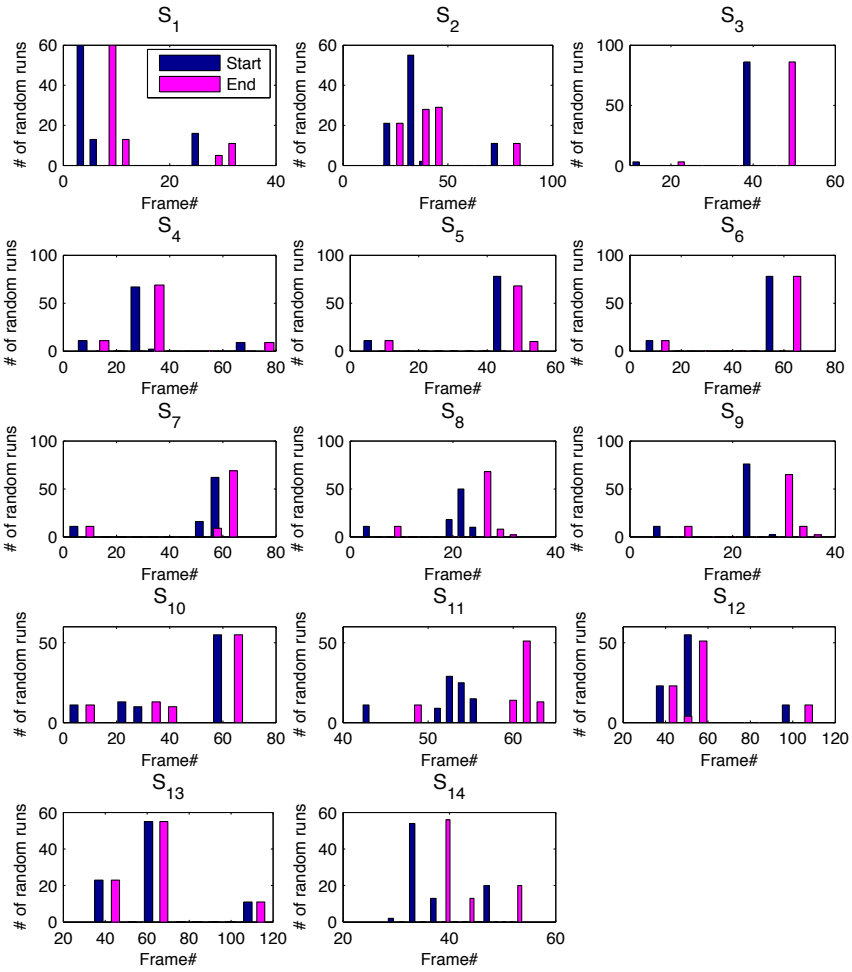


Figure 8: Histograms showing the start and end locations of signs extracted from 14 different sentences using multiple ICM runs. The initial parameter vector for each ICM run was chosen independently using uniform random sampling. As it can be seen the start and end points found by most of the runs converge to the same solution (denoted by single high bars in most of sentences). The legend shown in the plot for the first sentence, S_1, holds for other sentences as well.

- BUY - 11 sentences
- SECURITY - 16 sentences
- HAVE - 6 sentences
- MOVE - 11 sentences
- TIME - 14 sentences

- FUTURE - 12 sentences
- TABLE - 13 sentences
- PASSPORT - 14 sentences
- TICKET - 16 sentences

This data set was used to extract 12 common subsequences when we searched for the first most common sign, and 24 common subsequences when we searched for the second most common sign. We also organized the video sequences into 10 groups by combining two 'pure' groups of sentences as described above. This was used to investigate the power of our framework for selecting the common sequences in a 'mixed' collection. The breakdown of these 'mixed' groups and the number of sentences in each are as follows:

- DEPART (14 sentences) + BAGGAGE (14 sentences)
- CANT (14 sentences) + BUY (11 sentences)
- TIME (14 sentences) + TABLE (13 sentences)
- PASSPORT (14 sentences) + TICKET (16 sentences)
- SECURITY (16 sentences) + FUTURE (12 sentences)
- MOVE (11 sentences) + HAVE (6 sentences)
- BUY (11 sentences) + TABLE (13 sentences)
- DEPART (14 sentences) + FUTURE (12 sentences)
- BAGGAGE (14 sentences) + TICKET (16 sentences)
- SECURITY (16 sentences) + PASSPORT (14 sentences)

All of the signs were performed by the same signer with plain clothing and background. The video sequences were captured at 25 frames per second with a frame resolution of 490×370.

4.2 Common Pattern Extraction Results

In this section, we present the results of our method for extracting common patterns from sign language sentences. We first present results for extracting the single most common sign and multiple common signs from the 'pure' sentence groups, followed by results for the most common patterns from the 'mixed' groups.

4.2.1 EXTRACTING THE MOST COMMON PATTERN

We perform extraction of the most common patterns from the 'pure' sentence groups. We possess a priori knowledge of the most common word due to the organization of the sentence groups. However, our goal is to extract the most common sequences automatically. As an example, Figure 9 depicts the result of extraction of the sign 'DEPART' from 14 video sequences. It plots the SoRD first dimension coefficients of the frames vs. the frame number for each sentence. The highlighted

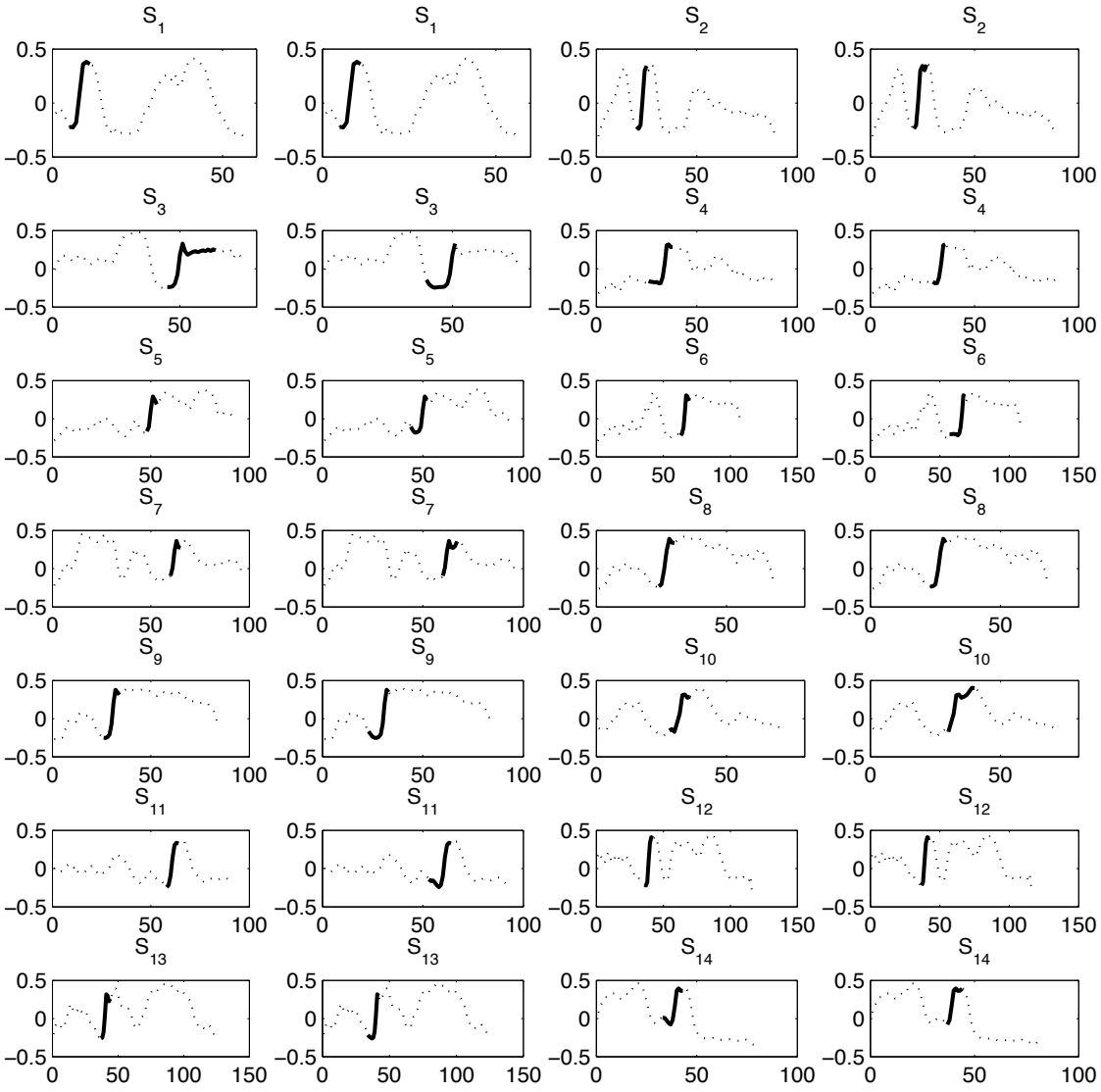
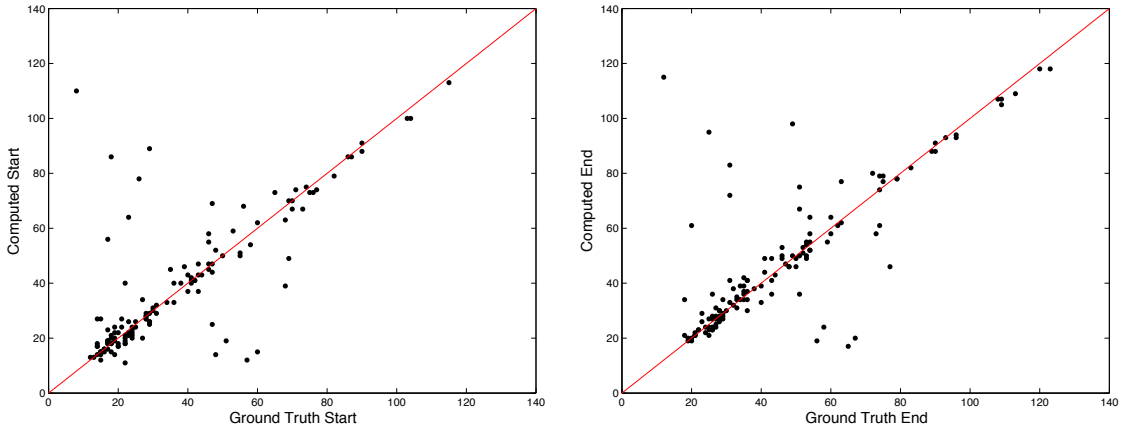


Figure 9: The first dimension of the video sequences containing a common sign 'DEPART'. The sequences are indicated by the dotted curves and the solid lines on each of them indicate the common pattern or signeme. The odd columns represent the ground truth and the even columns show the results.

portions represent the signeme. The odd columns show the ground truth and the even columns show the corresponding results. As can be seen, the extracted patterns and the corresponding ground truth patterns are quite similar, except for a few frames at the beginning and end of the some of the patterns. Note that since we deal with continuous video sequences, a difference of one or two frames between the ground truth and the extracted pattern is not considered a problem.



(a) Video Start Point Estimation

(b) Video End Point Estimation

Figure 10: Extraction of the most common patterns or signemes from the 'pure' sentence groups. The closer the points are to the diagonal, the closer the result is to the ground truth.

Figure 10(a) shows the scatter plot of the ground truth start positions vs. the estimated start positions of the pattern extracted from each of the 155 sentences in the video data set. Figure 10(b) shows the corresponding scatter plot for the end position of the patterns in the sentences. As can be seen most of the points in the scatter plots lie along the diagonal. This indicates that very few of the extracted patterns are wrong. Incorrect results correspond to the points positioned far from the diagonal. Figures 11 and 12 show one instance of the signeme extracted from group of sentences.

4.2.2 EXTRACTING MULTIPLE COMMON SIGNS

In this section we present some visual results for the extraction of the two most common signs from the 'pure' groups of sentences. We focused on extracting only two signs because the shortest ASL sentence contained two signs. Figure 13 shows the results for the two most common signs extracted from the sentence 'BAGGAGE THERE NOT MINE THERE'. The extracted subsequences correspond to the ASL words 'BAGGAGE' and 'MINE'. Consequently, the word 'BAGGAGE' appears in all the 14 sentences of the group, whereas the word 'MINE' (or 'MY') shows up in 11 sentences coinciding with what was expected. Similarly, Figure 14 shows the results for the two most common signs extracted from the sentence 'MY PASSPORT THERE STILL GOOD THERE'. The extracted subsequences correspond to the ASL words 'MY' and 'PASSPORT'. The word 'MY' appears in all the 11 sentences of the group, whereas the word 'PASSPORT' appears in all 14 sentences. These results are encouraging.

4.2.3 EXTRACTING THE MOST COMMON PATTERNS FROM MIXED SENTENCES

We perform extraction of the most common patterns from the collection of 'mixed' sentences as outlined in Section 4.1. Figure 15(a) shows the scatter plot of the ground truth start positions vs. the estimated start positions of the pattern extracted from each of the sentences. Similarly, Figure 15(b)

(a) BUY

(b) CANT

(c) DEPART

(d) FUTURE

(e) MOVE

Figure 11: Signemes extracted from sentences

shows the corresponding scatter plot for the end position of the patterns in the sentences. As can be seen, the points are more scattered as compared to the results shown in Figure 10 where the sentences used were known to contain common words. However, this result is still encouraging. A large proportion of the extracted patterns are incorrect, but there are many relatively near the diagonal. This result demonstrates the robustness of our algorithm for finding similarities in the presence of great dissimilarity. We believe that the incorrect patterns extracted are due to the differences in the frame width ranges for the mixed sentence sets. For example, sentences containing the word 'MOVE' were combined with sentences containing the word 'HAVE'. The frame width range for the sign 'HAVE' is between 4 and 6 frames with 4 being the minimum width and 6 being the maximum width. On the other hand, the frame width range for the sign 'MOVE' is between 19 and 27 frames. Combining these width ranges could be done using an average of the two or by selecting the minimum and maximum values between the two. However, these methods produced similar results. The correct combination of these range widths is a priority for future work.

(f) PASSPORT

(g) SECURITY

(h) TICKET

(i) TIME

(j) TABLE

Figure 12: Signemes extracted from sentences

4.3 Sign Localization

We used the extracted signemes to localize or spot signs in test sentences. The same process that is used for training sign models is used for sign localization. However, rather than randomly assigning initial parameter values, we use the parameters learned. We tested with 12 test sentences from the 'pure' group specified in Section 4.1 and their lengths varied from 4 to 12 signs. These test sentences were not used during training. The set of points representing the signeme were matched with the segments of the SoRD points from the test sentences to find the segment with the minimum matching score, which would represent the sign in the test sentence. The SoRD points of the signeme retrieved from the test sentence are mapped to their nearest frames and compared with the ground truth frame series representing the sign in the sentence. Localization performance is characterized as follows. Let a_1 and b_1 denote the start and end frame numbers of the underlying ground truth sign in the test sentence, and a_2 and b_2 denote the start and end frame numbers of the subsequence retrieved as the

(a) Frames corresponding to the word 'BAGGAGE'

(b) Frames corresponding to the word 'MINE'

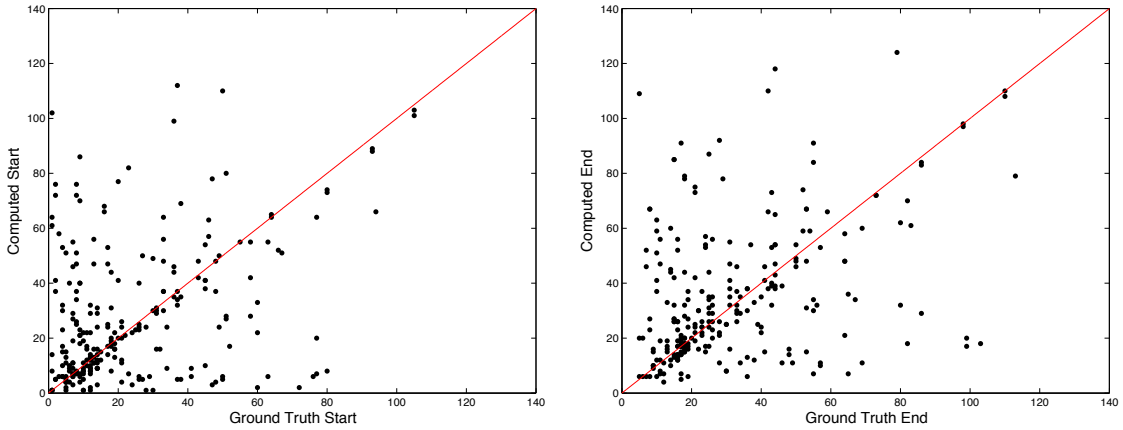
Figure 13: Extraction of the two most common patterns or signemes from the sentence 'BAGGAGE THERE NOT MINE THERE'.

(a) Frames corresponding to the word 'MY'

(b) Frames corresponding to the word 'PASSPORT'

Figure 14: Extraction of the two most common patterns or signemes from the sentence 'MY PASSPORT THERE STILL GOOD THERE'.

signeme for the test sentence. We calculate the precision and recall values of each test sentence as $\frac{m}{a_2-a_1+1}$ and $\frac{m}{b_2-a_2+1}$ respectively where m is the number of overlapping frames. Table 2 displays the results acquired. The 'Baggage', 'Cant', 'Have', and 'Table' test sequences were failure cases where there was no overlap between the extracted model frames and the localization frames (see Figure 16). Notice that the localization results heavily depend on the extracted signeme models. For a visual representation of this information, we define the Start Offset, ΔS, and End Offset, ΔE, as $\Delta S = a_1 - a_2$ and $\Delta E = b_1 - b_2$. The plot of the Start Offset vs. the End Offset is shown in Figure 16. Ideally, both the offsets should be zero. The points for different signs are scattered in the four quadrants depending on the nature of the overlap between the ground truth sign and the retrieved



(a) Video Start Point Estimation (b) Video End Point Estimation

Figure 15: Extraction of the most common patterns or signemes from the 'mixed' sentence groups. The closer the points are to the diagonal, the closer the result is to the ground truth.

Test Group	Precision	Recall
Buy	1.0	0.70
Depart	1.0	0.64
Future	0.71	0.756
Move	1.0	0.60
Passport	1.0	0.47
Security	0.57	0.67
Ticket	1.0	0.58
Time	0.63	1.0

Table 2: Localization Performance

signeme. Each point in the plot corresponds to a separate test sign. Its distance from the origin indicates the localizing quality of the signeme in its test sentence. The closer it is to the origin, the better the quality.

5. Conclusion And Future Work

We presented a novel algorithm to extract signemes, that is, the common pattern representing a sign, from multiple long video sequences of American Sign Language (ASL). A signeme is a part of the sign that is robust to the variations of the adjacent signs and the associated movement epenthesis. We first represent each sequence as a series of points in a low dimensional space of relational distributions, and then use a probabilistic framework to locate the signemes in each sequence concurrently. We use iterative conditional modes (ICM) to sample the parameters, that is, the starting location and width of the signemes in each sentence in a sequential manner. We show results on ASL video sequences that do not involve using any magnetic trackers or gloves for extracting the

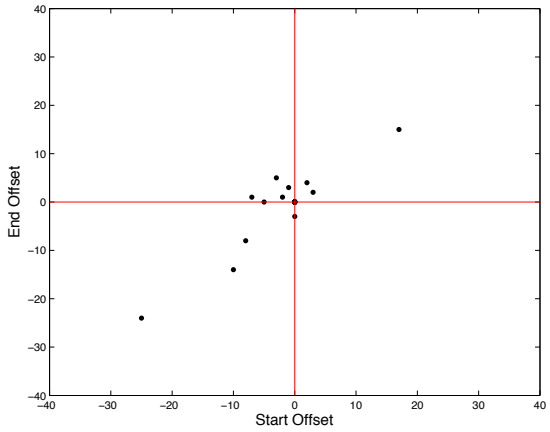


Figure 16: Start Offset vs. End Offset of Localized Signs

most common signs. The extracted signemes demonstrate that our approach is robust to some extent to the variations produced within a sign due to different contexts.

The approach in this paper can be used to speed up training set generation for ASL algorithms by drastically reducing the manual aspect of the process. Rather than manually demarcating signs in continuous sentences, which for our work took an expert approximately 5 minutes, we would just need instances of sentences containing the sign whose model is sought and based on our experiments this can be generated in approximately 2 minutes. Another contribution of this work is an empirically derived robust representation of the sign that is stable with respect to the variations due to neighboring signs and sentence context. These stable representations could be useful for detection of signs and gestures in extended gesture sequences.

There are some ways we can advance the work in this paper. One issue is the precision of the features used for representing the video sequences. Relational distributions when used as fixed size histograms perform well for discriminating global motion. However, optimizing the bin size of the histograms to the required precision might improve the accuracy. Additionally, we plan to extend our work to address the challenge of handling the large variations encountered when automatically recognizing signemes across different signers. Also, the algorithm is dependent to a large extent on the distance measure and conventional dynamic time warping cannot deal with the amplitude variations in the signs, which are very common across signers. We plan to work on a variation of dynamic time warping that is robust to amplitude differences between various instances of signs.

Acknowledgments

This work was supported in part by funds from University of South Florida's College of Engineering Interdisciplinary Scholarship Program and the National Science Foundation under ITR grant IIS 0312993.

References

O. Al-Jarrah and A. Halawani. Recognition of gestures in Arabic Sign Language using neuro-fuzzy systems. *Artificial Intelligence*, 133:117–138, 2001.

V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. Boostmap: A method for efficient approximate similarity rankings. *IEEE Conference On Computer Vision And Pattern Recognition*, pages 268–275, 2004.

T Bailey and C. Elkan. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, 21:51–80, 1995.

B. Bauer and H. Hienz. Relevant features for video-based continuous sign language recognition. In *Automatic Face And Gesture Recognition*, pages 440–445, 2000.

B. Bauer and K. F. Kraiss. Video-based sign recognition using self-organizing subunits. In *International Conference On Pattern Recognition*, volume 2, pages 434–437, 2002.

J. Besag. On the statistical analysis of dirty pictures. *Journal Of The Royal Statistical Society*, pages 259–302, 1986.

R. Bowden, D. Windridge, T. Kadir, A. Zisserman, and M. Brady. A linguistic feature vector for the visual interpretation of sign language. In *European Conference On Computer Vision*, volume 1, pages 390–401, 2004.

P. Buehler, A. Zisserman, and M. Everingham. Learning sign language by watching tv (using weakly aligned subtitles). In *IEEE Conference On Computer Vision And Pattern Recognition*, pages 2961–2968, June 2009.

G. Casella and E.I. George. Explaining the Gibbs sampler. *The American Statistician*, 46:167–174, 1992.

S. Chib and E. Greenberg. Understanding the Metropolis–Hastings algorithm. *The American Statistician*, 49:327–335, 1995.

B. Chiu, E. Keogh, and S. Lonardi. Probabilistic discovery of time series motifs. *ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*, pages 493–498, 2003.

H. Cooper and R. Bowden. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *IEEE Conference On Computer Vision And Pattern Recognition*, pages 2568–2574, June 2009.

Y. Cui and J. Weng. Appearance-based hand sign recognition from intensity image sequences. *Computer Vision And Image Understanding*, 78:157–176, 2000.

A. Denton. Kernel-density-based clustering of time series subsequences using a continuous random-walk noise model. *International Conference On Data Mining*, 2005.

K.G. Derpanis, R.R. Wildes, and J.K. Tsotsos. Hand gesture recognition within a linguistics-based framework. *European Conference On Computer Vision*, pages 282–296, 2004.

F. Duchne, C. Garbay, and V. Rialle. Learning recurrent behaviors from heterogeneous multivariate time-series. *Artificial Intelligence In Medicine*, (1):25–47, 2007.

G. Fang, X. Gao, W. Gao, and Y. Chen. A novel approach to automatically extracting basic units from Chinese Sign Language. *International Conference On Pattern Recognition*, 4:454–457, 2004.

A. Farhadi, D.A. Forsyth, and R. White. Transfer learning in sign language. In *Computer Vision And Pattern Recognition*, pages 1–8, 2007.

W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov chain Monte Carlo in practice*. Chapman and Hall, 1998.

J. Han, G. Awad, and A. Sutherland. Modelling and segmenting subunits for sign language recognition based on hand motion analysis. *Pattern Recognition Letters*, 30(6):623–633, 2009.

C.E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 1993.

S.K. Liddell and R.E. Johnson. American Sign Language: The phonological base. *Sign Language Studies*, pages 195–277, 1989.

J. Ma, W. Gao, C. Wang, and J. Wu. A continuous Chinese Sign Language recognition system. *International Conference On Automatic Face And Gesture Recognition*, pages 428–433, 2000.

D. Minnen, C.L. Isbell, I. Essa, and T. Starner. Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. *Conference On Artificial Intelligence*, 2007.

S. Nayak, S. Sarkar, and B. Loeding. Unsupervised modeling of signs embedded in continuous sentences. *IEEE Workshop On Vision For Human-Computer Interaction*, 2005.

S. Nayak, S. Sarkar, and B. Loeding. Automated extraction of signs from continuous sign language sentences using iterated conditional modes. In *IEEE Conference On Computer Vision And Pattern Recognition*, pages 2583–2590, June 2009a.

S. Nayak, S. Sarkar, and B. Loeding. Distribution-based dimensionality reduction applied to articulated motion recognition. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 31 (5):795–810, May 2009b.

T. Oates. PERUSE: An unsupervised algorithm for finding recurring patterns in time series. *International Conference On Data Mining*, pages 330–337, 2002.

S.C.W. Ong and S. Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 27:873–891, June 2005.

M. Oszust and M. Wysocki. Determining subunits for sign language recognition by evolutionary cluster-based segmentation of time series. In *Artifical Intelligence And Soft Computing*, volume 6114 of *Lecture Notes In Computer Science*, pages 189–196. Springer Berlin / Heidelberg, 2010.

P. A. Pevzner and S. H. Sze. Combinatorial approaches to finding subtle signals in DNA sequences. *International Conference On Intelligent Systems For Molecular Biology*, pages 269–278, 2000.

S.L. Phung, A. Bouzerdoum, and D. Chai. Skin segmentation using color pixel classification: Analysis and comparison. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 27: 148–154, January 2005.

I. Rigoutsos and A. Floratos. Combinatorial pattern discovery in biological sequences: The Teiresias algorithm. *Bioinformatics*, 14:55–67, 1998.

C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. New York: Springer-Verlag, 2004.

T. Starner and A. Pentland. Real-time American Sign Language recognition from video using Hidden Markov Models. *Computational Imaging And Vision*, 9:227–244, 1997.

T. Starner, J. Weaver, and A. Pentland. Real-time American Sign Language recognition using desk and wearable computer based video. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 20(12):1371–1375, 1998.

Yoshiki Tanaka, Kazuhisa Iwamoto, and Kuniaki Uehara. Discovery of time-series motif from multidimensional data based on MDL principle. *Machine Learning*, 58(2-3):269–300, 2005.

I.R. Vega and S. Sarkar. Statistical motion model based on the change of feature relationships: Human gait-based recognition. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 25:1323–1328, October 2003.

C. Vogler and D. Metaxas. Parallel Hidden Markov Models for American Sign Language Recognition. *International Conference On Computer Vision*, 1:116–122, 1999.

C. Vogler and D. Metaxas. A framework of recognizing the simultaneous aspects of American Sign Language. *Computer Vision And Image Understanding*, 81:358–384, 2001.

C. Wang, W. Gao, and S. Shan. An approach based on phonemes to large vocabulary Chinese Sign Language recognition. *International Conference On Automatic Face And Gesture Recognition*, pages 393–398, 2002.

Q. Wang, X. Chen, L.G. Zhang, C. Wang, and W Gao. Viewpoint invariant sign language recognition. In *Computer Vision And Image Understanding*, volume 108, pages 87–97, 2007.

M.H. Yang, N. Ahuja, and M. Tabb. Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 24: 1061–1074, 2002.

R. Yang, S. Sarkar, and B. Loeding. Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 32(3):462–477, March 2010.

Journal of Machine Learning Research 13 (2012) 2617-2654 Submitted 8/11; Revised 5/12; Published 9/12

Static Prediction Games for Adversarial Learning Problems

Michael Brückner MIBRUECK@CS.UNI-POTSDAM.DE
Department of Computer Science
University of Potsdam
August-Bebel-Str. 89
14482 Potsdam, Germany

Christian Kanzow KANZOW@MATHEMATIK.UNI-WUERZBURG.DE
Institute of Mathematics
University of Würzburg
Emil-Fischer-Str. 30
97074 Würzburg, Germany

Tobias Scheffer SCHEFFER@CS.UNI-POTSDAM.DE
Department of Computer Science
University of Potsdam
August-Bebel-Str. 89
14482 Potsdam, Germany

Editor: Nicolò Cesa-Bianchi

Abstract

The standard assumption of identically distributed training and test data is violated when the test data are generated in response to the presence of a predictive model. This becomes apparent, for example, in the context of email spam filtering. Here, email service providers employ spam filters, and spam senders engineer campaign templates to achieve a high rate of successful deliveries despite the filters. We model the interaction between the learner and the data generator as a static game in which the cost functions of the learner and the data generator are not necessarily antagonistic. We identify conditions under which this prediction game has a unique Nash equilibrium and derive algorithms that find the equilibrial prediction model. We derive two instances, the Nash logistic regression and the Nash support vector machine, and empirically explore their properties in a case study on email spam filtering.

Keywords: static prediction games, adversarial classification, Nash equilibrium

1. Introduction

A common assumption on which most learning algorithms are based is that training and test data are governed by identical distributions. However, in a variety of applications, the distribution that governs data at application time may be influenced by an adversary whose interests are in conflict with those of the learner. Consider, for instance, the following three scenarios. In computer and network security, scripts that control attacks are engineered with botnet and intrusion detection systems in mind. Credit card defrauders adapt their unauthorized use of credit cards—in particular, amounts charged per transactions and per day and the type of businesses that amounts are charged from—to avoid triggering alerting mechanisms employed by credit card companies. Email spam senders design message templates that are instantiated by nodes of botnets. These templates are

specifically designed to produce a low spam score with popular spam filters. The domain of email spam filtering will serve as a running example throughout the paper. In all of these applications, the party that creates the predictive model and the adversarial party that generates future data are aware of each other, and factor the possible actions of their opponent into their decisions.

The interaction between learner and data generators can be modeled as a game in which one player controls the predictive model whereas another exercises some control over the process of data generation. The adversary's influence on the generation of the data can be formally modeled as a transformation that is imposed on the distribution that governs the data at training time. The transformed distribution then governs the data at application time. The optimization criterion of either player takes as arguments both the predictive model chosen by the learner and the transformation carried out by the adversary.

Typically, this problem is modeled under the worst-case assumption that the adversary desires to impose the highest possible costs on the learner. This amounts to a zero-sum game in which the loss of one player is the gain of the other. In this setting, both players can maximize their expected outcome by following a minimax strategy. Lanckriet et al. (2002) study the minimax probability machine (MPM). This classifier minimizes the maximal probability of misclassifying new instances for a given mean and covariance matrix of each class. Geometrically, these class means and covariances define two hyper-ellipsoids which are equally scaled such that they intersect; their common tangent is the minimax probabilistic decision hyperplane. Ghaoui et al. (2003) derive a minimax model for input data that are known to lie within some hyper-rectangles around the training instances. Their solution minimizes the worst-case loss over all possible choices of the data in these intervals. Similarly, worst-case solutions to classification games in which the adversary deletes input features (Globerson and Roweis, 2006; Globerson et al., 2009) or performs an arbitrary feature transformation (Teo et al., 2007; Dekel and Shamir, 2008; Dekel et al., 2010) have been studied.

Several applications motivate problem settings in which the goals of the learner and the data generator, while still conflicting, are not necessarily entirely antagonistic. For instance, a defrauder's goal of maximizing the profit made from exploiting phished account information is not the inverse of an email service provider's goal of achieving a high spam recognition rate at close-to-zero false positives. When playing a minimax strategy, one often makes overly pessimistic assumptions about the adversary's behavior and may not necessarily obtain an optimal outcome.

Games in which a leader—typically, the learner—commits to an action first whereas the adversary can react after the leader's action has been disclosed are naturally modeled as a *Stackelberg competition*. This model is appropriate when the follower—the data generator—has full information about the predictive model. This assumption is usually a pessimistic approximation of reality because, for instance, neither email service providers nor credit card companies disclose a comprehensive documentation of their current security measures. Stackelberg equilibria of adversarial classification problems can be identified by solving a bilevel optimization problem (Brückner and Scheffer, 2011).

This paper studies *static* prediction games in which both players act simultaneously; that is, without prior information on their opponent's move. When the optimization criterion of both players depends not only on their own action but also on their opponent's move, then the concept of a player's optimal action is no longer well-defined. Therefore, we resort to the concept of a *Nash equilibrium* of static prediction games. A Nash equilibrium is a pair of actions chosen such that no player benefits from unilaterally selecting a different action. If a game has a unique Nash equi-

librium and is played by rational players that aim at maximizing their optimization criteria, it is reasonable for each player to assume that the opponent will play according to the Nash equilibrium strategy. If one player plays according to the equilibrium strategy, the optimal move for the other player is to play this equilibrium strategy as well. If, however, multiple equilibria exist and the players choose their strategy according to distinct ones, then the resulting combination may be arbitrarily disadvantageous for either player. It is therefore interesting to study whether adversarial prediction games have a unique Nash equilibrium.

Our work builds on an approach that Brückner and Scheffer (2009) developed for finding a Nash equilibrium of a static prediction game. We will discuss a flaw in Theorem 1 of Brückner and Scheffer (2009) and develop a revised version of the theorem that identifies conditions under which a unique Nash equilibrium of a prediction game exists. In addition to the inexact linesearch approach to finding the equilibrium that Brückner and Scheffer (2009) develop, we will follow a modified extragradient approach and develop Nash logistic regression and the Nash support vector machine. This paper also develops a kernelized version of these methods. An extended empirical evaluation explores the applicability of the Nash instances in the context of email spam filtering. We empirically verify the assumptions made in the modeling process and compare the performance of Nash instances with baseline methods on several email corpora including a corpus from an email service provider.

The rest of this paper is organized as follows. Section 2 introduces the problem setting. We formalize the Nash prediction game and study conditions under which a unique Nash equilibrium exists in Section 3. Section 4 develops strategies for identifying equilibrial prediction models, and in Section 5, we detail on two instances of the Nash prediction game. In Section 6, we report on experiments on email spam filtering; Section 7 concludes.

2. Problem Setting

We study static prediction games between two players: The *learner* ($v=-1$) and an adversary, the *data generator* ($v=+1$). In our running example of email spam filtering, we study the competition between recipient and senders, not competition among senders. Therefore, $v=-1$ refers to the recipient whereas $v=+1$ models the entirety of all legitimate and abusive email senders as a single, amalgamated player.

At *training time*, the data generator $v=+1$ produces a sample $D=\{(x_i,y_i)\}_{i=1}^n$ of n training instances $x_i \in \mathcal{X}$ with corresponding class labels $y_i \in \mathcal{Y}=\{-1,+1\}$. These object-class pairs are drawn according to a training distribution with density function $p(x,y)$. By contrast, *at application time* the data generator produces object-class pairs according to some test distribution with density $\dot{p}(x,y)$ which may differ from $p(x,y)$.

The task of the learner $v=-1$ is to select the parameters $\mathbf{w} \in \mathcal{W} \subset \mathbb{R}^m$ of a predictive model $h(x)=\operatorname{sign} f_{\mathbf{w}}(x)$ implemented in terms of a generalized linear decision function $f_{\mathbf{w}}: \mathcal{X} \to \mathbb{R}$ with $f_{\mathbf{w}}(x)=\mathbf{w}^{\mathsf{T}}\phi(x)$ and feature mapping $\phi: \mathcal{X} \to \mathbb{R}^m$. The learner's theoretical *costs* at application time are given by

$$\theta_{-1}(\mathbf{w},\dot{p})=\sum_{\mathcal{Y}}\int_{\mathcal{X}} c_{-1}(x,y)\ell_{-1}(f_{\mathbf{w}}(x),y)\dot{p}(x,y)\mathrm{d}x,$$

where weighting function $c_{-1}: \mathcal{X}\times\mathcal{Y} \to \mathbb{R}$ and loss function $\ell_{-1}: \mathbb{R}\times\mathcal{Y} \to \mathbb{R}$ compose the *weighted loss* $c_{-1}(x,y)\ell_{-1}(f_{\mathbf{w}}(x),y)$ that the learner incurs when the predictive model classifies

instance x as $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ while the true label is y. The positive class- and instance-specific weighting factors $c_{-1}(x,y)$ with $\mathbb{E}_{\mathbf{X},\mathbf{Y}}[c_{-1}(x,y)] = 1$ specify the importance of minimizing the loss $\ell_{-1}(f_{\mathbf{w}}(x),y)$ for the corresponding object-class pair (x,y). For instance, in spam filtering, the correct classification of non-spam messages can be business-critical for email service providers while failing to detect spam messages runs up processing and storage costs, depending on the size of the message.

The data generator $v = +1$ can modify the data generation process at application time. In practice, spam senders update their campaign templates which are disseminated to the nodes of botnets. Formally, the data generator transforms the training distribution with density p to the test distribution with density $\dot{p}$. The data generator incurs *transformation costs* by modifying the data generation process which is quantified by $\Omega_{+1}(p,\dot{p})$. This term acts as a regularizer on the transformation and may implicitly constrain the possible difference between the distributions at training and application time, depending on the nature of the application that is to be modeled. For instance, the email sender may not be allowed to alter the training distribution for non-spam messages, or to modify the nature of the messages by changing the label from *spam* to *non-spam* or vice versa. Additionally, changing the training distribution for spam messages may incur costs depending on the extent of distortion inflicted on the informational payload. The theoretical *costs* of the data generator at application time are the sum of the expected prediction costs and the transformation costs,

$$\theta_{+1}(\mathbf{w},\dot{p}) = \sum_{\mathcal{Y}} \int_{\mathcal{X}} c_{+1}(x,y)\ell_{+1}(f_{\mathbf{w}}(x),y)\dot{p}(x,y)\mathrm{d}x + \Omega_{+1}(p,\dot{p}),$$

where, in analogy to the learner's costs, $c_{+1}(x,y)\ell_{+1}(f_{\mathbf{w}}(x),y)$ quantifies the weighted loss that the data generator incurs when instance x is labeled as $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ while the true label is y. The weighting factors $c_{+1}(x,y)$ with $\mathbb{E}_{\mathbf{X},\mathbf{Y}}[c_{+1}(x,y)] = 1$ express the significance of (x,y) from the perspective of the data generator. In our example scenario, this reflects that costs of correctly or incorrectly classified instances may vary greatly across different physical senders that are aggregated into the amalgamated player.

Since the theoretical costs of both players depend on the test distribution, they can, for all practical purposes, not be calculated. Hence, we focus on a regularized, empirical counterpart of the theoretical costs based on the training sample D. The empirical counterpart $\hat{\Omega}_{+1}(D,\dot{D})$ of the data generator's regularizer $\Omega_{+1}(p,\dot{p})$ penalizes the divergence between training sample $D = \{(x_i,y_i)\}_{i=1}^n$ and a perturbated training sample $\dot{D} = \{(\dot{x}_i,y_i)\}_{i=1}^n$ that would be the outcome of applying the transformation that translates p into $\dot{p}$ to sample D. The learner's cost function, instead of integrating over $\dot{p}$, sums over the elements of the perturbated training sample $\dot{D}$. The players' empirical cost functions can still only be evaluated after the learner has committed to parameters $\mathbf{w}$ and the data generator to a transformation. However this transformation needs only be represented in terms of the effects that it will have on the training sample D. The transformed training sample $\dot{D}$ must not be mistaken for test data; test data are generated under $\dot{p}$ at application time after the players have committed to their actions.

The empirical costs incurred by the predictive model $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ with parameters $\mathbf{w}$ and the shift from p to $\dot{p}$ amount to

$$\hat{\theta}_{-1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{-1,i} \ell_{-1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{-1} \hat{\Omega}_{-1}(\mathbf{w}), \tag{1}$$

$$\hat{\theta}_{+1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{+1,i} \ell_{+1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{+1} \hat{\Omega}_{+1}(D, \dot{D}), \tag{2}$$

where we have replaced the weighting terms $\frac{1}{n} c_v(\dot{x}_i, y_i)$ by constant cost factors $c_{v,i} > 0$ with $\sum_i c_{v,i} = 1$. The learner's regularizer $\hat{\Omega}_{-1}(\mathbf{w})$ in (1) accounts for the fact that $\dot{D}$ does not constitute the test data itself, but is merely a training sample transformed to reflect the test distribution and then used to learn the model parameters $\mathbf{w}$. The trade-off between the empirical loss and the regularizer is controlled by each player's regularization parameter $\rho_v > 0$ for $v \in \{-1, +1\}$.

Note that either player's empirical costs $\hat{\theta}_v$ depend on both players' actions: $\mathbf{w} \in \mathcal{W}$ and $\dot{D} \subseteq \mathcal{X} \times \mathcal{Y}$. Because of the potentially conflicting players' interests, the decision process for $\mathbf{w}$ and $\dot{D}$ becomes a non-cooperative two-player game, which we call a *prediction game*. In the following section, we will refer to the *Nash prediction game* (NPG) which identifies the concept of an optimal move of the learner and the data generator under the assumption of simultaneously acting players.

3. The Nash Prediction Game

The outcome of a prediction game is one particular combination of actions $(\mathbf{w}^*, \dot{D}^*)$ that incurs costs $\hat{\theta}_v(\mathbf{w}^*, \dot{D}^*)$ for the players. Each player is aware that this outcome is affected by *both players' action* and that, consequently, their potential to choose an action can have an impact on the other player's decision. In general, there is no action that minimizes one player's cost function independent of the other player's action. In a non-cooperative game, the players are not allowed to communicate while making their decisions and therefore they have no information about the other player's strategy. In this setting, any concept of an optimal move requires additional assumptions on how the adversary will act.

We model the decision process for $\mathbf{w}^*$ and $\dot{D}^*$ as a *static* two-player game with *complete information*. In a *static* game, both players commit to an action simultaneously, without information about their opponent's action. In a game with *complete information*, both players know their opponent's cost function and action space.

When $\hat{\theta}_{-1}$ and $\hat{\theta}_{+1}$ are known and *antagonistic*, the assumption that the adversary will seek the greatest advantage by inflicting the greatest damage on $\hat{\theta}_{-1}$ justifies the *minimax strategy*: $\operatorname{argmin}_{\mathbf{w}} \max_{\dot{D}} \hat{\theta}_{-1}(\mathbf{w}, \dot{D})$. However, when the players' cost functions are not antagonistic, assuming that the adversary will inflict the greatest possible damage is overly pessimistic. Instead assuming that the adversary acts rationally in the sense of seeking the greatest possible personal advantage leads to the concept of a *Nash equilibrium*. An equilibrium strategy is a steady state of the game in which neither player has an incentive to unilaterally change their plan of actions.

In static games, equilibrium strategies are called Nash equilibria, which is why we refer to the resulting predictive model as *Nash prediction game* (NPG). In a two-player game, a Nash equilibrium is defined as a pair of actions such that no player can benefit from changing their action

unilaterally; that is,

$$\begin{aligned}\hat{\theta}_{-1}(\mathbf{w}^*,\dot{D}^*) &= \min_{\mathbf{w}\in\mathcal{W}} \hat{\theta}_{-1}(\mathbf{w},\dot{D}^*),\\ \hat{\theta}_{+1}(\mathbf{w}^*,\dot{D}^*) &= \min_{\dot{D}\subseteq\mathcal{X}\times\mathcal{Y}} \hat{\theta}_{+1}(\mathbf{w}^*,\dot{D}),\end{aligned}$$

where $\mathcal{W}$ and $\mathcal{X}\times\mathcal{Y}$ denote the players' action spaces.

However, a static prediction game may not have a Nash equilibrium, or it may possess multiple equilibria. If $(\mathbf{w}^*,\dot{D}^*)$ and $(\mathbf{w}',\dot{D}')$ are distinct Nash equilibria and each player decides to act according to a different one of them, then combinations $(\mathbf{w}^*,\dot{D}')$ and $(\mathbf{w}',\dot{D}^*)$ may incur arbitrarily high costs for both players. Hence, one can argue that it is rational for an adversary to play a Nash equilibrium only when the following assumption is satisfied.

Assumption 1 *The following statements hold:*

1. *both players act simultaneously;*
2. *both players have full knowledge about both (empirical) cost functions* $\hat{\theta}_v(\mathbf{w},\dot{D})$ *defined in* (1) *and* (2)*, and both action spaces* $\mathcal{W}$ *and* $\mathcal{X}\times\mathcal{Y}$*;*
3. *both players act rational with respect to their cost function in the sense of securing their lowest possible costs;*
4. *a unique Nash equilibrium exists.*

Whether Assumptions 1.1-1.3 are adequate—especially the assumption of simultaneous actions—strongly depends on the application. For example, in some applications, the data generator may unilaterally be able to acquire information about the model $f_{\mathbf{w}}$ before committing to $\dot{D}$. Such situations are better modeled as a *Stackelberg competition* (Brückner and Scheffer, 2011). On the other hand, when the learner is able to treat any executed action as part of the training data D and update the model $\mathbf{w}$, the setting is better modeled as repeated executions of a static game with simultaneous actions. The adequateness of Assumption 1.4, which we discuss in the following sections, depends on the chosen loss functions, the cost factors, and the regularizers.

3.1 Existence of a Nash Equilibrium

Theorem 1 of Brückner and Scheffer (2009) identifies conditions under which a unique Nash equilibrium exists. Kanzow located a flaw in the proof of this theorem: The proof argues that the pseudo-Jacobian can be decomposed into two (strictly) positive stable matrices by showing that the real part of every eigenvalue of those two matrices is positive. However, this does not generally imply that the sum of these matrices is positive stable as well since this would require a common Lyapunov solution (cf. Problem 2.2.6 of Horn and Johnson, 1991). But even if such a solution exists, the positive definiteness cannot be concluded from the positiveness of all eigenvalues as the pseudo-Jacobian is generally non-symmetric.

Having "unproven" prior claims, we will now derive sufficient conditions for the existence of a Nash equilibrium. To this end, we first define

$$\begin{aligned}\mathbf{x} &:= \left[\phi(x_1)^\mathsf{T},\phi(x_2)^\mathsf{T},\ldots,\phi(x_n)^\mathsf{T}\right]^\mathsf{T} \in \phi(\mathcal{X})^n \subset \mathbb{R}^{m\cdot n},\\ \dot{\mathbf{x}} &:= \left[\phi(\dot{x}_1)^\mathsf{T},\phi(\dot{x}_2)^\mathsf{T},\ldots,\phi(\dot{x}_n)^\mathsf{T}\right]^\mathsf{T} \in \phi(\mathcal{X})^n \subset \mathbb{R}^{m\cdot n},\end{aligned}$$

as long, concatenated, column vectors induced by feature mapping ϕ, training sample $D = \{(x_i, y_i)\}_{i=1}^n$, and transformed training sample $\dot{D} = \{(\dot{x}_i, y_i)\}_{i=1}^n$, respectively. For terminological harmony, we refer to vector $\dot{\mathbf{x}}$ as the data generator's action with corresponding action space $\phi(\mathcal{X})^n$.

We make the following assumptions on the action spaces and the cost functions which enables us to state the main result on the existence of at least one Nash equilibrium in Lemma 1.

Assumption 2 *The players' cost functions defined in Equations 1 and 2, and their action sets $\mathcal{W}$ and $\phi(\mathcal{X})^n$ satisfy the properties:*

1. *loss functions $\ell_v(z,y)$ with $v \in \{-1,+1\}$ are convex and twice continuously differentiable with respect to $z \in \mathbb{R}$ for all fixed $y \in \mathcal{Y}$;*

2. *regularizers $\hat{\Omega}_v$ are uniformly strongly convex and twice continuously differentiable with respect to $\mathbf{w} \in \mathcal{W}$ and $\dot{\mathbf{x}} \in \phi(\mathcal{X})^n$, respectively;*

3. *action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$ are non-empty, compact, and convex subsets of finite-dimensional Euclidean spaces $\mathbb{R}^m$ and $\mathbb{R}^{m \cdot n}$, respectively.*

Lemma 1 *Under Assumption 2, at least one equilibrium point $(\mathbf{w}^*, \dot{\mathbf{x}}^*) \in \mathcal{W} \times \phi(\mathcal{X})^n$ of the Nash prediction game defined by*

$$\begin{array}{l|l} \min\limits_{\mathbf{w}} \quad \hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}^*) & \min\limits_{\dot{\mathbf{x}}} \quad \hat{\theta}_{+1}(\mathbf{w}^*, \dot{\mathbf{x}}) \\ \text{s.t.} \quad \mathbf{w} \in \mathcal{W} & \text{s.t.} \quad \dot{\mathbf{x}} \in \phi(\mathcal{X})^n \end{array} \tag{3}$$

exists.

Proof. Each player v's cost function is a sum over n loss terms resulting from loss function ℓ_v and regularizer $\hat{\Omega}_v$. By Assumption 2, these loss functions are convex and continuous, and the regularizers are uniformly strongly convex and continuous. Hence, both cost functions $\hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}})$ and $\hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}})$ are continuous in all arguments and uniformly strongly convex in $\mathbf{w} \in \mathcal{W}$ and $\dot{\mathbf{x}} \in \phi(\mathcal{X})^n$, respectively. As both action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$ are non-empty, compact, and convex subsets of finite-dimensional Euclidean spaces, a Nash equilibrium exists—see Theorem 4.3 of Basar and Olsder (1999). ■

3.2 Uniqueness of the Nash Equilibrium

We will now derive conditions for the uniqueness of an equilibrium of the Nash prediction game defined in (3). We first reformulate the two-player game into an $(n+1)$-player game. In Lemma 2, we then present a sufficient condition for the uniqueness of the Nash equilibrium in this game, and by applying Proposition 4 and Lemma 5-7 we verify whether this condition is met. Finally, we state the main result in Theorem 8: The Nash equilibrium is unique under certain properties of the loss functions, the regularizers, and the cost factors which all can be verified easily.

Taking into account the Cartesian product structure of the data generator's action space $\phi(\mathcal{X})^n$, it is not difficult to see that $(\mathbf{w}^*, \dot{\mathbf{x}}^*)$ with $\dot{\mathbf{x}}^* = \left[\dot{\mathbf{x}}_1^{*\mathsf{T}}, \ldots, \dot{\mathbf{x}}_n^{*\mathsf{T}}\right]^\mathsf{T}$ and $\dot{\mathbf{x}}_i^* := \phi(\dot{x}_i^*)$ is a solution of the

two-player game if, and only if, $(\mathbf{w}^*, \dot{\mathbf{x}}_1^*, \ldots, \dot{\mathbf{x}}_n^*)$ is a Nash equilibrium of the $(n+1)$-player game defined by

$$\begin{array}{c|c|c|c} \min\limits_{\mathbf{w}} \ \hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) & \min\limits_{\dot{\mathbf{x}}_1} \ \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) & \cdots & \min\limits_{\dot{\mathbf{x}}_n} \ \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) \\ \text{s.t.} \ \ \mathbf{w} \in \mathcal{W} & \text{s.t.} \ \ \dot{\mathbf{x}}_1 \in \phi(\mathcal{X}) & \cdots & \text{s.t.} \ \ \dot{\mathbf{x}}_n \in \phi(\mathcal{X}) \end{array}, \tag{4}$$

which results from (3) by repeating n times the cost function $\hat{\theta}_{+1}$ and minimizing this function with respect to $\dot{\mathbf{x}}_i \in \phi(\mathcal{X})$ for $i = 1, \ldots, n$. Then the *pseudo-gradient* (in the sense of Rosen, 1965) of the game in (4) is defined by

$$\mathbf{g}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}) := \begin{bmatrix} r_0 \nabla_{\mathbf{w}} \hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) \\ r_1 \nabla_{\dot{\mathbf{x}}_1} \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) \\ r_2 \nabla_{\dot{\mathbf{x}}_2} \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) \\ \vdots \\ r_n \nabla_{\dot{\mathbf{x}}_n} \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) \end{bmatrix} \in \mathbb{R}^{m+m \cdot n}, \tag{5}$$

with any fixed vector $\mathbf{r} = [r_0, r_1, \ldots, r_n]^\mathsf{T}$ where $r_i > 0$ for $i = 0, \ldots, n$. The derivative of $\mathbf{g}_{\mathbf{r}}$—that is, the *pseudo-Jacobian* of (4)—is given by

$$\mathbf{J}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}) = \Lambda_{\mathbf{r}} \begin{bmatrix} \nabla^2_{\mathbf{w},\mathbf{w}} \hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) & \nabla^2_{\mathbf{w},\dot{\mathbf{x}}} \hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) \\ \nabla^2_{\dot{\mathbf{x}},\mathbf{w}} \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) & \nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}} \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) \end{bmatrix}, \tag{6}$$

where

$$\Lambda_{\mathbf{r}} := \begin{bmatrix} r_0 \mathbf{I}_m & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & r_1 \mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & r_n \mathbf{I}_m \end{bmatrix} \in \mathbb{R}^{(m+m \cdot n) \times (m+m \cdot n)}. \tag{7}$$

Note that the pseudo-gradient $\mathbf{g}_{\mathbf{r}}$ and the pseudo-Jacobian $\mathbf{J}_{\mathbf{r}}$ exist when Assumption 2 is satisfied. The above definition of the pseudo-Jacobian enables us to state the following result about the uniqueness of a Nash equilibrium.

Lemma 2 *Let Assumption 2 hold and suppose there exists a fixed vector* $\mathbf{r} = [r_0, r_1, \ldots, r_n]^\mathsf{T}$ *with* $r_i > 0$ *for all* $i = 0, 1, \ldots, n$ *such that the corresponding pseudo-Jacobian* $\mathbf{J}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}})$ *is positive definite for all* $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$. *Then the Nash prediction game in* (3) *has a unique equilibrium.*

Proof. The existence of a Nash equilibrium follows from Lemma 1. Recall from our previous discussion that the original Nash game in (3) has a unique solution if, and only if, the game from (4) with one learner and n data generators admits a unique solution. In view of Theorem 2 of Rosen (1965), the latter attains a unique solution if the pseudo-gradient $\mathbf{g}_{\mathbf{r}}$ is strictly monotone; that is, if for all actions $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$ and $\dot{\mathbf{x}}, \dot{\mathbf{x}}' \in \phi(\mathcal{X})^n$, the inequality

$$\big(\mathbf{g}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}) - \mathbf{g}_{\mathbf{r}}(\mathbf{w}', \dot{\mathbf{x}}')\big)^\mathsf{T} \left(\begin{bmatrix} \mathbf{w} \\ \dot{\mathbf{x}} \end{bmatrix} - \begin{bmatrix} \mathbf{w}' \\ \dot{\mathbf{x}}' \end{bmatrix} \right) > 0$$

holds. A sufficient condition for this pseudo-gradient being strictly monotone is the positive definiteness of the pseudo-Jacobian $\mathbf{J}_{\mathbf{r}}$ (see, e.g., Theorem 7.11 and Theorem 6, respectively, in Geiger

and Kanzow, 1999; Rosen, 1965). ■

To verify whether the positive definiteness condition of Lemma 2 is satisfied, we first derive the pseudo-Jacobian $\mathbf{J}_{\mathbf{r}}(\mathbf{w},\dot{\mathbf{x}})$. We subsequently decompose it into a sum of three matrices and analyze the definiteness of these matrices for the particular choice of vector $\mathbf{r}$ with $r_0 := 1$, $r_i := \frac{c_{-1,i}}{c_{+1,i}} > 0$ for all $i = 1,\ldots,n$, with corresponding matrix

$$\Lambda_{\mathbf{r}} := \begin{bmatrix} \mathbf{I}_m & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \frac{c_{-1,1}}{c_{+1,1}}\mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \frac{c_{-1,n}}{c_{+1,n}}\mathbf{I}_m \end{bmatrix}. \tag{8}$$

This finally provides us with sufficient conditions which ensure the uniqueness of the Nash equilibrium.

3.2.1 DERIVATION OF THE PSEUDO-JACOBIAN

Throughout this section, we denote by $\ell'_v(z,y)$ and $\ell''_v(z,y)$ the first and second derivative of the mapping $\ell_v(z,y)$ with respect to $z \in \mathbb{R}$ and use the abbreviations

$$\begin{aligned} \ell'_{v,i} &:= \ell'_v(\dot{\mathbf{x}}_i^\mathsf{T}\mathbf{w}, y_i), \\ \ell''_{v,i} &:= \ell''_v(\dot{\mathbf{x}}_i^\mathsf{T}\mathbf{w}, y_i), \end{aligned}$$

for both players $v \in \{-1,+1\}$ and $i = 1,\ldots,n$.

To state the pseudo-Jacobian for the empirical costs given in (1) and (2), we first derive their first-order partial derivatives,

$$\nabla_{\mathbf{w}}\hat{\theta}_{-1}(\mathbf{w},\dot{\mathbf{x}}) = \sum_{i=1}^{n} c_{-1,i}\ell'_{-1,i}\dot{\mathbf{x}}_i + \rho_{-1}\nabla_{\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}), \tag{9}$$

$$\nabla_{\dot{\mathbf{x}}_i}\hat{\theta}_{+1}(\mathbf{w},\dot{\mathbf{x}}) = c_{+1,i}\ell'_{+1,i}\mathbf{w} + \rho_{+1}\nabla_{\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}). \tag{10}$$

This allows us to calculate the entries of the pseudo-Jacobian given in (6),

$$\begin{aligned} \nabla^2_{\mathbf{w},\mathbf{w}}\hat{\theta}_{-1}(\mathbf{w},\dot{\mathbf{x}}) &= \sum_{i=1}^{n} c_{-1,i}\ell''_{-1,i}\dot{\mathbf{x}}_i\dot{\mathbf{x}}_i^\mathsf{T} + \rho_{-1}\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}), \\ \nabla^2_{\mathbf{w},\dot{\mathbf{x}}_i}\hat{\theta}_{-1}(\mathbf{w},\dot{\mathbf{x}}) &= c_{-1,i}\ell''_{-1,i}\dot{\mathbf{x}}_i\mathbf{w}^\mathsf{T} + c_{-1,i}\ell'_{-1,i}\mathbf{I}_m, \\ \nabla^2_{\dot{\mathbf{x}}_i,\mathbf{w}}\hat{\theta}_{+1}(\mathbf{w},\dot{\mathbf{x}}) &= c_{+1,i}\ell''_{+1,i}\mathbf{w}\dot{\mathbf{x}}_i^\mathsf{T} + c_{+1,i}\ell'_{+1,i}\mathbf{I}_m, \\ \nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_j}\hat{\theta}_{+1}(\mathbf{w},\dot{\mathbf{x}}) &= \delta_{ij}c_{+1,i}\ell''_{+1,i}\mathbf{w}\mathbf{w}^\mathsf{T} + \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_j}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}), \end{aligned}$$

where δ_{ij} denotes Kronecker's delta which is 1 if i equals j and 0 otherwise.

We can express these equations more compact as matrix equations. Therefore, we use the diagonal matrix $\Lambda_{\mathbf{r}}$ as defined in (7) and set $\Gamma_v := \operatorname{diag}(c_{v,1}\ell''_{v,1},\ldots,c_{v,n}\ell''_{v,n})$. Additionally, we define $\dot{\mathbf{X}} \in \mathbb{R}^{n\times m}$ as the matrix with rows $\dot{\mathbf{x}}_1^\mathsf{T},\ldots,\dot{\mathbf{x}}_n^\mathsf{T}$, and n matrices $\mathbf{W}_i \in \mathbb{R}^{n\times m}$ with all entries set to zero

except for the i-th row which is set to $\mathbf{w}^\mathsf{T}$. Then,

$$\begin{aligned}
\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\theta}_{-1}(\mathbf{w},\dot{\mathbf{x}}) &= \dot{\mathbf{X}}^\mathsf{T}\Gamma_{-1}\dot{\mathbf{X}} + \rho_{-1}\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}),\\
\nabla^2_{\mathbf{w},\dot{\mathbf{x}}_i}\hat{\theta}_{-1}(\mathbf{w},\dot{\mathbf{x}}) &= \dot{\mathbf{X}}^\mathsf{T}\Gamma_{-1}\mathbf{W}_i + c_{-1,i}\ell'_{-1,i}\mathbf{I}_m,\\
\nabla^2_{\dot{\mathbf{x}}_i,\mathbf{w}}\hat{\theta}_{+1}(\mathbf{w},\dot{\mathbf{x}}) &= \mathbf{W}_i^\mathsf{T}\Gamma_{+1}\dot{\mathbf{X}} + c_{+1,i}\ell'_{+1,i}\mathbf{I}_m,\\
\nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_j}\hat{\theta}_{+1}(\mathbf{w},\dot{\mathbf{x}}) &= \mathbf{W}_i^\mathsf{T}\Gamma_{+1}\mathbf{W}_j + \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_j}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}).
\end{aligned}$$

Hence, the pseudo-Jacobian in (6) can be stated as follows,

$$\begin{aligned}
\mathbf{J}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) \;=\; \Lambda_\mathbf{r}&\begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix}^\mathsf{T} \begin{bmatrix} \Gamma_{-1} & \Gamma_{-1} \\ \Gamma_{+1} & \Gamma_{+1} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix} + \\
\Lambda_\mathbf{r}&\begin{bmatrix} \rho_{-1}\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) & c_{-1,1}\ell'_{-1,1}\mathbf{I}_m & \cdots & c_{-1,n}\ell'_{-1,n}\mathbf{I}_m \\ c_{+1,1}\ell'_{+1,1}\mathbf{I}_m & \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_1,\dot{\mathbf{x}}_1}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) & \cdots & \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_1,\dot{\mathbf{x}}_n}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) \\ \vdots & \vdots & \ddots & \vdots \\ c_{+1,n}\ell'_{+1,n}\mathbf{I}_m & \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_n,\dot{\mathbf{x}}_1}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) & \cdots & \rho_{+1}\nabla^2_{\dot{\mathbf{x}}_n,\dot{\mathbf{x}}_n}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) \end{bmatrix}.
\end{aligned}$$

We now aim at decomposing the right-hand expression in order to verify the definiteness of the pseudo-Jacobian.

3.2.2 DECOMPOSITION OF THE PSEUDO-JACOBIAN

To verify the positive definiteness of the pseudo-Jacobian, we further decompose the second summand of the above expression into a positive semi-definite and a strictly positive definite matrix. Therefore, let us denote the smallest eigenvalues of the Hessians of the regularizers on the corresponding action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$ by

$$\lambda_{-1} := \inf_{\mathbf{w}\in\mathcal{W}} \lambda_{\min}\left(\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w})\right), \tag{11}$$

$$\lambda_{+1} := \inf_{\dot{\mathbf{x}}\in\phi(\mathcal{X})^n} \lambda_{\min}\left(\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})\right), \tag{12}$$

where $\lambda_{\min}(\mathbf{A})$ denotes the smallest eigenvalue of the symmetric matrix $\mathbf{A}$.

Remark 3 Note that the minimum in (11) and (12) is attained and is strictly positive: The mapping $\lambda_{\min} : \mathcal{M}^{k\times k} \to \mathbb{R}$ is concave on the set of symmetric matrices $\mathcal{M}^{k\times k}$ of dimension $k\times k$ (cf. Example 3.10 in Boyd and Vandenberghe, 2004), and in particular, it therefore follows that this mapping is continuous. Furthermore, the mappings $u_{-1} : \mathcal{W} \to \mathcal{M}^{m\times m}$ with $u_{-1}(\mathbf{w}) := \nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w})$ and $u_{+1} : \phi(\mathcal{X})^n \to \mathcal{M}^{m\cdot n\times m\cdot n}$ with $u_{+1}(\dot{\mathbf{x}}) := \nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})$ are continuous (for any fixed $\mathbf{x}$) by Assumption 2. Hence, the mappings $\mathbf{w} \mapsto \lambda_{\min}(u_{-1}(\mathbf{w}))$ and $\dot{\mathbf{x}} \mapsto \lambda_{\min}(u_{+1}(\dot{\mathbf{x}}))$ are also continuous since each is precisely the composition $\lambda_{\min}\circ u_v$ of the continuous functions $\lambda_{\min}$ and u_v for $v\in\{-1,+1\}$. Taking into account that a continuous mapping on a non-empty compact set attains its minimum, it follows that there exist elements $\mathbf{w}\in\mathcal{W}$ and $\dot{\mathbf{x}}\in\phi(\mathcal{X})^n$ such that

$$\begin{aligned}
\lambda_{-1} &= \lambda_{\min}\left(\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w})\right),\\
\lambda_{+1} &= \lambda_{\min}\left(\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})\right).
\end{aligned}$$

Moreover, since the Hessians of the regularizers are positive definite by Assumption 2, we see that $\lambda_v > 0$ holds for $v\in\{-1,+1\}$. ◇

By the above definitions, we can decompose the regularizers' Hessians as follows,

$$\begin{aligned}\nabla^2_{\mathbf{w},\mathbf{w}}\Omega_{-1}(\mathbf{w}) &= \lambda_{-1}\mathbf{I}_m + (\nabla^2_{\mathbf{w},\mathbf{w}}\Omega_{-1}(\mathbf{w}) - \lambda_{-1}\mathbf{I}_m),\\ \nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\Omega_{+1}(\mathbf{x},\dot{\mathbf{x}}) &= \lambda_{+1}\mathbf{I}_{m\cdot n} + (\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\Omega_{+1}(\mathbf{x},\dot{\mathbf{x}}) - \lambda_{+1}\mathbf{I}_{m\cdot n}).\end{aligned}$$

As the regularizers are strictly convex, λ_v are positive so that for each of the above equations the first summand is positive definite and the second summand is positive semi-definite.

Proposition 4 *The pseudo-Jacobian has the representation*

$$\mathbf{J}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) = \mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) + \mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) + \mathbf{J}^{(3)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) \tag{13}$$

where

$$\begin{aligned}\mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) &= \Lambda_\mathbf{r}\begin{bmatrix}\mathbf{X} & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n\end{bmatrix}^\mathsf{T}\begin{bmatrix}\Gamma_{-1} & \Gamma_{-1}\\ \Gamma_{+1} & \Gamma_{+1}\end{bmatrix}\begin{bmatrix}\mathbf{X} & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n\end{bmatrix},\\ \mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) &= \Lambda_\mathbf{r}\begin{bmatrix}\rho_{-1}\lambda_{-1}\mathbf{I}_m & c_{-1,1}\ell'_{-1,1}\mathbf{I}_m & \cdots & c_{-1,n}\ell'_{-1,n}\mathbf{I}_m\\ c_{+1,1}\ell'_{+1,1}\mathbf{I}_m & \rho_{+1}\lambda_{+1}\mathbf{I}_m & \cdots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ c_{+1,n}\ell'_{+1,n}\mathbf{I}_m & \mathbf{0} & \cdots & \rho_{+1}\lambda_{+1}\mathbf{I}_m\end{bmatrix},\\ \mathbf{J}^{(3)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}}) &= \Lambda_\mathbf{r}\begin{bmatrix}\rho_{-1}\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) - \rho_{-1}\lambda_{-1}\mathbf{I}_m & \mathbf{0}\\ \mathbf{0} & \rho_{+1}\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) - \rho_{+1}\lambda_{+1}\mathbf{I}_{m\cdot n}\end{bmatrix}.\end{aligned}$$

The above proposition restates the pseudo-Jacobian as a sum of the three matrices $\mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$, $\mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$, and $\mathbf{J}^{(3)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$. Matrix $\mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ contains all $\ell''_{v,i}$ terms, $\mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ is a composition of scaled identity matrices, and $\mathbf{J}^{(3)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ contains the Hessians of the regularizers where the diagonal entries are reduced by $\rho_{-1}\lambda_{-1}$ and $\rho_{+1}\lambda_{+1}$, respectively. We further analyze these matrices in the following section.

3.2.3 DEFINITENESS OF THE SUMMANDS OF THE PSEUDO-JACOBIAN

Recall, that we want to investigate whether the pseudo-Jacobian $\mathbf{J}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ is positive definite for each pair of actions $(\mathbf{w},\dot{\mathbf{x}}) \in \mathcal{W}\times\phi(\mathcal{X})^n$. A sufficient condition is that $\mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$, $\mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$, and $\mathbf{J}^{(3)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ are positive semi-definite and at least one of these matrices is positive definite. From the definition of λ_v, it becomes apparent that $\mathbf{J}^{(3)}_\mathbf{r}$ is positive semi-definite. In addition, $\mathbf{J}^{(2)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ obviously becomes positive definite for sufficiently large ρ_v as, in this case, the main diagonal dominates the non-diagonal entries. Finally, $\mathbf{J}^{(1)}_\mathbf{r}(\mathbf{w},\dot{\mathbf{x}})$ becomes positive semi-definite under some mild conditions on the loss functions.

In the following we derive these conditions, state lower bounds on the regularization parameters ρ_v, and provide formal proofs of the above claims. Therefore, we make the following assumptions on the loss functions ℓ_v and the regularizers $\hat{\Omega}_v$ for $v\in\{-1,+1\}$. Instances of these functions satisfying Assumptions 2 and 3 will be given in Section 5. A discussion on the practical implications of these assumptions is given in the subsequent section.

Assumption 3 *For all* $\mathbf{w}\in\mathcal{W}$ *and* $\dot{\mathbf{x}}\in\phi(\mathcal{X})^n$ *with* $\dot{\mathbf{x}} = \left[\dot{\mathbf{x}}_1^\mathsf{T},\ldots,\dot{\mathbf{x}}_n^\mathsf{T}\right]^\mathsf{T}$ *the following conditions are satisfied:*

1. *the second derivatives of the loss functions are equal for all $y \in \mathcal{Y}$ and $i = 1, \ldots, n$,*

$$\ell''_{-1}(f_{\mathbf{w}}(\dot{\mathbf{x}}_i), y) = \ell''_{+1}(f_{\mathbf{w}}(\dot{\mathbf{x}}_i), y),$$

2. *the players' regularization parameters satisfy*

$$\rho_{-1}\rho_{+1} > \tau^2 \frac{1}{\lambda_{-1}\lambda_{+1}} \mathbf{c}_{-1}^{\mathsf{T}} \mathbf{c}_{+1},$$

where λ_{-1}, λ_{+1} are the smallest eigenvalues of the Hessians of the regularizers specified in (11) *and* (12)*, $\mathbf{c}_v = [c_{v,1}, c_{v,2}, \ldots, c_{v,n}]^{\mathsf{T}}$, and*

$$\tau = \sup_{(\mathbf{x},y) \in \phi(\mathcal{X}) \times \mathcal{Y}} \frac{1}{2} \left| \ell'_{-1}(f_{\mathbf{w}}(\mathbf{x}), y) + \ell'_{+1}(f_{\mathbf{w}}(\mathbf{x}), y) \right|, \tag{14}$$

3. *for all $i = 1, \ldots, n$ either both players have equal instance-specific cost factors, $c_{-1,i} = c_{+1,i}$, or the partial derivative $\nabla_{\dot{\mathbf{x}}_i} \Omega_{+1}(\mathbf{x}, \dot{\mathbf{x}})$ of the* data generator's *regularizer is independent of $\dot{\mathbf{x}}_j$ for all $j \neq i$.*

Notice, that τ in Equation 14 can be chosen to be finite as the set $\phi(\mathcal{X}) \times \mathcal{Y}$ is assumed to be compact, and consequently, the values of both continuous mappings $\ell'_{-1}(f_{\mathbf{w}}(\mathbf{x}), y)$ and $\ell'_{+1}(f_{\mathbf{w}}(\mathbf{x}), y)$ are finite for all $(\mathbf{x}, y) \in \phi(\mathcal{X}) \times \mathcal{Y}$.

Lemma 5 *Let $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ be arbitrarily given. Under Assumptions 2 and 3, the matrix $\mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}})$ is symmetric positive semi-definite (but not positive definite) for $\Lambda_{\mathbf{r}}$ defined as in Equation 8.*

Proof. The special structure of $\Lambda_{\mathbf{r}}$, $\dot{\mathbf{X}}$, and $\mathbf{W}_i$ gives

$$\mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}}) = \begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} r_0\Gamma_{-1} & r_0\Gamma_{-1} \\ \Upsilon\Gamma_{+1} & \Upsilon\Gamma_{+1} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix},$$

with $\Upsilon := \mathrm{diag}(r_1, \ldots, r_n)$. From the assumption $\ell''_{-1,i} = \ell''_{+1,i}$ and the definition $r_0 = 1$, $r_i = \frac{c_{-1,i}}{c_{+1,i}} > 0$ for all $i = 1, \ldots, n$ it follows that $\Gamma_{-1} = \Upsilon\Gamma_{+1}$, such that

$$\mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}}) = \begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \Gamma_{-1} & \Gamma_{-1} \\ \Gamma_{-1} & \Gamma_{-1} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{W}_n \end{bmatrix},$$

which is obviously a symmetric matrix. Furthermore, we show that $\mathbf{z}^{\mathsf{T}} \mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}}) \mathbf{z} \geq 0$ holds for all vectors $\mathbf{z} \in \mathbb{R}^{m+m \cdot n}$. To this end, let $\mathbf{z}$ be arbitrarily given, and partition this vector in $\mathbf{z} = \left[\mathbf{z}_0^{\mathsf{T}}, \mathbf{z}_1^{\mathsf{T}}, \ldots, \mathbf{z}_n^{\mathsf{T}}\right]^{\mathsf{T}}$ with $\mathbf{z}_i \in \mathbb{R}^m$ for all $i = 0, 1, \ldots, n$. Then a simple calculation shows that

$$\mathbf{z}^{\mathsf{T}} \mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}}) \mathbf{z} = \sum_{i=1}^{n} \left(\mathbf{z}_0^{\mathsf{T}} \mathbf{x}_i + \mathbf{z}_i^{\mathsf{T}} \mathbf{w} \right)^2 c_{-1,i} \ell''_{-1,i} \geq 0$$

since $\ell''_{-1,i} \geq 0$ for all $i = 1, \ldots, n$ in view of the assumed convexity of mapping $\ell_{-1}(z, y)$. Hence, $\mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}})$ is positive semi-definite. This matrix cannot be positive definite since we have $\mathbf{z}^{\mathsf{T}} \mathbf{J}_{\mathbf{r}}^{(1)}(\mathbf{w}, \dot{\mathbf{x}}) \mathbf{z} = 0$ for the particular vector $\mathbf{z}$ defined by $\mathbf{z}_0 := -\mathbf{w}$ and $\mathbf{z}_i := \mathbf{x}_i$ for all $i = 1, \ldots, n$. ■

Lemma 6 *Let $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ be arbitrarily given. Under Assumptions 2 and 3, the matrix $\mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}})$ is positive definite for $\Lambda_{\mathbf{r}}$ defined as in Equation 8.*

Proof. A sufficient and necessary condition for the (possibly asymmetric) matrix $\mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}})$ to be positive definite is that the Hermitian matrix

$$\mathbf{H}(\mathbf{w}, \dot{\mathbf{x}}) := \mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}}) + \mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}})^{\mathsf{T}}$$

is positive definite, that is, all eigenvalues of $\mathbf{H}(\mathbf{w}, \dot{\mathbf{x}})$ are positive. Let $\Lambda_{\mathbf{r}}^{\frac{1}{2}}$ denote the square root of $\Lambda_{\mathbf{r}}$ which is defined in such a way that the diagonal elements of $\Lambda_{\mathbf{r}}^{\frac{1}{2}}$ are the square roots of the corresponding diagonal elements of $\Lambda_{\mathbf{r}}$. Furthermore, we denote by $\Lambda_{\mathbf{r}}^{-\frac{1}{2}}$ the inverse of $\Lambda_{\mathbf{r}}^{\frac{1}{2}}$. Then, by Sylvester's law of inertia, the matrix

$$\bar{\mathbf{H}}(\mathbf{w}, \dot{\mathbf{x}}) := \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \mathbf{H}(\mathbf{w}, \dot{\mathbf{x}}) \Lambda_{\mathbf{r}}^{-\frac{1}{2}}$$

has the same number of positive, zero, and negative eigenvalues as matrix $\mathbf{H}(\mathbf{w}, \dot{\mathbf{x}})$ itself. Hence, $\mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}})$ is positive definite if, and only if, all eigenvalues of

$$\begin{aligned}
\bar{\mathbf{H}}(\mathbf{w}, \dot{\mathbf{x}}) &= \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \left(\mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}}) + \mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w}, \dot{\mathbf{x}})^{\mathsf{T}} \right) \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \\
&= \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \Lambda_{\mathbf{r}} \begin{bmatrix} \rho_{-1}\lambda_{-1}\mathbf{I}_m & c_{-1,1}\ell'_{-1,1}\mathbf{I}_m & \cdots & c_{-1,n}\ell'_{-1,n}\mathbf{I}_m \\ c_{+1,1}\ell'_{+1,1}\mathbf{I}_m & \rho_{+1}\lambda_{+1}\mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ c_{+1,n}\ell'_{+1,n}\mathbf{I}_m & \mathbf{0} & \cdots & \rho_{+1}\lambda_{+1}\mathbf{I}_m \end{bmatrix} \Lambda_{\mathbf{r}}^{-\frac{1}{2}} + \\
&\quad \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \begin{bmatrix} \rho_{-1}\lambda_{-1}\mathbf{I}_m & c_{+1,1}\ell'_{+1,1}\mathbf{I}_m & \cdots & c_{+1,n}\ell'_{+1,n}\mathbf{I}_m \\ c_{-1,1}\ell'_{-1,1}\mathbf{I}_m & \rho_{+1}\lambda_{+1}\mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ c_{-1,n}\ell'_{-1,n}\mathbf{I}_m & \mathbf{0} & \cdots & \rho_{+1}\lambda_{+1}\mathbf{I}_m \end{bmatrix} \Lambda_{\mathbf{r}} \Lambda_{\mathbf{r}}^{-\frac{1}{2}} \\
&= \begin{bmatrix} 2\rho_{-1}\lambda_{-1}\mathbf{I}_m & \tilde{c}_1\mathbf{I}_m & \cdots & \tilde{c}_n\mathbf{I}_m \\ \tilde{c}_1\mathbf{I}_m & 2\rho_{+1}\lambda_{+1}\mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{c}_n\mathbf{I}_m & \mathbf{0} & \cdots & 2\rho_{+1}\lambda_{+1}\mathbf{I}_m \end{bmatrix}
\end{aligned}$$

are positive, where $\tilde{c}_i := \sqrt{c_{-1,i}c_{+1,i}}(\ell'_{-1,i} + \ell'_{+1,i})$. Each eigenvalue λ of this matrix satisfies

$$\left(\bar{\mathbf{H}}(\mathbf{w}, \dot{\mathbf{x}}) - \lambda \mathbf{I}_{m+m\cdot n}\right) \mathbf{v} = \mathbf{0}$$

for the corresponding eigenvector $\mathbf{v}^{\mathsf{T}} = \left[\mathbf{v}_0^{\mathsf{T}}, \mathbf{v}_1^{\mathsf{T}}, \ldots, \mathbf{v}_n^{\mathsf{T}}\right]$ with $\mathbf{v}_i \in \mathbb{R}^m$ for $i = 0, 1, \ldots, n$. This eigenvalue equation can be rewritten block-wise as

$$(2\rho_{-1}\lambda_{-1} - \lambda)\mathbf{v}_0 + \sum_{i=1}^{n} \tilde{c}_i \mathbf{v}_i = \mathbf{0}, \tag{15}$$

$$(2\rho_{+1}\lambda_{+1} - \lambda)\mathbf{v}_i + \tilde{c}_i \mathbf{v}_0 = \mathbf{0} \quad \forall i = 1, \ldots, n. \tag{16}$$

To compute all possible eigenvalues, we consider two cases. First, assume that $\mathbf{v}_0 = \mathbf{0}$. Then (15) and (16) reduce to

$$\sum_{i=1}^{n} \tilde{c}_i \mathbf{v}_i = \mathbf{0} \quad \text{and} \quad (2\rho_{+1}\lambda_{+1} - \lambda)\mathbf{v}_i = \mathbf{0} \quad \forall i = 1, \ldots, n.$$

Since $\mathbf{v}_0 = \mathbf{0}$ and eigenvector $\mathbf{v} \neq \mathbf{0}$, at least one $\mathbf{v}_i$ is non-zero. This implies that $\lambda = 2\rho_{+1}\lambda_{+1}$ is an eigenvalue. Using the fact that the null space of the linear mapping $\mathbf{v} \mapsto \sum_{i=1}^{n} \tilde{c}_i \mathbf{v}_i$ has dimension $(n-1) \cdot m$ (we have $n \cdot m$ degrees of freedom counting all components of $\mathbf{v}_1, \ldots, \mathbf{v}_n$ and m equations in $\sum_{i=1}^{n} \tilde{c}_i \mathbf{v}_i = \mathbf{0}$), it follows that $\lambda = 2\rho_{+1}\lambda_{+1}$ is an eigenvalue of multiplicity $(n-1) \cdot m$.

Now we consider the second case where $\mathbf{v}_0 \neq \mathbf{0}$. We may further assume that $\lambda \neq 2\rho_{+1}\lambda_{+1}$ (since otherwise we get the same eigenvalue as before, just with a different multiplicity). We then get from (16) that

$$\mathbf{v}_i = -\frac{\tilde{c}_i}{2\rho_{+1}\lambda_{+1} - \lambda} \mathbf{v}_0 \quad \forall i = 1, \ldots, n, \tag{17}$$

and when substituting this expression into (15), we obtain

$$\left((2\rho_{-1}\lambda_{-1} - \lambda) - \sum_{i=1}^{n} \frac{\tilde{c}_i^2}{2\rho_{+1}\lambda_{+1} - \lambda} \right) \mathbf{v}_0 = \mathbf{0}.$$

Taking into account that $\mathbf{v}_0 \neq \mathbf{0}$, this implies

$$0 = 2\rho_{-1}\lambda_{-1} - \lambda - \frac{1}{2\rho_{+1}\lambda_{+1} - \lambda} \sum_{i=1}^{n} \tilde{c}_i^2$$

and, therefore,

$$0 = \lambda^2 - 2(\rho_{-1}\lambda_{-1} + \rho_{+1}\lambda_{+1})\lambda + 4\rho_{-1}\rho_{+1}\lambda_{-1}\lambda_{+1} - \sum_{i=1}^{n} \tilde{c}_i^2.$$

The roots of this quadratic equation are

$$\lambda = \rho_{-1}\lambda_{-1} + \rho_{+1}\lambda_{+1} \pm \sqrt{(\rho_{-1}\lambda_{-1} - \rho_{+1}\lambda_{+1})^2 + \sum_{i=1}^{n} \tilde{c}_i^2}, \tag{18}$$

and these are the remaining eigenvalues of $\tilde{\mathbf{H}}(\mathbf{w}, \dot{\mathbf{x}})$, each of multiplicity m since there are precisely m linearly independent vectors $\mathbf{v}_0 \neq \mathbf{0}$ whereas the other vectors $\mathbf{v}_i$ ($i = 1, \ldots, n$) are uniquely defined by (17) in this case. In particular, this implies that the dimensions of all three eigenspaces together is $(n-1)m + m + m = (n+1)m$, hence other eigenvalues cannot exist. Since the eigenvalue $\lambda = 2\rho_{+1}\lambda_{+1}$ is positive by Remark 3, it remains to show that the roots in (18) are positive as well. By Assumption 3, we have

$$\sum_{i=1}^{n} \tilde{c}_i^2 = \sum_{i=1}^{n} c_{-1,i} c_{+1,i} (\ell'_{-1,i} + \ell'_{+1,i})^2 \leq 4\tau^2 \mathbf{c}_{-1}^{\mathsf{T}} \mathbf{c}_{+1} < 4\rho_{-1}\rho_{+1}\lambda_{-1}\lambda_{+1},$$

where $\mathbf{c}_v = [c_{v,1}, c_{v,2}, \cdots, c_{v,n}]^{\mathsf{T}}$. This inequality and Equation 18 give

$$\begin{aligned} \lambda &= \rho_{-1}\lambda_{-1} + \rho_{+1}\lambda_{+1} \pm \sqrt{(\rho_{-1}\lambda_{-1} - \rho_{+1}\lambda_{+1})^2 + \sum_{i=1}^{n} \tilde{c}_i^2} \\ &> \rho_{-1}\lambda_{-1} + \rho_{+1}\lambda_{+1} - \sqrt{(\rho_{-1}\lambda_{-1} - \rho_{+1}\lambda_{+1})^2 + 4\rho_{-1}\rho_{+1}\lambda_{-1}\lambda_{+1}} \;=\; 0. \end{aligned}$$

As all eigenvalues of $\bar{\mathbf{H}}(\mathbf{w},\dot{\mathbf{x}})$ are positive, matrix $\mathbf{H}(\mathbf{w},\dot{\mathbf{x}})$ and, consequently, also the matrix $\mathbf{J}_{\mathbf{r}}^{(2)}(\mathbf{w},\dot{\mathbf{x}})$ are positive definite. ■

Lemma 7 *Let $(\mathbf{w},\dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ be arbitrarily given. Under Assumptions 2 and 3, the matrix $\mathbf{J}_{\mathbf{r}}^{(3)}(\mathbf{w},\dot{\mathbf{x}})$ is positive semi-definite for $\Lambda_{\mathbf{r}}$ defined as in Equation 8.*

Proof. By Assumption 3, either both players have equal instance-specific costs, or the partial gradient $\nabla_{\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})$ of the *sender's* regularizer is independent of $\dot{\mathbf{x}}_j$ for all $j \neq i$ and $i = 1,\ldots,n$. Let us consider the first case where $c_{-1,i} = c_{+1,i}$, and consequently $r_i = 1$, for all $i = 1,\ldots,n$, such that

$$\mathbf{J}_{\mathbf{r}}^{(3)}(\mathbf{w},\dot{\mathbf{x}}) = \begin{bmatrix} \rho_{-1}\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) - \rho_{-1}\lambda_{-1}\mathbf{I}_m & \mathbf{0} \\ \mathbf{0} & \rho_{+1}\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) - \rho_{+1}\lambda_{+1}\mathbf{I}_{m\cdot n} \end{bmatrix}.$$

The eigenvalues of this block diagonal matrix are the eigenvalues of the matrix $\rho_{-1}(\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) - \lambda_{-1}\mathbf{I}_m)$ together with those of $\rho_{+1}(\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) - \lambda_{+1}\mathbf{I}_{m\cdot n})$. From the definition of λ_v in (11) and (12) follows that these matrices are positive semi-definite for $v \in \{-1,+1\}$. Hence, $\mathbf{J}_{\mathbf{r}}^{(3)}(\mathbf{w},\dot{\mathbf{x}})$ is positive semi-definite as well.

Now, let us consider the second case where we assume that $\nabla_{\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})$ is independent of $\dot{\mathbf{x}}_j$ for all $j \neq i$. Hence, $\nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_j}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) = \mathbf{0}$ for all $j \neq i$ such that

$$\mathbf{J}_{\mathbf{r}}^{(3)}(\mathbf{w},\dot{\mathbf{x}}) = \begin{bmatrix} \rho_{-1}\tilde{\Omega}_{-1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \rho_{+1}\frac{c_{-1,1}}{c_{+1,1}}\tilde{\Omega}_{+1,1} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \rho_{+1}\frac{c_{-1,n}}{c_{+1,n}}\tilde{\Omega}_{+1,n} \end{bmatrix},$$

where $\tilde{\Omega}_{-1} := \nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) - \lambda_{-1}\mathbf{I}_m$ and $\tilde{\Omega}_{+1,i} = \nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) - \lambda_{+1}\mathbf{I}_m$. The eigenvalues of this block diagonal matrix are again the union of the eigenvalues of the single blocks $\rho_{-1}\tilde{\Omega}_{-1}$ and $\rho_{+1}\frac{c_{-1,i}}{c_{+1,i}}\tilde{\Omega}_{+1,i}$ for $i = 1,\ldots,n$. As in the first part of the proof, $\tilde{\Omega}_{-1}$ is positive semi-definite. The eigenvalues of $\nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})$ are the union of all eigenvalues of $\nabla^2_{\dot{\mathbf{x}}_i,\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}})$. Hence, each of these eigenvalues is larger or equal to λ_{+1} and thus, each block $\tilde{\Omega}_{+1,i}$ is positive semi-definite. The factors $\rho_{-1} > 0$ and $\rho_{+1}\frac{c_{-1,i}}{c_{+1,i}} > 0$ are multipliers that do not affect the definiteness of the blocks, and consequently, $\mathbf{J}_{\mathbf{r}}^{(3)}(\mathbf{w},\dot{\mathbf{x}})$ is positive semi-definite as well. ■

The previous results guarantee the existence and uniqueness of a Nash equilibrium under the stated assumptions.

Theorem 8 *Let Assumptions 2 and 3 hold. Then the Nash prediction game in* (3) *has a unique equilibrium.*

Proof. The existence of an equilibrium of the Nash prediction game in (3) follows from Lemma 1. Proposition 4 and Lemma 5 to 7 imply that there is a positive diagonal matrix $\Lambda_{\mathbf{r}}$ such that $\mathbf{J}_{\mathbf{r}}(\mathbf{w},\dot{\mathbf{x}})$ is positive definite for all $(\mathbf{w},\dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$. Hence, the uniqueness follows from Lemma 2. ■

3.2.4 PRACTICAL IMPLICATIONS OF ASSUMPTIONS 2 AND 3

Theorem 8 guarantees the uniqueness of the equilibrium only if the cost functions of learner and data generator relate in a certain way that is defined by Assumption 3. In addition, each of the cost functions has to satisfy Assumption 2. This section discusses the practical implication of these assumptions.

The conditions of Assumption 2 impose rather technical limitations on the cost functions. The requirement of convexity is quite ordinary in the machine learning context. In addition, the loss function has to be twice continuously differentiable, which restricts the family of eligible loss functions. However, this condition can still be met easily; for instance, by smoothed versions of the hinge loss. The second requirement of uniformly strongly convex and twice continuously differentiable regularizers is, again, only a week restriction in practice. These requirements are met by standard regularizers; they occur, for instance, in the optimization criteria of SVMs and logistic regression. The requirement of non-empty, compact, and convex action spaces may be a restriction when dealing with binary or multinomial attributes. However, relaxing the action spaces of the data generator would typically result in a strategy that is more defensive than would be optimal but still less defensive than a worst-case strategy.

The first condition of Assumptions 3 requires the cost functions of learner and data generator to have the same curvatures. This is a crucial restriction; if the cost functions differ arbitrarily the Nash equilibrium may not be unique. The requirement of identical curvatures is met, for instance, if one player chooses a loss function $\ell(f_{\mathbf{w}}(\dot{\mathbf{x}}_i), y)$ which only depends on the term $y f_{\mathbf{w}}(\dot{\mathbf{x}}_i)$, such as for SVM's hinge loss or the logistic loss. In this case, the condition is met when the other player chooses the $\ell(-f_{\mathbf{w}}(\dot{\mathbf{x}}_i), y)$. This loss is in some sense the opposite of $\ell(f_{\mathbf{w}}(\dot{\mathbf{x}}_i), y)$ as it approaches zero when the other goes to infinity and vice versa. In this case, the cost functions may still be non-antagonistic because the player's cost functions may contain instance-specific cost factors $c_{v,i}$ that can be modeled independently for the players.

The second part of Assumptions 3 couples the degree of regularization of the players. If the data generator produces instances at application time that differ greatly from the instances at training time, then the learner is required to regularize strongly for a unique equilibrium to exist. If the distributions at training and application time are more similar, the equilibrium is unique for smaller values of the learner's regularization parameters. This requirement is in line with the intuition that when the training instances are a poor approximation of the distribution at application time, then imposing only weak regularization on the loss function will result in a poor model.

The final requirement of Assumptions 3 is, again, rather a technical limitation. It states that the interdependencies between the players' instance-specific costs must be either captured by the regularizers, leading to a full Hessian, or by cost factors. These cost factors of learner and data generator may differ arbitrarily if the gradient of the data generator's costs of transforming an instance $\mathbf{x}_i$ into $\dot{\mathbf{x}}_i$ are independent of all other instances $\dot{\mathbf{x}}_j$ with $j \neq i$. This is met, for instance, by cost models that only depend on some measure of the distance between $\mathbf{x}_i$ and $\dot{\mathbf{x}}_i$.

4. Finding the Unique Nash Equilibrium

According to Theorem 8, a unique equilibrium of the Nash prediction game in (3) exists for suitable loss functions and regularizers. To find this equilibrium, we derive and study two distinct methods: The first is based on the Nikaido-Isoda function that is constructed such that a minimax solution of this function is an equilibrium of the Nash prediction game and vice versa. This problem is then

solved by inexact linesearch. In the second approach, we reformulate the Nash prediction game into a variational inequality problem which is solved by a modified extragradient method.

The data generator's action of transforming the input distribution manifests in a concatenation of transformed training instances $\dot{\mathbf{x}} \in \phi(\mathcal{X})^n$ mapped into the feature space $\dot{\mathbf{x}}_i := \phi(\dot{x}_i)$ for $i = 1, \ldots, n$, and the learner's action is to choose weight vector $\mathbf{w} \in \mathcal{W}$ of classifier $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ with linear decision function $f_{\mathbf{w}}(x) = \mathbf{w}^\mathsf{T}\phi(x)$.

4.1 An Inexact Linesearch Approach

To solve for a Nash equilibrium, we again consider the game from (4) with one learner and n data generators. A solution of this game can be identified with the help of the weighted *Nikaido-Isoda* function (Equation 19). For any two combinations of actions $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ and $(\mathbf{w}', \dot{\mathbf{x}}') \in \mathcal{W} \times \phi(\mathcal{X})^n$ with $\dot{\mathbf{x}} = \left[\dot{\mathbf{x}}_1^\mathsf{T}, \ldots, \dot{\mathbf{x}}_n^\mathsf{T}\right]^\mathsf{T}$ and $\dot{\mathbf{x}}' = \left[\dot{\mathbf{x}}_1'^\mathsf{T}, \ldots, \dot{\mathbf{x}}_n'^\mathsf{T}\right]^\mathsf{T}$, this function is the weighted sum of relative cost savings that the $n+1$ players can enjoy by changing from strategy $\mathbf{w}$ to $\mathbf{w}'$ and $\dot{\mathbf{x}}_i$ to $\dot{\mathbf{x}}_i'$, respectively, while the other players continue to play according to $(\mathbf{w}, \dot{\mathbf{x}})$; that is,

$$\vartheta_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}, \mathbf{w}', \dot{\mathbf{x}}') := r_0 \left(\hat{\theta}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) - \hat{\theta}_{-1}(\mathbf{w}', \dot{\mathbf{x}})\right) + \sum_{i=1}^{n} r_i \left(\hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) - \hat{\theta}_{+1}(\mathbf{w}, \dot{\mathbf{x}}^{(i)})\right), \tag{19}$$

where $\dot{\mathbf{x}}^{(i)} := \left[\dot{\mathbf{x}}_1^\mathsf{T}, \ldots, \dot{\mathbf{x}}_i'^\mathsf{T}, \ldots, \dot{\mathbf{x}}_n^\mathsf{T}\right]^\mathsf{T}$. Let us denote the weighted sum of greatest possible cost savings with respect to any given combination of actions $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ by

$$\bar{\vartheta}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}) := \max_{(\mathbf{w}', \dot{\mathbf{x}}') \in \mathcal{W} \times \phi(\mathcal{X})^n} \vartheta_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}, \mathbf{w}', \dot{\mathbf{x}}'), \tag{20}$$

where $\bar{\mathbf{w}}(\mathbf{w}, \dot{\mathbf{x}})$, $\bar{\mathbf{x}}(\mathbf{w}, \dot{\mathbf{x}})$ denotes the corresponding pair of maximizers. Note that the maximum in (20) is attained for any $(\mathbf{w}, \dot{\mathbf{x}})$, since $\mathcal{W} \times \phi(\mathcal{X})^n$ is assumed to be compact and $\vartheta_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}, \mathbf{w}', \dot{\mathbf{x}}')$ is continuous in $(\mathbf{w}', \dot{\mathbf{x}}')$.

By these definitions, a combination $(\mathbf{w}^*, \dot{\mathbf{x}}^*)$ is an equilibrium of the Nash prediction game if, and only if, $\bar{\vartheta}_{\mathbf{r}}(\mathbf{w}^*, \dot{\mathbf{x}}^*)$ is a global minimum of mapping $\bar{\vartheta}_{\mathbf{r}}$ with $\bar{\vartheta}_{\mathbf{r}}(\mathbf{w}^*, \dot{\mathbf{x}}^*) = 0$ for any fixed weights $r_i > 0$ and $i = 0, \ldots, n$, see Proposition 2.1(b) of von Heusinger and Kanzow (2009). Equivalently, a Nash equilibrium simultaneously satisfies both equations $\bar{\mathbf{w}}(\mathbf{w}^*, \dot{\mathbf{x}}^*) = \mathbf{w}^*$ and $\bar{\mathbf{x}}(\mathbf{w}^*, \dot{\mathbf{x}}^*) = \dot{\mathbf{x}}^*$.

The significance of this observation is that the equilibrium problem in (3) can be reformulated into a minimization problem of the continuous mapping $\bar{\vartheta}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}})$. To solve this minimization problem, we make use of Corollary 3.4 (von Heusinger and Kanzow, 2009). We set the weights $r_0 := 1$ and $r_i := \frac{c_{-1,i}}{c_{+1,i}}$ for all $i = 1, \ldots, n$ as in (8), which ensures the main condition of Corollary 3.4; that is, the positive definiteness of the Jacobian $\mathbf{J}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}})$ in (13) (cf. proof of Theorem 8). According to this corollary, vectors

$$\mathbf{d}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) := \bar{\mathbf{w}}(\mathbf{w}, \dot{\mathbf{x}}) - \mathbf{w} \quad \text{and} \quad \mathbf{d}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) := \bar{\mathbf{x}}(\mathbf{w}, \dot{\mathbf{x}}) - \dot{\mathbf{x}}$$

form a descent direction $\mathbf{d}(\mathbf{w}, \dot{\mathbf{x}}) := [\mathbf{d}_{-1}(\mathbf{w}, \dot{\mathbf{x}})^\mathsf{T}, \mathbf{d}_{+1}(\mathbf{w}, \dot{\mathbf{x}})^\mathsf{T}]^\mathsf{T}$ of $\bar{\vartheta}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}})$ at any position $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ (except for the Nash equilibrium where $\mathbf{d}(\mathbf{w}^*, \dot{\mathbf{x}}^*) = \mathbf{0}$), and consequently, there exists $t \in [0, 1]$ such that

$$\bar{\vartheta}_{\mathbf{r}}(\mathbf{w} + t\mathbf{d}_{-1}(\mathbf{w}, \dot{\mathbf{x}}), \dot{\mathbf{x}} + t\mathbf{d}_{+1}(\mathbf{w}, \dot{\mathbf{x}})) < \bar{\vartheta}_{\mathbf{r}}(\mathbf{w}, \dot{\mathbf{x}}).$$

Since, $(\mathbf{w}, \dot{\mathbf{x}})$ and $(\bar{\mathbf{w}}(\mathbf{w}, \dot{\mathbf{x}}), \bar{\mathbf{w}}(\mathbf{w}, \dot{\mathbf{x}}))$ are feasible combinations of actions, the convexity of the action spaces ensures that $(\mathbf{w} + t\mathbf{d}_{-1}(\mathbf{w}, \dot{\mathbf{x}}), \dot{\mathbf{x}} + t\mathbf{d}_{+1}(\mathbf{w}, \dot{\mathbf{x}}))$ is a feasible combination for any $t \in [0, 1]$ as well. The following algorithm exploits these properties.

Algorithm 1 ILS: Inexact Linesearch Solver for Nash Prediction Games

Require: Cost functions $\hat{\theta}_v$ as defined in (1) and (2), and action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$.

1: Select initial $\mathbf{w}^{(0)} \in \mathcal{W}$, set $\dot{\mathbf{x}}^{(0)} := \mathbf{x}$, set $k := 0$, and select $\sigma \in (0,1)$ and $\beta \in (0,1)$.

2: Set $r_0 := 1$ and $r_i := \frac{c_{-1,i}}{c_{+1,i}}$ for all $i = 1, \ldots, n$.

3: **repeat**

4: Set $\mathbf{d}_{-1}^{(k)} := \bar{\mathbf{w}}^{(k)} - \mathbf{w}^{(k)}$ where $\bar{\mathbf{w}}^{(k)} := \operatorname{argmax}_{\mathbf{w}' \in \mathcal{W}} \vartheta_{\mathbf{r}}\left(\mathbf{w}^{(k)}, \dot{\mathbf{x}}^{(k)}, \mathbf{w}', \dot{\mathbf{x}}^{(k)}\right)$.

5: Set $\mathbf{d}_{+1}^{(k)} := \bar{\mathbf{x}}^{(k)} - \dot{\mathbf{x}}^{(k)}$ where $\bar{\mathbf{x}}^{(k)} := \operatorname{argmax}_{\dot{\mathbf{x}}' \in \phi(\mathcal{X})^n} \vartheta_{\mathbf{r}}\left(\mathbf{w}^{(k)}, \dot{\mathbf{x}}^{(k)}, \mathbf{w}^{(k)}, \dot{\mathbf{x}}'\right)$.

6: Find maximal step size $t^{(k)} \in \left\{\beta^l \mid l \in \mathbb{N}\right\}$ with

$$\bar{\vartheta}_{\mathbf{r}}\left(\mathbf{w}^{(k)}, \dot{\mathbf{x}}^{(k)}\right) - \bar{\vartheta}_{\mathbf{r}}\left(\mathbf{w}^{(k)} + t^{(k)}\mathbf{d}_{-1}^{(k)}, \dot{\mathbf{x}}^{(k)} + t^{(k)}\mathbf{d}_{+1}^{(k)}\right) \geq \sigma t^{(k)} \left(\left\|\mathbf{d}_{-1}^{(k)}\right\|_2^2 + \left\|\mathbf{d}_{+1}^{(k)}\right\|_2^2\right).$$

7: Set $\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + t^{(k)}\mathbf{d}_{-1}^{(k)}$.

8: Set $\dot{\mathbf{x}}^{(k+1)} := \dot{\mathbf{x}}^{(k)} + t^{(k)}\mathbf{d}_{+1}^{(k)}$.

9: Set $k := k + 1$.

10: **until** $\left\|\mathbf{w}^{(k)} - \mathbf{w}^{(k-1)}\right\|_2^2 + \left\|\dot{\mathbf{x}}^{(k)} - \dot{\mathbf{x}}^{(k-1)}\right\|_2^2 \leq \varepsilon$.

The convergence properties of Algorithm 1 are discussed by von Heusinger and Kanzow (2009), so we skip the details here.

4.2 A Modified Extragradient Approach

In Algorithm 1, line 4 and 5, as well as the linesearch in line 6, require to solve a concave maximization problem within each iteration. As this may become computationally demanding, we derive a second approach based on extragradient descent. Therefore, instead of reformulating the equilibrium problem into a minimax problem, we directly address the first-order optimality conditions of each players' minimization problem in (4): Under Assumption 2, a combination of actions $(\mathbf{w}^*, \dot{\mathbf{x}}^*)$ with $\dot{\mathbf{x}}^* = \left[\dot{\mathbf{x}}_1^{*\mathsf{T}}, \ldots, \dot{\mathbf{x}}_n^{*\mathsf{T}}\right]^\mathsf{T}$ satisfies each player's first-order optimality conditions if, and only if, for all $(\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n$ the following inequalities hold,

$$\begin{aligned} \nabla_{\mathbf{w}} \hat{\theta}_{-1}(\mathbf{w}^*, \dot{\mathbf{x}}^*)^\mathsf{T}(\mathbf{w} - \mathbf{w}^*) &\geq 0, \\ \nabla_{\dot{\mathbf{x}}_i} \hat{\theta}_{+1}(\mathbf{w}^*, \dot{\mathbf{x}}^*)^\mathsf{T}(\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_i^*) &\geq 0 \quad \forall i = 1, \ldots, n. \end{aligned}$$

As the joint action space of all players $\mathcal{W} \times \phi(\mathcal{X})^n$ is precisely the full Cartesian product of the learner's action set $\mathcal{W}$ and the n data generators' action sets $\phi(\mathcal{X})$, the (weighted) sum of those individual optimality conditions is also a sufficient and necessary optimality condition for the equilibrium problem. Hence, a Nash equilibrium $(\mathbf{w}^*, \dot{\mathbf{x}}^*) \in \mathcal{W} \times \phi(\mathcal{X})^n$ is a solution of the *variational inequality problem*,

$$\mathbf{g}_{\mathbf{r}}(\mathbf{w}^*, \dot{\mathbf{x}}^*)^\mathsf{T} \left(\begin{bmatrix} \mathbf{w} \\ \dot{\mathbf{x}} \end{bmatrix} - \begin{bmatrix} \mathbf{w}^* \\ \dot{\mathbf{x}}^* \end{bmatrix} \right) \geq 0 \quad \forall (\mathbf{w}, \dot{\mathbf{x}}) \in \mathcal{W} \times \phi(\mathcal{X})^n \tag{21}$$

and vice versa (cf. Proposition 7.1 of Harker and Pang, 1990). The pseudo-gradient $\mathbf{g}_{\mathbf{r}}$ in (21) is defined as in (5) with fixed vector $\mathbf{r} = [r_0, r_1, \ldots, r_n]^\mathsf{T}$ where $r_0 := 1$ and $r_i := \frac{c_{-1,i}}{c_{+1,i}}$ for all $i = 1, \ldots, n$

(cf. Equation 8). Under Assumption 3, this choice of $\mathbf{r}$ ensures that the mapping $\mathbf{g}_{\mathbf{r}}(\mathbf{w},\dot{\mathbf{x}})$ is continuous and strictly monotone (cf. proof of Lemma 2 and Theorem 8). Hence, the variational inequality problem in (21) can be solved by *modified extragradient descent* (see, for instance, Chapter 7.2.3 of Geiger and Kanzow, 1999). Before presenting Algorithm 2, which is an extragradient-based algorithm for the Nash prediction game, let us denote the L^2-projection of $\mathbf{a}$ into the non-empty, compact, and convex set $\mathcal{A}$ by

$$\Pi_{\mathcal{A}}(\mathbf{a}) := \arg\min_{\mathbf{a}'\in\mathcal{A}} \|\mathbf{a}-\mathbf{a}'\|_2^2.$$

Notice, that if $\mathcal{A} := \{\mathbf{a}\in\mathbb{R}^m \mid \|\mathbf{a}\|_2 \leq \kappa\}$ is the closed l^2-ball of radius $\kappa > 0$ and $\mathbf{a}\notin\mathcal{A}$, this projection simply reduces to a rescaling of vector $\mathbf{a}$ to length κ.

Based on this definition of $\Pi_{\mathcal{A}}$, we can now state an iterative method (Algorithm 2), which—apart from back projection steps—does not require solving an optimization problem in each iteration. The proposed algorithm converges to a solution of the variational inequality problem in 21; that is, the unique equilibrium of the Nash prediction game, if Assumptions 2 and 3 hold (cf. Theorem 7.40 of Geiger and Kanzow, 1999).

Algorithm 2 EDS: Extragradient Descent Solver for Nash Prediction Games

Require: Cost functions $\hat{\theta}_v$ as defined in (1) and (2), and action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$.

1: Select initial $\mathbf{w}^{(0)}\in\mathcal{W}$, set $\dot{\mathbf{x}}^{(0)} := \mathbf{x}$, set $k := 0$, and select $\sigma\in(0,1)$ and $\beta\in(0,1)$.

2: Set $r_0 := 1$ and $r_i := \frac{c_{-1,i}}{c_{+1,i}}$ for all $i = 1,\ldots,n$.

3: **repeat**

4: Set $\begin{bmatrix} \mathbf{d}_{-1}^{(k)} \\ \mathbf{d}_{+1}^{(k)} \end{bmatrix} := \Pi_{\mathcal{W}\times\phi(\mathcal{X})^n}\left(\begin{bmatrix} \mathbf{w}^{(k)} \\ \dot{\mathbf{x}}^{(k)} \end{bmatrix} - \mathbf{g}_{\mathbf{r}}\left(\mathbf{w}^{(k)},\dot{\mathbf{x}}^{(k)}\right)\right) - \begin{bmatrix} \mathbf{w}^{(k)} \\ \dot{\mathbf{x}}^{(k)} \end{bmatrix}$.

5: Find maximal step size $t^{(k)}\in\left\{\beta^l \mid l\in\mathbb{N}\right\}$ with

$$-\mathbf{g}_{\mathbf{r}}\left(\mathbf{w}^{(k)}+t^{(k)}\mathbf{d}_{-1}^{(k)},\dot{\mathbf{x}}^{(k)}+t^{(k)}\mathbf{d}_{+1}^{(k)}\right)^{\mathsf{T}}\begin{bmatrix} \mathbf{d}_{-1}^{(k)} \\ \mathbf{d}_{+1}^{(k)} \end{bmatrix} \geq \sigma\left(\left\|\mathbf{d}_{-1}^{(k)}\right\|_2^2 + \left\|\mathbf{d}_{+1}^{(k)}\right\|_2^2\right).$$

6: Set $\begin{bmatrix} \bar{\mathbf{w}}^{(k)} \\ \bar{\mathbf{x}}^{(k)} \end{bmatrix} := \begin{bmatrix} \mathbf{w}^{(k)} \\ \dot{\mathbf{x}}^{(k)} \end{bmatrix} + t^{(k)}\begin{bmatrix} \mathbf{d}_{-1}^{(k)} \\ \mathbf{d}_{+1}^{(k)} \end{bmatrix}$.

7: Set step size of extragradient

$$\gamma^{(k)} := -\frac{t^{(k)}}{\left\|\mathbf{g}_{\mathbf{r}}\left(\bar{\mathbf{w}}^{(k)},\bar{\mathbf{x}}^{(k)}\right)\right\|_2^2}\mathbf{g}_{\mathbf{r}}\left(\bar{\mathbf{w}}^{(k)},\bar{\mathbf{x}}^{(k)}\right)^{\mathsf{T}}\begin{bmatrix} \mathbf{d}_{-1}^{(k)} \\ \mathbf{d}_{+1}^{(k)} \end{bmatrix}.$$

8: Set $\begin{bmatrix} \mathbf{w}^{(k+1)} \\ \dot{\mathbf{x}}^{(k+1)} \end{bmatrix} := \Pi_{\mathcal{W}\times\phi(\mathcal{X})^n}\left(\begin{bmatrix} \mathbf{w}^{(k)} \\ \dot{\mathbf{x}}^{(k)} \end{bmatrix} - \gamma^{(k)}\mathbf{g}_{\mathbf{r}}\left(\bar{\mathbf{w}}^{(k)},\bar{\mathbf{x}}^{(k)}\right)\right)$.

9: Set $k := k+1$.

10: **until** $\left\|\mathbf{w}^{(k)}-\mathbf{w}^{(k-1)}\right\|_2^2 + \left\|\dot{\mathbf{x}}^{(k)}-\dot{\mathbf{x}}^{(k-1)}\right\|_2^2 \leq \varepsilon$.

5. Instances of the Nash Prediction Game

In this section, we present two instances of the Nash prediction game and investigate under which conditions those games possess unique Nash equilibria. We start by specifying both players' loss function and regularizer. An obvious choice for the loss function of the learner $\ell_{-1}(z,y)$ is the *zero-one loss* defined by

$$\ell_{0/1}(z,y) := \begin{cases} 1 & \text{, if } yz < 0 \\ 0 & \text{, if } yz \geq 0 \end{cases} .$$

A possible choice for the data generator's loss is $\ell_{0/1}(z,-1)$ which penalizes positive decision values z, independently of the class label. The rationale behind this choice is that the data generator experiences costs when the learner blocks an event, that is, assigns an instance to the positive class. For instance, a legitimate email sender experiences costs when a legitimate email is erroneously blocked just like an abusive sender, also amalgamated into the *data generator*, experiences costs when spam messages are blocked. However, the zero-one loss violates Assumption 2 as it is neither convex nor twice continuously differentiable. In the following sections, we therefore approximate the zero-one loss by the *logistic loss* and a newly derived *trigonometric loss*, which both satisfy Assumption 2.

Recall that $\hat{\Omega}_{+1}(D,\dot{D})$ is an estimate of the transformation costs that the data generator incurs when transforming the distribution that generates the instances $\mathbf{x}_i$ at training time into the distribution that generates the instances $\dot{\mathbf{x}}_i$ at application time. In our analysis, we approximate these costs by the average squared l^2-distance between x_i and $\dot{x}_i$ in the feature space induced by mapping ϕ, that is,

$$\hat{\Omega}_{+1}(D,\dot{D}) := \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2} \|\phi(\dot{x}_i) - \phi(x_i)\|_2^2 . \tag{22}$$

The learner's regularizer $\hat{\Omega}_{-1}(\mathbf{w})$ penalizes the complexity of the predictive model $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$. We consider Tikhonov regularization, which, for linear decision functions $f_{\mathbf{w}}$, reduces to the squared l^2-norm of $\mathbf{w}$,

$$\hat{\Omega}_{-1}(\mathbf{w}) := \frac{1}{2}\|\mathbf{w}\|_2^2 . \tag{23}$$

Before presenting the *Nash logistic regression* (NLR) and the *Nash support vector machine* (NSVM), we turn to a discussion on the applicability of general kernel functions.

5.1 Applying Kernels

So far, we assumed the knowledge of feature mapping $\phi : \mathcal{X} \to \phi(\mathcal{X})$ such that we can compute an explicit feature representation $\phi(x_i)$ of the training instances x_i for all $i = 1,\ldots,n$. However, in some applications, such a feature mapping is unwieldy or hard to identify. Instead, one is often equipped with a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which measures the similarity between two instances. Generally, kernel function k is assumed to be a positive-semidefinite kernel such that it can be stated in terms of a scalar product in the corresponding reproducing kernel Hilbert space, that is, $\exists\phi$ with $k(x,x') = \phi(x)^{\mathsf{T}}\phi(x')$.

To apply the representer theorem (see, e.g., Schölkopf et al., 2001) we assume that the transformed instances lie in the span of the mapped training instances, that is, we restrict the data generator's action space such that the transformed instances $\dot{x}_i$ are mapped into the same subspace of the reproducing kernel Hilbert space as the unmodified training instances x_i. By this assumption, the

weight vector $\mathbf{w} \in \mathcal{W}$ and the transformed instances $\phi(\dot{x}_i) \in \phi(\mathcal{X})$ for $i = 1,\ldots,n$ can be expressed as linear combinations of the mapped training instances, that is, $\exists \alpha_i, \Xi_{ij}$ such that

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(x_i) \quad \text{and} \quad \phi(\dot{x}_j) = \sum_{i=1}^{n} \Xi_{ij} \phi(x_i) \quad \forall\, j = 1,\ldots,n.$$

Further, let us assume that the action spaces $\mathcal{W}$ and $\phi(\mathcal{X})^n$ can be adequately translated into dual action spaces $\mathcal{A} \subset \mathbb{R}^n$ and $\mathcal{Z} \subset \mathbb{R}^{n\times n}$, which is possible, for instance, if $\mathcal{W}$ and $\phi(\mathcal{X})^n$ are closed l^2-balls. Then, a kernelized variant of the Nash prediction game is obtained by inserting the above equations into the players' cost functions in (1) and (2) with regularizers in (22) and (23),

$$\hat{\theta}_{-1}(\alpha,\Xi) = \sum_{i=1}^{n} c_{-1,i}\ell_{-1}(\alpha^\top \mathbf{K}\Xi\mathbf{e}_i, y_i) + \rho_{-1}\frac{1}{2}\alpha^\top \mathbf{K}\alpha, \tag{24}$$

$$\hat{\theta}_{+1}(\alpha,\Xi) = \sum_{i=1}^{n} c_{+1,i}\ell_{+1}(\alpha^\top \mathbf{K}\Xi\mathbf{e}_i, y_i) + \rho_{+1}\frac{1}{2n}\mathrm{tr}\left((\Xi-\mathbf{I}_n)^\top \mathbf{K}(\Xi-\mathbf{I}_n)\right), \tag{25}$$

where $\mathbf{e}_i \in \{0,1\}^n$ is the i-th unit vector, $\alpha \in \mathcal{A}$ is the dual weight vector, $\Xi \in \mathcal{Z}$ is the dual transformed data matrix, and $\mathbf{K} \in \mathbb{R}^{n\times n}$ is the kernel matrix with $K_{ij} := k(x_i, x_j)$. In the dual Nash prediction game with cost functions (24) and (25), the learner chooses the dual weight vector $\alpha = [\alpha_1,\ldots,\alpha_n]^\top$ and classifies a new instance x by $h(x) = \mathrm{sign}\, f_\alpha(x)$ with $f_\alpha(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$. In contrast, the data generator chooses the dual transformed data matrix Ξ, which implicitly reflects the change of the training distribution.

Their transformation costs are in proportion to the deviation of Ξ from the identity matrix $\mathbf{I}_n$, where if Ξ equals $\mathbf{I}_n$, the learner's task reduces to standard kernelized empirical risk minimization. The proposed Algorithms 1 and 2 can be readily applied when replacing $\mathbf{w}$ by α and $\dot{\mathbf{x}}_i$ by $\Xi\mathbf{e}_i$ for all $i = 1,\ldots,n$.

An alternative approach to a kernelization of the Nash prediction game is to first construct an explicit feature representation with respect to the given kernel function k and the training instances and then to train the Nash model by applying this feature mapping. Here, we again assume that the transformed instances $\phi(\dot{x}_i)$ as well as the weight vector $\mathbf{w}$ lie in the span of the explicitly mapped training instances $\phi(x)$. Let us consider the kernel PCA map (see, e.g., Schölkopf and Smola, 2002) defined by

$$\phi_{\mathrm{PCA}} : x \mapsto \Lambda^{\frac{1}{2}^+} \mathbf{V}^\top [k(x_1,x),\ldots,k(x_n,x)]^\top, \tag{26}$$

where $\mathbf{V}$ is the column matrix of eigenvectors of kernel matrix $\mathbf{K}$, Λ is the diagonal matrix with the corresponding eigenvalues such that $\mathbf{K} = \mathbf{V}\Lambda\mathbf{V}^\top$, and $\Lambda^{\frac{1}{2}^+}$ denotes the pseudo-inverse of the square root of Λ with $\Lambda = \Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}$.

Remark 9 Notice that for any positive-semidefinite kernel function $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ and fixed training instances $x_1,\ldots,x_n \in \mathcal{X}$, the PCA map is a uniquely defined real function with $\phi_{\mathrm{PCA}} : \mathcal{X} \to \mathbb{R}^n$ such that $k(x_i,x_j) = \phi_{\mathrm{PCA}}(x_i)^\top \phi_{\mathrm{PCA}}(x_j)$ for any $i,j \in \{1,\ldots,n\}$: We first show that ϕ_{PCA} is a real mapping from the input space $\mathcal{X}$ to the Euclidean space $\mathbb{R}^n$. As $x \mapsto [k(x_1,x),\ldots,k(x_n,x)]^\top$ is a real vector-valued function and $\mathbf{V}$ is a real $n\times n$ matrix, it remains to show that the pseudo-inverse of $\Lambda^{\frac{1}{2}}$ is real as well. Since the kernel function is positive-semidefinite, all eigenvalues λ_i of $\mathbf{K}$ are non-negative, and hence, $\Lambda^{\frac{1}{2}}$ is a diagonal matrix with real diagonal entries $\sqrt{\lambda_i}$ for $i = 1,\ldots,n$. The

pseudo-inverse of this matrix is the uniquely defined diagonal matrix $\Lambda^{\frac{1}{2}^+}$ with real non-negative diagonal entries $\frac{1}{\sqrt{\lambda_i}}$ if $\lambda_i > 0$ and zero otherwise. This proves the first claim. The PCA map also satisfies $k(x_i,x_j) = \phi_{\mathrm{PCA}}(x_i)^\mathsf{T}\phi_{\mathrm{PCA}}(x_j)$ for any pair of training instances x_i and x_j as

$$\begin{aligned}\phi_{PCA}(x_i) &= \Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\left[k(x_1,x_i),\ldots,k(x_n,x_i)\right]^\mathsf{T}\\ &= \Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\mathbf{K}\mathbf{e}_i\\ &= \Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\mathbf{V}\Lambda\mathbf{V}^\mathsf{T}\mathbf{e}_i\\ &= \Lambda^{\frac{1}{2}^+}\Lambda\mathbf{V}^\mathsf{T}\mathbf{e}_i\end{aligned}$$

for all $i = 1,\ldots,n$ and consequently

$$\begin{aligned}\phi_{PCA}(x_i)^\mathsf{T}\phi_{PCA}(x_j) &= \mathbf{e}_i^\mathsf{T}\mathbf{V}\Lambda\Lambda^{\frac{1}{2}^+}\Lambda^{\frac{1}{2}^+}\Lambda\mathbf{V}^\mathsf{T}\mathbf{e}_j\\ &= \mathbf{e}_i^\mathsf{T}\mathbf{V}\Lambda\Lambda^{+}\Lambda\mathbf{V}^\mathsf{T}\mathbf{e}_j\\ &= \mathbf{e}_i^\mathsf{T}\mathbf{V}\Lambda\mathbf{V}^\mathsf{T}\mathbf{e}_j\\ &= \mathbf{e}_i^\mathsf{T}\mathbf{K}\mathbf{e}_j = \mathbf{K}_{ij} = k(x_i,x_j)\end{aligned}$$

which proves the second claim. ◇

An equilibrium strategy pair $\mathbf{w}^* \in \mathcal{W}$ and $[\phi_{\mathrm{PCA}}(\dot{x}_1^*)^\mathsf{T},\ldots,\phi_{\mathrm{PCA}}(\dot{x}_n^*)^\mathsf{T}]^\mathsf{T} \in \phi(\mathcal{X})^n$ can be identified by applying the PCA map together with Algorithms 1 or 2. To classify a new instance $x \in \mathcal{X}$ we may first map x into the PCA map-induced feature space and apply the linear classifier $h(x) = \operatorname{sign} f_{\mathbf{w}^*}(x)$ with $f_{\mathbf{w}^*}(x) = \mathbf{w}^{*\mathsf{T}}\phi_{\mathrm{PCA}}(x)$. Alternatively, we can derive a dual representation of $\mathbf{w}^*$ such that $\mathbf{w}^* = \sum_{i=1}^n \alpha_i^*\phi_{\mathrm{PCA}}(x_i)$, and consequently $f_{\mathbf{w}^*}(x) = f_{\alpha^*}(x) = \sum_{i=1}^n \alpha_i^* k(x_i,x)$, where $\alpha^* = [\alpha_1^*,\ldots,\alpha_n^*]^\mathsf{T}$ is a not necessarily uniquely defined dual weight vector of $\mathbf{w}^*$. Therefore, we have to identify a solution α^* of the linear system

$$\mathbf{w}^* = \Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\mathbf{K}\alpha^*. \tag{27}$$

A direct calculation shows that

$$\alpha^* := \mathbf{V}\Lambda^{\frac{1}{2}^+}\mathbf{w}^* \tag{28}$$

is a solution of (27) provided that either all elements λ_i of the diagonal matrix Λ are positive or that $\lambda_i = 0$ implies that the same component of the vector $\mathbf{w}^*$ is also equal to zero (in which case the solution is non-unique). In fact, inserting (28) in (27) then gives

$$\Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\mathbf{K}\alpha^* = \Lambda^{\frac{1}{2}^+}\mathbf{V}^\mathsf{T}\mathbf{V}\Lambda\mathbf{V}^\mathsf{T}\mathbf{V}\Lambda^{\frac{1}{2}^+}\mathbf{w}^* = \Lambda^{\frac{1}{2}^+}\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}^+}\mathbf{w}^* = \mathbf{w}^*$$

whereas in the other cases the linear system (27) is obviously inconsistent. The advantage of the latter approach is that classifying a new instances $x \in \mathcal{X}$ requires the computation of the scalar product $\sum_{i=1}^n \alpha_i^* k(x_i,x)$ rather than a matrix multiplication when mapping x into the PCA map-induced feature space (cf. Equation 26).

When implementing a kernelized solution, the data generator has to generate instances in the input space with dual representation $\mathbf{K}\Xi^*\mathbf{e}_1,\ldots,\mathbf{K}\Xi^*\mathbf{e}_n$ and $\phi_{\mathrm{PCA}}(\dot{x}_1^*),\ldots,\phi_{\mathrm{PCA}}(\dot{x}_n^*)$, respectively. To this end, the data generator must solve a pre-image problem which typically has a non-unique solution. However, as every solution of this problem incurs the same costs to both players the data generator is free to select any of them. To find such a solution, the data generator may solve a non-convex optimization problem as proposed by Mika et al. (1999), or may apply a non-iterative method (Kwok and Tsang, 2003) based on multidimensional scaling.

5.2 Nash Logistic Regression

In this section we study the particular instance of the Nash prediction game where each players' loss function rests on the negative logarithm of the logistic function $\sigma(a) := \frac{1}{1+e^{-a}}$, that is, the *logistic loss*

$$\ell^{\mathrm{l}}(z,y) := -\log\sigma(yz) = \log\left(1+e^{-yz}\right). \tag{29}$$

We consider the regularizers in (22) and (23), respectively, which give rise to the following definition of the *Nash logistic regression* (NLR).

In the following definition, column vectors $\mathbf{x} := [\mathbf{x}_1^\mathsf{T},\ldots,\mathbf{x}_n^\mathsf{T}]^\mathsf{T}$ and $\dot{\mathbf{x}} := [\dot{\mathbf{x}}_1^\mathsf{T},\ldots,\dot{\mathbf{x}}_n^\mathsf{T}]^\mathsf{T}$ again denote the concatenation of the original and the transformed training instances, respectively, which are mapped into the feature space by $\mathbf{x}_i := \phi(x_i)$ and $\dot{\mathbf{x}}_i := \phi(\dot{x}_i)$.

Definition 10 *The* Nash logistic regression *(NLR) is an instance of the Nash prediction game with non-empty, compact, and convex action spaces $\mathcal{W} \subset \mathbb{R}^m$ and $\phi(\mathcal{X})^n \subset \mathbb{R}^{m\cdot n}$ and cost functions*

$$\begin{aligned}
\hat{\theta}^{\mathrm{l}}_{-1}(\mathbf{w},\dot{\mathbf{x}}) &:= \sum_{i=1}^n c_{-1,i}\ell^{\mathrm{l}}(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i, y_i) + \rho_{-1}\frac{1}{2}\|\mathbf{w}\|_2^2 \\
\hat{\theta}^{\mathrm{l}}_{+1}(\mathbf{w},\dot{\mathbf{x}}) &:= \sum_{i=1}^n c_{+1,i}\ell^{\mathrm{l}}(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i, -1) + \rho_{+1}\frac{1}{n}\sum_{i=1}^n \frac{1}{2}\|\dot{\mathbf{x}}_i - \mathbf{x}_i\|_2^2
\end{aligned}$$

where ℓ^{l} is specified in (29)*.*

As in our introductory discussion, the data generator's loss function $\ell_{+1}(z,y) := \ell^{\mathrm{l}}(z,-1)$ penalizes positive decision values independently of the class label y. In contrast, instances that pass the classifier, that is, instances with negative decision values, incur little or almost no costs. By the above definition, the Nash logistic regression obviously satisfies Assumption 2, and according to the following corollary, also satisfies Assumption 3 for suitable regularization parameters.

Corollary 11 *Let the Nash logistic regression be specified as in Definition 10 with positive regularization parameters ρ_{-1} and ρ_{+1} which satisfy*

$$\rho_{-1}\rho_{+1} \geq n\mathbf{c}_{-1}^\mathsf{T}\mathbf{c}_{+1}, \tag{30}$$

then Assumption 2 and 3 hold, and consequently, the Nash logistic regression possess a unique Nash equilibrium.

Proof. By Definition 10, both players employ the logistic loss with $\ell_{-1}(z,y) := \ell^{\mathrm{l}}(z,y)$ and $\ell_{+1}(z,y) := \ell^{\mathrm{l}}(z,-1)$ and the regularizers in (22) and (23), respectively. Let

$$\begin{array}{rcl|rcl}
\ell'_{-1}(z,y) &=& -y\frac{1}{1+e^{yz}} & \ell'_{+1}(z,y) &=& \frac{1}{1+e^{-z}} \\
\ell''_{-1}(z,y) &=& \frac{1}{1+e^{z}}\frac{1}{1+e^{-z}} & \ell''_{+1}(z,y) &=& \frac{1}{1+e^{z}}\frac{1}{1+e^{-z}}
\end{array} \tag{31}$$

denote the first and second derivatives of the players' loss functions with respect to $z \in \mathbb{R}$. Further, let

$$\begin{array}{rcl|rcl}
\nabla_{\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) &=& \mathbf{w} & \nabla_{\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) &=& \frac{1}{n}(\dot{\mathbf{x}}-\mathbf{x}) \\
\nabla^2_{\mathbf{w},\mathbf{w}}\hat{\Omega}_{-1}(\mathbf{w}) &=& \mathbf{I}_m & \nabla^2_{\dot{\mathbf{x}},\dot{\mathbf{x}}}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) &=& \frac{1}{n}\mathbf{I}_{m\cdot n}
\end{array}$$

denote the gradients and Hessians of the players' regularizers. Assumption 2 holds as:

1. The the second derivatives of $\ell_{-1}(z,y)$ and $\ell_{+1}(z,y)$ are positive and continuous for all $z \in \mathbb{R}$ and $y \in \mathcal{Y}$. Consequently, $\ell_v(z,y)$ is convex and twice continuously differentiable with respect to z for $v \in \{-1,+1\}$ and fixed y.

2. The Hessians of the players' regularizers are fixed, positive definite matrices and consequently both regularizers are twice continuously differentiable and uniformly strongly convex in $\mathbf{w} \in \mathcal{W}$ and $\dot{\mathbf{x}} \in \phi(\mathcal{X})^n$ (for any fixed $\mathbf{x} \in \phi(\mathcal{X})^n$), respectively.

3. By Definition 10, the players' action sets are non-empty, compact, and convex subsets of finite-dimensional Euclidean spaces.

Assumption 3 holds as for all $z \in \mathbb{R}$ and $y \in \mathcal{Y}$:

1. The second derivatives of $\ell_{-1}(z,y)$ and $\ell_{+1}(z,y)$ in (31) are equal.

2. The sum of the first derivatives of the loss functions is bounded,

$$\ell'_{-1}(z,y) + \ell'_{+1}(z,y) = -y\frac{1}{1+e^{yz}} + \frac{1}{1+e^{-z}} = \begin{cases} \frac{1-e^{-z}}{1+e^{-z}} & \text{, if } y = +1 \\ \frac{2}{1+e^{-z}} & \text{, if } y = -1 \end{cases} \in (-1,2),$$

   which together with Equation 14 gives

$$\tau = \sup_{(\mathbf{x},y) \in \phi(\mathcal{X}) \times \mathcal{Y}} \frac{1}{2} \left| \ell'_{-1}(f_{\mathbf{w}}(\mathbf{x}),y) + \ell'_{+1}(f_{\mathbf{w}}(\mathbf{x}),y) \right| < 1.$$

   The supremum τ is strictly less than 1 since $f_{\mathbf{w}}(\mathbf{x})$ is finite for compact action sets $\mathcal{W}$ and $\phi(\mathcal{X})^n$. The smallest eigenvalues of the players' regularizers are $\lambda_{-1} = 1$ and $\lambda_{+1} = \frac{1}{n}$, such that inequalities

$$\rho_{-1}\rho_{+1} \geq n\mathbf{c}_{-1}^{\mathsf{T}}\mathbf{c}_{+1} > \tau^2 \frac{1}{\lambda_{-1}\lambda_{+1}} \mathbf{c}_{-1}^{\mathsf{T}}\mathbf{c}_{+1}$$

   hold.

3. The partial gradient $\nabla_{\dot{\mathbf{x}}_i} \hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) = \frac{1}{n}(\dot{\mathbf{x}}_i - \mathbf{x}_i)$ of the data generator's regularizer is independent of $\dot{\mathbf{x}}_j$ for all $j \neq i$ and $i = 1,\ldots,n$.

As Assumptions 2 and 3 are satisfied, the existence of a unique Nash equilibrium follows immediately from Theorem 8. ■

Recall, that the weighting factors $c_{v,i}$ are strictly positive with $\sum_{i=1}^n c_{v,i} = 1$ for both players $v \in \{-1,+1\}$. In particular, it therefore follows that in the unweighted case where $c_{v,i} = \frac{1}{n}$ for all $i = 1,\ldots,n$ and $v \in \{-1,+1\}$, a sufficient condition to ensure the existence of a unique Nash equilibrium is to set the learner's regularization parameter to $\rho_{-1} \geq \frac{1}{\rho_{+1}}$.

5.3 Nash Support Vector Machine

The Nash logistic regression tends to non-sparse solutions. This becomes particularly apparent if the Nash equilibrium $(\mathbf{w}^*,\dot{\mathbf{x}}^*)$ is an interior point of the joint action set $\mathcal{W} \times \phi(\mathcal{X})^n$ in which case the (partial) gradients in (9) and (10) are zero at $(\mathbf{w}^*,\dot{\mathbf{x}}^*)$. For regularizer (23), this implies that $\mathbf{w}^*$ is

a linear combination of the transformed instances $\dot{\mathbf{x}}_i$ where all weighting factors are non-zero since the first derivative of the logistic loss as well as the cost factors $c_{-1,i}$ are non-zero for all $i = 1,\ldots,n$. The *support vector machine* (SVM), which employs the *hinge loss*,

$$\ell^{\mathrm{h}}(z,y) := \max(0, 1-yz) = \begin{cases} 1-yz & , \text{if } yz < 1 \\ 0 & , \text{if } yz \geq 1 \end{cases},$$

does not suffer from non-sparsity, however, the hinge loss obviously violates Assumption 2 as it is not twice continuously differentiable. Therefore, we propose a twice continuously differentiable loss function that we call *trigonometric loss*, which satisfies Assumptions 2 and 3.

Definition 12 *For any fixed smoothness factor $s > 0$, the* trigonometric loss *is defined by*

$$\ell^{\mathrm{t}}(z,y) := \begin{cases} -yz & , \textit{if } yz < -s \\ \frac{s-yz}{2} - \frac{s}{\pi}\cos\left(\frac{\pi}{2s}yz\right) & , \textit{if } |yz| \leq s \\ 0 & , \textit{if } yz > s \end{cases}. \tag{32}$$

The trigonometric loss is similar to the hinge loss in that it, except around the decision boundary, penalizes misclassifications in proportion to the decision value $z \in \mathbb{R}$ and attains zero for correctly classified instances. Analogous to the once continuously differentiable Huber loss where a polynomial is embedded into the hinge loss, the trigonometric loss combines the perceptron loss $\ell^{\mathrm{p}}(z,y) := \max(0,-yz)$ with a trigonometric function. This trigonometrical embedding yields a convex, twice continuously differentiable function.

Lemma 13 *The trigonometric loss $\ell^{\mathrm{t}}(z,y)$ is convex and twice continuously differentiable with respect to $z \in \mathbb{R}$ for any fixed $y \in \mathcal{Y}$.*

Proof. Let

$$\ell^{\mathrm{t}\prime}(z,y) = \begin{cases} -y & , \text{if } yz < -s \\ -\frac{1}{2}y + \frac{1}{2}y\sin\left(\frac{\pi}{2s}yz\right) & , \text{if } |yz| \leq s \\ 0 & , \text{if } yz > s \end{cases}$$

$$\ell^{\mathrm{t}\prime\prime}(z,y) = \begin{cases} 0 & , \text{if } yz < -s \\ \frac{\pi}{4s}\cos\left(\frac{\pi}{2s}yz\right) & , \text{if } |yz| \leq s \\ 0 & , \text{if } yz > s \end{cases}$$

denote the first and second derivatives of $\ell^{\mathrm{t}}(z,y)$, respectively, with respect to $z \in \mathbb{R}$. The trigonometric loss $\ell^{\mathrm{t}}(z,y)$ is convex in $z \in \mathbb{R}$ (for any fixed $y \in \mathcal{Y}$) as the second derivative $\ell^{\mathrm{t}\prime\prime}(z,y)$ is strictly positive if $|z| = |yz| < s$ and zero otherwise. Moreover, since the second derivative is continuous,

$$\lim_{|z|\to s-} \ell^{\mathrm{t}\prime\prime}(z,y) = \frac{\pi}{4s}\cos\left(\pm\frac{\pi}{2}\right) = 0 = \lim_{|z|\to s+} \ell^{\mathrm{t}\prime\prime}(z,y),$$

the trigonometric loss is also twice continuously differentiable. ■

Because of the similarities of the loss functions, we call the Nash prediction game that is based upon the trigonometric loss *Nash support vector machine* (NSVM) where we again consider the regularizers in (22) and (23).

Definition 14 *The* Nash support vector machine *(NSVM) is an instance of the Nash prediction game with non-empty, compact, and convex action spaces $\mathcal{W} \subset \mathbb{R}^m$ and $\phi(\mathcal{X})^n \subset \mathbb{R}^{m \cdot n}$ and cost functions*

$$\begin{aligned} \hat{\theta}^{\mathrm{t}}_{-1}(\mathbf{w}, \dot{\mathbf{x}}) &:= \sum_{i=1}^{n} c_{-1,i} \ell^{\mathrm{t}}(\mathbf{w}^{\mathsf{T}} \dot{\mathbf{x}}_i, y_i) + \rho_{-1} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \hat{\theta}^{\mathrm{t}}_{+1}(\mathbf{w}, \dot{\mathbf{x}}) &:= \sum_{i=1}^{n} c_{+1,i} \ell^{\mathrm{t}}(\mathbf{w}^{\mathsf{T}} \dot{\mathbf{x}}_i, -1) + \rho_{+1} \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \|\dot{\mathbf{x}}_i - \mathbf{x}_i\|_2^2 \end{aligned} \tag{33}$$

where ℓ^{t} is specified in (32).

The following corollary states sufficient conditions under which the Nash support vector machine satisfies Assumptions 2 and 3, and consequently has a unique Nash equilibrium.

Corollary 15 *Let the Nash support vector machine be specified as in Definition 14 with positive regularization parameters ρ_{-1} and ρ_{+1} which satisfy*

$$\rho_{-1} \rho_{+1} > n \mathbf{c}_{-1}^{\mathsf{T}} \mathbf{c}_{+1}, \tag{34}$$

then Assumptions 2 and 3 hold, and consequently, the Nash support vector machine has a unique Nash equilibrium.

Proof. By Definition 14, both players employ the trigonometric loss with $\ell_{-1}(z,y) := \ell^{\mathrm{t}}(z,y)$ and $\ell_{+1}(z,y) := \ell^{\mathrm{t}}(z,-1)$ and the regularizers in (22) and (23), respectively. Assumption 2 holds:

1. According to Lemma 13, $\ell^{\mathrm{t}}(z,y)$, and consequently $\ell_{-1}(z,y)$ and $\ell_{+1}(z,y)$, are convex and twice continuously differentiable with respect to $z \in \mathbb{R}$ (for any fixed $y \in \{-1,+1\}$).

2. The regularizers of the Nash support vector machine are equal to that of the Nash logistic regression and possess the same properties as in Theorem 11.

3. By Definition 14, the players' action sets are non-empty, compact, and convex subsets of finite-dimensional Euclidean spaces.

Assumption 3 holds:

1. The second derivatives of $\ell_{-1}(z,y)$ and $\ell_{+1}(z,y)$ are equal for all $z \in \mathbb{R}$ since

$$\ell^{\mathrm{t}\prime\prime}(z,y) = \begin{cases} \frac{\pi}{4s} \cos\left(\frac{\pi}{2s} z\right) & , \text{if } |z| \leq s \\ 0 & , \text{if } |z| > s \end{cases}$$

   does not dependent on $y \in \mathcal{Y}$.

2. The sum of the first derivatives of the loss functions is bounded as for $y = -1$:

$$\ell'_{-1}(z,-1) + \ell'_{+1}(z,-1) = 2\ell^{\mathrm{t}\prime}(z,-1) = \begin{cases} 0 & , \text{if } z < -s \\ 1 - \sin\left(-\frac{\pi}{2s} z\right) & , \text{if } |z| \leq s \\ 2 & , \text{if } z > s \end{cases} \in [0,2],$$

and for $y = +1$:

$$\ell'_{-1}(z,+1) + \ell'_{+1}(z,+1) = \begin{cases} -1 & \text{, if } z < -s \\ \sin\left(\frac{\pi}{2s}z\right) & \text{, if } |z| \leq s \\ 1 & \text{, if } z > s \end{cases} \in [-1,1].$$

Together with Equation 14, it follows that

$$\tau = \sup_{(\mathbf{x},y)\in\phi(\mathcal{X})\times\mathcal{Y}} \frac{1}{2}\left|\ell'_{-1}(f_{\mathbf{w}}(\mathbf{x}),y) + \ell'_{+1}(f_{\mathbf{w}}(\mathbf{x}),y)\right| \leq 1.$$

The smallest eigenvalues of the players' regularizers are $\lambda_{-1} = 1$ and $\lambda_{+1} = \frac{1}{n}$, such that inequalities

$$\rho_{-1}\rho_{+1} > n\mathbf{c}_{-1}^{\mathsf{T}}\mathbf{c}_{+1} \geq \tau^2 \frac{1}{\lambda_{-1}\lambda_{+1}}\mathbf{c}_{-1}^{\mathsf{T}}\mathbf{c}_{+1}$$

hold.

3. As for Nash logistic regression, the partial gradient $\nabla_{\dot{\mathbf{x}}_i}\hat{\Omega}_{+1}(\mathbf{x},\dot{\mathbf{x}}) = \frac{1}{n}(\dot{\mathbf{x}}_i - \mathbf{x}_i)$ of the data generator's regularizer is independent of $\dot{\mathbf{x}}_j$ for all $j \neq i$ and $i = 1,\ldots,n$.

Because Assumptions 2 and 3 are satisfied, the existence of a unique Nash equilibrium follows immediately from Theorem 8. ■

6. Experimental Evaluation

The goal of this section is to explore the relative strengths and weaknesses of the discussed instances of the Nash prediction game and existing reference methods in the context of email spam filtering. We compare a regular *support vector machine* (SVM), *logistic regression* (LR), the *SVM with trigonometric loss* (SVMT, a variant of the SVM which minimizes (33) for the given training data), the worst-case solution *SVM for invariances with feature removal* (Invar-SVM, Globerson and Roweis, 2006; Teo et al., 2007), and the Nash equilibrium strategies *Nash logistic regression* (NLR) and *Nash support vector machine* (NSVM).

data set	*instances*	*features*	*delivery period*
ESP	169,612	541,713	01/06/2007 - 27/04/2010
Mailinglist	128,117	266,378	01/04/1999 - 31/05/2006
Private	108,178	582,100	01/08/2005 - 31/03/2010
TREC 2007	75,496	214,839	04/08/2007 - 07/06/2007

Table 1: Data sets used in the experiments.

We use four corpora of chronologically sorted emails detailed in Table 1: The first data set contains emails of an email service provider (ESP) collected between 2007 and 2010. The second (Mailinglist) is a collection of emails from publicly available mailing lists augmented by spam emails from Bruce Guenter's spam trap of the same time period. The third corpus (Private) contains

newsletters and spam and non-spam emails of the authors. The last corpus is the NIST TREC 2007 spam corpus.

Feature mapping $\phi(x)$ is defined such that email $x \in \mathcal{X}$ is tokenized with the X-tokenizer (Siefkes et al., 2004) and converted into the m-dimensional binary bag-of-word vector $\mathbf{x} := [0,1]^m$. The value of m is determined by the number of distinct terms in the data set where we have removed all terms which occur only once. For each experiment and each repetition, we then construct the PCA mapping (26) with respect to the corresponding n training emails using the linear kernel $k(\mathbf{x},\mathbf{x}') := \mathbf{x}^\top\mathbf{x}'$ resulting in n-dimensional training instances $\phi_{\text{PCA}}(x_i) \in \mathbb{R}^n$ for $i = 1,\ldots,n$. To ensure the convexity as well as the compactness requirement in Assumption 2, we notionally restrict the players' action sets by defining $\phi(\mathcal{X}) := \{\phi_{\text{PCA}}(x) \in \mathbb{R}^n \mid \|\phi_{\text{PCA}}(x)\|_2^2 \leq \kappa\}$ and $\mathcal{W} := \{\mathbf{w} \in \mathbb{R}^n \mid \|\mathbf{w}\|_2^2 \leq \kappa\}$ for some fixed constant κ. Note that by choosing an arbitrarily large κ, the players' action sets become effectively unbounded.

For both algorithms, ILS and EDS, we set $\sigma := 0.001$, $\beta := 0.2$, and $\varepsilon := 10^{-14}$. The algorithms are stopped if l exceeds 30 in line 6 of ILS and line 5 in EDS, respectively; in this case, no convergence is achieved. In all experiments, we use the F-measure—that is, the harmonic mean of precision and recall—as evaluation measure and tune all parameters with respect to likelihood. The particular protocol and results of each experiment are detailed in the following sections.

6.1 Convergence

Corollaries 11 (for Nash logistic regression) and 15 (for the Nash support vector machine) specify conditions on the regularization parameters ρ_{-1} and ρ_{+1} under which a unique Nash equilibrium necessarily exists. When this is the case, both the ILS and EDS algorithms will converge on that Nash equilibrium. In the first set of experiments, we study whether repeated restarts of the algorithm converge on the same equilibrium when the bounds in Equations 30 and 34 are satisfied, and when they are violated to increasingly large degrees.

We set $c_{v,i} := \frac{1}{n}$ for $v \in \{-1,+1\}$ and $i = 1,\ldots,n$, such that for $\rho_{-1} > \frac{1}{\rho_{+1}}$ both bounds (Equations 30 and 34) are satisfied. For each value of ρ_{-1} and ρ_{+1} and each of 10 repetitions, we randomly draw 400 emails from the data set and run EDS with randomly chosen initial solutions $(\mathbf{w}^{(0)},\dot{\mathbf{x}}^{(0)})$ until convergence. We run ILS on the same training set; in each repetition, we randomly choose a distinct initial solution, and after each iteration k we compute the Euclidean distance between the EDS solution and the current ILS iterate $\mathbf{w}^{(k)}$. Figure 1 reports on these average Euclidean distances between distinctly initialized runs. The blue curves ($\rho_{-1} = 2\frac{1}{\rho_{+1}}$) satisfy Equations 30 and 34, the yellow curves ($\rho_{-1} = \frac{1}{\rho_{+1}}$) lie exactly on the boundary; all other curves violate the bounds. Dotted lines show the Euclidean distance between the Nash equilibrium and the solution of logistic regression.

Our findings are as follows. Logistic regression and regular SVM never coincide with the Nash equilibrium—the Euclidean distances lie in the range between 10^{-2} and 2. ILS and EDS always converge to identical equilibria when (30) and (34) are satisfied (blue and yellow curves). The Euclidean distances lie at the threshold of numerical computing accuracy. When Equations 30 and 34 are violated by a factor up to 4 (turquoise and red curves), all repetitions still converge on the same equilibrium, indicating that the equilibrium is either still unique or a secondary equilibrium is unlikely to be found. When the bounds are violated by a factor of 8 or 16 (green and purple curves), then some repetitions of the learning algorithms do not converge or start to converge to

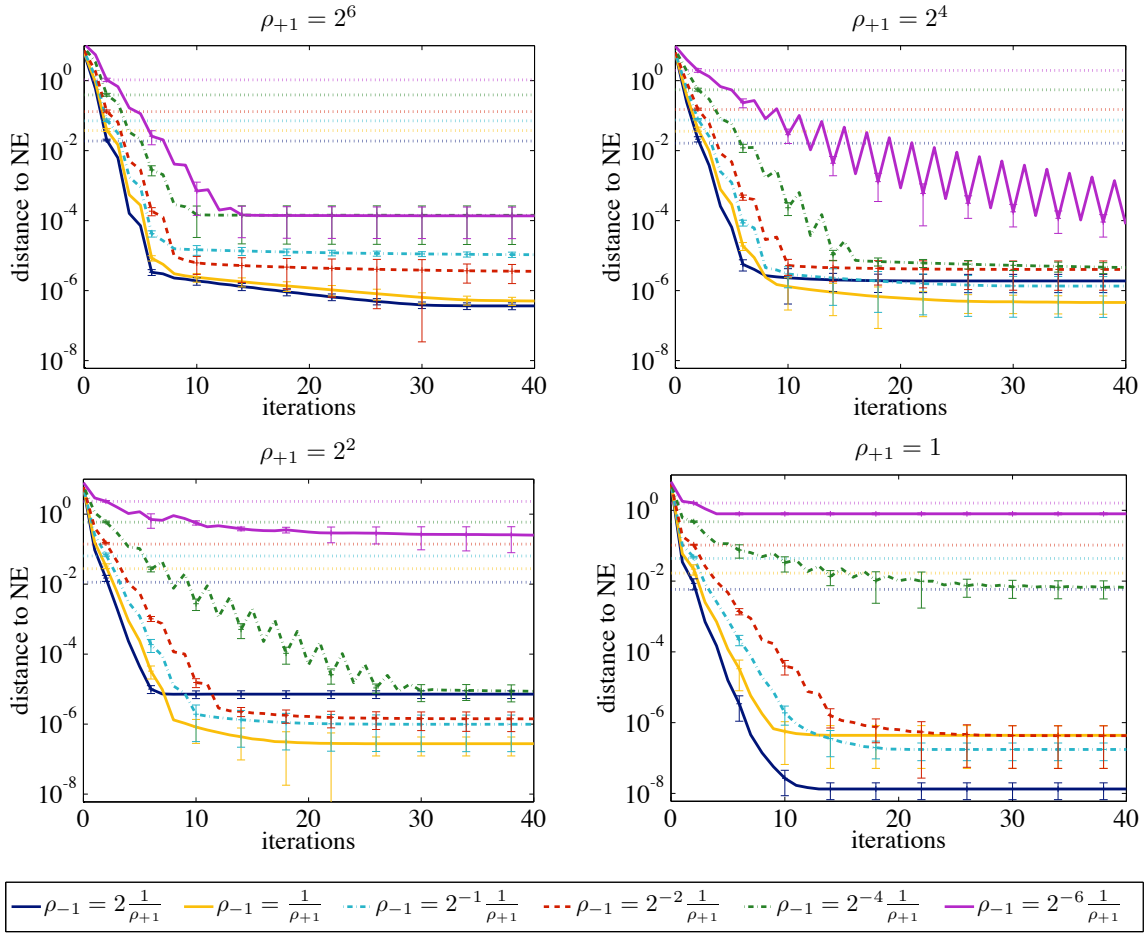


Figure 1: Average Euclidean distance between the EDS solution and the ILS solution at iteration $k = 0, \ldots, 40$ for Nash logistic regression on the ESP corpus. The dotted lines show the distance between the EDS solution and the solution of logistic regression. Error bars indicate standard deviation.

distinct equilibria. In the latter case, learner and data generator may attain distinct equilibria and may experience an arbitrarily poor outcome when playing a Nash equilibrium.

6.2 Regularization Parameters

The regularization parameters ρ_v of the players $v \in \{-1, +1\}$ play a major role in the prediction game. The learner's regularizer determines the generalization ability of the predictive model and the data generator's regularizer controls the amount of change in the data generation process. In order to tune these parameter, one would need to have access to labeled data that are governed by the transformed input distribution. In our second experiment, we will explore to which extent those parameters can be estimated using a portion of the newest training data. Intuitively, the most recent training data may be more similar to the test data than older training data.

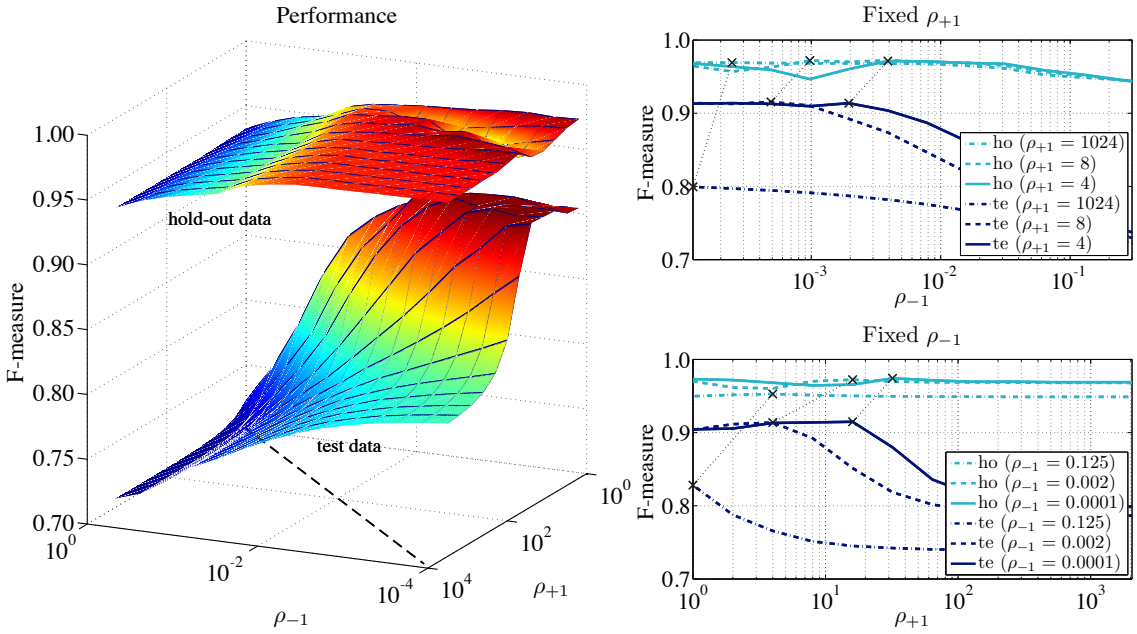


Figure 2: Left: Performance of NLR on the hold-out and the test data with respect to regularization parameters. Right: Performance of NLR on the hold-out data (ho) and the test data (te) for fixed values of ρ_v.

We split the data set into three parts: The 2,000 oldest emails constitute the training portion, we use the next 2,000 emails as hold-out portion on which the parameters are tuned, and the remaining emails are used as test set. We randomly draw 200 spam and 200 non-spam messages from the training portion and draw another subset of 400 emails from the hold-out portion. Both NPG instances are trained on the 400 training emails and evaluated against all emails of the test portion. To tune the parameters, we conduct a grid search maximizing the likelihood on the 400 hold-out emails. We repeat this experiment 10 times for all four data sets and report on the resulting parameters as well as the "optimal" reference parameters according to the maximal value of F-measure on the test set. Those optimal regularization parameters are not used in later experiments. The intuition of the experiment is that the data generation process has already been changed between the oldest and the latest emails. This change may cause a distribution shift which is reflected in the hold-out portion. We expect that one can tune each players' regularization parameter by tuning with respect to this hold-out set.

In Figure 2 (left) we plot the performance of the Nash logistic regression (NLR) on the hold-out and the test data against the regularization parameters ρ_{-1} and ρ_{+1}. The dashed line visualizes the bound in (30) on the regularization parameters for which NLR is guaranteed to possess a unique Nash equilibrium. Figure 2 (right) shows sectional views of the left plot along the ρ_{-1}-axis (upper diagram) and the ρ_{+1}-axis (lower diagram) for several values of ρ_{+1} and ρ_{-1}, respectively. As expected, the effect of the regularization parameters on the test data is much stronger than on the hold-out data.

It turns out that the data generator's ρ_{+1} has almost no impact on the value of F-measure on the hold-out data set (see lower right diagram of Figure 2). Hence, we conclude that estimating ρ_{+1} without access to labeled data from the test distribution or additional knowledge about the data generator is difficult for this application; the most recent training data are still too different from the test data. In all remaining experiments and for all data sets we set $\rho_{+1} = 8$ for NLR and $\rho_{+1} = 2$ for NSVM. For those choices the Nash models performed generally best on the hold-out set for a large variety of values of ρ_{-1}. For Invar-SVM the regularization of the data generator's transformation is controlled explicitly by the number K of modifiable attributes per positive instance. We conducted the same experiment for Invar-SVM resulting in an optimal value of $K = 25$; that is, the data generator is allowed to remove up to 25 tokens of each spam email of the training data set.

From the upper right diagram of Figure 2 we see that estimating ρ_{-1} for any fixed ρ_{+1} seems possible. Even if we slightly overestimate the learner's optimal regularization parameter—to compensate for the distributional difference between the transformed training sample and the marginal shifted hold-out set—the determined value of ρ_{-1} is close to the optimum for all four data sets.

6.3 Evaluation for Adversary Following an Equilibrium Strategy

We evaluate both a regular classifier trained under the *i.i.d.* assumption and a model that follows a Nash equilibrium strategy against both an adversary who does not transform the input distribution and an adversary who executes the Nash-equilibrial transformation on the input distribution. Since we cannot be certain that actual spam senders play a Nash equilibrium, we use the following semi-artificial setting.

The learner observes a sample of 200 spam and 200 non-spam emails drawn from the training portion of the data and estimates the Nash-optimal prediction model with parameters $\dot{\mathbf{w}}$; the trivial baseline solution of regularized empirical risk minimization (ERM) is denoted by $\mathbf{w}$. The data generator observes a *distinct* sample D of 200 spam and 200 non-spam messages, also drawn from the training portion, and computes their Nash-optimal response $\dot{D}$.

We again set $c_{v,i} := \frac{1}{n}$ for $v \in \{-1,+1\}$ and $i = 1,\ldots,n$ and study the following four scenarios:

- $(\mathbf{w}, D)$: Both players ignore the presence of an opponent; that is, the learner employs a regular classifier and the sender does not change the data generation process.
- $(\mathbf{w}, \dot{D})$: The learner ignores the presence of an active data generator who changes the data generation process such that D evolves to $\dot{D}$ by playing a Nash strategy.
- $(\dot{\mathbf{w}}, D)$: The learner expects a rational data generator and chooses a Nash-equilibrial prediction model. However, the data generator does not change the input distribution.
- $(\dot{\mathbf{w}}, \dot{D})$: Both players are aware of the opponent and play a Nash-equilibrial action to secure lowest costs.

We repeat this experiment 100 times for all four data sets. Table 2 reports on the average values of F-measure over all repetitions and both NPG instances and corresponding baselines; numbers in boldface indicate significant differences ($\alpha = 0.05$) between the F-measures of $f_{\mathbf{w}}$ and $f_{\dot{\mathbf{w}}}$ for fixed sample D and $\dot{D}$, respectively.

As expected, when the data generator does not alter the input distribution, the regularized empirical risk minimization baselines, logistic regression and the SVM, are generally best. However,

NLR vs. LR	ESP		Mailinglist		Private		TREC 2007	
	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$
D	**0.957**	0.924	**0.987**	0.984	**0.961**	0.944	**0.980**	**0.979**
$\dot{D}$	0.912	**0.925**	0.958	**0.976**	0.903	**0.912**	0.955	**0.961**

NSVM vs. SVM	ESP		Mailinglist		Private		TREC 2007	
	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$	$\mathbf{w}$	$\dot{\mathbf{w}}$
D	**0.955**	0.939	**0.987**	0.985	**0.961**	0.957	0.979	**0.981**
$\dot{D}$	0.928	**0.939**	0.961	**0.976**	0.932	**0.936**	0.960	**0.968**

Table 2: Nash predictor and regular classifier against passive and Nash-equilibrial data generator.

the performance of those baselines drops substantially when the data generator plays the Nash-equilibrial action $\dot{D}$. The Nash-optimal prediction models are more robust against this transformation of the input distribution and significantly outperform the reference methods for all four data sets.

6.4 Case Study on Email Spam Filtering

To study the performance of the Nash prediction models and the baselines for email spam filtering, we evaluate all methods *into the future* by processing the test set in chronological order. The test portion of each data set is split into 20 chronologically sorted disjoint subsets. We average the value of F-measure on each of those subsets over the 20 models (trained on different samples drawn from the training portion) for each method and perform a paired t-test. In the absence of information about player and instance-specific costs, we again set $c_{v,i} := \frac{1}{n}$ for $v \in \{-1,+1\}$, $i = 1,\ldots,n$. Note, that the chosen loss functions and regularizers would allow us to select any positive cost factors without violating Assumption 1.

Figure 3 shows that, for all data sets, the NPG instances outperform logistic regression (LR), SVM, and SVMT that do not explicitly factor the adversary into the optimization criterion. Especially for the ESP corpus, the Nash logistic regression (NLR) and the Nash support vector machine (NSVM) are superior. On the TREC 2007 data set, the methods behave comparably with a slight advantage for the Nash support vector machine. The period over which the TREC 2007 data have been collected is very short; we believe that the training and test instances are governed by nearly identical distributions. Consequently, for this data set, the game theoretic models do not gain a significant advantage over logistic regression and the SVM that assume *i.i.d.* samples. With respect to the non-game theoretic baselines, the regular SVM outperforms LR and SVMT for most of the data sets.

Table 3 shows aggregated results over all four data sets. For each point in each of the diagrams of Figure 3, we conduct a pairwise comparison of all methods based on a paired t-test at a confidence level of $\alpha = 0.05$. When a difference is significant, we count this as a win for the method that achieves a higher value of F-measure. Each line of Table 3 details the wins and, set in italics, the *losses* of one method against all other methods.

The Nash logistic regression and the Nash support vector machine have more wins than they have losses against each of the other methods. The ranking continues with Invar-SVM, the regular SVM, logistic regression and the trigonometric loss SVM which loses more frequently than it wins against all other methods.

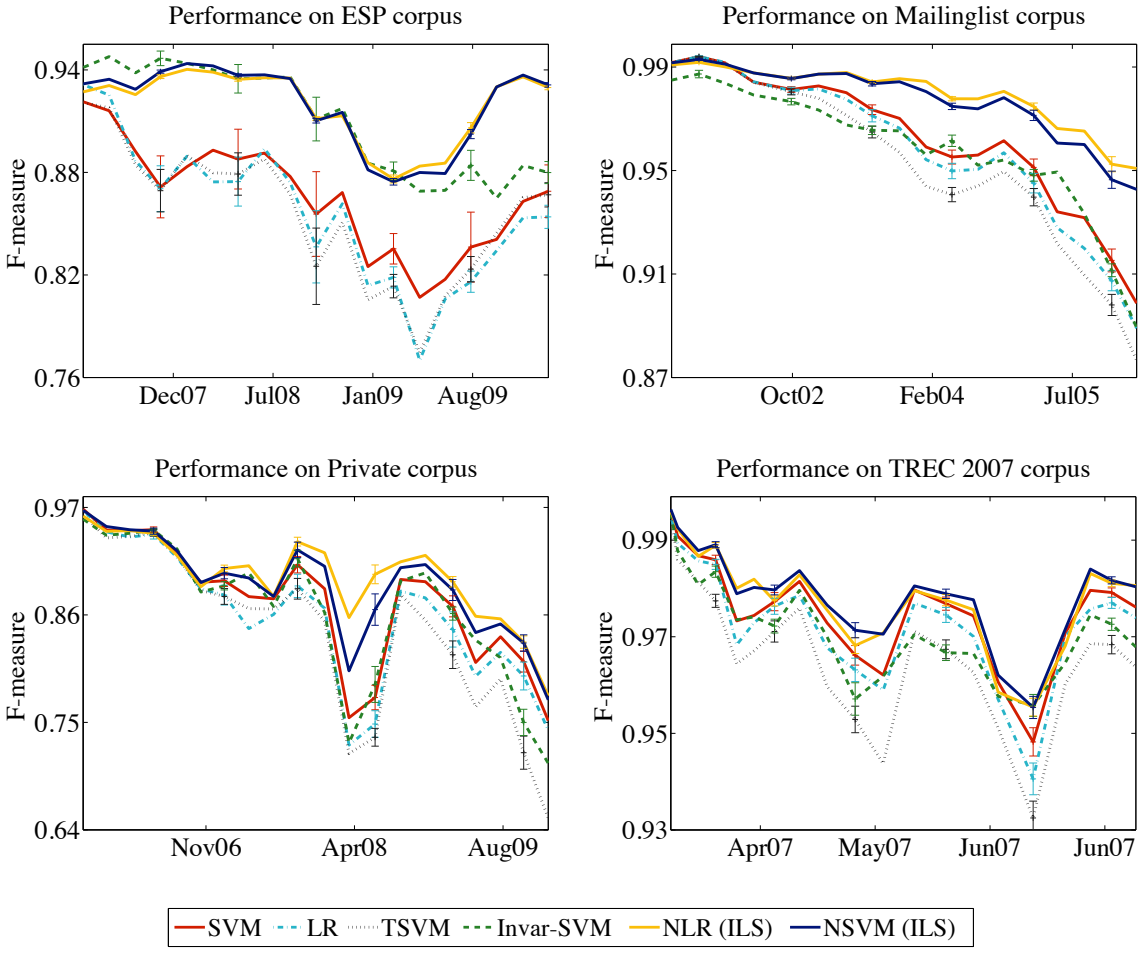


Figure 3: Value of F-measure of predictive models. Error bars indicate standard errors.

method vs. *method*	*SVM*	*LR*	*SVMT*	*Invar-SVM*	*NLR*	*NSVM*
SVM	0:*0*	40:*2*	53:*0*	30:*20*	8:*57*	2:*65*
LR	2:*40*	0:*0*	49:*5*	19:*29*	5:*59*	2:*71*
SVMT	0:*53*	5:*49*	0:*0*	9:*47*	2:*70*	2:*74*
Invar-SVM	20:*30*	29:*19*	47:*9*	0:*0*	5:*57*	3:*57*
NLR	57:*8*	59:*5*	70:*2*	57:*5*	0:*0*	22:*30*
NSVM	65:*2*	71:*2*	74:*2*	57:*3*	30:*22*	0:*0*

Table 3: Results of paired t-test over all corpora: Number of trials in which each method (row) has significantly outperformed each other *method (column)* vs. number of times it *was outperformed*.

6.5 Efficiency versus Effectiveness

To assess the predictive performance as well as the execution time as a function of the sample size, we train the baselines and the two NPG instances for a varying number of training examples. We

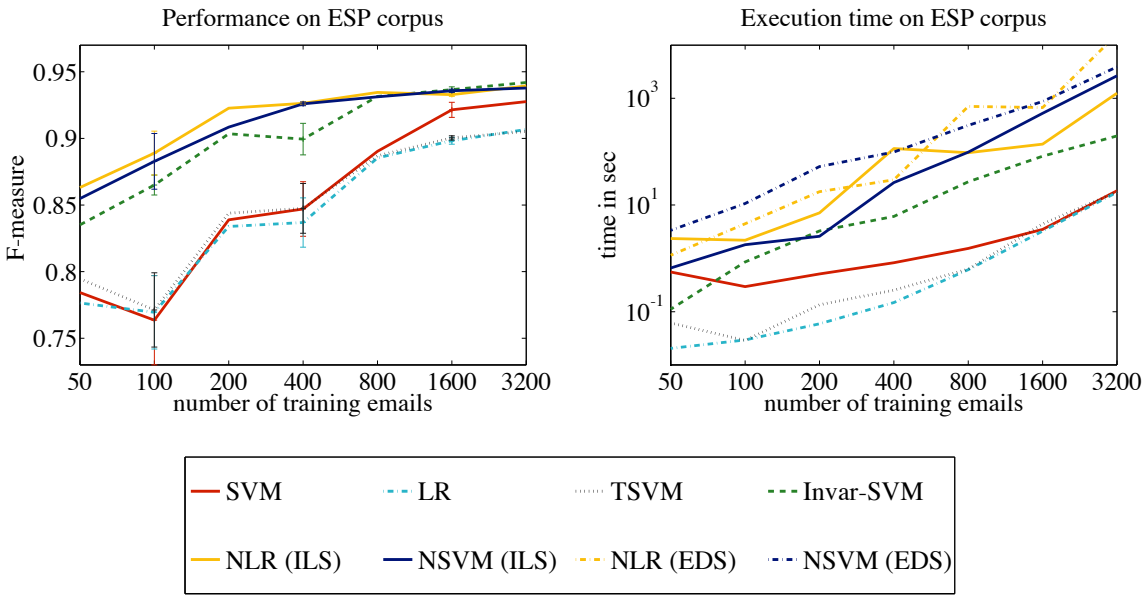


Figure 4: Predictive performance (left) and execution time (right) for varying sizes of the training data set.

report on the results for the ESP data set in Figure 4. The game theoretic models significantly outperform the trivial baseline methods logistic regression, the SVM and the SVMT, especially for small data sets. However, this comes at the price of considerably higher computational cost. The ILS algorithm requires in general only a couple of iterations to converge; however in each iteration several optimization problems have to be solved so that the total execution time is up to a factor 150 larger than that of the corresponding ERM baseline. In contrast to the ILS algorithm, a single iteration of the EDS algorithm does not require solving nested optimization problems. However, the execution time of the EDS algorithm is still higher as it often requires several thousand iterations to fully converge. For larger data sets, the discrepancy in predictive performance between game theoretic models and *i.i.d.* baseline decreases. Our results do not provide conclusive evidence whether ILS or EDS is faster at solving the optimization problems. We conclude that the benefit of the NPG prediction models over the classification baseline is greatest for small to medium sample sizes.

6.6 Nash-Equilibrial Transformation

In contrast to Invar-SVM, the Nash models allow the data generator to modify non-spam emails. However in practice most senders of legitimate messages do not deliberately change their writing behavior in order to bypass spam filters, perhaps with the exception of senders of newsletters who must be careful not to trigger filtering mechanisms.

In a final experiment, we want to study whether the Nash model reflects this aspect of reality, and how the data generator's regularizer effects this transformation. The training portion contains again $n_{+1} = 200$ spam and $n_{-1} = 200$ non-spam instances randomly chosen from the oldest $4,000$

emails. We determine the Nash equilibrium and measure the number of additions and deletions to spam and non-spam emails in $\dot{D}$:

$$\Delta_{-1}^{\text{add}} := \frac{1}{n_{-1}} \sum_{i:y_i=-1} \sum_{j=1}^{m} \max(0, \dot{\mathbf{x}}_{i,j} - \mathbf{x}_{i,j}) \qquad \Delta_{+1}^{\text{add}} := \frac{1}{n_{+1}} \sum_{i:y_i=+1} \sum_{j=1}^{m} \max(0, \dot{\mathbf{x}}_{i,j} - \mathbf{x}_{i,j})$$

$$\Delta_{-1}^{\text{del}} := \frac{1}{n_{-1}} \sum_{i:y_i=-1} \sum_{j=1}^{m} \max(0, \mathbf{x}_{i,j} - \dot{\mathbf{x}}_{i,j}) \qquad \Delta_{+1}^{\text{del}} := \frac{1}{n_{+1}} \sum_{i:y_i=+1} \sum_{j=1}^{m} \max(0, \mathbf{x}_{i,j} - \dot{\mathbf{x}}_{i,j})$$

where $\mathbf{x}_{i,j}$ indicates the presence of token j in the i-th training email, that is, Δ_v^{add} and Δ_v^{del} denote the average number of word additions and deletions per spam and non-spam email performed by the sender.

Figure 5 shows the number of additions and deletions of the Nash transformation as a function of the adversary's regularization parameter for the ESP data set. Table 4 reports on the average number of word additions and deletions for all data sets. For Invar-SVM, we set the number of possible deletions to $K = 25$.

ESP

game	*non-spam*		*spam*	
model	*add*	*del*	*add*	*del*
Invar-SVM	0.0	0.0	0.0	24.8
NLR	0.7	1.0	22.5	31.2
NSVM	0.4	0.5	17.9	23.8

Mailinglist

game	*non-spam*		*spam*	
model	*add*	*del*	*add*	*del*
Invar-SVM	0.0	0.0	0.0	23.9
NLR	0.3	0.4	8.6	10.9
NSVM	0.3	0.3	6.9	8.4

Private

game	*non-spam*		*spam*	
model	*add*	*del*	*add*	*del*
Invar-SVM	0.0	0.0	0.0	24.2
NLR	0.4	0.2	24.3	11.2
NSVM	0.1	0.1	15.6	7.3

TREC 2007

game	*non-spam*		*spam*	
model	*add*	*del*	*add*	*del*
Invar-SVM	0.0	0.0	0.0	24.7
NLR	0.2	0.2	15.0	11.4
NSVM	0.2	0.1	11.1	8.4

Table 4: Average number of word additions and deletions per training email.

The Nash-equilibrial transformation imposes almost no changes on any non-spam email; the number of modifications declines as the regularization parameter grows (see Figure 5). We observe for all data sets that even if the total amount of transformation differs for NLR and NSVM, both instances behave similarly insofar as the number of word additions and deletions continues to grow when the adversary's regularizer decreases.

7. Conclusion

We studied prediction games in which the learner and the data generator have conflicting but not necessarily directly antagonistic cost functions. We focused on static games in which learner and data generator have to commit simultaneously to a predictive model and a transformation of the input distribution, respectively.

The cost-minimizing action of each player depends on the opponent's move; in the absence of information about the opponent's move, players may choose to play a Nash equilibrium strategy

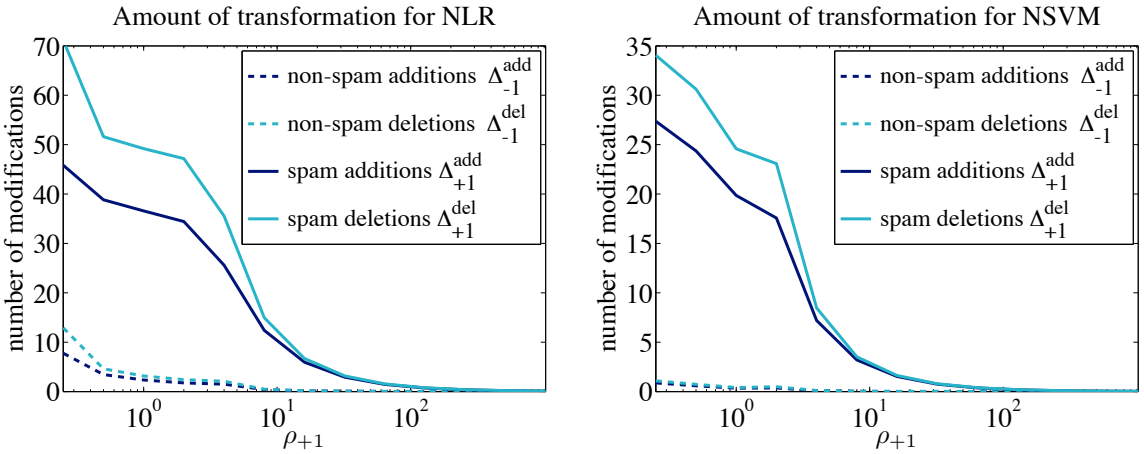


Figure 5: Average number of additions and deletions per spam/non-spam email for NLR (left) and NSVM (right) with respect to the adversary's regularization parameter ρ_{+1} for fixed $\rho_{-1} = n^{-1}$.

which constitutes a cost-minimizing move for each player *if* the other player follows the equilibrium as well. Because a combination of actions from distinct equilibria may lead to arbitrarily high costs for either player, we have studied conditions under which a prediction game can be guaranteed to possess a unique Nash equilibrium. Lemma 1 identifies conditions under which at least one equilibrium exists, and Theorem 8 elaborates on when this equilibrium is unique. We propose an inexact linesearch approach and a modified extragradient approach for identifying this unique equilibrium. Empirically, both approaches perform quite similarly.

We derived Nash logistic regression and Nash support vector machine models and kernelized versions of these methods. Corollaries 11 and 15 specialize Theorem 8 and expound conditions on the player's regularization parameters under which the Nash logistic regression and the support vector machine converge on a unique Nash equilibrium. Empirically, we find that both methods identify unique Nash equilibria when the bounds laid out in Corollaries 11 and 15 are satisfied or violated by a factor of up to 4. From our experiment on several email corpora we conclude that Nash logistic regression and the support vector machine outperform their *i.i.d.* baselines and Invar-SVM for the problem of classifying future emails based on training data from the past.

Acknowledgments

This work was supported by the German Science Foundation (DFG) under grant SCHE540/12-1 and by STRATO AG. We thank Niels Landwehr and Christoph Sawade for constructive comments and suggestions, and the anonymous reviewer for helpful contributions and careful proofreading of the manuscript!

References

Tamer Basar and Geert J. Olsder. *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics, 1999.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Michael Brückner and Tobias Scheffer. Nash equilibria of static prediction games. In *Advances in Neural Information Processing Systems*. MIT Press, 2009.

Michael Brückner and Tobias Scheffer. Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Diego, CA, USA*. ACM, 2011.

Ofer Dekel and Ohad Shamir. Learning to classify with missing and corrupted features. In *Proceedings of the International Conference on Machine Learning*. ACM, 2008.

Ofer Dekel, Ohad Shamir, and Lin Xiao. Learning to classify with missing and corrupted features. *Machine Learning*, 81(2):149–178, 2010.

Carl Geiger and Christian Kanzow. *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer, 1999.

Laurent El Ghaoui, Gert R. G. Lanckriet, and Georges Natsoulis. Robust classification with interval data. Technical Report UCB/CSD-03-1279, EECS Department, University of California, Berkeley, 2003.

Amir Globerson and Sam T. Roweis. Nightmare at test time: Robust learning by feature deletion. In *Proceedings of the International Conference on Machine Learning*. ACM, 2006.

Amir Globerson, Choon Hui Teo, Alex J. Smola, and Sam T. Roweis. An adversarial view of covariate shift and a minimax approach. In *Dataset Shift in Machine Learning*, pages 179–198. MIT Press, 2009.

Patrick T. Harker and Jong-Shi Pang. Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications. *Mathematical Programming*, 48(2):161–220, 1990.

Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, 1991.

James T. Kwok and Ivor W. Tsang. The pre-image problem in kernel methods. In *Proceedings of International Conference on Machine Learning*, pages 408–415. AAAI Press, 2003. ISBN 1-57735-189-4.

Gert R. G. Lanckriet, Laurent El Ghaoui, Chiranjib Bhattacharyya, and Michael I. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2002.

Sebastian Mika, Bernhard Schölkopf, Alex J. Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel PCA and de–noising in feature spaces. In *Advances in Neural Information Processing Systems*. MIT Press, 1999.

J. Ben Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33(3):520–534, 1965.

Bernhard Schölkopf and Alex J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 2001.

Christian Siefkes, Fidelis Assis, Shalendra Chhabra, and William S. Yerazunis. Combining Winnow and orthogonal sparse bigrams for incremental spam filtering. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, volume 3202 of *Lecture Notes in Artificial Intelligence*, pages 410–421. Springer, 2004.

Choon Hui Teo, Amir Globerson, Sam T. Roweis, and Alex J. Smola. Convex learning with invariances. In *Advances in Neural Information Processing Systems*. MIT Press, 2007.

Anna von Heusinger and Christian Kanzow. Relaxation methods for generalized Nash equilibrium problems with inexact line search. *Journal of Optimization Theory and Applications*, 143(1): 159–183, 2009.

Journal of Machine Learning Research 13 (2012) 2655-2697 Submitted 12/11; Published 9/12

Selective Sampling and Active Learning from Single and Multiple Teachers

Ofer Dekel OFERD@MICROSOFT.COM
Microsoft Research
One Microsoft Way
Redmond, WA, 98052, USA

Claudio Gentile CLAUDIO.GENTILE@UNINSUBRIA.IT
DiSTA, Università dell'Insubria
via Mazzini 5
21100 Varese, Italy

Karthik Sridharan SKARTHIK@WHARTON.UPENN.EDU
Department of Statistics of the Wharton School
University of Pennsylvania
3730 Walnut Street
Philadelphia, PA, 19104, USA

Editor: Sanjoy Dasgupta

Abstract

We present a new online learning algorithm in the selective sampling framework, where labels must be actively queried before they are revealed. We prove bounds on the regret of our algorithm and on the number of labels it queries when faced with an adaptive adversarial strategy of generating the instances. Our bounds both generalize and strictly improve over previous bounds in similar settings. Additionally, our selective sampling algorithm can be converted into an efficient statistical active learning algorithm. We extend our algorithm and analysis to the multiple-teacher setting, where the algorithm can choose which subset of teachers to query for each label. Finally, we demonstrate the effectiveness of our techniques on a real-world Internet search problem.

Keywords: online learning, regret, label-efficient, crowdsourcing

1. Introduction

Human-generated labels are expensive. The *active learning* paradigm is built around the idea that we should only acquire labels that actually improve our ability to make accurate predictions. *Online selective sampling* (Cohn et al., 1990; Freund et al., 1997) is an active learning setting that is modeled as a repeated game between a *learner* and an *adversary*. On round t of the game, the adversary presents the learner with an instance $\mathbf{x}_t \in \mathbb{R}^d$ and the learner responds by predicting a binary label $\hat{y}_t \in \{-1,+1\}$. The learner has access to a *teacher*,[1] who knows the correct label for each instance. The learner must now decide whether or not to pay a unit cost and query the teacher for the correct binary label $y_t \in \{-1,+1\}$ from the teacher. If the learner decides to issue a query, he observes the correct label and uses it to improve his future predictions. However, when we analyze the accuracy

1. Most previous publications do not distinguish between the *adversary* and the *teacher*. We make this distinction explicitly and intentionally, in anticipation of the multiple-teacher variant of the problem.

of the learner's predictions, we account for all labels, regardless of whether they were observed by the learner or not. The learner has two conflicting goals: to make accurate predictions and to issue a small number of queries.

To motivate the selective sampling setting, consider an Internet search company that uses online learning techniques to construct a (simplified) search engine. In this case, the instance $\mathbf{x}_t$ represents the pairing of a search-engine query with a candidate web page and the task is to predict whether this pair is a good match or a bad match. Clearly, there is no way to manually label the millions of daily search engine queries along with all of their candidate web pages. Instead, an intelligent mechanism of choosing which instances to label is required. Search engine queries arrive in an online manner and a search engine uses its index of the web to match each query with potential candidate URLs, making this problem well suited for the selective sampling problem setting.

The first part of this paper is devoted to the selective sampling framework described above. In Section 2 we present a selective sampling learning algorithm inspired by known ridge regression algorithms (Hoerl and Kennard, 1970; Lai and Wei, 1982; Vovk, 2001; Azoury and Warmuth, 2001; Cesa-Bianchi et al., 2003, 2005a; Li et al., 2008; Strehl and Littman, 2008; Cavallanti et al., 2009; Cesa-Bianchi et al., 2009). To analyze this algorithm, we adopt the model introduced in Cavallanti et al. (2009), Cesa-Bianchi et al. (2009) and Strehl and Littman (2008), where the adversary may choose arbitrary instances, but the teacher is stochastic and samples each label from an instance-dependent distribution. We evaluate the accuracy of the learner using the game-theoretic notion of *regret*, which measures the extent to which the learner's predictions disagree with the teacher's labels. We prove both an upper bound on the regret and an upper bound on the number of queries issued by the learner.

Our algorithm is an online learning algorithm, designed to incrementally make binary predictions on a sequence of adversarially-generated instances. However, we can also convert our algorithm into an efficient statistical active learning algorithm, which receives a sample of instances from some unknown distribution, queries the teacher for a subset of the labels, and outputs a hypothesis with a small risk. The risk of a hypothesis is its error rate on new instances sampled form the same underlying distribution. We present the details of this conversion in Section 2.5.

In the setting described above, we assumed the learner has access to a single all-knowing teacher. To make things more interesting, we introduce multiple teachers, each with a different area of expertise and a different level of overall competence. On round t, some of the teachers may be experts on $\mathbf{x}_t$ while others may not be. A teacher who is an expert on $\mathbf{x}_t$ is likely to provide the correct label, while a teacher who isn't may give the wrong label. To make this setting as realistic as possible, we assume that the areas of expertise and the overall competence levels of the different teachers are unknown to the learner, and any characterization of a teacher must be inferred from the observed labels.

On round t, the learner receives the instance $\mathbf{x}_t$ from the adversary and makes the binary prediction $\hat{y}_t$. Then, the learner has the option to query any subset of teachers: each teacher charges a unit cost per query and provides a binary label, without any indication of his confidence and without the option of abstaining. The labels received from the queried teachers may disagree, and the learner has no a-priori way of knowing which teacher to trust. If the learner queries the wrong teachers, their labels may agree but still be wrong. The algorithm's goal remains to make accurate predictions using a small number of queries. However, in the absence of a ground truth labeling, it is unclear how to define what it means to make an accurate prediction. To resolve this problem, we formalize

the assumption that different teachers have different areas of expertise, which allows us to compare each predicted label with the labels provided by experts on the relevant topic.

Recalling the motivating example given above, assume that the Internet search company employs multiple human teachers. Some teachers may be better than others across the board and some teachers may be experts on specific topics, such as sports or politics. Some teachers may know the right answer, while others may think they know the right answers but in fact do not—for this reason we do not rely on the teachers themselves to reveal their expertise regions. For example, say that the search engine receives the web query "nhl new york team" and a candidate url is "kings.nhl.com"; a teacher who is a hockey expert would know that this is a bad match (since New York's NHL hockey team is called the Rangers and not the Kings) while a non-expert may not know the answer. The learner has no a-priori knowledge of which teacher to query for the label; yet, in our analysis we would like to compare the learner's prediction to the label given by the expert teacher.

The multiple-teacher selective sampling setting is the focus of the second half of this paper. Specifically, in Section 3 we present a multiple-teacher extension of the (single-teacher) adversarial-stochastic model mentioned earlier, along with two new learning algorithms in this setting. Our model of the teachers' expertise regions enables our algorithms to gradually identify the expertise region of each teacher. Roughly speaking, the algorithm attempts to measure the consistency of the binary labels provided by each teacher in different regions of the instance space. Our first multiple-teacher algorithm has the property that it either queries all of the teachers or does not query any teacher, on each round. Our second algorithm is more sophisticated and queries only those teachers it believes to be experts on $\mathbf{x}_t$. Again, we provide a theoretical analysis that bounds both regret and number of queries issued to the teachers.

Since our results rely on the specific stochastic model of the teachers, it is natural to question how well this model approximates the real-world. To gain some confidence in our assumptions and in our algorithms, in Section 4 we present a simple empirical study on real data that both validate our theoretical results and demonstrates the effectiveness of our approach.

1.1 Related Work in the Single Teacher Setting

Single-teacher selective sampling lies between passive learning (where the algorithm has no control over the learning sequence) and fully active learning (where the learning algorithm is allowed to select the instances $\mathbf{x}_t$). The literature on active learning is vast, and we can hardly do it justice here. Recent papers on active learning include the works by Balcan et al. (2006), Bach (2006), Balcan et al. (2007), Balcan et al. (2008), Castro and Nowak (2008), Dasgupta et al. (2005), Dasgupta et al. (2008), Hanneke (2007), Hanneke (2009) and Koltchinskii (2010). All of these papers consider the case where instances are drawn i.i.d. from a fixed distribution (either known or unknown). In particular, Dasgupta et al. (2005) gives an efficient Perceptron-like algorithm for learning within accuracy ε the class of homogeneous d-dimensional half-spaces under the uniform distribution over the unit ball, with label complexity of the form $d \log \frac{1}{\varepsilon}$. Still in the i.i.d. setting, more general results are given by Balcan et al. (2007). A neat analysis of previously proposed general active learning schemes (Balcan et al., 2006; Dasgupta et al., 2008) is provided by the aforementioned paper by Hanneke (2009). Even more recently, a general Rademacher complexity-based analysis of active learning is given by Koltchinskii (2010). Due to their generality, many of the above results rely on schemes that are computationally prohibitive, exceptions being the results by Dasgupta et al. (2005) and the realizable cases analyzed by Balcan et al. (2007). For instance, the general algorithms

proposed by Hanneke (2009); Koltchinskii (2010) do actually imply estimating ε-minimal sets (or disagreement sets) from empirical data and (local) Rademacher complexities, which makes them computationally hard even for simple function classes, like linear-threshold functions. Finally, pool-based active learning scenarios are considered by Bach (2006) (and the references therein), though the analysis therein is only asymptotic in nature and no quantification is given of the trade-off between risk and number of labels.

To contrast our work with the papers mentioned above, it is worth stressing that our results hold with no stochastic assumption on the source of the instances—in fact, we assume that the instances may be generated by an adaptive adversary. However, as mentioned above, we also show how our online learning algorithm can be converted into a statistical active learning algorithm, with a formal risk bound. Our results in the online selective sampling setting are more in line with the worst-case analyses by Cesa-Bianchi et al. (2006), Strehl and Littman (2008), Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011). These papers present variants of Recursive Least Squares algorithms that operate on arbitrary instance sequences. The analysis by Cesa-Bianchi et al. (2006) is completely worst case: the authors make no assumptions whatsoever on the mechanism generating instances or labels; however, they are unable to prove bounds on the label query rate. The setups by Strehl and Littman (2008), Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011) are closest to ours in that they assume the same stochastic model of the teacher. Our bounds can be shown to be optimal with respect to certain parameters and, unlike competing works on this subject, we are able to face the case when the instance sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots$ is generated by an *adaptive adversary*, rather than the weaker *oblivious adversary*, as by, for example, Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011). It is actually this difference that makes it possible the selective sampling-to-active learning conversion. A detailed comparison of our results in the single-teacher setting with the results of the predominant papers on this topic is given in Section 2.6, after our results are presented.

1.2 Related Work in the Multiple Teacher Setting

There is also much related work in the multiple-teacher setting, which is often motivated within recent *crowdsourcing* applications. We can map the current state-of-the-art on this topic along various interesting axes.

First, we distinguish between techniques that attempt to find the ground-truth labeling (and evaluate each teacher's quality) independent of the learning algorithm, and techniques that combine the ground-truth-finding and the actual learning into a single algorithm. In the first category are the classical work of Dawid and Skeene (1979), which presents techniques of reconciling conflicting responses on medical questionnaires, the one of Spiegelhalter and Stovin (1983) which handles conflicting information from repeated biopsies, the one by Smyth et al. (1995), where the authors infer a ground truth from multiple annotations of astronomical images, and the one by Hui and Zhou (1998) which examines the more general problem of evaluation in the absence of a ground truth. Still in the first category, Dekel and Shamir (2009a) and Chen et al. (2010) both present general techniques for identifying and rejecting low quality teachers. Papers in the second category discuss supervised learning algorithms that can handle multiple-teacher input. In this category, Dekel and Shamir (2009b) present an SVM variant that is less sensitive to bad labels generated by a small set of malicious teachers, Raykar et al. (2010) use EM to jointly establish a ground truth labeling and learn a maximum-likelihood estimator, Argall et al. (2009) dynamically choose which human

demonstrator to use when teaching a policy to a robot, and Groot et al. (2011) integrate multiple-teacher support into Gaussian process regression learning. Our work in the current paper falls in the second category.

We can also distinguish between algorithms that rely on repeated labeling (where multiple teachers label each example), versus techniques that assume that each example is labeled only once. Sheng et al. (2008), Snow et al. (2008) and Donmez et al. (2009) collect repeated labels and aggregate them (e.g., using a majority vote) to simulate the ground-truth labeling. Some of these papers balance an explore-exploit tradeoff, which determines how many repeated labels are needed for each example. At the opposite end of the spectrum, Dekel and Shamir (2009a) identify low-quality teachers and labels without any repeated labeling. The technique presented in this paper falls in the latter category, since we actively determine which subset of teachers to query on each online round. However, while we do query multiple teachers, we do not assume that the majority vote, or any other aggregate label, is accurate. Still, we do compare to some majority vote of teachers in both our analysis and our experiments.

Next, we distinguish between papers that consider the overall quality score of each teacher (over the entire input space) from papers that assume that each teacher has a specific area of expertise. Most of the papers mentioned above fall in the first category. In the second category, Yan et al. (2010) extend the work in Raykar et al. (2010) (again, maximizing likelihood and using EM) to handle the case where different teachers have knowledge about different parts of the input space. In the present paper, we also model each teacher as an expert on a different subtopic. A closely related research topic is multi-domain adaptation (Mansour et al., 2009a,b), where multiple hypotheses must be optimally combined, under the assumption that each hypothesis makes accurate predictions with respect to a different distribution. Another closely related topic is learning from multiple sources (Crammer et al., 2008), where multiple data sets are sampled from different distributions, and the goal is to optimally combine them with a given target distribution in mind. However, in both of these related problems we are given some prior information on the various distributions, whereas in the multiple-teacher setting we must infer the expertise of each teacher from data.

Another interesting distinction can be made between passive multiple-teacher techniques, which process a static data set that was collected beforehand, and active techniques that route each example to the appropriate teacher. Most of the aforementioned work follows the static approach. The proactive learning setting (Domnez, 2010; Yang and Carbonell, 2009a,b) assumes that the learner has access to teachers of different global quality, with associated costs per label. Yang and Carbonell (2009a) present a theoretical analysis of proactive learning, under the assumption that each teacher gives the correct label most of the time. However, note that the active category fits quite nicely with the assumption that each teacher has an area of expertise (as opposed to measuring the global quality of each teacher): once the algorithm identifies the area of expertise of a teacher, it seems only natural to actively route the relevant examples to that teacher. The approach presented in this paper does precisely that. At the time of writing the extended version of our paper, other works have been published that considered the problem of active learning from multiple annotators. The one whose goal is closest to ours is perhaps the paper by Yan et al. (2011), where a probabilistic multi-labeler model is formulated that allows one to learn the expertise of the labelers and to single out the most uncertain sample (within a given pool of unlabeled instances) whose label is useful to query. Though that paper is similar in spirit to ours, it does mainly focus on modeling and empirical investigations. Finally, we note that Melville et al. (2005) study the closely related problem of actively acquiring individual feature values.

An interesting variation on the multiple-teacher theme involves allowing each teacher's quality to vary with time (Donmez et al., 2010).

2. The Single Teacher Case

In this section, we focus on the standard online selective sampling setting, where the learner has to learn an accurate predictor while determining whether or not to query the label of each instance it observes. We formally define the problem setting in Section 2.1 and introduce our algorithm in Section 2.2. We prove upper bounds on the regret and on the number of queries in Section 2.3. We briefly mention how to convert our online learning algorithm into a statistical active learning algorithm in Section 2.4 and Section 2.5, and we compare our results to related work in Section 2.6.

2.1 Preliminaries and Notation

As mentioned above, on round t of the online selective sampling game, the learner receives an instance $\mathbf{x}_t \in \mathbb{R}^d$, predicts a binary label $\hat{y}_t \in \{-1,+1\}$, and chooses whether or not to query the correct label $y_t \in \{-1,+1\}$. We set $Z_t = 1$ if a query is issued on round t and $Z_t = 0$ otherwise. The only assumption we make on the process that generates $\mathbf{x}_t$ is that $\|\mathbf{x}_t\| \leq 1$; for all we know, instances may be generated by an *adaptive* adversary (an adversary that reacts to our previous actions). Note that most of the previous work on this topic makes stronger assumptions on the process that generates $\mathbf{x}_t$, resulting in a less powerful setting. As for the labels provided by the teacher, we adopt the standard stochastic linear noise model for this problem (Cesa-Bianchi et al., 2003; Cavallanti et al., 2009; Cesa-Bianchi et al., 2009; Strehl and Littman, 2008) and assume that each $y_t \in \{-1,+1\}$ is sampled according to the law

$$P(y_t = 1 \,|\mathbf{x}_t\,) = \frac{1+\mathbf{u}^\top \mathbf{x}_t}{2}, \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^d$ is a fixed but unknown vector with $\|\mathbf{u}\| \leq 1$. Note that $\mathbb{E}\,[y_t \,|\mathbf{x}_t\,] = \mathbf{u}^\top \mathbf{x}_t$, and we denote this value by Δ_t. Unlike much of the recent literature on active learning (see Section 1.1), this simple noise model has the advantage of delivering time-efficient algorithms of practical use.

The learner constructs a sequence of linear predictors $\mathbf{w}_0, \mathbf{w}_1, \ldots$, where each $\mathbf{w}_t \in \mathbb{R}^d$, and predicts $\hat{y}_t = \mathrm{sign}(\hat{\Delta}_t)$ where $\hat{\Delta}_t = {\mathbf{w}_{t-1}}^\top \mathbf{x}_t$. The desirable outcome is for the sequence $\mathbf{w}_0, \mathbf{w}_1, \ldots$ to quickly converge to $\mathbf{u}$. Let P_t denote the conditional probability $\mathbb{P}(\,\cdot\,|\,\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t, y_1, \ldots, y_{t-1})$. We evaluate the accuracy of the learner's predictions using its *regret*, defined as

$$R_T \;=\; \sum_{t=1}^{T} \big(P_t(y_t \hat{\Delta}_t < 0) - P_t(y_t \Delta_t < 0)\big) \quad .$$

Additionally, we are interested in the number of queries issued by the learner $N_T = \sum_{t=1}^{T} Z_t$. Our goal is to simultaneously bound the regret R_T and the number of queries N_T with high probability over the random draw of labels.

Remark 1 *At first glance, the linear noise model (1) might seem too restrictive. However, this model can be made implicitly nonlinear by running our algorithm in a Reproducing Kernel Hilbert Space $\mathcal{H}$. This entails that the linear operation $\mathbf{u}^\top \mathbf{x}_t$ in (1) is replaced by $h(\mathbf{x}_t)$, for some (typically nonlinear) function $h \in \mathcal{H}$. See also the comments at the end of Section 2.2, and those surrounding Theorem 2.*

2.2 Algorithm

The single teacher algorithm is a margin-based selective sampling procedure. The algorithm "Selective Sampler" (Algorithm 1) depends on a confidence parameter $\delta \in (0,1]$. As in known online ridge-regression-like algorithms (Hoerl and Kennard, 1970; Vovk, 2001; Azoury and Warmuth, 2001; Cesa-Bianchi et al., 2003, 2005a; Li et al., 2008; Strehl and Littman, 2008; Cavallanti et al., 2009; Cesa-Bianchi et al., 2009), our algorithm maintains a weight vector $\mathbf{w}_t$ (initialized as $\mathbf{w}_0 = \mathbf{0}$) and a data correlation matrix A_t (initialized as $A_0 = I$). After receiving $\mathbf{x}_t$ and predicting $\hat{y}_t = \mathrm{sign}(\hat{\Delta}_t)$, the algorithm computes an adaptive data-dependent threshold θ_t, defined as

$$\theta_t^2 \;=\; \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t\left(1+4\sum_{i=1}^{t-1} Z_i r_i + 36\log\frac{t}{\delta}\right) \;,$$

where $r_i = \mathbf{x}_i^\top A_i^{-1}\mathbf{x}_i$. The definition of θ_t follows from our analysis, and can be interpreted as the algorithm's uncertainty in its own predictions. More precisely, the learner believes that $|\hat{\Delta}_t - \Delta_t| \leq \theta_t$. A query is issued only if $|\hat{\Delta}_t| \leq \theta_t$, or in other words, when the algorithm is unsure about the sign of Δ_t. In Algorithm 1, this is denoted by $Z_t = \mathbb{1}\left\{\hat{\Delta}_t^2 \leq \theta_t^2\right\}$, where $\mathbb{1}\left\{\,\cdot\,\right\}$ denotes the indicator function.

If the label is not queried, ($Z_t = 0$) then the algorithm does not update its internal state (and $\mathbf{x}_t$ is discarded). If the label is queried ($Z_t = 1$), then the algorithm computes the intermediate vector $\mathbf{w}'_{t-1}$ in such a way that $\hat{\Delta}'_t = {\mathbf{w}'_{t-1}}^\top \mathbf{x}_t$ is at most one in magnitude. Observe that $\hat{\Delta}_t$ and $\hat{\Delta}'_t$ have the same sign and only their magnitudes can differ. In particular, it holds that

$$\hat{\Delta}'_t = \begin{cases} \mathrm{sgn}(\hat{\Delta}_t) & \text{if } |\hat{\Delta}_t| > 1 \\ \hat{\Delta}_t & \text{otherwise} \;. \end{cases}$$

Next, the algorithm defines the new vector $\mathbf{w}_t$ so that $A_t\mathbf{w}_t$ undergoes an additive update, where A_t is a rank-one adjustment of A_{t-1}.

The algorithm can be run both in primal form (as in the pseudocode in Algorithm 1) and in dual form (i.e., in a Reproducing Kernel Hilbert Space). It is not hard to show that the algorithm has a quadratic running time per round, where quadratic means $O(d^2)$ if it is run in primal form, and $O(N_t^2)$ if it is run in dual form, where $N_t = \sum_{i \leq t} Z_i$ is the number of labels requested by the algorithm up to time t. In the dual case, since the algorithm updates only when $Z_t = 1$, the number of labels N_t also corresponds to the number of support vectors used to define the current hypothesis.

2.3 Analysis

Before diving into a formal analysis of Algorithm 1, we attempt to give some intuition regarding our choice of θ_t. Recall that θ_t is the radius of the algorithm's confidence interval, and therefore a small value of θ_t implies that the algorithm is highly confident that Δ_t and $\hat{\Delta}_t$ are close. If, additionally, Δ_t is large, then $\mathrm{sign}(\hat{\Delta}_t)$ is likely to equal $\mathrm{sign}(\Delta_t)$, and the algorithm's prediction is correct. Therefore, we want to show that θ_t can be kept small without issuing an excessive number of queries. To see this, we notice that θ_t depends on the three terms: $\sum_{i=1}^{t-1} Z_i r_i$, $\log(t/\delta)$, and $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$. Later in this section, we prove that $\sum_{i=1}^{t} Z_i r_i$ grows logarithmically with the number of queries N_t, and obviously $\log(t/\delta)$ grows logarithmically with t. To show that θ_t remains small, we must show that the third term, $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$, decreases quickly when labels are queried. $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$ depends on the relationship between the current instance $\mathbf{x}_t$ and the previous instances on rounds where a query was issued.

If $\mathbf{x}_t$ lies along the directions spanned by the previous instances, we show that $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$ tends to shrink as $1/N_t$. As a result, θ_t is on the order of $\log(t/\delta)/N_t$, and N_t only needs to grow at a slow logarithmic rate. On the other hand, if the adversary chooses $\mathbf{x}_t$ outside of the subspace spanned by the previous examples, then the term $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$ causes θ_t to be large, and the algorithm becomes more likely to issue a query. Overall, to ensure a small value of θ_t across the instance space spanned by the $\mathbf{x}_t$ produced by the adversary, the algorithm must query $O\big(\log(t)\big)$ labels in each direction of this instance space.

As noted above, the adversary can arbitrarily inflate our regret by choosing instances that induce small values of Δ_t. Recall that a small value of Δ_t implies that the teacher guesses the label y_t almost at random. Following Cesa-Bianchi et al. (2009), the bounds we prove depend on how many of the instances $\mathbf{x}_t$ are chosen such that Δ_t is very small. Formally, for any $\varepsilon > 0$, define

$$T_\varepsilon = \sum_{t=1}^{T} \mathbf{1}\{|\Delta_t| \le \varepsilon\} \ . \tag{2}$$

The following theorem is the main result of this section, and is stated so as to emphasize both the data-dependent and the time-dependent aspects of our bounds.

Theorem 2 *Assume that Selective Sampler is run with confidence parameter $\delta \in (0,1]$. Then with probability at least $1-\delta$ it holds that for all $T \ge 3$*

$$\begin{aligned}
R_T &\le \inf_{\varepsilon>0}\Big\{\varepsilon T_\varepsilon + \frac{2+8\log|A_T|+144\log(T/\delta)}{\varepsilon}\Big\} = \inf_{\varepsilon>0}\Big\{\varepsilon T_\varepsilon + O\Big(\frac{d\log T+\log(T/\delta)}{\varepsilon}\Big)\Big\} \\
N_T &\le \inf_{\varepsilon>0}\Big\{T_\varepsilon + O\Big(\frac{\log|A_T|\log(T/\delta)+\log^2|A_T|}{\varepsilon^2}\Big)\Big\} = \inf_{\varepsilon>0}\Big\{T_\varepsilon + O\Big(\frac{d^2\log^2(T/\delta)}{\varepsilon^2}\Big)\Big\} \ ,
\end{aligned}$$

where $|A_T|$ is the determinant of the matrix A_T.

Note that the bounds above depend on d the dimension of the instance space. In the case of a (possibly infinite-dimensional) Reproducing Kernel Hilbert Space, d is replaced by a quantity that depends on the spectrum of the data's Gram matrix.

The proof of Theorem 2 splits into a series of lemmas. For every $T > 0$ and $\varepsilon > 0$, we define

$$\begin{aligned}
U_{T,\varepsilon} &= \sum_{t=1}^{T} \bar{Z}_t \ \mathbf{1}\big\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\big\}, \\
Q_{T,\varepsilon} &= \sum_{t=1}^{T} Z_t \ \mathbf{1}\big\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\big\}|\Delta_t| \ ,
\end{aligned}$$

where $\bar{Z}_t = 1 - Z_t$. In the above, $U_{T,\varepsilon}$ deals with rounds where the algorithm does not make a query, while $Q_{T,\varepsilon}$ deals with rounds where the algorithm does make a query. The proof exploits the potential-based method for online ridge-regression-like algorithms we learned from Azoury and Warmuth (2001). See also the works of Hazan et al. (2007), Dani et al. (2008) and Crammer and Gentile (2011) for a similar use in different contexts. The potential function we use is the (quadratic) Bregman divergence $d_t(\mathbf{u},\mathbf{w}) = \frac{1}{2}(\mathbf{u}-\mathbf{w})^\top A_t(\mathbf{u}-\mathbf{w})$, where A_t is the matrix computed by Selective Sampler at time t.

The proof structure is as follows. First, Lemma 3 below decomposes the regret R_T into three parts:

$$R_T \le \varepsilon T_\varepsilon + U_{T,\varepsilon} + Q_{T,\varepsilon} \ .$$

Algorithm 1: Selective Sampler

input confidence level $\delta \in (0,1]$
initialize $\mathbf{w}_0 = \mathbf{0},\ A_0 = I$
for $t = 1, 2, \ldots$
  receive $\mathbf{x}_t \in \mathbb{R}^d : ||\mathbf{x}_t|| \leq 1,$ and set $\hat{\Delta}_t = \mathbf{w}_{t-1}{}^\top \mathbf{x}_t$
  predict $\hat{y}_t = \mathrm{sgn}(\hat{\Delta}_t) \in \{-1,+1\}$
  $\theta_t^2 = \mathbf{x}_t^\top A_{t-1}^{-1} \mathbf{x}_t \left(1 + 4\sum_{i=1}^{t-1} Z_i r_i + 36\log(t/\delta)\right)$
  $Z_t = \mathbb{1}\left\{\hat{\Delta}_t^2 \leq \theta_t^2\right\} \in \{0,1\}$
  if $Z_t = 1$
    query $y_t \in \{-1,+1\}$
    $$\mathbf{w}'_{t-1} = \begin{cases} \mathbf{w}_{t-1} - \mathrm{sgn}(\hat{\Delta}_t)\left(\frac{|\hat{\Delta}_t|-1}{\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}\right) A_{t-1}^{-1}\mathbf{x}_t & \text{if } |\hat{\Delta}_t| > 1 \\ \mathbf{w}_{t-1} & \text{otherwise} \end{cases}$$
    $A_t = A_{t-1} + \mathbf{x}_t\mathbf{x}_t^\top,\quad r_t = \mathbf{x}_t^\top A_t^{-1}\mathbf{x}_t,\quad \mathbf{w}_t = A_t^{-1}(A_{t-1}\mathbf{w}'_{t-1} + y_t\mathbf{x}_t)$
  else
    $A_t = A_{t-1},\ \mathbf{w}_t = \mathbf{w}_{t-1},\ r_t = 0$

The bound on $U_{T,\varepsilon}$ is given by Lemma 4. For the bound on $Q_{T,\varepsilon}$ and the bound on the number of queries N_T, we use Lemmas 5 and 6, respectively. However, both of these lemmas require that $(\Delta_t - \hat{\Delta}_t)^2 \leq \theta_t^2$ for all t. This assumption is taken care of by the subsequent Lemma 7. Since ε is a positive free parameter, we can take the infimum over $\varepsilon > 0$ to get the required results. In turn, many of these lemmas rely on technical lemmas given in Appendix A and Appendix B.

Lemma 3 *For any $\varepsilon > 0$ it holds that* $R_T \leq \varepsilon T_\varepsilon + U_{T,\varepsilon} + Q_{T,\varepsilon}$.

Proof We have

$$\begin{aligned}
&P_t(\hat{\Delta}_t y_t < 0) - P_t(\Delta_t y_t < 0) \\
&\quad\leq \mathbb{1}\{\hat{\Delta}_t\Delta_t \leq 0\}\left|2P_t(y_t = 1) - 1\right| \\
&\quad= \mathbb{1}\{\hat{\Delta}_t\Delta_t \leq 0\}|\Delta_t| \\
&\quad= \mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 \leq \varepsilon^2\}|\Delta_t| + \mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\}|\Delta_t| \\
&\quad\leq \varepsilon\,\mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 \leq \varepsilon^2\} + \mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\}|\Delta_t| \qquad (3) \\
&\quad= \varepsilon\,\mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 \leq \varepsilon^2\} + \mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2, Z_t = 0\}|\Delta_t| \\
&\qquad\qquad + \mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2, Z_t = 1\}|\Delta_t| \\
&\quad\leq \varepsilon\,\mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 \leq \varepsilon^2\} + \bar{Z}_t\,\mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\} \qquad (4) \\
&\qquad\qquad + Z_t\,\mathbb{1}\{\Delta_t\hat{\Delta}_t < 0, \Delta_t^2 > \varepsilon^2\}|\Delta_t| \,.
\end{aligned}$$

The inequality in Equation (4) follows directly from $|\Delta_t| \leq 1$. Summing over $t = 1 \ldots T$ completes the proof. ■

Lemma 4 *For any $\varepsilon > 0$ and $T \geq 3$, with probability at least $1-\delta$, it holds that*

$$Q_{T,\varepsilon} \;\leq\; \frac{2+8\log|A_T|+144\log(T/\delta)}{\varepsilon} \;=\; O\left(\frac{d\log T+\log(T/\delta)}{\varepsilon}\right) .$$

Proof We begin with

$$\begin{aligned} Q_{T,\varepsilon} \;&=\; \sum_{t=1}^{T} Z_t \,\mathbb{1}\{\Delta_t\hat{\Delta}_t<0\}\,\mathbb{1}\{\Delta_t^2>\varepsilon^2\}|\Delta_t| \\ &\leq\; \frac{1}{\varepsilon}\sum_{t=1}^{T} Z_t \,\mathbb{1}\{\hat{\Delta}_t\Delta_t<0\}\Delta_t^2 \\ &=\; \frac{1}{\varepsilon}\sum_{t=1}^{T} Z_t \,\mathbb{1}\{\hat{\Delta}'_t\Delta_t<0\}\Delta_t^2 \;\; . \end{aligned}$$

$\hat{\Delta}'_t\Delta_t<0$ implies that $\Delta_t^2 \leq (\Delta_t-\hat{\Delta}'_t)^2$, and therefore the above can be upper bounded by

$$\frac{1}{\varepsilon}\sum_{t=1}^{T} Z_t(\Delta_t-\hat{\Delta}'_t)^2.$$

Next we rely on some standard technical results that are given in the appendix. Lemma 23 (i) upper bounds the above by

$$\frac{2}{\varepsilon}\sum_{t=1}^{T} Z_t\,\left((y_t-\hat{\Delta}'_t)^2-(y_t-\Delta_t)^2\right)+\frac{144}{\varepsilon}\log(T/\delta).$$

Lemma 25 (iv) further bounds this term by

$$\frac{4}{\varepsilon}\sum_{t=1}^{T} Z_t\left(d_{t-1}(\mathbf{u},\mathbf{w}'_{t-1})-d_t(\mathbf{u},\mathbf{w}'_t)+2\log\frac{|A_t|}{|A_{t-1}|}\right)+\frac{144}{\varepsilon}\log(T/\delta).$$

After telescoping and using the facts that $d_0(\mathbf{u},\mathbf{w}'_0)=d_0(\mathbf{u},\mathbf{w}_0)=||\mathbf{u}||^2/2\leq 1/2$ and $|A_0|=1$, the above is bounded by

$$\frac{2+8\log|A_T|+144\log(T/\delta)}{\varepsilon},$$

which is in fact $O\left(\frac{d\log T+\log(T/\delta)}{\varepsilon}\right)$ in the finite-dimensional case. This concludes the proof. ■

Lemma 5 *Assume that $(\Delta_t-\hat{\Delta}_t)^2\leq\theta_t^2$ holds for all t. Then, for any $\varepsilon>0$, we have $U_{T,\varepsilon}=0$*

Proof We rewrite our assumption $(\Delta_t-\hat{\Delta}_t)^2\leq\theta_t^2$ as

$$\Delta_t\hat{\Delta}_t \geq \frac{\hat{\Delta}_t^2+\Delta^2-\theta_t^2}{2} \geq \frac{\hat{\Delta}_t^2-\theta_t^2}{2} .$$

However, if $\bar{Z}_t=1$, then $\hat{\Delta}_t^2>\theta_t^2$ and so $\Delta_t\hat{\Delta}_t\geq 0$. Hence, under the above assumption, we can guarantee that for any t, $\bar{Z}_t\,\mathbb{1}\{\Delta_t\hat{\Delta}_t<0\}=0$, thereby implying $U_{T,\varepsilon}=\sum_{t=1}^{T}\bar{Z}_t\,\mathbb{1}\{\Delta_t\hat{\Delta}_t<0,\Delta_t^2>\varepsilon^2\}=0$. ■

In the proof of the next two lemmas, we use the shorthand $g(t)=\sum_{i=1}^{t} Z_i r_i$.

Lemma 6 *Assume that* $(\Delta_t - \hat{\Delta}_t)^2 \le \theta_t^2$ *holds for all* t*. Then, for any* $\varepsilon > 0$*, and* $T > 0$ *we have*

$$N_T \;\le\; T_\varepsilon + O\left(\frac{\log|A_T|\log(T/\delta) + \log^2|A_T|}{\varepsilon^2}\right) \;=\; T_\varepsilon + O\left(\frac{d^2\log^2(T/\delta)}{\varepsilon^2}\right).$$

Proof Let us rewrite our assumption $(\Delta_t - \hat{\Delta}_t)^2 \le \theta_t^2$ as $|\Delta_t - \hat{\Delta}_t| \le \theta_t$. Then $|\hat{\Delta}_t| \le \theta_t$ implies $|\Delta_t| \le 2\theta_t$. We can write

$$\begin{aligned}
Z_t &= \mathbb{1}\{\hat{\Delta}_t^2 \le \theta_t^2\} \le \mathbb{1}\{\hat{\Delta}_t^2 \le \theta_t^2, \Delta_t^2 \le 4\theta_t^2\} \\
&= \mathbb{1}\left\{\hat{\Delta}_t^2 \le \theta_t^2, \Delta_t^2 \le 4\theta_t^2, \theta_t^2 \ge \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\} \\
&\quad + \mathbb{1}\left\{\hat{\Delta}_t^2 \le \theta_t^2, \Delta_t^2 \le 4\theta_t^2, \theta_t^2 < \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\} \\
&\le \mathbb{1}\left\{\hat{\Delta}_t^2 \le \theta_t^2, \theta_t^2 \ge \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\} + \mathbb{1}\left\{\Delta_t^2 \le 4\theta_t^2, \theta_t^2 < \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\}. \qquad (5)
\end{aligned}$$

By Lemma 24 (i) we have $\mathbf{x}_t^T A_{t-1}^{-1}\mathbf{x}_t \le 2\,r_t$, hence

$$\mathbb{1}\left\{\Delta_t^2 \le 4\theta_t^2, \theta_t^2 < \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\} \le \mathbb{1}\{\Delta_t^2 \le \varepsilon^2\}.$$

Plugging back into (5) and summing over t shows that, for any $\varepsilon > 0$,

$$N_T \le T_\varepsilon + \sum_{t=1}^{T} \mathbb{1}\left\{\hat{\Delta}_t^2 \le \theta_t^2, \theta_t^2 \ge \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\}.$$

Now observe that, by definition of Z_t and θ_t

$$\begin{aligned}
&\sum_{t=1}^{T}\left\{\hat{\Delta}_t^2 \le \theta_t^2, \theta_t^2 \ge \frac{\varepsilon^2 \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}{8\,r_t}\right\} \\
&\qquad = \sum_{t=1}^{T} Z_t\,\mathbb{1}\left\{8\,r_t\left(1 + 4g(t-1) + 36\log(t/\delta)\right) \ge \varepsilon^2\right\} \\
&\qquad \le \frac{8}{\varepsilon^2}\sum_{t=1}^{T} Z_t\,r_t\left(1 + 4g(t-1) + 36\log(t/\delta)\right).
\end{aligned}$$

Using Lemma 24 (ii), the above is upper bounded by

$$\frac{8}{\varepsilon^2}\,(1 + 36\log(T/\delta))\log|A_T| + \frac{32}{\varepsilon^2}\sum_{t=1}^{T} Z_t r_t g(t-1),$$

which is in turn upper bounded by

$$\frac{8}{\varepsilon^2}\,(1 + 36\log(T/\delta))\log|A_T| + \frac{16}{\varepsilon^2}\sum_{t=1}^{T}\left(g^2(t) - g^2(t-1)\right).$$

Again using Lemma 24 (ii), we upper bound the above by

$$\frac{8}{\varepsilon^2}\,(1+36\log(T/\delta))\log|A_T| + \frac{16}{\varepsilon^2}\,\log^2|A_T|.$$

This term is $O\left(\frac{\log|A_T|\log(T/\delta)+\log^2|A_T|}{\varepsilon^2}\right)$, and specifically $O\left(\frac{d^2\log^2(T/\delta)}{\varepsilon^2}\right)$ in the finite-dimensional case. Since this above holds for any $\varepsilon > 0$, it also holds for the best choice of ε. ■

Lemma 7 *If Selective Sampler is run with confidence parameter $\delta \in (0,1]$, then with probability at least $1-\delta$, the inequality $(\Delta_t - \hat{\Delta}_t)^2 \le \theta_t^2$ holds simultaneously for all $t \ge 3$.*

Proof First note that by Hölder's inequality,

$$(\Delta_t - \hat{\Delta}_t)^2 = ((\mathbf{w}_{t-1}-\mathbf{u})^\top \mathbf{x}_t)^2 \le 2\,\mathbf{x}_t^T A_{t-1}^{-1}\mathbf{x}_t\; d_{t-1}(\mathbf{w}_{t-1},\mathbf{u})\;. \tag{6}$$

Now let $t' := \operatorname{argmax}_{j\le t-1 : Z_j=1} j$, that is, t' is the last round (up to time $t-1$) on which the algorithm issued a query. Then Lemma 25 (i), (ii), (iii), allows us to write

$$\begin{aligned}\frac{1}{2}\sum_{i=1}^{t'} Z_i\left((y_i-\hat{\Delta}_i')^2-(y_i-\Delta_i)^2\right) &\le \sum_{i=1}^{t'-1} Z_i\left(d_{i-1}(\mathbf{u},\mathbf{w}_{i-1}')-d_i(\mathbf{u},\mathbf{w}_i')+2Z_i r_i\right)\\ &\quad + d_{t'-1}(\mathbf{u},\mathbf{w}_{t'-1}')-d_{t'}(\mathbf{u},\mathbf{w}_{t'})+2r_{t'}\\ &\le \frac{1}{2}-d_{t'}(\mathbf{u},\mathbf{w}_{t'})+2g(t')\;,\end{aligned}$$

where the last step comes from the telescoping sum and the fact that

$$d_0(\mathbf{u},\mathbf{w}_0') = d_0(\mathbf{u},\mathbf{w}_0) = \frac{1}{2}\|\mathbf{u}\|^2 \le 1/2.$$

Moreover, by definition of t' we see that $g(t') = g(t-1)$ and $Z_j = 0$ for any $j \in [t'+1, t-1]$. Hence for any such j we have $\mathbf{w}_j = \mathbf{w}_{t'}$. This yields

$$\frac{1}{2}\sum_{i=1}^{t-1} Z_i\left((y_i-\hat{\Delta}_i')^2-(y_i-\Delta_i)^2\right) \le \frac{1}{2}-d_{t-1}(\mathbf{u},\mathbf{w}_{t-1})+2g(t-1)\;.$$

Plugging back into (6) results in

$$(\Delta_t-\hat{\Delta}_t)^2 \le 2\,\mathbf{x}_t^T A_{t-1}^{-1}\mathbf{x}_t\left(1/2+2\,g(t-1)-\frac{1}{2}\sum_{i=1}^{t-1} Z_i\left((y_i-\hat{\Delta}_i')^2-(y_i-\Delta_i)^2\right)\right).$$

A direct application of Lemma 23 (ii) shows that for any given $t \ge 3$, with probability at least $1-\delta/t^2$,

$$(\Delta_t-\hat{\Delta}_t)^2 \le \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t\,(1+4\,g(t-1)+36\log(t/\delta)) = \theta_t^2\;.$$

Finally, a union bound allows us to conclude that $(\Delta_t-\hat{\Delta}_t)^2 \le \theta_t^2$ holds simultaneously for all $t \ge 3$ with probability at least $1-\delta$. ■

Remark 8 *Computing the intermediate vector $\mathbf{w}'_{t-1}$ from $\mathbf{w}_{t-1}$, as defined in Algorithm 1, corresponds to projecting $\mathbf{w}_{t-1}$ onto the convex set $C_t = \{\mathbf{w} \in \mathbb{R}^d : |\mathbf{w}^\top \mathbf{x}_t| \leq 1\}$ w.r.t. the Bregman divergence d_{t-1}, that is, $\mathbf{w}'_{t-1} = \operatorname{argmin}_{\mathbf{u}\in C_t} d_{t-1}(\mathbf{u}, \mathbf{w}_{t-1})$. Notice that C_t includes the unit ball since $\mathbf{x}_t$ is normalized. This projection step is needed for technical purposes during the construction of a suitable bounded-variance martingale difference sequence (see Lemma 23 in Appendix A). Unlike similar constructions (Hazan et al., 2007; Dani et al., 2008), we do not project onto the unit ball. In fact, computing the latter would involve a line search over matrices, which would significantly slow down the algorithm. On the other hand, it is also interesting to observe that Selective Sampler performs the projection onto C_t only a logarithmic number of times. This is because*

$$\begin{aligned}\sum_{t=1}^{T} \mathbf{1}\{\hat{\Delta}_t^2 \leq \theta_t^2, |\hat{\Delta}_t| > 1\} &\leq \sum_{t=1}^{T} Z_t \hat{\Delta}_t^2 \\ &\leq \sum_{t=1}^{T} Z_t \theta_t^2 \\ &\leq 2\sum_{t=1}^{T} Z_t\, r_t\Big(1 + 4\,g(t-1) + 36\log(t/\delta)\Big),\end{aligned}$$

which is $O\left(d^2 \log^2(T/\delta)\right)$ by Lemma 24 (iii).

2.4 An Online-to-Batch Conversion

It is instructive to see what the bound in Theorem 2 looks like when we assume that the instances $\mathbf{x}_t$ are drawn i.i.d. according to an unknown distribution over the Euclidean unit sphere, and to compare this bound to standard statistical learning bounds. We model the distribution of the instances near the hyperplane $\{\mathbf{x} : \mathbf{u}^\top \mathbf{x} = 0\}$ using the well-known *Mammen-Tsybakov low noise condition* (Tsybakov, 2004):[2]

$$\text{There exist } c > 0 \text{ and } \alpha \geq 0 \text{ such that } P\big(|\mathbf{u}^\top \mathbf{x}| < \varepsilon\big) \leq c\,\varepsilon^\alpha \text{ for all } \varepsilon > 0.$$

We now describe a simple randomized algorithm which, with high probability over the sampling of the data, returns a linear predictor with a small expected risk (expectation is taken over the randomization of the algorithm). The algorithm is as follows:

1. Run Algorithm 1 with confidence level δ on the data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$, and obtain the sequence of predictors $\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{T-1}$

2. Pick $r \in \{0, 1, \ldots, T-1\}$ uniformly at random and return $\mathbf{w}_r$.

Due to the unavailability of all labels, standard conversion techniques that return a single deterministic hypothesis (Cesa-Bianchi and Gentile, 2008) do not readily apply here. The following theorem, whose proof is given in Appendix C, states a high probability bound on the risk and the label complexity of our algorithm.

2. The constant c might actually depend on the input dimension d. For notational simplicity, Theorem 9 regards c as a constant, hence it is hidden in the big-oh notation.

Theorem 9 *Let $\mathbf{w}_r$ be the linear hypothesis returned by the above algorithm. Then with probability at least $1-\delta$ we have*

$$\mathbb{E}_r\left[P'_r(y\,\mathbf{w}_r^\top\mathbf{x}<0)\right] \le P(y\,\mathbf{u}^\top\mathbf{x}<0) + O\left(\frac{(d\log(T/\delta))^{\frac{\alpha+1}{\alpha+2}}}{T^{\frac{\alpha+1}{\alpha+2}}} + \frac{\log\left(\frac{\log T}{\delta}\right)}{T}\right) \ ,$$

$$N_T = O\left((d^2\log^2(T/\delta))^{\frac{\alpha}{\alpha+2}}\,T^{\frac{2}{\alpha+2}} + \log(1/\delta)\right) \ ,$$

where $\mathbb{E}_r$ is the expectation over the randomization in the algorithm, and $P'_r(\cdot)$ denotes the conditional probability $P(\cdot\,|\,\mathbf{x}_1,\ldots,\mathbf{x}_{r-1},y_1,\ldots,y_{r-1})$.[3]

As α goes from 0 (no assumptions on the noise) to ∞ (hard separation assumption), the above bound on the average regret roughly interpolates between $1/\sqrt{T}$ and $1/T$. Correspondingly, the bound on the number of labels N_T goes from T to $\log^2 T$. In particular, observe that, viewed as a function of N_T (and disregarding log factors), the instantaneous regret is of the form $N_T^{-\frac{\alpha+1}{2}}$. These bounds are sharper than those by Cavallanti et al. (2009) and, in fact, no further improvement is generally possible (Castro and Nowak, 2008). The same rates are obtained by Hanneke (2009) under much more general conditions, for less efficient algorithms that are based on empirical risk minimization.

2.5 Statistical Active Learning

We now briefly show how to turn our algorithm into a standard statistical active learning algorithm.

Following Koltchinskii (2010), we consider a sequential learning protocol for active learning where on round t the algorithm has to choose a subset S_t of the instance space $\{\mathbf{x}\in R^d : ||\mathbf{x}|| \le 1\}$ (the Euclidean sphere) from which the next instance $\mathbf{x}_t$ is sampled from. Specifically, $\mathbf{x}_t$ is sampled from the conditional distribution $P(\cdot\,|\mathbf{x}\in S_t)$, being $P(\cdot)$ an unknown distribution over the Euclidean sphere. The algorithm then observes the associated label y_t, generated according to the linear noise model of Section 2.1. Notice that the set S_t is typically depending on past examples $(\mathbf{x}_1,y_1),\ldots(\mathbf{x}_{t-1},y_{t-1})$. Again, the goal is to study the high probability behavior of the regret as a function of the number of observed labels (which now coincides with the number of sampled instances $\mathbf{x}_t$).

The analysis developed in Section 2.4 immediately accommodates this model of learning, once we let S_t be the querying region of Algorithm 1, that is,

$$S_t = \{\mathbf{x} : (\mathbf{w}_{t-1}^\top\mathbf{x})^2 \le \theta_t^2\},$$

and think of the randomized i.i.d. algorithm described in that section as operating as follows. We sample an independent new instance $\mathbf{x}$ from the Euclidean sphere, and check whether $\mathbf{x}\in S_t$ or not. In the former case, the associated label y is sampled, and the subset S_t is updated into S_{t+1} according to the rules of Algorithm 1 for updating $\mathbf{w}_{t-1}$ into $\mathbf{w}_t$ and θ_t into θ_{t+1}. In the latter case, $\mathbf{x}$ is discarded, S_t remains unchanged, and a new independent instance is drawn. Notice that this is precisely what Algorithm 1 does when running with an i.i.d. sequence of examples. The same conclusions we have drawn from Theorem 9 can be repeated here.

3. Notice the difference with the conditional probability $P_r(\cdot)$ defined in Section 2.1.

2.6 Related Work

As we mentioned in Section 1.1, the results of Theorem 2 are more in line with the worst-case analyses by Strehl and Littman (2008), Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011). These papers present variants of Recursive Least Squares algorithms that operate on arbitrary instance sequences, but assuming the same linear stochastic noise-model used in our analysis. The algorithm presented by Strehl and Littman (2008) approximates the Bayes margin to within a given accuracy ε, and queries $\tilde{O}(d^3/\varepsilon^4)$ labels; this bound is significantly inferior to our bound, and it seems to hold only in the finite-dimensional case. A more precise comparison can be made to the (expectation) bounds presented by Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011), which are of the form $R_T \leq \min_{0<\varepsilon<1}\left(\varepsilon T_\varepsilon + \frac{T^{1-\kappa}}{\varepsilon} + \frac{d}{\varepsilon^2}\ln T\right)$, and $N_T = O(dT^\kappa \ln T)$, where $\kappa \in [0,1]$ is a tunable parameter of their algorithm. After a proper setting of κ, this gives rise to an instantaneous regret which is still (up to log factors) in the form $N_T^{-\frac{\alpha+1}{2}}$ under the same low-noise assumptions as in Section 2.4. On the other hand, our bound here does not require tuning of parameters. More importantly, whereas the analysis of Cesa-Bianchi et al. (2009) and Orabona and Cesa-Bianchi (2011) only holds for oblivious adversaries, we cover the case where the instances can be generated adaptively.[4] We emphasize that it is just the adaptivity of the adversary that enabled us to convert our selective sampling algorithm to the statistical active learning algorithm presented in Section 2.5.

Another relevant line of research that came to our attention at the time of writing the extended version of our paper is the importance sampling-based active learning schemes followed by Beygelzimer et al. (2010, 2011). These papers are interesting in that they give up with the version space approach followed by their predecessors (Dasgupta et al., 2008; Hanneke, 2007, 2009; Koltchinskii, 2010) which might deliver time-efficient active learning schemes. A direct comparison to Beygelzimer et al. (2010, 2011) is not straightforward. While we can see that their label selection mechanism (e.g., Algorithm 1 in Beygelzimer et al., 2010) gets similar to the one in our Selective Sampler (once it is adapted to square loss and the class conditional distribution is (1)), their analysis (e.g., Theorem 4 therein) seems to provide suboptimal results. For instance, under hard separation assumptions, their bound on N_T never gets as small as a logarithmic function in T. In short, we suspect that their algorithm (or variants thereof) is a strict generalization of ours but, being more general, the associated analysis is also significantly looser.

3. The Multiple Teacher Case

The problem is still online binary classification, where on each round $t = 1, 2, \ldots$ the learner receives an input $\mathbf{x}_t \in \mathbb{R}^d$, with $\|\mathbf{x}_t\| \leq 1$, and outputs a binary prediction $\hat{y}_t$. However, there are now K available teachers, each with his own area of expertise. The expertise area of each teacher is unknown to the algorithm, and can only be inferred indirectly from the binary labels provided by that teacher and by other teachers. If $\mathbf{x}_t$ falls within the expertise region of teacher j, then that teacher can provide an accurate label. After making each binary prediction, the learner chooses if to issue a query to one or more of the K teachers. The learner is free to query any subset of teachers, but each

4. It is fair to say that Orabona and Cesa-Bianchi (2011) have further improvements over both Cesa-Bianchi et al. (2009) and this paper. In particular, the DGS-Mod algorithm therein is able to handle the case when the vector $\mathbf{u}$ generating the labels has unknown length $||\mathbf{u}||$. However, it does so at the cost of an exponential dependence of R_T on $||\mathbf{u}||$.

teacher charges a unit cost per label. We emphasize that a queried teacher provides only a binary label, and does not indicate his level of confidence in that label.

From the point of view of our learning algorithm, these confidence levels have to be interpreted as *reliability* rates of the teachers. Since these rates play a major role in weighting the relative importance of the teachers, it looks wiser to let the algorithm compute these rates as a function of past interactions among teachers, rather than relying on human "self-judgement".

Formally, we assume that teacher j is associated with a weight vector $\mathbf{u}_j \in \mathbb{R}^d$, where $\|\mathbf{u}_j\| \leq 1$. If teacher j is queried on round t, he stochastically generates the binary label $y_{j,t}$ according to $P_t(y_{j,t} = 1|\mathbf{x}_t) = (1+\Delta_{j,t})/2$, where $\Delta_{j,t} = \mathbf{u}_j{}^\top \mathbf{x}_t$ and, as in Section 2, $\mathbf{x}_t$ can be chosen adversarially depending on previous $\mathbf{x}$'s and y_j's. We consider $|\Delta_{j,t}|$ to be the (hidden) *confidence* of teacher j in his label for $\mathbf{x}_t$. When the learner issues a query, he receives nothing other than the binary label itself, and the confidence is only part of our theoretical model of the teacher. If $\mathbf{x}_t$ is almost orthogonal to $\mathbf{u}_j$ then teacher j has a very low confidence in his label, and we say that $\mathbf{x}_t$ lies outside the expertise region of teacher j.

It is no longer clear how we should evaluate the performance of the learner, since the K teachers will often give inconsistent labels on the given $\mathbf{x}_t$, and we do not have a well-defined ground-truth to compare against. Intuitively, we would like the learner to predict the label of $\mathbf{x}_t$ as accurately as the teachers who are experts on $\mathbf{x}_t$. To formalize this intuition,[5] define the average margin of a generic subset of teachers $C \subseteq [K]$ as $\Delta_{C,t} = \frac{1}{|C|}\sum_{i \in C} \Delta_{i,t}$. We define the set of experts for each instance using a user-specified parameter $\tau > 0$. Define

$$j_t^\star \;=\; \text{argmax}_j |\Delta_{j,t}| \quad \text{and} \quad C_t \;=\; \{i : |\Delta_{i,t}| \geq |\Delta_{j_t^\star,t}| - \tau\} \;. \tag{7}$$

In words, $j_t^\star$ is the *most confident teacher* at time t, and C_t is the *set of confident teachers* at time t. Again, recall that C_t is unknown to the learning algorithm. In this setting, τ is a tolerance parameter that defines how confident a teacher must be, compared to the most confident teacher, to be considered a confident teacher. Although τ does not appear explicitly in the notation C_t, the reader should keep in mind that C_t and other sets defined later on in this section all depend on τ. Using the definitions above, $\Delta_{C_t,t}$ is the average margin of the confident teachers, and we abbreviate $\Delta_t = \Delta_{C_t,t}$.

Now, let y_t be the random variable that takes values in $\{-1,1\}$, with $P_t(y_t = 1|\mathbf{x}_t) = (1+\Delta_t)/2$. In words, y_t is the binary label generated according to the average margin of the confident teachers. We consider the sequence $y_1, \ldots, y_T$ to be our ad-hoc ground-truth, and the goal of our algorithm is to accurately predict this sequence. Note that an equivalent way of generating y_t is to pick a confident teacher j uniformly at random from C_t and to set $y_t = y_{j,t}$. Indeed there are other reasonable ways to define the ground-truth for this problem, however, we feel that our definition coincides with our intuitions on learning from teachers with different areas of expertise. If τ is set to 1, the learner is compared against the average margin of all K teachers, while if $\tau = 0$, the learner is compared against the single most confident teacher.

Remark 10 *The reader might wonder whether the framework just described could be accommodated by a standard experts setting (e.g., Cesa-Bianchi and Lugosi, 2006) or, perhaps, by a label-efficient version thereof (e.g., Helmbold and Panizza, 1997; Cesa-Bianchi et al., 2005b). Due to the absence of a ground truth, the answer is negative. Of course, we might be tempted to apply*

5. Here and throughout, $[K] = \{1, 2, \ldots, K\}$.

a label-efficient expert algorithm by pretending that the missing ground-truth is provided by some function of the teachers we query. Unfortunately, the above references contain results which are too general to yield tight bounds for our specific noise model. Indeed, our ambition here is to leverage the side information provided by the instance vectors so as to outperform the best single expert in hindsight while, at the same time, querying just a small fraction of the available teachers' labels.

We now describe and analyze two algorithms within the multiple teacher setting. We call these algorithms "first version" and "second version". In the first version, the algorithm queries either all of the teachers or none of them. The second version is more refined in that the algorithm may query a different subset of teachers on each round. In Section 4 we present experiments on real-world data with the second version of the algorithm.

3.1 Algorithm, First Version

The learner attempts to model each weight vector $\mathbf{u}_j$ with a corresponding weight vector $\mathbf{w}_{j,t}$. As in the single teacher case, the learner maintains a variable threshold θ_t, which can be interpreted as the learner's confidence in its current set of weight vectors. The learner attempts to mimic the process of generating y_t by choosing its own set of confident teachers on each round. Denoting $\hat{\Delta}_{j,t} = \mathbf{w}_{j,t}{}^{\top}\mathbf{x}_t$, the learner defines

$$\hat{j}_t \;=\; \text{argmax}_j|\hat{\Delta}_{j,t}| \quad \text{and} \quad \hat{C}_t \;=\; \{i : |\hat{\Delta}_{i,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau - 2\theta_t\} \;\;,$$

where $\hat{j}_t$ is the learner's estimate of the most confident teacher, and $\hat{C}_t$ is the learner's estimate of the set of confident teachers. Note that the definition of $\hat{C}_t$ is more inclusive than the definition of C_t in Equation (7), in that it also includes teachers whose confidence falls below $|\hat{\Delta}_{\hat{j}_t,t}| - \tau$. This accounts for the uncertainty regarding the learner's set of weight vectors.

As above, we define the notation $\hat{\Delta}_{C,t} = \frac{1}{|C|}\sum_{i\in C}\hat{\Delta}_{i,t}$, and abbreviate $\hat{\Delta}_t = \hat{\Delta}_{\hat{C}_t,t}$. The learner predicts the binary label $\hat{y}_t = \text{sgn}(\hat{\Delta}_t)$. Let P_t denote the conditional probability $P_t(\cdot) = \mathbb{P}(\cdot|\mathbf{x}_1, y_{1,1}\ldots, y_{K,1}, \mathbf{x}_2, y_{1,2}\ldots, y_{K,2}, \ldots \mathbf{x}_{t-1}, y_{1,t-1}, \ldots y_{K,t-1}, \mathbf{x}_t)$, and define the regret of the learner as

$$R_T \;=\; \sum_{t=1}^{T}\left(P_t(y_t\hat{\Delta}_t < 0) - P_t(y_t\Delta_t < 0)\right) \;\;. \tag{8}$$

Next, we proceed to describe our criterion for querying teachers. We present a simple criterion that either sets $Z_t = 1$ and queries all of the teachers or sets $Z_t = 0$ and queries none of them. Therefore, the learner either incurs a cost of K or a cost of 0 on each round. We partition the set of confident teachers $\hat{C}_t$ into two sets,

$$\begin{aligned}\hat{H}_t \;&=\; \{i \;:\; |\hat{\Delta}_{i,t}| \;\geq\; |\hat{\Delta}_{\hat{j}_t,t}| - \tau + 2\theta_t\},\\ \hat{B}_t \;&=\; \{i \;:\; |\hat{\Delta}_{\hat{j}_t,t}| - \tau - 2\theta_t \;\leq\; |\hat{\Delta}_{i,t}| \;<\; |\hat{\Delta}_{\hat{j}_t,t}| - \tau + 2\theta_t\}\;.\end{aligned}$$

In words, $\hat{H}_t$ is the set of teachers with especially high confidence, while $\hat{B}_t$ is the set of teachers with borderline confidence. Intuitively, the learner is unsure whether the teachers in $\hat{B}_t$ should or should not be included in $\hat{C}_t$. The learner issues a query (to all K teachers) in one of two cases. The first case is when there exists a subset of borderline teachers $S \subseteq \hat{B}_t$ that causes the predicted label to flip, namely, $\hat{\Delta}_t\hat{\Delta}_{\hat{H}_t\cup S,t} < 0$. The second case is when there exists a subset of borderline teachers

Algorithm 2: Multiple Teacher Selective Sampler—first version

input confidence level $\delta \in (0,1]$, tolerance parameter $\tau \geq 0$

initialize $A_0 = I$, $\forall j \in [K]$ $\mathbf{w}_{j,0} = \mathbf{0}$

for $t = 1, 2, \ldots$

receive $\mathbf{x}_t \in \mathbb{R}^d$: $||\mathbf{x}_t|| \leq 1$

$\theta_t^2 = \mathbf{x}_t^\top A_{t-1}^{-1} \mathbf{x}_t \left(1 + 4\sum_{i=1}^{t-1} Z_i r_i + 36\log(Kt/\delta)\right)$

$\forall j \in [K]$ $\hat{\Delta}_{j,t} = \mathbf{w}_{j,t-1}{}^\top \mathbf{x}_t$ and $\hat{j}_t = \operatorname{argmax}_j |\hat{\Delta}_{j,t}|$

predict $\hat{y}_t = \operatorname{sgn}(\hat{\Delta}_t) \in \{-1, +1\}$

$$Z_t = \begin{cases} 1 & \text{if } \exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S \cup \hat{H}_t, t} < 0 \text{ or } |\hat{\Delta}_{S \cup \hat{H}_t, t}| \leq \theta_t \\ 0 & \text{otherwise} \end{cases}$$

if $Z_t = 1$

query $y_{1,t}, \ldots, y_{K,t}$

$A_t = A_{t-1} + \mathbf{x}_t \mathbf{x}_t^\top$, $r_t = \mathbf{x}_t^\top A_t^{-1} \mathbf{x}_t$

for $j = 1, \ldots, K$

$$\mathbf{w}'_{j,t-1} = \begin{cases} \mathbf{w}_{j,t-1} - \operatorname{sgn}(\hat{\Delta}_{j,t}) \left(\frac{|\hat{\Delta}_{j,t}| - 1}{\mathbf{x}_t^\top A_{t-1}^{-1} \mathbf{x}_t} \right) A_{t-1}^{-1} \mathbf{x}_t & \text{if } |\hat{\Delta}_{j,t}| > 1, \\ \mathbf{w}_{j,t-1} & \text{otherwise} \end{cases}$$

$$\mathbf{w}_{j,t} = A_t^{-1}(A_{t-1}\mathbf{w}'_{j,t-1} + y_{j,t}\mathbf{x}_t)$$

else

$A_t = A_{t-1}$, $r_t = 0$ and $\mathbf{w}_{j,t} = \mathbf{w}_{j,t-1}$ $\forall j \in [K]$

$S \subseteq \hat{B}_t$ that causes the margin to be too small, namely $|\hat{\Delta}_{\hat{H}_t \cup S, t}| \leq \theta_t$. In either of these cases, we say that the set of (estimated) confident teachers is *unstable*. If a query is issued, each weight vector $\mathbf{w}_{j,t}$ is updated as in the single teacher case. The pseudocode of this algorithm is given in Algorithm 2.

Remark 11 *At first sight, it may seem that computing Z_t causes an exponential explosion due to the need to check all possible subsets $S \subseteq \hat{B}_t$. The same implementation issue arises in Algorithm 3 (Section 3.3). As a matter of fact, this check can be computed efficiently by first sorting the teachers according to their estimated confidence $|\hat{\Delta}_{j,t}|$, and then greedily growing the subset S by following this order.*

3.2 Analysis, First Version

Our learning algorithm relies on labels it receives from a set of teachers, and therefore our bounds should naturally depend on the ability of those teachers to provide accurate labels for the sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$. For example, if an input $\mathbf{x}_t$ lies outside the expertise regions of all teachers, we cannot hope to learn anything from the labels provided by the teachers for this input. Similarly, there

is nothing we can do on rounds where the set of confident teachers is split between two equally confident but conflicting opinions. We count these difficult rounds by defining, for any $\varepsilon > 0$,

$$T_\varepsilon \;=\; \sum_{t=1}^{T} \mathbf{1}\{|\Delta_t| \le \varepsilon\} \;. \tag{9}$$

The above is just a multiple teacher counterpart to (2). However it is interesting to note that even in a case where most teachers have low confidence in their prediction on any given round, T_ε can still be small provided that the experts in the field have a confident opinion.

A more subtle difficulty presents itself when the collective opinion expressed by the set of confident teachers changes qualitatively with a small perturbation of the input $\mathbf{x}_t$ or one of the weight vectors $\mathbf{u}_j$. To state this formally, define for any $\varepsilon > 0$

$$\begin{aligned} H_{\varepsilon,t} \;&=\; \{i \;:\; |\Delta_{i,t}| \;\ge\; |\Delta_{j_t^\star,t}| - \tau + \varepsilon\}, \\ B_{\varepsilon,t} \;&=\; \{i \;:\; |\Delta_{j_t^\star,t}| - \tau - \varepsilon \;\le\; |\Delta_{i,t}| \;<\; |\Delta_{j_t^\star,t}| - \tau + \varepsilon\} \;. \end{aligned}$$

The set $H_{\varepsilon,t}$ is the subset of teachers in C_t with especially high confidence, ε higher than the minimal confidence required for inclusion in C_t. In contrast, the set $B_{\varepsilon,t}$ is the set of teachers with borderline confidence: either teachers in C_t that would be excluded if their margin were smaller by ε, or teachers that are not in C_t that would be included if their margin were larger by ε. We say that the average margin of the confident teachers is *unstable* with respect to τ and ε if $|\Delta_t| > \varepsilon$ but we can find a subset $S \subseteq B_{\varepsilon,t}$ such that either $\Delta_t \Delta_{S \cup H_{\varepsilon,t},t} < 0$ or $|\Delta_{S \cup H_{\varepsilon,t},t}| < \varepsilon$. In other words, we are dealing with the situation where Δ_t is sufficiently confident, but a small ε-perturbation to the margins of the individual teachers can cause its sign to flip, or its confidence to fall below ε. We count the unstable rounds by defining, for any $\varepsilon > 0$,[6]

$$T'_\varepsilon \;=\; \sum_{t=1}^{T} \mathbf{1}\big\{|\Delta_t| > \varepsilon\big\}\, \mathbf{1}\big\{\exists S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t},t} < 0 \;\vee\; |\Delta_{S \cup H_{\varepsilon,t},t}| \le \varepsilon\big\} \;. \tag{10}$$

Intuitively T'_ε counts the number of rounds on which an ε-perturbation of $\Delta_{t,j}$ either changes the sign of the average margin or results in an average margin close to zero. Like T_ε, this quantity measures an inherent hardness of the multiple teacher problem.

The following theorem is the main theoretical result of this section. It provides an upper bound on the regret of the learner, as defined in Equation (8), and on the total cost of queries, $N_T = K \sum_{t=1}^{T} Z_t$. Again, we emphasize both the data and the time-dependent aspects of the bound.

6. Notice that, up to degenerate cases, both T_ε and T'_ε tend to vanish as $\varepsilon \to 0$. Hence, as in the single teacher case, the free parameter ε trades-off hardness terms against large deviation terms.

Theorem 12 *Assume Algorithm 2 is run with a confidence parameter $\delta > 0$. Then with probability at least $1-\delta$ it holds for all $T \geq 3$ that*

$$\begin{aligned}
R_T &\leq \inf_{\varepsilon>0}\left\{\varepsilon T_\varepsilon + T'_\varepsilon + O\left(\frac{\log|A_T|\,\log(KT/\delta)+\log^2|A_T|}{\varepsilon^2}\right)\right\} \\
&= \inf_{\varepsilon>0}\left\{\varepsilon T_\varepsilon + T'_\varepsilon + O\left(\frac{d^2\log^2(KT/\delta)}{\varepsilon^2}\right)\right\} \ , \\
N_T &\leq K \inf_{\varepsilon>0}\left\{T_\varepsilon + T'_\varepsilon + O\left(\frac{\log|A_T|\,\log(KT/\delta)+\log^2|A_T|}{\varepsilon^2}\right)\right\} \\
&= K \inf_{\varepsilon>0}\left\{T_\varepsilon + T'_\varepsilon + O\left(\frac{d^2\log^2(KT/\delta)}{\varepsilon^2}\right)\right\} \ .
\end{aligned}$$

As in the proof of Theorem 2, we begin by decomposing the regret and the number of queries. Recall the definitions of T_ε and T'_ε in Equation (9) and Equation (10), respectively. Additionally, define for any $\varepsilon > 0$

$$\begin{aligned}
U_T &= \textstyle\sum_{t=1}^T \bar{Z}_t \, \mathbb{1}\{\Delta_t \hat{\Delta}_t < 0\}, \\
Q_{T,\varepsilon} &= \textstyle\sum_{t=1}^T Z_t \, \mathbb{1}\{\forall S \subseteq B_{\varepsilon,t} \ : \ \Delta_t \Delta_{S \cup H_{\varepsilon,t}} \geq 0, \ |\Delta_{S \cup H_{\varepsilon,t}}| > \varepsilon\} \ .
\end{aligned} \tag{11}$$

T_ε and T'_ε deal with rounds on which the ground truth itself is unreliable, U_T sums over rounds where the learner does not issue a query, and $Q_{T,\varepsilon}$ sums over rounds where the learner does issue a query. Using these definitions, we state the following decomposition lemmas.

Lemma 13 *For any $\varepsilon > 0$ it holds that $R_T \ \leq \ \varepsilon T_\varepsilon + T'_\varepsilon + U_T + Q_{T,\varepsilon}$.*

Lemma 14 *For any $\varepsilon > 0$, it holds that $N_T \leq K\,(T_\varepsilon + T'_\varepsilon + Q_{T,\varepsilon})$.*

The proofs of these lemmas are given in Appendix C. To conclude the proof of Theorem 12, it remains to upper bound U_T and $Q_{T,\varepsilon}$.

Lemma 15 *If $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_t^2$ holds for all $j \in [K]$ and $t \in [T]$, then*

$$Q_{T,\varepsilon} = O\left(\frac{\log|A_T|\,\log(KT/\delta)+\log^2|A_T|}{\varepsilon^2}\right) = O\left(\frac{d^2\log^2(KT/\delta)}{\varepsilon^2}\right) \ .$$

Lemma 16 *If $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_t^2$ for all $j \in [K]$ and $t \in [T]$, then $U_T = 0$.*

The proofs of these lemmas are also given in Appendix C. Both lemmas rely on the assumption that $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_t^2$ for all $t \in [T]$ and $j \in [K]$. A straightforward stratification of Lemma 7 in Section 2 over the K teachers verifies that this condition holds with high probability. This concludes the proof of Theorem 2.

Algorithm 3: Multiple Teacher Selective Sampler—second version

input confidence level $\delta \in (0,1]$, tolerance parameter $\tau \geq 0$

initialize $A_{j,0} = I$, $\mathbf{w}_{j,0} = \mathbf{0}$, $\forall j \in [K]$

for $t = 1, 2, \ldots$

receive $\mathbf{x}_t \in \mathbb{R}^d$: $||\mathbf{x}_t|| \leq 1$

$\forall j \in [K]$, $\theta^2_{j,t} = \mathbf{x}_t^\top A^{-1}_{j,t-1} \mathbf{x}_t \big(1 + 4\sum_{i=1}^{t-1} Z_i r_{j,i} + 36\log(Kt/\delta)\big)$

$\forall j \in [K]$, $\hat{\Delta}_{j,t} = \mathbf{w}_{j,t-1}{}^\top \mathbf{x}_t$ and $\hat{j}_t = \text{argmax}_j |\hat{\Delta}_{j,t}|$

predict $\hat{y}_t = \text{sgn}(\hat{\Delta}_t) \in \{-1, +1\}$

$$Z_t = \begin{cases} 1 & \text{if } \exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S \cup \hat{H}_t, t} < 0 \quad \text{or} \quad |\hat{\Delta}_{S \cup \hat{H}_t, t}| \leq \theta_{S \cup \hat{H}_t, t} \\ 0 & \text{otherwise} \end{cases}$$

if $Z_t = 1$ and $j \in \hat{C}_t$

query $y_{j,t}$

$$A_{j,t} = A_{j,t-1} + \mathbf{x}_t \mathbf{x}_t^\top, \quad r_{j,t} = \mathbf{x}_t^\top A^{-1}_{j,t} \mathbf{x}_t$$

$$\mathbf{w}'_{j,t-1} = \begin{cases} \mathbf{w}_{j,t-1} - \text{sgn}(\hat{\Delta}_{j,t}) \left(\frac{|\hat{\Delta}_{j,t}| - 1}{\mathbf{x}_t^\top A^{-1}_{j,t-1} \mathbf{x}_t} \right) A^{-1}_{j,t-1} \mathbf{x}_t & \text{if } |\hat{\Delta}_{j,t}| > 1, \\ \mathbf{w}_{j,t-1} & \text{otherwise} \end{cases}$$

$$\mathbf{w}_{j,t} = A^{-1}_{j,t} (A_{j,t-1} \mathbf{w}'_{j,t-1} + y_{j,t} \mathbf{x}_t)$$

else

$A_{j,t} = A_{j,t-1}$, $r_{j,t} = 0$ and $\mathbf{w}_{j,t} = \mathbf{w}_{j,t-1}$

3.3 Algorithm, Second Version

The second version differs from the first one in that now each teacher j has its own threshold $\theta_{j,t}$, and also its own matrix $A_{j,t}$. As a consequence, the set of confident teachers $\hat{C}_t$ and the partition of $\hat{C}_t$ into highly confident ($\hat{H}_t$) and borderline ($\hat{B}_t$) teachers have to be redefined as follows:

$$\begin{aligned} \hat{C}_t &= \{j : |\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau - \theta_{j,t} - \theta_{\hat{j}_t,t}\}, \qquad \text{where } \hat{j}_t = \text{argmax}_j |\hat{\Delta}_{j,t}|, \\ \hat{H}_t &= \{i \,:\, |\hat{\Delta}_{i,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau + \theta_{j,t} + \max_{j \in \hat{C}_t} \theta_{j,t}\}, \\ \hat{B}_t &= \left\{i : |\hat{\Delta}_{\hat{j}_t,t}| - \tau - \theta_{j,t} - \theta_{\hat{j}_t,t} \leq |\hat{\Delta}_{i,t}| < |\hat{\Delta}_{\hat{j}_t,t}| - \tau + \theta_{j,t} + \max_{j \in \hat{C}_t} \theta_{j,t}\right\}. \end{aligned}$$

The pseudocode is given in Algorithm 3. Notice that the query condition defining Z_t now depends on an *average threshold* $\theta_{S \cup \hat{H}_t, t} = \frac{1}{|S \cup \hat{H}_t|} \sum_{j \in S \cup \hat{H}_t} \theta_{j,t}$.

3.4 Analysis, Second Version

The following theorem bounds the regret and the total number of queries issued by the second version of our algorithm, with high probability. The proof is similar to the proof of Theorem 12. We keep the definitions of the sets $H_{\varepsilon,t}$ and $B_{\varepsilon,t}$ as given in Section 3.2, but in the bound on N_T in

Theorem 17, we replace T'_ε with the more refined quantity T''_ε, defined as

$$T''_\varepsilon \;=\; \sum_{t=1}^{T} \frac{|H_{\varepsilon,t} \cup B_{\varepsilon,t}|}{K}\, \mathbf{1}\big\{|\Delta_t| > \varepsilon\big\}\, \mathbf{1}\big\{\exists S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t},t} < 0 \;\vee\; |\Delta_{S \cup H_{\varepsilon,t},t}| \le \varepsilon\big\}\;.$$

Note that T''_ε is similar to T'_ε except that while T'_ε only counts the number of times that perturbations to the $\Delta_{j,t}$'s lead to conflict or low confidence predictions, T''_ε also accounts for the fraction of confident teachers involved in the conflict. We state the following bound on regret and on the overall number of queries.

Theorem 17 *Assume Algorithm 3 is run with a confidence parameter $\delta > 0$. Then with probability at least $1-\delta$ it holds for all $T \ge 3$ that*

$$\begin{aligned}
R_T &\le \inf_{\varepsilon>0}\left\{\varepsilon T_\varepsilon + T'_\varepsilon + O\left(\frac{K \log|A_T| \log(KT/\delta) + K\log^2|A_T|}{\varepsilon^2}\right)\right\} \\
&= \inf_{\varepsilon>0}\left\{\varepsilon T_\varepsilon + T'_\varepsilon + O\left(\frac{K d^2 \log^2(KT/\delta)}{\varepsilon^2}\right)\right\}\;, \\
N_T &\le K \inf_{\varepsilon>0}\left\{T_\varepsilon + T''_\varepsilon + O\left(\frac{K \log|A_T| \log(KT/\delta) + K\log^2|A_T|}{\varepsilon^2}\right)\right\} \\
&= K \inf_{\varepsilon>0}\left\{T_\varepsilon + T''_\varepsilon + O\left(\frac{K d^2 \log^2(KT/\delta)}{\varepsilon^2}\right)\right\}\;.
\end{aligned}$$

The bounds above resemble the bounds stated in Theorem 12 for the first version of the algorithm; all of these bounds contain two kinds of terms: hardness terms (T_ε, T'_ε, and T''_ε) and large deviation terms ($d \log T$-like factors). The regret bound for the second version of the algorithm is strictly inferior to the regret bound for the first version, as an additional factor of K multiplies the large deviation term. However, the bounds on the number of queries of the two algorithms are not directly comparable. On one hand, if a typical example only has a few confident teachers, we expect T''_ε to be much smaller than T'_ε, which could make the bound on N_T in Theorem 17 much smaller than its counterpart in Theorem 12. On the other hand, the bound in Theorem 17 has an additional factor of K multiplying its large deviation term.

As in the proofs of Theorem 2 and Theorem 12, to analyze the regret and number of queries made by the algorithm, we start by decomposing these terms. To decompose the regret, we note that Lemma 13 applies as before, and we have that for any $\varepsilon > 0$,

$$R_T \;\le\; \varepsilon T_\varepsilon + T'_\varepsilon + U_T + Q_{T,\varepsilon},$$

where T_ε is as defined in Equation (9), T'_ε is as defined in Equation (10), and U_T and $Q_{T,\varepsilon}$ are defined in Equation (11). To decompose the total number of queries, we require a new lemma.

Lemma 18 *If $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \le \theta^2_{j,t}$ holds for all $j \in [K]$ and $t \in [T]$, then for any $\varepsilon > 0$, it holds that*

$$\begin{aligned}
N_T &\le K\left(T_\varepsilon + T''_\varepsilon + O\left(\frac{\sum_{j=1}^K \left(\log|A_{j,T}| \log(KT/\delta) + \log^2|A_{j,T}|\right)}{\varepsilon^2}\right)\right) \\
&\le K\left(T_\varepsilon + T''_\varepsilon + O\left(\frac{Kd^2 \log^2(KT/\delta)}{\varepsilon^2}\right)\right)\;.
\end{aligned}$$

Once again, proofs are given in Appendix C. We are left with the task of bounding U_T and $Q_{T,\varepsilon}$.

Lemma 19 *If* $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_{j,t}^2$ *holds for all* $j \in [K]$ *and* $t \in [T]$*, then*

$$Q_{T,\varepsilon} = O\left(\frac{\sum_{j=1}^{K}\left(\log|A_{j,T}|\,\log(KT/\delta) + \log^2|A_{j,T}|\right)}{\varepsilon^2}\right) = O\left(\frac{K\,d^2\,\log^2(KT/\delta)}{\varepsilon^2}\right) \quad .$$

Lemma 20 *If* $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_{j,t}^2$ *for all* $j \in [K]$ *and* $t \in [T]$*, then* $U_T = 0$*.*

Proofs of these lemmas are also given in Appendix C. As before, these lemmas hold under the condition that $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_{j,t}^2$ for all $t \in [T]$ and $j \in [K]$. Again as done previously, a straightforward union bound over Lemma 7 in Section 2 applied to each of the K teachers verifies that this condition holds with high probability which in turn concludes the proof of Theorem 17.

Remark 21 *It should be clear that a low noise analysis, akin to the one presented in Sections 2.4 and 2.5 can be attempted, once low noise conditions in the vein of Tsybakov (2004) are formulated which take into account both the conflicting region defining* T_ε*, and the unstable regions defining* T'_ε*, and* T''_ε*. Rather than presenting explicit theoretical results of this sort here, we do prefer quantifying the label saving capability implied by teacher aggregation by the nontrivial experimental results contained in the next section.*

4. Experiments in the Multiple Teacher Setting

We report on the results of an empirical study carried out on a medium-size real-world data set. The goal of our experiments is to validate the theory and to quantify the effectiveness of our multiple teacher query selection mechanism in different multiple-teacher scenarios. Due to our difficulty in finding genuine multiple-teacher data sets of a significant size, we resorted to *simulating* the teachers through learning. This also allowed us to obtain a much more controlled experimental setup.

4.1 Data Set and Tasks

Our data are taken from a subset of the learning-to-rank data set MSLR-WEB10K.[7] This data set is a collection of (anonymized) query-url pairs collected from a commercial search engine (Microsoft Bing). Each query-url pair is represented by a feature vector and a human-generated relevance label between 0 (irrelevant) to 4 (perfectly relevant). Each feature vector is made up of 136 real or integer valued features.[8] MSLR-WEB10K is partitioned into five subsets, named S1 through S5: we only used S1 in our experiments. The S1 subset contains 241988 query-url pairs, with 2000 distinct queries, and about 121 urls per query (with a maximum of 809 urls and a minimum of 1 url per query). As a preprocessing step, we randomly shuffled the examples within each query and normalized the feature vectors to unit length.

We generated a binary classification data set by assigning the binary label "-1" to all query-url pairs with a relevance label of 0 and the binary label "+1" to all remaining pairs. This gave rise to a data set with balanced classes (roughly, 48% positive and 52% negative). We then simulated

7. Available at `http://research.microsoft.com/en-us/projects/mslr`.
8. After a quick scrutiny of the semantics of the features, we decided to drop features 126 through 131. Hence, we ended up with 130 usable features.

four different multiple-teacher scenarios, distinguished by the number of teachers involved ("few" or "many") and the amount of overlap between expertise regions ("nonoverlapping teachers" vs. "overlapping teachers").

This binary classification data set simply provides a binary label per example, and does not specify the identity of the teacher that provided that label. We simulated multiple teachers as follows: we grouped queries together in various ways (see below) and trained a linear classifier using half of the urls associated with each query in the group. The training was done using a single random-order pass of a full-information second-order Perceptron algorithm (Cesa-Bianchi et al., 2005a). The result is a linear classifier per query-group: we view each of these linear classifiers as a teacher. The specific subset of training queries in each group determines the expertise region of the respective teacher. The 119507 query-url pairs that were not used to simulate the teachers were later used to test our algorithm.

We defined the query groups in four different ways, to simulate four different multiple-teacher scenarios.

- **Few nonoverlapping teachers.** We generated 5 teachers by partitioning the 2000 queries into 5 sets (the first teacher is defined by (half of) the first 400 queries, the second teacher by (half of) the second 400 queries, and so on). Hence each teacher has acquired expertise in the subset of 400 queries seen during training.

- **Few overlapping teachers.** We generated 5 teachers by defining 5 overlapping sets of queries. Specifically, the first 500 queries are common to all teachers, and the remaining 1500 queries are partitioned equally among the teachers. Hence, each teacher is trained on examples from 800 queries.

- **Many nonoverlapping teachers.** We generated 100 teachers by partitioning the queries into 100 disjoint sets, each containing 20 queries. The resulting teachers turned out to be quite unreliable; some of them gave labels that were not far from random guessing at test time.

- **Many overlapping teachers.** We generated 100 teachers with partially overlapping expertise. All teachers share the first 100 queries, and the remaining 1900 queries are partitioned equally. Hence, each teacher is trained on examples from 100+19 = 119 queries.

Due to the variance introduced by the randomized training/test splits and the random order in which training examples were presented to the second-order Perceptron, we repeated the above process 10 times per scenario and averaged the results.

The reader should observe that the way we generated teachers makes our results comparable even across scenarios. In fact, despite the actual training/test split differs over scenarios, in all four scenarios and for all 2000 queries, half the urls (and associated labels) are used for training and half are used for test. So, in a sense, the set of teachers we generated in all scenarios collectively encode the same amount of information. That is, the data used for training the teachers are of the same size and query mixture across scenarios.

4.2 Algorithm and Baselines

On any given scenario, a teacher is then just a linear-threshold function. We generated teachers' opinions on the test set just by evaluating such functions on the test set instances. Table 1 gives

SCENARIO	BEST	WORST	AVG	STDDEV
FEW NONOVERLAPPING TEACHERS	19.9%	31.7%	24.9%	4.6%
FEW OVERLAPPING TEACHERS	20.5%	29.6%	24.5%	3.6%
MANY NONOVERLAPPING TEACHERS	16.3%	54.8%	25.5%	7.4%
MANY OVERLAPPING TEACHERS	17.0%	42.3%	24.5%	5.1%

Table 1: Performance (test set mistake rate) of the generated teachers in the four simulated scenarios. Results are averaged over 10 repetitions. "best", "worst", and "avg" are the (average) mistake rate of the best, worst and average performing teacher, respectively. "stddev" is the standard deviation of the mistake rates, and gives an idea of the difference in performance *across teachers* (not across repetitions).

relevant statistics about the teachers' performance on the test set (notice that such figures can only be computed after knowing the true labels on the test set—this information was never made available to the multiple teacher algorithm). As expected, best and worst teachers are farther apart in the nonoverlapping scenarios (correspondingly, "stddev" figures are larger), with a larger variability in the many teacher settings. Moreover, throughout the 10 repetitions, it often happened that among the many poorly trained classifiers (as are those produced within the "many nonoverlapping" setting), a few of them turned out to be significantly accurate on the test set. Likewise, some of them happened to be even worse than random guessing.

After simulating the teachers, we implemented a simplified version of our second-version multiple teacher algorithm (Algorithm 3), where the thresholds $\theta^2_{j,t}$ are simplified to

$$\theta^2_{j,t} = \alpha \mathbf{x}_t^\top A^{-1}_{j,t-1} \mathbf{x}_t \log(1+t), \tag{12}$$

and $\alpha > 0$ is a tunable parameter (independent of j and t). Hence our algorithm now has two parameters: $\tau \in [0,1]$ and $\alpha > 0$. The reason for this simplified $\theta_{j,t}$ is that the actual expression for $\theta_{j,t}$, as it appears in Algorithm 3, is the one suggested by the theory after significant mathematical overapproximations (large deviations, Hölder's inequality, etc.). This suggests that the exact expression for $\theta_{j,t}$ given in the pseudocode may be too conservative to work well in practice. In any event, observe that the factor $\alpha \log(1+t)$ in (12) is a good proxy for the factor $1+4\sum_{i=1}^{t-1} Z_i r_i + 36\log(Kt/\delta)$ in the algorithm's pseudocode, once we let α range over the positive reals.

The following three baselines were used in our comparative study.

- BEST TEACHER in hindsight on the test set. This is the predictor that would be learned on-the-fly by a standard expert algorithm (e.g., Weighted Majority—see Littlestone and Warmuth, 1994; Cesa-Bianchi and Lugosi, 2006), where teachers are experts, and the algorithm has at its disposal both the true labels of the test set and the prediction of all teachers. Recall that the true labels of the test set are not available to our algorithm. Because this algorithm is expected to make at least as many mistakes as the best expert, the "best" column in Table 1 delivers optimistic approximations to the actual performance of this algorithm in the four scenarios. The associated number of queries made to the teachers is the largest possible, that is, the size of the test set (119507) times the number of teachers (119507×5 in the "few teacher" scenarios, and 119507×100 in the "many teacher" scenarios).

SCENARIO	BEST TEACH.	FLAT MAJORITY	FULL-INFO ALG. 3
FEW NONOVERLAPPING TEACHERS	19.9%	16.7%	15.8%
FEW OVERLAPPING TEACHERS	20.5%	17.9%	15.8%
MANY NONOVERLAPPING TEACHERS	16.3%	15.6%	15.6%
MANY OVERLAPPING TEACHERS	17.0%	15.7%	15.7%

Table 2: Performance (test set mistake rate) of the three tested baselines in the four simulated scenarios. Results are averaged over 10 repetitions. The "Best Teacher" figures are taken from Table 1.

- FLAT MAJORITY of teachers. This algorithm asks all teachers and predicts with their flat majority.[9] Like the BEST TEACHER baseline, this algorithm queries all of the teachers all the time. Unlike BEST TEACHER, this algorithm does not receive any feedback on the true labels of the test set.

- FULL-INFORMATION version of our second-version Algorithm (Algorithm (3)). This is our algorithm with $\theta^2_{j,t}$ fixed to the value ∞ for all j and t. Since $\theta^2_{j,t} = \infty$ implies $Z_t = 1$ and $\hat{C}_t = [K]$ for all t (thereby making τ immaterial), this algorithm predicts by aggregating all teachers via a margin-based majority and, as before, querying all labels from all teachers. Again, no ground-truth feedback is given. Hence, this baseline is just a weighted version of FLAT MAJORITY, where the weights are given by the estimated margins $\hat{\Delta}_{j,t}$ computed by Algorithm 3 operating in a "full information" mode.

4.3 Results and Comments

We measured the error rate on the test set and the average number of requested labels per example. Figure 1 shows test error rate as a function of the per-teacher query rate (i.e., the average fraction of times we query the teachers). The figure displays the test error rate of our algorithm compared to the three baselines mentioned above, in each of the four scenarios, with $\tau = 0.3$ and different values of α in $[0.01, 10]$. Very similar plots are obtained for other values of τ.[10] Increasing α causes a steady increase in the (average) per-teacher query rate, but surprisingly enough, has a negligible effect on test error rate across most of its range (hence the flattish plots in Figure 1). In particular, a query rate of about 1% is already sufficient to get very close to the smallest test error rate achieved by the algorithm. As for comparison to the baselines, the following comments can be made.

- Our algorithm significantly outperforms all baselines in the "few teacher" scenarios, but is about the same as the two majority baselines in the "many teacher" scenarios. Notice, however, that this comparison is unfairly penalizing our algorithm in that the baselines do achieve their results by asking all of the teachers all of the time. Moreover, it is worth stressing that

9. Alternatively, this algorithm picks a teacher uniformly at random and goes with its label. In our experiments, we did not test this randomized version due to the high variance of the results, especially in the "many teacher" scenarios—see the last two rows in Table 1.

10. For instance, in the "few nonoverlapping" setting, when $\tau = 0.0$ and α ranges over [0.01,10] the test error rate of our algorithm ranges between 15.5% and 15.7%; when $\tau = 0.7$ the test error ranges between 15.6% and 15.7%. In the "many overlapping" setting, when $\tau = 0.0$ and $\alpha \in [0.01,10]$ we obtain a test error between 15.6% and 15.9%; When $\tau = 0.7$, the range is between 15.4% and 15.5%.

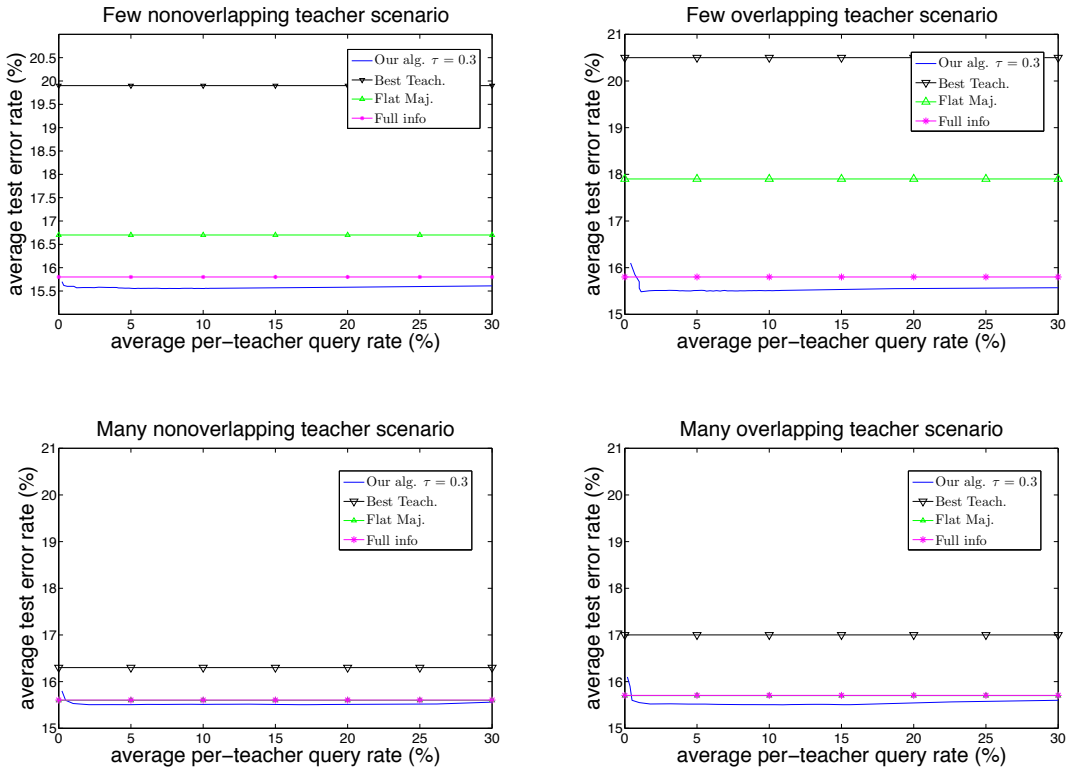


Figure 1: Average per-teacher query rate vs. test error rates in the four scenarios. Results are averaged over 10 repetitions. The query rate of the multiple selective sampler - second version ("Our alg.") is obtained by setting $\tau = 0.3$, and letting α in (12) vary across the range $[0.01, 10]$. In any given scenario, the per-teacher query rate of our algorithm is the average fraction of labels requested to the teachers out of the total number of labels available in that scenario. For instance, an average per-teacher query rate of 10% achieved in a few teacher scenario means that, averaged over the 10 repetitions, the total no. of queries made to the five teachers was $119507 \times 5 \times 10\% = 59753.5$. Hence, each of the 5 teachers received on average 11950.7 queries. The test error rates of the three baselines are taken from Table 2, and are plotted (as horizontal lines) just for reference.

though our plots display average rates over repetitions (i.e., over train/test splits), the above comparative behavior did consistently occur in *every* single repetition.

- We found somewhat surprising that FULL-INFO does not improve on FLAT MAJORITY. Moreover, since FULL-INFO can be obtained by our algorithm, just by setting $\alpha = \infty$ in (12), we see that more teacher labels can even be detrimental. This phenomenon is statistically significant only in the "few teacher" scenarios.

- The more teachers we have at our disposal, the more beneficial is the process of averaging over them. Notably, in our data set, many unreliable (but nonoverlapped) teachers queried all the times and aggregated by a flat average (aka FLAT MAJORITY) is about as good in terms

of accuracy as running more sophisticated weighted averages. Still, our experiments show that there is no need to query all of the teachers in order to achieve this accuracy.

- Aggregating opinions of teachers with a good amount of overlapping expertise (as in the two "overlapping teacher" settings), might be detrimental, as evinced by comparing the first row in Table 2 to the second one, and the third to the fourth one. Similar conclusions are suggested by the behavior of our algorithm as presented in Figure 1.

Finally, we make a few comments on the role of the parameter τ in our algorithm. As mentioned above, we observed that the value of τ does not have a significant influence on the algorithm's test error rate or label query rate. In a sense, this is a lucky circumstance, since we initially expected the tuning of τ to be a nontrivial task.[11] The value of τ does however play an important role in the "degree of aggregation" of teachers: When $Z_t = 1$, setting τ close to 0 makes the algorithm query only the (estimated) most confident teacher at time t, whereas setting τ close to 1 causes the algorithm to query all teachers. For instance, in the "many nonoverlapping teachers" scenario, if $\tau = 0.3$ (as in the plot in Figure 1 (d)), and $\alpha = 0.1$, the most queried teacher receives 5440 queries (out of 119507), and the least queried teacher receives 3319. In the same scenario with $\tau = 0.0$ and the same value of α, the most queried teacher receives 10849 queries while the least queried teacher gets only 1050. Hence, our algorithm exhibits a desirable fine-grained selection capability of the subsets of teachers to query, thereby making it significantly different from the all-or-none strategy followed by the first version of our algorithm (Algorithm 2), which we do not expect to work as well in practice.

5. Conclusions and Open Questions

We introduced a new algorithm in the online selective sampling framework, where instances are chosen by an adaptive adversary and labels are sampled from a linear stochastic model. We gave sharp bounds on the regret and on the number of queries made by this algorithm, improving over previous algorithms and closing some important open questions on this topic. The same machinery can also be used to build efficient active learning algorithms working under standard statistical assumptions. We then lifted the above to the more involved setting where multiple unreliable teachers are available. We presented two algorithms and corresponding analyses. We concluded with a preliminary empirical study that demonstrates how the second version of our algorithm outperforms various intuitive baselines, both in terms of accuracy and total number of queries.

We leave some open problems for future research: The bound on N_T in Theorem 2 is tight w.r.t. ε (see the lower bound by Cesa-Bianchi et al., 2009), but need not be tight w.r.t. d. This might be due to the way we constructed our martingale argument to prove Lemma 7. Resolving this issue remains an open problem. Second, it would be interesting to generalize our results to other stochastic label models, such as logistic models, and to understand how closely each model matches the true behavior of human teachers. Third, the bounds in the multiple teacher setting (Theorems 12 and 17) are likely to be suboptimal, and might perhaps be improved by exploiting the interaction structure among teachers. Fourth, it would be interesting to extend our work to a setting where different teachers charge different rates. For example, one could imagine a setting where the cost of each label depends on each teacher's confidence in his own answer. This setting is closer to the

11. Consider that the absence of ground-truth feedback makes standard cross-validation techniques somewhat problematic.

proactive learning setting (Donmez and Carbonell, 2008; Yang and Carbonell, 2009a,b). These and other open problems provide many opportunities for interesting future research on this topic.

Acknowledgments

We thank the Action Editor for his timely handling of this paper. We also thank the anonymous reviewers for their helpful comments. This research was done while the second and the third authors were visiting Microsoft Research at Redmond. The second author acknowledges the PASCAL2 Network of Excellence under EC grant 216886 for supporting travel expenses to the conference.

Appendix A.

This appendix contains the large deviation inequalities we use throughout the paper.

Lemma 22 *(Kakade and Tewari, 2008)*
Suppose $X_1, X_2, ..., X_T$ is a martingale difference sequence with $|X_t| \leq b$. Let $\mathrm{Var}_t(X_t) = \mathrm{Var}(X_t|X_1, ..., X_{t-1})$, and $V = \sum_{t=1}^{T} \mathrm{Var}_t(X_t)$. Then for any $\delta < 1/e$ and $T \geq 3$, we have

$$P\left(\sum_{t=1}^{T} X_t > \max\left\{\sqrt{4V \log\frac{4\log T}{\delta}}, 3b\log\frac{4\log T}{\delta}\right\}\right) \leq \delta.$$

Lemma 23 *With the notation introduced in Section 2, define*

$$\mu_t = \sum_{i=1}^{t} Z_i(\Delta_i - \hat{\Delta}'_i)^2, \qquad \Sigma_t = \sum_{i=1}^{t} Z_i\left((y_i - \hat{\Delta}'_i)^2 - (y_i - \Delta_i)^2\right).$$

Assume that Selective Sampler in Section 2 is run with confidence parameter $\delta \in (0,1]$, and let $t \geq 3$. Then

(i) with probability at least $1 - \delta/t^2$ we have $\mu_t \leq 2\Sigma_t + 144\log(t/\delta)$;

(ii) with probability at least $1 - \delta/t^2$ we have $-\frac{1}{2}\Sigma_t \leq 36\log(t/\delta)$.

Proof Set $M_i = Z_i(\Delta_i - y_i)(\Delta_i - \hat{\Delta}'_i)$, and observe that M_i can be rewritten as

$$M_i = \frac{1}{2} Z_i\left((\Delta_i - \hat{\Delta}'_i)^2 - \left((y_i - \hat{\Delta}'_i)^2 - (y_i - \Delta_i)^2\right)\right),$$

which implies $\frac{1}{2}(\mu_t - \Sigma_t) = \sum_{t=1}^{t} M_i$. Now, $M_1, ..., M_t$ is a martingale difference sequence w.r.t. history and current $\mathbf{x}_i$. This is because $\mathbf{E}_i[M_i] = Z_i(\Delta_i - \mathbf{E}_i[y_i])(\Delta_i - \hat{\Delta}'_i) = 0$. Since $|\Delta_t|, |\hat{\Delta}_t| \leq 1$, we also have that $|M_i| \leq 4$. Let $\mathrm{Var}_i(\cdot)$ denote the conditional variance $\mathrm{Var}(\cdot \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_i, y_1, \ldots, y_{i-1})$. Observing that

$$\mathrm{Var}_i(M_i) = Z_i(\Delta_i - \hat{\Delta}'_i)^2 \mathrm{Var}_i\left((\Delta_i - y_i)^2\right) \leq \frac{4}{3} Z_i(\Delta_i - \hat{\Delta}'_i)^2$$

holds, an application of Lemma 22 yields

$$\frac{1}{2}(\mu_t - \Sigma_t) \leq \max\left\{\sqrt{6\,\mu_t\,\log\left(\frac{4t^2\log t}{\delta}\right)}, 12\log\left(\frac{4t^2\log t}{\delta}\right)\right\}. \tag{13}$$

We now use the inequality $\sqrt{ab} \leq \frac{a+b}{2}$ to (13) with $a = \mu_t/2$ and $b = 12\log\left(\frac{4t^2\log t}{\delta}\right)$. This implies

$$\frac{1}{2}\left(\mu_t - \Sigma_t\right) \leq \mu_t/4 + 12\log\left(\frac{4t^2\log t}{\delta}\right)$$

which in turn implies (i). To prove (ii), we again apply $\sqrt{ab} \leq \frac{a+b}{2}$ to (13), this time with $a = \mu_t$ and $b = 6\log\left(\frac{4t^2\log t}{\delta}\right)$. ■

Appendix B.

Most of the steps in the proofs of these lemmas appear in the papers by Azoury and Warmuth (2001) and Cesa-Bianchi et al. (2005a). The proofs are provided here for completeness.

Lemma 24 *With the notation introduced in Section 2, we have that for each $t = 1, 2, \ldots$ the following inequalities hold :*

(i) $\mathbf{x}^\top A_{t-1}\mathbf{x}_t \leq 2r_t$;

(ii) $Z_t r_t \leq \log\frac{|A_t|}{|A_{t-1}|}$;

(iii) $\sum_{i=1}^t Z_i r_i \leq \log|A_t| \leq d\log(1+N_t) = O(d\log t)$.

Proof To prove (i), note that on the rounds we do not query, $A_t = A_{t-1}$ and so $r_t = \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$. On the rounds we do query, $A_t = A_{t-1} + \mathbf{x}_t\mathbf{x}_t^\top$, and so by the matrix inversion formula

$$A_t^{-1} = A_{t-1}^{-1} - \frac{A_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top A_{t-1}^{-1}}{1+\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}$$

we see that

$$r_t = \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t - \frac{(\mathbf{x}_t^\top A_{t-1}\mathbf{x}_t)^2}{1+\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t}\ .$$

This automatically gives us that $r_t \leq \mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$. Further since $\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t \leq 1$, we can conclude that $r_t \geq \frac{1}{2}\mathbf{x}_t^\top A_{t-1}^{-1}\mathbf{x}_t$. Hence we conclude that for any t, $r_t \leq \mathbf{x}^\top A_{t-1}\mathbf{x}_t \leq 2r_t$.

Now to prove (ii), note that since whenever we query, $A_t = A_{t-1} + \mathbf{x}_t\mathbf{x}_t^\top$, using the identity, $\mathbf{x}_t^\top(A_{t-1} + \mathbf{x}_t\mathbf{x}_t^\top)^{-1}\mathbf{x}_t = 1 - \frac{|A_t|}{|A_{t-1}|}$ and the fact that $1 - x \leq \log(x)$, we see that

$$r_t \leq \log\frac{|A_t|}{|A_{t-1}|}.$$

To get (iii), we sum up and resolve the telescoping sum as

$$\sum_{i=1}^t Z_i r_i \leq \sum_{i=1}^t Z_i \log\frac{|A_i|}{|A_{i-1}|} = \log|A_t| \leq d\ \log(1+N_t) = O(d\ \log t)\ .$$

■

Lemma 25 *With the notation introduced in Section 2, the following holds for any* $\mathbf{u} : ||\mathbf{u}|| \leq 1$*:*

(i) If t *is such that* $Z_t = 1$ *we have*

$$\frac{1}{2}\left((y_t - \mathbf{w}'_{t-1}{}^{\top}\mathbf{x}_t)^2 - (y_t - \mathbf{u}^{\top}\mathbf{x}_t)^2\right) \;=\; d_{t-1}(\mathbf{u}, \mathbf{w}'_{t-1}) - d_t(\mathbf{u}, \mathbf{w}_t) + d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t)\;;$$

(ii) If t *is such that* $Z_t = 1$ *we have* $d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t) \leq 2\,r_t$;

(iii) If t *is such that* $Z_t = 1$ *we have* $d_t(\mathbf{u}, \mathbf{w}'_t) \leq d_t(\mathbf{u}, \mathbf{w}_t)$;

(iv) For any $t = 1, 2, \ldots$*, we have*

$$\frac{Z_t}{2}\left((y_t - \mathbf{w}'_{t-1}{}^{\top}\mathbf{x}_t)^2 - (y_t - \mathbf{u}^{\top}\mathbf{x}_t)^2\right) \leq Z_t\,\left(d_{t-1}(\mathbf{u}, \mathbf{w}'_{t-1}) - d_t(\mathbf{u}, \mathbf{w}'_t)\right) + 2\log\frac{|A_t|}{|A_{t-1}|}.$$

Proof To prove (i), define $\alpha_t := d_{t-1}(\mathbf{u}, \mathbf{w}'_{t-1}) - d_t(\mathbf{u}, \mathbf{w}_t) + d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t)$. Using the definition of d_t, we have that

$$\alpha_t \;=\; \frac{1}{2}\mathbf{u}^{\top}(A_{t-1} - A_t)\mathbf{u} + \mathbf{u}^{\top}(A_t\mathbf{w}_t - A_{t-1}\mathbf{w}'_{t-1}) + \frac{1}{2}\mathbf{w}'_{t-1}(A_{t-1} + A_t)\mathbf{w}'_{t-1} - \mathbf{w}'_{t-1}A_t\mathbf{w}_t \;\;.$$

Using the recursive definition $A_t = A_{t-1} + \mathbf{x}_t\mathbf{x}_t^{\top}$ and rearranging terms in the right-hand side above, we get

$$\begin{aligned}\alpha_t \;&=\; \frac{1}{2}\left((\mathbf{w}'_{t-1})^{\top}\mathbf{x}_t\mathbf{x}_t^{\top}\mathbf{w}'_{t-1} - \mathbf{u}^{\top}\mathbf{x}_t\mathbf{x}_t^{\top}\mathbf{u}\right) + (\mathbf{u}^{\top} - \mathbf{w}'_{t-1})(A_t\mathbf{w}_t - A_{t-1}\mathbf{w}'_{t-1}) \\ &=\; \frac{1}{2}\left((\mathbf{w}'_{t-1}{}^{\top}\mathbf{x}_t)^2 - (\mathbf{u}^{\top}\mathbf{x}_t)^2\right) + (\mathbf{u}^{\top} - \mathbf{w}'_{t-1})(A_t\mathbf{w}_t - A_{t-1}\mathbf{w}'_{t-1}) \;\;.\end{aligned}$$

By definition, $A_t\mathbf{w}_t = A_{t-1}\mathbf{w}'_{t-1} + y_t\mathbf{x}_t$. Plugging this equality into the equation above gives

$$\begin{aligned}\alpha_t \;&=\; \frac{1}{2}\left((\mathbf{w}'_{t-1}{}^{\top}\mathbf{x}_t)^2 - (\mathbf{u}^{\top}\mathbf{x}_t)^2\right) + y_t(\mathbf{u}^{\top} - \mathbf{w}'_{t-1})\mathbf{x}_t \\ &=\; \frac{1}{2}\left((y_t - \mathbf{w}'_{t-1}{}^{\top}\mathbf{x}_t)^2 - (y_t - \mathbf{u}^{\top}\mathbf{x}_t)^2\right)\;,\end{aligned}$$

thereby proving (i).

To prove (ii), we rewrite $d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t)$ as

$$\begin{aligned}d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t) \;&=\; \frac{1}{2}(\mathbf{w}'_{t-1} - \mathbf{w}_t)^{\top}A_t(\mathbf{w}'_{t-1} - \mathbf{w}_t) \\ &=\; \frac{1}{2}(A_t\mathbf{w}'_{t-1} - A_t\mathbf{w}_t)^{\top}A_t^{-1}(A_t\mathbf{w}'_{t-1} - A_t\mathbf{w}_t) \;\;.\end{aligned}$$

Using $A_t\mathbf{w}_t = A_{t-1}\mathbf{w}'_{t-1} + y_t\mathbf{x}_t$, the above becomes

$$d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t) \;=\; \frac{1}{2}\left((A_t - A_{t-1})\mathbf{w}'_{t-1} - y_t\mathbf{x}_t\right)^{\top}A_t^{-1}\left((A_t - A_{t-1})\mathbf{w}'_{t-1} - y_t\mathbf{x}_t\right) \;\;.$$

Using $A_t = A_{t-1} + \mathbf{x}_t \mathbf{x}_t^\top$, we have

$$\begin{aligned} d_t(\mathbf{w}'_{t-1}, \mathbf{w}_t) &= \frac{1}{2}\left(\mathbf{x}_t \mathbf{x}_t^\top \mathbf{w}'_{t-1} - y_t \mathbf{x}_t\right) A_t^{-1} \left(\mathbf{x}_t \mathbf{x}_t^\top \mathbf{w}'_{t-1} - y_t \mathbf{x}_t\right) \\ &= \frac{(\mathbf{w}'_{t-1}{}^\top \mathbf{x}_t - y_t)^2}{2} \mathbf{x}_t^\top A_t^{-1} \mathbf{x}_t \\ &= \frac{(\mathbf{w}'_{t-1}{}^\top \mathbf{x}_t - y_t)^2}{2} r_t \\ &= \frac{(\hat{\Delta}'_t - y_t)^2}{2} r_t \\ &\le 2 r_t \ , \end{aligned}$$

where the last step uses $|\hat{\Delta}'_t| \le 1$

To prove (iii) we observe that, $\mathbf{w}'_t$, as defined in Algorithm 1, is the projection of $\mathbf{w}_t$ onto the convex set $C_t = \{\mathbf{w} : |\mathbf{w}^\top \mathbf{x}_t| \le 1\}$ w.r.t. Bregman divergence d_t. By the theorem of generalized projections we have that

$$0 \le d_t(\mathbf{w}'_t, \mathbf{w}_t) \le d_t(\mathbf{u}, \mathbf{w}_t) - d_t(\mathbf{u}, \mathbf{w}'_t) \ .$$

holds for any $\mathbf{u} \in C_t$. Since C_t includes the unit ball $\{\mathbf{u} : ||\mathbf{u}|| \le 1\}$ the claim follows.

Finally, to prove (iv), observe that when t is such that $Z_t = 0$ then both sides of the inequality are 0 (since $A_t = A_{t-1}$). On the other hand, when $Z_t = 1$ we just combine (i), (ii), (iii), and Lemma 24 (ii) to give the required inequality. ■

Appendix C.

Proof sketch of Theorem 9. We rely on Theorem 2, where the role of T_ε is neatly handled by the low-noise assumption combined with a standard Chernoff bound. In particular, since $\mathbb{E}[T_\varepsilon] \le cT\varepsilon^\alpha$, we can easily conclude that for any $\delta > 0$, with probability at least $1-\delta$ over sample $\mathbf{x}_1, \ldots, \mathbf{x}_T$ we have $T_\varepsilon \le \frac{3c}{2} T\varepsilon^\alpha + O(\log(1/\delta))$. We optimize over ε the bounds on R_T and N_T contained in Theorem 2. We obtain that, with the same probability,

$$\begin{aligned} R_T &= O\left((d \log(T/\delta))^{\frac{\alpha+1}{\alpha+2}} \, T^{\frac{1}{\alpha+2}} + \log(1/\delta) \right) \ , \\ N_T &= O\left((d^2 \log^2(T/\delta))^{\frac{\alpha}{\alpha+2}} \, T^{\frac{2}{\alpha+2}} + \log(1/\delta) \right) \ . \end{aligned} \tag{14}$$

Now define

$$K_t = \left(P'_t(y_t \, \hat{\Delta}_t < 0) - P'_t(y_t \, \Delta_t < 0)\right) - \left(P_t(y_t \, \hat{\Delta}_t < 0) - P_t(y_t \, \Delta_t < 0)\right) \ ,$$

and note that $K_1, ..., K_T$ forms a martingale difference sequence. Let $\mathbf{E'}_t[\cdot]$ denote the conditional expectation $\mathbf{E}[\cdot \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, y_1, \ldots, y_{t-1}]$ and $\mathrm{Var'}_t(\cdot)$ be the conditional variance

$\mathrm{Var}(\cdot \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, y_1, \ldots, y_{t-1})$.[12] We have

$$\begin{aligned}
\mathrm{Var}'_t[K_t] &= \mathbf{E'}_t\left[K_t^2\right] \\
&\le 2\left(\left(P'_t(y_t\ \hat{\Delta}_t < 0) - P'_t(y_t\ \Delta_t < 0)\right)\right)^2 \\
&\quad + 2\,\mathbf{E'}_t\left[\left(P_t(y_t\ \hat{\Delta}_t < 0) - P_t(y_t\ \Delta_t < 0)\right)^2\right] \\
&\quad \text{(using } (a-b)^2 \le 2a^2 + 2b^2\text{)} \\
&\le 2\left(P'_t(y_t\ \hat{\Delta}_t < 0) - P'_t(y_t\ \Delta_t < 0)\right) + 2\,\mathbf{E'}_t\left[P_t(y_t\ \hat{\Delta}_t < 0) - P_t(y_t\ \Delta_t < 0)\right] \\
&\quad \text{(using } P'_t(y_t\ \hat{\Delta}_t < 0) \ge P'_t(y_t\ \Delta_t < 0) \text{ and } P_t(y_t\ \hat{\Delta}_t < 0) \ge P_t(y_t\ \Delta_t < 0)\text{)} \\
&= 4\left(P'_t(y_t\ \hat{\Delta}_t < 0) - P'_t(y_t\ \Delta_t < 0)\right)\ .
\end{aligned}$$

Following Lemma 22 and overapproximating we have that, with probability at least $1-\delta$,

$$\begin{aligned}
\sum_{t=1}^{T}\left(P'_t(y_t\ \hat{\Delta}_t < 0) - P'_t(y_t\ \Delta_t < 0)\right) &\le 2\,R_T + O\left(\log\left(\frac{\log T}{\delta}\right)\right) \\
&= O\left((d\log(T/\delta))^{\frac{\alpha+1}{\alpha+2}}\ T^{\frac{1}{\alpha+2}} + \log\left(\frac{\log T}{\delta}\right)\right)\ ,
\end{aligned}$$

the last equality deriving from (14). Dividing by T concludes the proof.

Proof of Lemma 13. We upper bound each of the summands in Equation (8) individually. We begin as in Equation (3) in the proof of Lemma 3. This gives us

$$P_t(y\hat{\Delta}_t < 0) - P_t(y\Delta_t < 0)\ \le\ \varepsilon\,\mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0, |\Delta_t| \le \varepsilon\right\} + \mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0, |\Delta_t| > \varepsilon\right\}|\Delta_t|\ . \tag{15}$$

The first term on the right-hand side above is simply upper bounded by $\varepsilon\,\mathbb{1}\left\{|\Delta_t| \le \varepsilon\right\}$. To upper bound the second term, we recall that $|\Delta_t| \le 1$ and bound $\mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0, |\Delta_t| > \varepsilon\right\}$ by

$$\begin{aligned}
&\mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0, |\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} < 0\ \vee\ |\Delta_{S\cup H_{\varepsilon,t}}| \le \varepsilon\right\} \\
&\qquad + \mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0, |\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} \ge 0,\ |\Delta_{S\cup H_{\varepsilon,t}}| > \varepsilon\right\} \\
&\le\ \mathbb{1}\left\{|\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} < 0\ \vee\ |\Delta_{S\cup H_{\varepsilon,t}}| \le \varepsilon\right\} \\
&\qquad + \mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0\right\}\mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} \ge 0,\ |\Delta_{S\cup H_{\varepsilon,t}}| > \varepsilon\right\} \\
&\le\ \mathbb{1}\left\{|\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} < 0\ \vee\ |\Delta_{S\cup H_{\varepsilon,t}}| \le \varepsilon\right\} \\
&\qquad + \bar{Z}_t\,\mathbb{1}\left\{\Delta_t\hat{\Delta}_t < 0\right\} + Z_t\,\mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} \ge 0,\ |\Delta_{S\cup H_{\varepsilon,t}}| > \varepsilon\right\}\ .
\end{aligned} \tag{16}$$

We plug Equation (16) into the right-hand side of Equation (15) to obtain the desired upper-bound on $P_t(y\hat{\Delta}_t < 0) - P_t(y\Delta_t < 0)$. Summing over t completes the proof.

Proof of Lemma 14. It is straightforward to verify that

$$\begin{aligned}
Z_t\ &=\ Z_t\,\mathbb{1}\left\{|\Delta_t| \le \varepsilon\right\} + Z_t\,\mathbb{1}\left\{|\Delta_t| > \varepsilon\right\} \\
&\le\ Z_t\,\mathbb{1}\left\{|\Delta_t| \le \varepsilon\right\}\ +\ Z_t\,\mathbb{1}\left\{|\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} < 0\ \vee\ |\Delta_{S\cup H_{\varepsilon,t}}| \le \varepsilon\right\} \\
&\quad + Z_t\,\mathbb{1}\left\{|\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} \ge 0,\ |\Delta_{S\cup H_{\varepsilon,t}}| > \varepsilon\right\} \\
&\le\ \mathbb{1}\left\{|\Delta_t| \le \varepsilon\right\}\ +\ \mathbb{1}\left\{|\Delta_t| > \varepsilon\right\}\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} < 0\ \vee\ |\Delta_{S\cup H_{\varepsilon,t}}| \le \varepsilon\right\} \\
&\quad + Z_t\,\mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t}\ :\ \Delta_t\Delta_{S\cup H_{\varepsilon,t}} \ge 0,\ |\Delta_{S\cup H_{\varepsilon,t}}| > \varepsilon\right\}\ .
\end{aligned}$$

12. Notice the difference between the conditional expectation and conditional variance used here and those used in the proof of Lemma 23.

Summing over t proves the bound.

Proof of Lemma 15. First, note that, by the way Algorithm 2 is defined,

$$\begin{aligned}
Z_t &= \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0 \,\vee\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\} \\
&= \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\} \\
&\quad + \mathbf{1}\{\forall S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_t\}\, \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0\} \\
&\le \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\} \\
&\quad + \mathbf{1}\{|\hat{\Delta}_t| > \theta_t\}\, \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0,\ |\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_t\}.
\end{aligned}$$

We focus on the second term on the right-hand side above. Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \le \theta_t$ for all $j \in [K]$ together with Jensen's inequality, we have that $|\hat{\Delta}_t - \Delta_{\hat{C}_t,t}| \le \theta_t$ and $|\hat{\Delta}_{S\cup\hat{H}_t,t} - \Delta_{S\cup\hat{H}_t,t}| \le \theta_t$ for any S. Now, if S is such that $\hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0$, $|\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_t$, and $|\hat{\Delta}_t| > \theta_t$, then it also holds that $\Delta_{\hat{C}_t,t}\Delta_{S\cup\hat{H}_t,t} < 0$. Moreover, if there exists $S \subseteq \hat{B}_t$ such that $\Delta_{\hat{C}_t,t}\Delta_{S\cup\hat{H}_t,t} < 0$ then either $\Delta_t \Delta_{S\cup\hat{H}_t,t} < 0$ or $\Delta_t \Delta_{\hat{C}_t,t} < 0$. Since $\hat{C}_t = \hat{H}_t \cup \hat{B}_t$ we have that

$$\mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \Delta_{\hat{C}_t,t}\Delta_{S\cup\hat{H}_t,t} < 0\} \;\le\; \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\}\,.$$

Putting together, we can write

$$Z_t \;\le\; \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\} + \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\}\;.$$

Using the above, we can decompose Z_t as follows

$$\begin{aligned}
Z_t &= Z_t\, \mathbf{1}\{4\theta_t > \varepsilon\} + Z_t\, \mathbf{1}\{4\theta_t \le \varepsilon\} \\
&\le Z_t\, \mathbf{1}\{4\theta_t > \varepsilon\} + \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\}\, \mathbf{1}\{4\theta_t \le \varepsilon\} \\
&\quad + \mathbf{1}\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\}\, \mathbf{1}\{4\theta_t \le \varepsilon\}\;. \qquad (17)
\end{aligned}$$

Next, we show that $\hat{B}_t$ can be replaced with B_t in the equation above. To do so, we use the fact that $\hat{B}_t$ appears only in terms that are multiplied by $\mathbf{1}\{4\theta_t \le \varepsilon\}$. Using the definition of $\hat{B}_t$, the fact that $|\hat{\Delta}_{j_t^\star,t}| \le |\hat{\Delta}_{\hat{j}_t,t}|$ and $|\Delta_{\hat{j}_t,t}| \le |\Delta_{j_t^\star,t}|$, together with the assumption $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \le \theta$ for all $j \in [K]$ we get

$$\hat{B}_t \subseteq \{i \,:\, |\Delta_{j_t^\star,t}| - \tau - 4\theta_t \;\le\; |\Delta_{i,t}| \;\le\; |\Delta_{j_t^\star,t}| - \tau + 4\theta_t\}\;.$$

If $4\theta_t \le \varepsilon$ then the right-hand side above is a subset of $B_{\varepsilon,t}$, and therefore, under this condition, $\hat{B}_t \subseteq B_{\varepsilon,t}$. We conclude that $\hat{B}_t$ can be replaced by B_t in Equation (17), and

$$\begin{aligned}
Z_t &\le Z_t\, \mathbf{1}\{4\theta_t > \varepsilon\} + \mathbf{1}\{\exists S \subseteq B_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \theta_t\}\, \mathbf{1}\{4\theta_t \le \varepsilon\} \\
&\quad + \mathbf{1}\{\exists S \subseteq B_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\}\, \mathbf{1}\{4\theta_t \le \varepsilon\} \\
&\le Z_t\, \mathbf{1}\{4\theta_t > \varepsilon\} + \mathbf{1}\{\exists S \subseteq B_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \le \varepsilon/4\} + \mathbf{1}\{\exists S \subseteq B_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\}\;.
\end{aligned}$$

With the inequality above handy, we are now ready to upper-bound $Q_{T,\varepsilon}$. We have

$$\begin{aligned}
Q_{T,\varepsilon} &= \sum_{t=1}^{T} Z_t \, \mathbb{1}\big\{\forall S \subseteq B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup H_{\varepsilon,t},t} \geq 0,\ |\Delta_{S\cup H_{\varepsilon,t},t}| > \varepsilon\big\} \\
&\leq \sum_{t=1}^{T} Z_t \, \mathbb{1}\big\{4\theta_t > \varepsilon\big\} \\
&\quad + \underbrace{\mathbb{1}\big\{\exists S \subseteq B_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \varepsilon/4\big\}\, \mathbb{1}\big\{\forall S \subset B_{\varepsilon,t} \,:\, |\Delta_{S\cup H_{\varepsilon,t},t}| > \varepsilon\big\}}_{=0} \\
&\quad + \underbrace{\mathbb{1}\big\{\exists S \subseteq B_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\big\}\, \mathbb{1}\big\{\forall S \subset B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup H_{\varepsilon,t},t} \geq 0\big\}}_{=0} \\
&\leq \frac{16}{\varepsilon^2} \sum_{t=1}^{T} Z_t \theta_t^2 \ .
\end{aligned}$$

Recall that $\theta_t^2 = \mathbf{x}_t^\top A_{t-1}^{-1} \mathbf{x}_t \big(1 + 4\sum_{i=1}^{t-1} Z_i r_i + 36\log(Kt/\delta)\big)$. Using Lemma 24 (i), we obtain $Q_{T,\varepsilon} \leq \frac{32}{\varepsilon^2} \sum_{t=1}^{T} Z_t r_t \Big(1 + 4\sum_{i=1}^{t-1} Z_i r_i + 36\log(Kt/\delta)\Big)$. The conclusion of the proof follows along the lines of the proof of Lemma 6.

Proof of Lemma 16. We first prove that $\hat{H}_t \subseteq C_t \subseteq \hat{C}_t$. If $j \in C_t$, then $|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau \geq |\Delta_{\hat{j}_t,t}| - \tau$. Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_t$ and $|\Delta_{\hat{j}_t,t} - \hat{\Delta}_{\hat{j}_t,t}| \leq \theta_t$, we have that $|\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau - 2\theta_t$, and therefore $j \in \hat{C}_t$. Similarly, if $j \in \hat{H}_t$, then $|\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau + 2\theta_t \geq |\hat{\Delta}_{j_t^\star,t}| - \tau + 2\theta_t$. Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_t$ and $|\Delta_{j_t^\star,t} - \hat{\Delta}_{j_t^\star,t}| \leq \theta_t$, we get $|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau$, and therefore $j \in C_t$.

Now assume that $Z_t = 0$. By definition, $\hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} \geq 0$ and $|\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta$ for all $S \subseteq \hat{B}_t$, and particularly for $S = C_t \setminus \hat{H}_t$. Namely, $\hat{\Delta}_t \hat{\Delta}_{C_t,t} \geq 0$ and $|\hat{\Delta}_{C_t,t}| > \theta_t$. Once again using the assumption of the lemma, this time in conjunction with Jensen's inequality, we get that $(\Delta_t - \hat{\Delta}_{C_t,t})^2 \leq \theta_t^2$, which implies $\Delta_t \hat{\Delta}_{C_t,t} \geq \frac{1}{2}\Big(\hat{\Delta}_{C_t,t}^2 - \theta_t^2\Big)$. Plugging in $|\hat{\Delta}_{C_t,t}| > \theta_t$ gives $\Delta_t \hat{\Delta}_{C_t,t} > 0$ which, combined with $\hat{\Delta}_t \hat{\Delta}_{C_t,t} \geq 0$ gives $\Delta_t \hat{\Delta}_t \geq 0$. Overall we have shown that $Z_t = 0$ implies that $\Delta_t \hat{\Delta}_t \geq 0$. Therefore, $U_T = \sum_{t=1}^{T} \bar{Z}_t \, \mathbb{1}\big\{\Delta_t \hat{\Delta}_t < 0\big\} = 0$.

Proof of Lemma 19. First, note that, by the way Algorithm 2 is defined,

$$\begin{aligned}
Z_t &= \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0 \ \vee\ |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}\big\} \\
&= \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}\big\} \\
&\quad + \mathbb{1}\big\{\forall S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_{S\cup\hat{H}_t,t}\big\}\, \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0\big\} \\
&\leq \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}\big\} \\
&\quad + \mathbb{1}\big\{|\hat{\Delta}_t| > \theta_{\hat{C}_t,t}\big\}\, \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, \hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0,\ |\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_{S\cup\hat{H}_t,t}\big\}.
\end{aligned}$$

We focus on the second term on the right-hand side above. Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_{j,t}$ for all $j \in [K]$ together with Jensen's inequality, we have that $|\hat{\Delta}_t - \Delta_{\hat{C}_t,t}| \leq \theta_{\hat{C}_t,t}$ and $|\hat{\Delta}_{S\cup\hat{H}_t,t} - \Delta_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}$ for any S. Now, if S is such that $\hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} < 0$, $|\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_{S\cup\hat{H}_t,t}$, and $|\hat{\Delta}_t| > \theta_{\hat{C}_t,t}$, then it also holds that $\Delta_{\hat{C}_t,t} \Delta_{S\cup\hat{H}_t,t} < 0$. Moreover, if there exists $S \subseteq \hat{B}_t$ such that $\Delta_{\hat{C}_t,t} \Delta_{S\cup\hat{H}_t,t} < 0$ then either $\Delta_t \Delta_{S\cup\hat{H}_t,t} < 0$ or $\Delta_t \Delta_{\hat{C}_t,t} < 0$. Since $\hat{C}_t = \hat{H}_t \cup \hat{B}_t$ we have that

$$\mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, \Delta_{\hat{C}_t,t} \Delta_{S\cup\hat{H}_t,t} < 0\big\} \ \leq\ \mathbb{1}\big\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\big\} \ .$$

Putting together, we can write

$$Z_t \;\leq\; \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}\Big\} + \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\Big\} \;\;.$$

Using the above, we can decompose Z_t as follows

$$\begin{aligned}
Z_t \;&=\; Z_t\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} > \varepsilon\Big\} + Z_t\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\Big\} \\
&\leq\; Z_t\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} > \varepsilon\Big\} + \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \theta_{S\cup\hat{H}_t,t}\Big\}\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\Big\} \\
&\quad+\;\; \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\Big\}\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\Big\} \\
&\leq\; Z_t\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} > \varepsilon\Big\} + \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \varepsilon/4\Big\}\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\Big\} \\
&\quad+\;\; \mathbf{1}\Big\{\exists S \subseteq \hat{B}_t \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\Big\}\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\Big\} \;\;. \qquad (18)
\end{aligned}$$

where the last step is because $\max_{j\in\hat{C}_t}\theta_{j,t} \geq \theta_{S\cup\hat{H}_t,t}$. Next, we show that $\hat{B}_t$ can be replaced with B_t in the equation above. To do so, we use the fact that $\hat{B}_t$ appears only in terms that are multiplied by $\mathbf{1}\big\{4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon\big\}$. Using the definition of $\hat{B}_t$, the fact that $|\hat{\Delta}_{j_t^\star,t}| \leq |\hat{\Delta}_{\hat{j}_t,t}|$ and $|\Delta_{\hat{j}_t,t}| \leq |\Delta_{j_t^\star,t}|$, together with the assumption $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_{j,t}$ for all $j \in [K]$ we get

$$\hat{B}_t \subseteq \left\{ i \,:\, |\Delta_{j_t^\star,t}| - \tau - 4\max_{j\in\hat{C}_t}\theta_{j,t} \;\leq\; |\Delta_{i,t}| \;\leq\; |\Delta_{j_t^\star,t}| - \tau + 4\theta_{\hat{C}_t,t} \right\} \;\;.$$

Hence, when $4\max_{j\in\hat{C}_t}\theta_{j,t} \leq \varepsilon$ we are guaranteed that the right-hand side above is a subset of $B_{\varepsilon,t}$, and therefore, under this condition, $\hat{B}_t \subseteq B_{\varepsilon,t}$. We conclude that $\hat{B}_t$ can be replaced by $B_{\varepsilon,t}$ in Equation (18), and so Z_t is upper bounded by

$$Z_t\,\mathbf{1}\Big\{4\max_{j\in\hat{C}_t}\theta_{j,t} > \varepsilon\Big\} + \mathbf{1}\Big\{\exists S \subseteq B_{\varepsilon,t} \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \varepsilon/4\Big\} + \;\mathbf{1}\Big\{\exists S \subseteq B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\Big\} \;\;.$$

With the inequality above handy, we are now ready to upper-bound $Q_{T,\varepsilon}$. We have

$$\begin{aligned}
Q_{T,\varepsilon} &= \sum_{t=1}^{T} Z_t \, \mathbb{1}\left\{\forall S \subseteq B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup H_{\varepsilon,t},t} \geq 0,\ |\Delta_{S\cup H_{\varepsilon,t},t}| > \varepsilon\right\} \\
&\leq \sum_{t=1}^{T} Z_t \, \mathbb{1}\left\{4 \max_{j\in\hat{C}_t} \theta_{j,t} > \varepsilon\right\} \\
&\quad + \underbrace{\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t} \,:\, |\hat{\Delta}_{S\cup\hat{H}_t,t}| \leq \varepsilon/4\right\} \mathbb{1}\left\{\forall S \subset B_{\varepsilon,t} \,:\, |\Delta_{S\cup H_{\varepsilon,t},t}| > \varepsilon\right\}}_{=0} \\
&\quad + \underbrace{\mathbb{1}\left\{\exists S \subseteq B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup\hat{H}_t,t} < 0\right\} \mathbb{1}\left\{\forall S \subset B_{\varepsilon,t} \,:\, \Delta_t \Delta_{S\cup H_{\varepsilon,t},t} \geq 0\right\}}_{=0} \\
&\leq \frac{16}{\varepsilon^2} \sum_{t=1}^{T} Z_t \max_{j\in\hat{C}_t} \theta_{j,t}^2 \\
&\leq \frac{16 \sum_{t=1}^{T} Z_t \sum_{j\in\hat{C}_t} \theta_{j,t}^2}{\varepsilon^2} \\
&= \frac{16 \sum_{j\in[K]} \sum_{t=1}^{T} Z_t \, \mathbb{1}\left\{j \in \hat{C}_t\right\} \theta_{j,t}^2}{\varepsilon^2} \\
&= \frac{16 \sum_{j\in[K]} \sum_{t=1}^{T} Z_t \, \mathbb{1}\left\{j \in \hat{C}_t\right\} \mathbf{x}_t^\top A_{j,t-1}^{-1} \mathbf{x}_t \left(1 + 4\sum_{i=1}^{t-1} Z_i r_{j,i} + 36\log(Kt/\delta)\right)}{\varepsilon^2} \\
&= \frac{16 \sum_{j\in[K]} \sum_{t=1}^{T} Z_t r_{j,t} \left(1 + 4\sum_{i=1}^{t-1} Z_i r_{j,i} + 36\log(Kt/\delta)\right)}{\varepsilon^2} \,.
\end{aligned}$$

Now, proceeding along the same lines as in the proof of Lemma 6 (which in turn mainly relies on Lemma 24) we conclude that,

$$Q_{T,\varepsilon} \leq \frac{16 \sum_{j\in[K]} \left((1 + 36\log(KT/\delta)) \log|A_{j,T}| + 4\log^2|A_{j,T}|\right)}{\varepsilon^2} = O\left(\frac{Kd^2 \log^2(KT/\delta)}{\varepsilon^2}\right) .$$

This concludes the proof.

Proof of Lemma 20. The proof proceeds in the same way as the proof of Lemma 16. We first prove that $\hat{H}_t \subseteq C_t \subseteq \hat{C}_t$. If $j \in C_t$, then $|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau \geq |\Delta_{\hat{j}_t,t}| - \tau$. Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_{j,t}$ and $|\Delta_{\hat{j}_t,t} - \hat{\Delta}_{\hat{j}_t,t}| \leq \theta_{\hat{j}_t,t}$, we have that $|\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau - \theta_{j,t} - \theta_{\hat{j}_t,t}$, and therefore $j \in \hat{C}_t$. Similarly, if $j \in \hat{H}_t$, then

$$|\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau + \theta_{j,t} + \max_{j\in\hat{C}_t} \geq |\hat{\Delta}_{j_t^\star,t}| - \tau + \theta_{j,t} + \max_{j\in\hat{C}_t} .$$

Using the assumption that $|\Delta_{j,t} - \hat{\Delta}_{j,t}| \leq \theta_{j,t}$ and $|\Delta_{j_t^\star,t} - \hat{\Delta}_{j_t^\star,t}| \leq \theta_{j_t^\star,t} \leq \max_{j\in\hat{C}_t} \theta_{j,t}$, we get $|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau$, and therefore $j \in C_t$.

Now assume that $Z_t = 0$. By definition, $\hat{\Delta}_t \hat{\Delta}_{S\cup\hat{H}_t,t} \geq 0$ and $|\hat{\Delta}_{S\cup\hat{H}_t,t}| > \theta_{S\cup\hat{H}_t,t}$ for all $S \subseteq \hat{B}_t$, and particularly for $S = C_t \setminus \hat{H}_t$. Namely, $\hat{\Delta}_t \hat{\Delta}_{C_t,t} \geq 0$ and $|\hat{\Delta}_{C_t,t}| > \theta_{\hat{C}_t,t}$. Once again using the assumption of the lemma, this time in conjunction with Jensen's inequality, we get that $(\Delta_t - \hat{\Delta}_{C_t,t})^2 \leq \theta_t^2$, which implies $\Delta_t \hat{\Delta}_{C_t,t} \geq \frac{1}{2}\left(\hat{\Delta}_{C_t,t}^2 - \theta_{\hat{C}_t,t}^2\right)$. Plugging in $|\hat{\Delta}_{C_t,t}| > \theta_{\hat{C}_t,t}$ gives $\Delta_t \hat{\Delta}_{C_t,t} > 0$ which, combined

with $\hat{\Delta}_t \hat{\Delta}_{C_t,t} \geq 0$, gives $\Delta_t \hat{\Delta}_t \geq 0$. Overall we have shown that $Z_t = 0$ implies that $\Delta_t \hat{\Delta}_t \geq 0$. Therefore, $U_T \;=\; \sum_{t=1}^{T} \bar{Z}_t \, \mathbb{1}\{\Delta_t \hat{\Delta}_t < 0\} \;=\; 0$.

Proof of Lemma 18. We start just as in the proof of Lemma 14 and get,

$$\begin{aligned} Z_t \leq \;& \mathbb{1}\{|\Delta_t| \leq \varepsilon\} \;+\; Z_t \, \mathbb{1}\{|\Delta_t| > \varepsilon\} \, \mathbb{1}\{\exists S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \;\vee\; |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon\} \\ & + Z_t \, \mathbb{1}\{\forall S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} \geq 0, \; |\Delta_{S \cup H_{\varepsilon,t}}| > \varepsilon\} \;\; . \end{aligned}$$

Hence,

$$\begin{aligned} N_T = \;& \sum_{t=1}^{T} |\hat{C}_t| Z_t \\ \leq \;& \sum_{t=1}^{T} |\hat{C}_t| \, \mathbb{1}\{|\Delta_t| \leq \varepsilon\} \\ & + \sum_{t=1}^{T} |\hat{C}_t| Z_t \, \mathbb{1}\{|\Delta_t| > \varepsilon\} \, \mathbb{1}\{\exists S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \;\vee\; |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon\} \\ & + \sum_{t=1}^{T} |\hat{C}_t| Z_t \, \mathbb{1}\{\forall S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} \geq 0, \; |\Delta_{S \cup H_{\varepsilon,t}}| > \varepsilon\} \\ \leq \;& K \sum_{t=1}^{T} \mathbb{1}\{|\Delta_t| \leq \varepsilon\} \\ & + K \sum_{t=1}^{T} \frac{|\hat{C}_t|}{K} Z_t \, \mathbb{1}\{|\Delta_t| > \varepsilon\} \, \mathbb{1}\{\exists S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \;\vee\; |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon\} \\ & + K \sum_{t=1}^{T} Z_t \, \mathbb{1}\{\forall S \subseteq B_{\varepsilon,t} \;:\; \Delta_t \Delta_{S \cup H_{\varepsilon,t}} \geq 0, \; |\Delta_{S \cup H_{\varepsilon,t}}| > \varepsilon\} \\ = \;& K T_\varepsilon \\ & + K \sum_{t=1}^{T} \tfrac{|\hat{C}_t| Z_t}{K} \, \mathbb{1}\{|\Delta_t| > \varepsilon\} \, \mathbb{1}\{\exists S \subseteq B_{\varepsilon,t} : \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \vee |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon\} \\ & + K Q_{T,\varepsilon} \; . \end{aligned} \tag{19}$$

Now we note that by definition of $\hat{C}_t$, if $j \in \hat{C}_t$ then

$$|\hat{\Delta}_{j,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \tau - \theta_{j,t} - \theta_{\hat{j}_t,t} \geq |\hat{\Delta}_{j_t^\star,t}| - \tau - \theta_{j,t} - \theta_{\hat{j}_t,t} \; .$$

Combined with our assumption that $(\Delta_{j,t} - \hat{\Delta}_{j,t})^2 \leq \theta_{j,t}^2$ holds for all $j \in [K]$ this implies

$$|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau - 2\theta_{j,t} - \theta_{\hat{j}_t,t} - \theta_{j_t^\star,t} \; . \tag{20}$$

On the other hand, by definition of $j_t^\star$, we also have $|\Delta_{j_t^\star,t}| \geq |\Delta_{\hat{j}_t,t}|$ and, owing to our assumption, $|\hat{\Delta}_{j_t^\star,t}| \geq |\hat{\Delta}_{\hat{j}_t,t}| - \theta_{j_t^\star,t} - \theta_{\hat{j}_t,t}$. Hence we see that $j_t^\star \in \hat{C}_t$. Using this in Equation (20) gives, for any $j \in \hat{C}_t$,

$$|\Delta_{j,t}| \geq |\Delta_{j_t^\star,t}| - \tau - 4 \max_{j \in \hat{C}_t} \theta_{j,t} \; .$$

Thus we see that as long as $4 \max_{j \in \hat{C}_t} \theta_{j,t} \leq \varepsilon$, we have $\hat{C}_t \subset H_{\varepsilon,t} \cup B_{\varepsilon,t}$.

We now use this in Equation (19). We obtain

$$\begin{aligned}
N_T \leq & K T_\varepsilon + K Q_{T,\varepsilon} \\
& + K \left(\sum_{t=1}^{T} \frac{|\hat{C}_t| Z_t}{K} \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} \leq \varepsilon \right\} \mathbb{1}\left\{ |\Delta_t| > \varepsilon \right\} \times \right. \\
& \qquad \left. \times \mathbb{1}\left\{ \exists S \subseteq B_{\varepsilon,t} : \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \vee |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon \right\} \right) \\
& + K \left(\sum_{t=1}^{T} \frac{|\hat{C}_t| Z_t}{K} \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} > \varepsilon \right\} \mathbb{1}\left\{ |\Delta_t| > \varepsilon \right\} \times \right. \\
& \qquad \left. \times \mathbb{1}\left\{ \exists S \subseteq B_{\varepsilon,t} : \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \vee |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon \right\} \right) \\
\leq & \; K T_\varepsilon + K Q_{T,\varepsilon} \\
& + K \left(\sum_{t=1}^{T} \frac{|B_{\varepsilon,t} \cup H_{\varepsilon,t}|}{K} \mathbb{1}\left\{ |\Delta_t| > \varepsilon \right\} \mathbb{1}\left\{ \exists S \subseteq B_{\varepsilon,t} : \Delta_t \Delta_{S \cup H_{\varepsilon,t}} < 0 \vee |\Delta_{S \cup H_{\varepsilon,t}}| \leq \varepsilon \right\} \right) \\
& + K \sum_{t=1}^{T} Z_t \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} > \varepsilon \right\} \\
= & \; K \left(T_\varepsilon + T''_\varepsilon + Q_{T,\varepsilon} + \sum_{t=1}^{T} Z_t \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} > \varepsilon \right\} \right) . \qquad (21)
\end{aligned}$$

In order to bound last term, we notice that

$$\begin{aligned}
& \sum_{t=1}^{T} Z_t \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} > \varepsilon \right\} \\
& \quad \leq \frac{16 \sum_{t=1}^{T} Z_t \max_{j \in \hat{C}_t} \theta_{j,t}^2}{\varepsilon^2} \\
& \quad \leq \frac{16 \sum_{t=1}^{T} Z_t \sum_{j \in \hat{C}_t} \theta_{j,t}^2}{\varepsilon^2} \\
& \quad = \frac{16 \sum_{j \in [K]} \sum_{t=1}^{T} Z_t \mathbb{1}\left\{ j \in \hat{C}_t \right\} \theta_{j,t}^2}{\varepsilon^2} \\
& \quad = \frac{16 \sum_{j \in [K]} \sum_{t=1}^{T} Z_t \mathbb{1}\left\{ j \in \hat{C}_t \right\} \mathbf{x}_t^\top A_{j,t-1}^{-1} \mathbf{x}_t \left(1 + 4 \sum_{i=1}^{t-1} Z_i r_{j,i} + 36 \log(Kt/\delta) \right)}{\varepsilon^2} \\
& \quad = \frac{16 \sum_{j \in [K]} \sum_{t=1}^{T} Z_t r_{j,t} \left(1 + 4 \sum_{i=1}^{t-1} Z_i r_{j,i} + 36 \log(Kt/\delta) \right)}{\varepsilon^2} .
\end{aligned}$$

Now, proceeding along the lines of the proof of Lemma 6 (which in turn mainly relies on Lemma 24), we obtain

$$\begin{aligned}
\sum_{t=1}^{T} Z_t \mathbb{1}\left\{ 4 \max_{j \in \hat{C}_t} \theta_{j,t} > \varepsilon \right\} & \leq \frac{16 \sum_{j \in [K]} \left((1 + 36 \log(KT/\delta)) \log |A_{j,T}| + 4 \log^2 |A_{j,T}| \right)}{\varepsilon^2} \\
& = O\left(\frac{K d^2 \log^2(KT/\delta)}{\varepsilon^2} \right) .
\end{aligned}$$

Plugging these back into Equation (21) and applying the bound on $Q_{T,\varepsilon}$ from Lemma 19 concludes the proof.

References

B. D. Argall, B. Browning, and M. Veloso. Automatic weight learning for multiple data sources when learning from demonstration. In *Proc. of the 2009 IEEE International Conference on Robotics and Automation*, pages 226–231, 2009.

K. S. Azoury and M. K. Warmuth. Relative loss bounds for online density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

F. Bach. Active learning for misspecified generalized linear models. Technical Report N15/06/MM, Ecole des mines de Paris, June 2006.

M. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proc. of the 23th International Conference on Machine Learning*, pages 65–72, 2006.

M. Balcan, A. Broder, and T. Zhang. Margin-based active learning. In *Proc. of the 20th Annual Conference on Learning Theory*, pages 35–50, 2007.

M. Balcan, S. Hanneke, and J. Wortman. The true sample complexity of active learning. In *Proc. of the 21th Annual Conference on Learning Theory*, pages 45–56, 2008.

A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems 24*, pages 199–207, 2010.

A. Beygelzimer, D. Hsu, N. Karampatziakis, J. Langford, and T. Zhang. Efficient active learning. In *ICML 2011 Workshop on On-line Trading of Exploration and Exploitation*, 2011.

R. Castro and R. D. Nowak. Minimax bounds for active learning. *IEEE Transactions on Information Theory*, 54(5):2339–2353, 2008.

G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear classification and selective sampling under low noise conditions. In *Advances in Neural Information Processing Systems 21*, pages 249–256, 2009.

N. Cesa-Bianchi and C. Gentile. Improved risk tail bounds for on-line algorithms. *IEEE Transactions on Information Theory*, 54(1):386–390, 2008.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *Proc. of the 16th Annual Conference on Learning Theory*, pages 373–387, 2003.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order Perceptron algorithm. *SIAM Journal on Computing*, 43(3):640–668, 2005a.

N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Minimizing regret with label efficient prediction. *IEEE Transactions on Information Theory*, 51(6):2152–2162, 2005b.

N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 7:1025–1230, 2006.

N. Cesa-Bianchi, C. Gentile, and F. Orabona. Robust bounds for classification via selective sampling. In *Proc. of the 26th International Conference on Machine Learning*, pages 121–128, 2009.

S. Chen, J. Zhang, G. Chen, and C Zhang. What if the irresponsible teachers are dominating? A method of training on samples and clustering on teachers. In *Proc. of the Twenty-fourth AAAI Conference on Artificial Intelligence*, pages 419–424, 2010.

R. Cohn, L. Atlas, and R. Ladner. Training connectionist networks with queries and selective sampling. In *Advances in Neural Information Processing Systems 2*, pages 566–573, 1990.

K. Crammer and C. Gentile. Multiclass classification with bandit feedback using adaptive regularization. In *Proc. of the 28th International Conference on Machine Learning*, pages 273–280, 2011.

K. Crammer, M. Kearns, and J. Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9:1757–1774, 2008.

V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. In *Proc. of the 21th Annual Conference on Learning Theory*, pages 355–366, 2008.

S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. In *Proc. of the 18th Annual Conference on Learning Theory*, pages 249–263, 2005.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems 21*, pages 353–360, 2008.

A.P. Dawid and A.M. Skeene. Maximum likelihood estimation of observed error-rates using the em algorithm. *Applied statistics*, 28:20–28, 1979.

O. Dekel and O. Shamir. Vox populi: Collecting high-quality labels from a crowd. In *Proc. of the 22nd Annual Conference on Learning Theory*, 2009a.

O. Dekel and O. Shamir. Good learners for evil teachers. In *Proc. of the Twenty-Sixth International Conference on Machine Learning*, pages 216–223, 2009b.

P. Domnez. *Proactive Learning: Towards Learning with Multiple Imperfect Predictors*. PhD thesis, Carnegie Mellon University, 2010.

P. Donmez and J. G. Carbonell. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proc. of the 17th ACM Conference on Information and Knowledge Management*, pages 619–628, 2008.

P. Donmez, J.G. Carbonell, and J. Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. In *Proc. of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2009.

P. Donmez, J.G. Carbonell, and J. Schneider. A probabilistic framework to learn from multiple annotators with time-varying accuracy. In *Proc. of the SIAM International Conference on Data Mining*, pages 826–837, 2010.

Y. Freund, S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

P. Groot, A. Birlutiu, and T. Heskes. Learning from multiple annotators with Gaussian processes. In *Proc. of the 21st International Conference on Artificial Neural Networks*, pages 159–164, 2011.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *Proc. of the 24th International Conference on Machine Learning*, pages 353–360, 2007.

S. Hanneke. Adaptive rates of convergence in active learning. In *Proc. of the 22th Annual Conference on Learning Theory*, 2009.

E. Hazan, A. Kalai, S. Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 2:169–192, 2007.

D. Helmbold and S. Panizza. Some label efficient learning results. In *Proc. of the 10th Conference of Computationa Learning Theory*, pages 218–230, 1997.

A. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

S.L. Hui and X.H. Zhou. Evaluation of diagnostic tests without a gold standard. *Statistical Methods in Medical Research*, 7:354–370, 1998.

S. Kakade and A. Tewari. On the generalization ability of online strongly convex programming algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808, 2008.

V. Koltchinskii. Rademacher complexities and bounding the excess risk in active learning. *Journal of Machine Learning Research*, 11:2457–2485, 2010.

T. L. Lai and C. Z. Wei. Least squares estimates in stochastic regression models with applications to identification and control of dynamic systems. *The Annals of Statistics*, pages 154–166, 1982.

L. Li, M. Littman, and T. Walsh. Knows what it knows: a framework for self-aware learning. In *Proc. of the 25th International Conference on Machine Learning*, pages 568–575, 2008.

N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems 21*, pages 1041–1048, 2009a.

Y. Mansour, M. Mohri, and A. Rostamizadeh. Multiple source adaptation and the Renyi divergence. In *Proc. of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 367–374, 2009b.

P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. Economical active feature-value acquisition through expected utility estimation. In *KDD Workshop on Utility-based data mining*, 2005.

F. Orabona and N. Cesa-Bianchi. Better algorithms for selective sampling. In *Proc. of the 28th International Conference on Machine Learning*, pages 433–440, 2011.

V.C. Raykar, S. Yu, L.H. Zhao, G.H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning*, 11:1297–1322, 2010.

V.S. Sheng, F. Provost, and P.G. Ipeirotis. Get another label? Improving data quality and data mining using multiple noisy labelers. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 614–622, 2008.

P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. Inferring ground truth from subjective labelling of venus images. In *Advances in neural information processing systems*, pages 1085–1092, 1995.

R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, 2008.

D.J. Spiegelhalter and P. Stovin. An analysis of repeated biopsies following cardiac transplantation. *Statistics in Medicine*, 2(1):33–40, 1983.

A. Strehl and M. Littman. Online linear regression and its application to model-based reinforcement learning. In *Advances in Neural Information Processing Systems 20*, pages 1417–1424, 2008.

A. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32(1): 135–166, 2004.

V. Vovk. Competitive on-line statistics. *International Statistical Review*, 69:213–248, 2001.

Y. Yan, R. Rosales, L. Bogoni, G. Fung, L. Moy, M. Schmidt, and J.G. Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. In *Proc. of the 13th International Conference on Artificial Intelligence and Statistics*, pages 932–939, 2010.

Y. Yan, R. Rosales, G. Fung, and J. Dy. Active learning from crowds. In *Proc. of the 28th International Conference on Machine Learning*, pages 1161–1168, 2011.

L. Yang and J. Carbonell. Cost complexity of proactive learning via a reduction to realizable active learning. Technical Report CMU-ML-09-113, Carnegie Mellon University, 2009a.

L. Yang and J. Carbonell. Adaptive proactive learning with cost-reliability tradeoff. Technical Report CMU-ML-09-114, Carnegie Mellon University, 2009b.

Journal of Machine Learning Research 13 (2012) 2699-2703 Submitted 7/11; Revised 4/12; Published 9/12

PREA: Personalized Recommendation Algorithms Toolkit

Joonseok Lee JLEE716@GATECH.EDU
Mingxuan Sun MSUN3@GATECH.EDU
Guy Lebanon LEBANON@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332, USA

Editor: Soeren Sonnenburg

Abstract

Recommendation systems are important business applications with significant economic impact. In recent years, a large number of algorithms have been proposed for recommendation systems. In this paper, we describe an open-source toolkit implementing many recommendation algorithms as well as popular evaluation metrics. In contrast to other packages, our toolkit implements recent state-of-the-art algorithms as well as most classic algorithms.

Keywords: recommender systems, collaborative filtering, evaluation metrics

1. Introduction

As the demand for personalized services in E-commerce increases, recommendation systems are emerging as an important business application. Amazon.com, for example, provides personalized product recommendations based on previous purchases. Other examples include music recommendations in pandora.com, movie recommendations in netflix.com, and friend recommendations in facebook.com.

A wide variety of algorithms have been proposed by the research community for recommendation systems. Unlike classification where comprehensive packages are available, existing recommendation systems toolkits lag behind. They concentrate on implementing traditional algorithms rather than the rapidly evolving state-of-the-art. Implementations of modern algorithms are scattered over different sources which makes it hard to have a fair and comprehensive comparison.

In this paper we describe a new toolkit PREA (Personalized Recommendation Algorithms Toolkit), implementing a wide variety of recommendation systems algorithms. PREA offers implementation of modern state-of-the-art recommendation algorithms as well as most traditional ones. In addition, it provides many popular evaluation methods and data sets. As a result, it can be used to conduct fair and comprehensive comparisons between different recommendation algorithms. The implemented evaluation routines, data sets, and the open-source nature of PREA makes it easy for third parties to contribute additional implementations.

Not surprisingly, the performance of recommendation algorithms depends on the data characteristics. (Lee et al., 2012a,b) For example, some algorithms may work better or worse depending on the amount of missing data (sparsity), distribution of ratings, and the number of users and items. It is likely that an algorithm may perform better on one data set and worse on another. Furthermore, the different evaluation methods that have been proposed in the literature may have conflicting orderings over algorithms (Gunawardana and Shani, 2009). PREA's ability to compare different

algorithms using a variety of evaluation metrics may clarify which algorithms perform better under what circumstance.

2. Implementation

PREA is a multi-platform Java Software (version 1.6 or higher required). It is compatible with MS Windows, Linux, and Mac OS. The toolkit consists of three groups of classes, as shown in Figure 1. The top-level routines of the toolkit may be directly called from other programming environments like Matlab. The top two groups implement basic data structures and recommendation algorithms. The third group (bottom) implements evaluation metrics, statistical distributions, and other utility functions.

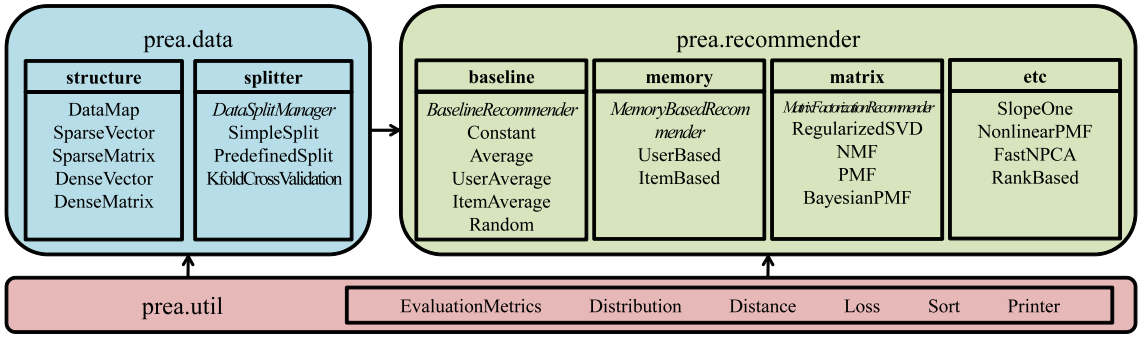


Figure 1: Class Structure of PREA

2.1 Input Data File Format

As in the WEKA[1] toolkit PREA accepts the data in ARFF[2] (Attribute-Relation File Format). Since virtually all recommendation systems data sets are sparse, PREA implements sparse (rather than dense) ARFF format.

2.2 Evaluation Framework

For ease of comparison, PREA provides the following unified interface for running the different recommendation algorithms.

- Instantiate a class instance according to the type of the recommendation algorithm.
- Build a model based on the specified algorithm and a training set.
- Given a test set, predict user ratings over unseen items.
- Given the above prediction and held-out ground truth ratings, evaluate the prediction using various evaluation metrics.

Note that lazy learning algorithms like the constant models or memory-based algorithms may skip the second step.

A simple train and test split is constructed by choosing a certain proportion of all ratings as test set and assigning the remaining ratings to the train set. In addition to this, K-fold cross-validation is also supported.

1. Official web site can be found at `http://www.cs.waikato.ac.nz/~ml/weka/`.
2. Full description about ARFF can be found at `http://www.cs.waikato.ac.nz/~ml/weka/arff.html`.

2.3 Data Structures

PREA uses a sparse matrix representation for the rating matrices which are generally extremely sparse (users provide ratings for only a small subset of items). Specifically, we use Java's data structure *HashMap* to create a *DataMap* class and a *SparseVector* class. We tried some other options including *TreeMap*, but we chose *HashMap* due to its superior performance over others. Figure 2 (left) shows an example of a sparse vector, containing data only in indices 1, 3, and 6.

We construct a *SparseMatrix* class using an array of *SparseVector* objects. To facilitate fast access to rows and columns, both row-oriented and column-oriented arrays are kept. This design is also useful for fast transposing of sparse matrices by interchanging rows and columns. Figure 2 (right) shows an example of sparse matrix.

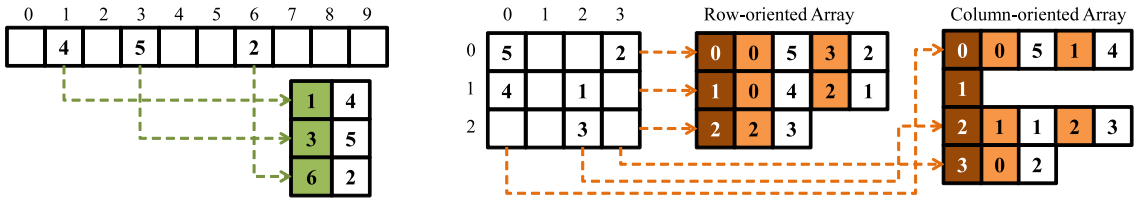


Figure 2: Sparse Vector (left) and Matrix (right) Implementation

PREA also uses dense matrices in some cases. For example, dense representations are used for low-rank matrix factorizations or other algebraic operations that do not maintain sparsity. The dense representations are based on the matrix implementations in the Universal Java Matrix Package (UJMP) (`http://www.ujmp.org/`).

2.4 Implemented Algorithms and Evaluation Metrics

PREA implements the following prediction algorithms:

- Baselines (constant, random, overall average, user average, item average): make little use of personalized information for recommending items.
- Memory-based Neighborhood algorithms (user-based, item-based collaborative filtering and their extensions including default vote, inverse user frequency): predict ratings of unseen items by referring those of similar users or items.
- Matrix Factorization methods (SVD, NMF, PMF, Bayesian PMF, Non-linear PMF): build low-rank user/item profiles by factorizing training data set with linear algebraic techniques.
- Others: recent state-of-the-art algorithms such as Fast NPCA and Rank-based collaborative filtering.

We provide popular evaluation metrics as follows:

- Accuracy for rating predictions (RMSE, MAE, NMAE): measure how much the predictions are similar to the actual ratings.
- Rank-based evaluation metrics (HLU, NDCG, Kendall's Tau, and Spearman): score depending on similarity between orderings of predicted ratings and those of ground truth.

3. Related Work

Several other open-source recommendation toolkits are available. Table 1 summarizes the implemented features in these toolkits and compares them to those in PREA. Mahout,[3] which incorporated Taste,[4] provides an implementation of many memory-based algorithms. It also supports powerful mathematical and statistical operations as it is a general-purpose machine learning toolkit. Duine[5] provides a way to combine multiple algorithms into a hybrid system and also addressed the cold-start situation. Cofi[6] implements several traditional algorithms with a simple design based on providing wrappers for publicly available data sets. MyMedia[7] is a C#-based recommendation toolkit which supports most traditional algorithms and several evaluation metrics.

As indicated in Table 1, existing toolkits widely provide simple memory-based algorithms, while recent state-of-the-art algorithms are often not supported. Also, these toolkits are generally limited in their evaluation metrics (with the notable exception of MyMedia). In contrast, PREA provides a wide coverage of the the most up-to-date algorithms as well as various evaluation metrics, facilitating a comprehensive comparison between state-of-the-art and newly proposed algorithms in the research community.

Category	Feature	PREA	Mahout	Duine	Cofi	MyMedia
Baselines	Constant	O			O	O
	User/Item Average	O	O	O	O	O
	Random	O				O
Memory-based CF	User-based CF (Su and Khoshgoftaar, 2009)	O	O	O	O	O
	Item-based CF (Sarwar et al., 2001)	O	O	O	O	O
	Default Vote, Inv-User-Freq (Breese et al., 1998)	O	O			
	Slope-One (Lemire and Maclachlan, 2005)	O	O			O
Matrix Factorization	SVD (Paterek, 2007)	O	O		O	O
	NMF (Lee and Seung, 2001)	O				
	PMF (Salakhutdinov and Mnih, 2008a)	O				
	Bayesian PMF (Salakhutdinov and Mnih, 2008b)	O				
	Non-linear PMF (Lawrence and Urtasun, 2009)	O				
Other methods	Fast NPCA (Yu et al., 2009)	O				
	Rank-based CF (Sun et al., 2011, 2012)	O				
Evaluation Metric	(N)MAE	O	O	O	O	O
	RMSE	O	O		O	O
	HLU/NDCG	O				O
	Kendall's Tau, Spearman	O				
	Precision/Recall/F1		O			O
Miscellaneous	Sparse Vector/Matrix	O	O	O	O	O
	Wrapper for other languages	O			O	O
	Item Recommender for Positive-only Data					O
	Release Year	2011	2005	2009	2004	2009
	Language	Java	Java	Java	Java	C#
	License	GPL	LGPL	LGPL	GPL	GPL

Table 1: Comparison of Features with other Collaborative Filtering Toolkits

4. Summary

PREA contributes to recommendation system research community and industry by (a) providing an easy and fair comparison among most traditional recommendation algorithms, (b) supporting state-of-the-art recommendation algorithms which have not been available on other toolkits, and (c) implementing different evaluation criteria emphasizing different performance aspects rather than a single evaluation measure like mean squared error.

The open-source nature of the software (available under GPL) may encourage other recommendation systems experts to add their own algorithms to PREA. More documentation for developers as well as user tutorials are available on the web (`http://prea.gatech.edu`).

3. Details can be found at `https://cwiki.apache.org/confluence/display/MAHOUT/`.
4. Details can be found at `http://taste.sourceforge.net/old.html`.
5. Details can be found at `http://www.duineframework.org/`.
6. Details can be found at `http://www.nongnu.org/cofi/`.
7. Details can be found at `http://www.ismll.uni-hildesheim.de/mymedialite/`.

References

J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of Uncertainty in Artificial Intelligence*, 1998.

A. Gunawardana and G. Shani. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, 10:2935–2962, 2009.

N. D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proc. of the International Conference of Machine Learning*, 2009.

D. Lee and H. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, 2001.

J. Lee, M. Sun, S. Kim, and G. Lebanon. Automatic feature induction for stagewise collaborative filtering. In *Advances in Neural Information Processing Systems*, 2012a.

J. Lee, M. Sun, and G. Lebanon. A comparative study of collaborative filtering algorithms. *ArXiv Report 1205.3193*, 2012b.

D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. *Society for Industrial Mathematics*, 5:471–480, 2005.

A. Paterek. Improving regularized singular value decomposition for collaborative filtering. *Statistics*, 2007:2–5, 2007.

R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, 2008a.

R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. of the International Conference on Machine Learning*, 2008b.

B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the International Conference on World Wide Web*, 2001.

X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4:2–4:2, January 2009.

M. Sun, G. Lebanon, and P. Kidwell. Estimating probabilities in recommendation systems. In *Proc. of the International Conference on Artificial Intelligence and Statistics*, 2011.

M. Sun, G. Lebanon, and P. Kidwell. Estimating probabilities in recommendation systems. *Journal of the Royal Statistical Society, Series C*, 61(3):471–492, 2012.

K. Yu, S. Zhu, J. Lafferty, and Y. Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proc. of ACM SIGIR Conference*, 2009.

Journal of Machine Learning Research 13 (2012) 2705-2734 Submitted 4/11; Revised 2/12; Published 9/12

Coherence Functions with Applications in Large-Margin Classification Methods

Zhihua Zhang ZHZHANG@ZJU.EDU.CN
Dehua Liu DEHUALIU0427@GMAIL.COM
Guang Dai GUANG.GDAI@GMAIL.COM
College of Computer Science and Technology
Zhejiang University
Hangzhou, Zhejiang 310027, China

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Xiaotong Shen

Abstract

Support vector machines (SVMs) naturally embody sparseness due to their use of hinge loss functions. However, SVMs can not directly estimate conditional class probabilities. In this paper we propose and study a family of *coherence functions*, which are convex and differentiable, as surrogates of the hinge function. The coherence function is derived by using the maximum-entropy principle and is characterized by a temperature parameter. It bridges the hinge function and the logit function in logistic regression. The limit of the coherence function at zero temperature corresponds to the hinge function, and the limit of the minimizer of its expected error is the minimizer of the expected error of the hinge loss. We refer to the use of the coherence function in large-margin classification as "$\mathcal{C}$*-learning*," and we present efficient coordinate descent algorithms for the training of regularized $\mathcal{C}$-learning models.

Keywords: large-margin classifiers, hinge functions, logistic functions, coherence functions, $\mathcal{C}$-learning

1. Introduction

Large-margin classification methods have become increasingly popular since the advent of boosting (Freund, 1995), support vector machines (SVM) (Vapnik, 1998) and their variants such as ψ-learning (Shen et al., 2003). Large-margin classification methods are typically devised based on a majorization-minimization procedure, which approximately solves an otherwise intractable optimization problem defined with the 0-1 loss. For example, the conventional SVM employs a hinge loss, the AdaBoost algorithm employs the exponential loss, and ψ-learning employs a so-called ψ-loss, as majorizations of the 0-1 loss.

Large-margin classification methods can be unified using the tools of regularization theory; that is, they can be expressed as the form of "loss" + "penalty" (Hastie et al., 2001). Sparseness has also emerged as a significant theme generally associated with large-margin methods. Typical approaches for achieving sparseness are to use either a non-differentiable penalty or a non-differentiable loss.

Recent developments in the former vein focus on the use of the ℓ_1 penalty (Tibshirani, 1996) or the elastic-net penalty (a mixture of the ℓ_1 and ℓ_2 penalties) (Zou and Hastie, 2005) instead of the ℓ_2 penalty which is typically used in large-margin classification methods. As for non-differentiable losses, the paradigm case is the hinge loss function that is used for the SVM and which leads to a sparse expansion of the discriminant function.

Unfortunately, the conventional SVM does not directly estimate a conditional class probability. Thus, the conventional SVM is unable to provide estimates of uncertainty in its predictions—an important desideratum in real-world applications. Moreover, the non-differentiability of the hinge loss also makes it difficult to extend the conventional SVM to multi-class classification problems. Thus, one seemingly natural approach to constructing a classifier for the binary and multi-class problems is to consider a smooth loss function, while an appropriate penalty is employed to maintain the sparseness of the classifier. For example, regularized logistic regression models based on logit losses (Friedman et al., 2010) are competitive with SVMs.

Of crucial concern are the statistical properties (Lin, 2002; Bartlett et al., 2006; Zhang, 2004) of the majorization function for the original 0-1 loss function. In particular, we analyze the statistical properties of extant majorization functions, which are built on the exponential, logit and hinge functions. This analysis inspires us to propose a new majorization function, which we call a *coherence function* due to a connection with statistical mechanics. We also define a loss function that we refer to as $\mathcal{C}$-loss based on the coherence function.

The $\mathcal{C}$-loss is smooth and convex, and it satisfies a Fisher-consistency condition—a desirable statistical property (Bartlett et al., 2006; Zhang, 2004). The $\mathcal{C}$-loss has the advantage over the hinge loss that it provides an estimate of the conditional class probability, and over the logit loss that one limiting version of it is just the hinge loss. Thus, the $\mathcal{C}$-loss as well as the coherence function have several desirable properties in the context of large-margin classifiers.

In this paper we show how the coherence function can be used to develop an effective approach to estimating the class probability of the conventional binary SVM. Platt (1999) first exploited a sigmoid link function to map the SVM outputs into probabilities, while Sollich (2002) used logarithmic scoring rules (Bernardo and Smith, 1994) to transform the hinge loss into the negative of a conditional log-likelihood (i.e., a predictive class probability). Recently, Wang et al. (2008) developed an interval estimation method. Theoretically, Steinwart (2003) and Bartlett and Tewari (2007) showed that the class probability can be asymptotically estimated by replacing the hinge loss with a differentiable loss. Our approach also appeals to asymptotics to derive a method for estimating the class probability of the conventional binary SVM.

Using the $\mathcal{C}$-loss, we devise new large-margin classifiers which we refer to as *$\mathcal{C}$-learning*. To maintain sparseness, we use the elastic-net penalty in addition to $\mathcal{C}$-learning. We in particular propose two versions. The first version is based on reproducing kernel Hilbert spaces (RKHSs) and it can automatically select the number of support vectors via penalization. The second version focuses on the selection of features again via penalization. The classifiers are trained by coordinate descent algorithms developed by Friedman et al. (2010) for generalized linear models.

The rest of this paper is organized as follows. In Section 2 we summarize the fundamental basis of large-margin classification. Section 3 presents $\mathcal{C}$-loss functions and their mathematical properties. Section 4 gives a method for class probability estimation of the conventional SVM outputs and Section 5 studies our $\mathcal{C}$-learning algorithms. We conduct an experimental analysis in Section 6 and conclude our work in Section 7. All proofs are deferred to the appendix.

2. Large-Margin Classifiers

We consider a *binary* classification problem with a set of training data $\mathcal{T} = \{\mathbf{x}_i, y_i\}_1^n$, where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ is an input vector and $y_i \in \mathcal{Y} = \{1, -1\}$ is the corresponding class label. Our goal is to find a decision function $f(\mathbf{x})$ over a measurable function class $\mathcal{F}$. Once such an $f(\mathbf{x})$ is obtained, the classification rule is $y = \text{sign}(f(\mathbf{x}))$ where $\text{sign}(a) = 1, 0, -1$ according to $a > 0$, $a = 0$ or $a < 0$. Thus, we have that $\mathbf{x}$ is misclassified if and only if $yf(\mathbf{x}) \leq 0$ (here we ignore the case that $f(\mathbf{x}) = 0$).

Let $\eta(\mathbf{x}) = \Pr(Y = 1|X = \mathbf{x})$ be the conditional probability of class 1 given $\mathbf{x}$ and let $P(X,Y)$ be the probability distribution over $\mathcal{X} \times \mathcal{Y}$. For a measurable decision function $f(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$, the expected error at $\mathbf{x}$ is then defined by

$$\Psi(f(\mathbf{x})) = \mathbb{E}(I_{[Yf(X)\leq 0]}|X = \mathbf{x}) = I_{[f(\mathbf{x})\leq 0]}\eta(\mathbf{x}) + I_{[f(\mathbf{x})>0]}(1 - \eta(\mathbf{x})),$$

where $I_{[\#]} = 1$ if # is true and 0 otherwise. The generalization error is

$$\Psi_f = \mathbb{E}_P I_{[Yf(X)\leq 0]} = \mathbb{E}_X \left[I_{[f(X)\leq 0]}\eta(X) + I_{[f(X)>0]}(1 - \eta(X)) \right],$$

where the expectation $\mathbb{E}_P$ is taken with respect to the distribution $P(X,Y)$ and $\mathbb{E}_X$ denotes the expectation over the input data $\mathcal{X}$. The optimal Bayes error is $\hat{\Psi} = \mathbb{E}_P I_{[Y(2\eta(X)-1)\leq 0]}$, which is the minimum of Ψ_f with respect to measurable functions f.

A classifier is a classification algorithm which finds a measurable function $f_{\mathcal{T}} : \mathcal{X} \to \mathbb{R}$ based on the training data $\mathcal{T}$. We assume that the $(\mathbf{x}_i, y_i)$ in $\mathcal{T}$ are independent and identically distributed from $P(X,Y)$. A classifier is said to be *universally consistent* if

$$\lim_{n\to\infty} \Psi_{f_{\mathcal{T}}} = \hat{\Psi}$$

holds in probability for any distribution P on $\mathcal{X}\times\mathcal{Y}$. It is strongly universally consistent if the condition $\lim_{n\to\infty} \Psi_{f_{\mathcal{T}}} = \hat{\Psi}$ is satisfied almost surely (Steinwart, 2005).

The empirical generalization error on the training data $\mathcal{T}$ is given by

$$\Psi_{emp} = \frac{1}{n} \sum_{i=1}^{n} I_{[y_i f(\mathbf{x}_i)\leq 0]}.$$

Given that the empirical generalization error Ψ_{emp} is equal to its minimum value zero when all training data are correctly classified, we wish to use Ψ_{emp} as a basis for devising classification algorithms. However, the corresponding minimization problem is computationally intractable.

2.1 Surrogate Losses

A wide variety of classifiers can be understood as minimizers of a continuous *surrogate loss* function $\phi(yf(\mathbf{x}))$, which upper bounds the 0-1 loss $I_{[yf(\mathbf{x})\leq 0]}$. Corresponding to $\Psi(f(\mathbf{x}))$ and Ψ_f, we denote $R(f(\mathbf{x})) = \phi(f(\mathbf{x}))\eta(\mathbf{x}) + \phi(-f(\mathbf{x}))(1 - \eta(\mathbf{x}))$ and

$$R_f = \mathbb{E}_P[\phi(Yf(X))] = \mathbb{E}_X \left[\phi(f(X))\eta(X) + \phi(-f(X))(1 - \eta(X)) \right].$$

For convenience, we assume that $\eta \in [0, 1]$ and define the notation

$$R(\eta, f) = \eta\phi(f) + (1 - \eta)\phi(-f).$$

Exponential Loss	Logit Loss	Hinge Loss	Squared Hinge Loss
$\exp[-yf(\mathbf{x})/2]$	$\log[1+\exp(-yf(\mathbf{x}))]$	$[1-yf(\mathbf{x})]_+$	$([1-yf(\mathbf{x})]_+)^2$

Table 1: Surrogate losses for margin-based classifier.

The surrogate ϕ is said to be Fisher consistent, if for every $\eta \in [0,1]$ the minimizer of $R(\eta, f)$ with respect to f exists and is unique and the minimizer (denoted $\hat{f}(\eta)$) satisfies $\text{sign}(\hat{f}(\eta)) = \text{sign}(\eta - 1/2)$ (Lin, 2002; Bartlett et al., 2006; Zhang, 2004). Since $\text{sign}(u) = 0$ is equivalent to $u = 0$, we have that $\hat{f}(1/2) = 0$. Substituting $\hat{f}(\eta)$ into $R(\eta, f)$, we also define the following notation:

$$\hat{R}(\eta) = \inf_f R(\eta, f) = R(\eta, \hat{f}(\eta)).$$

The difference between $R(\eta, f)$ and $\hat{R}(\eta)$ is

$$\triangle R(\eta, f) = R(\eta, f) - \hat{R}(\eta) = R(\eta, f) - R(\eta, \hat{f}(\eta)).$$

When regarding $f(\mathbf{x})$ and $\eta(\mathbf{x})$ as functions of $\mathbf{x}$, it is clear that $\hat{f}(\eta(\mathbf{x}))$ is the minimizer of $R(f(\mathbf{x}))$ among all measurable function class $\mathcal{F}$. That is,

$$\hat{f}(\eta(\mathbf{x})) = \operatorname*{argmin}_{f(\mathbf{x})\in\mathcal{F}} R(f(\mathbf{x})).$$

In this setting, the difference between R_f and $\mathbb{E}_X[R(\hat{f}(\eta(X)))]$ (denoted $R_{\hat{f}}$) is given by

$$\triangle R_f = R_f - R_{\hat{f}} = \mathbb{E}_X \triangle R(\eta(X), f(X)).$$

If $\hat{f}(\eta)$ is invertible, then the inverse function $\hat{f}^{-1}(f(\mathbf{x}))$ over $\mathcal{F}$ can be regarded as a class-conditional probability estimate given that $\eta(\mathbf{x}) = \hat{f}^{-1}(\hat{f}(\mathbf{x}))$. Moreover, Zhang (2004) showed that $\triangle R_f$ is the expected distance between the conditional probability $\hat{f}^{-1}(f(\mathbf{x}))$ and the true conditional probability $\eta(\mathbf{x})$. Thus, minimizing R_f is equivalent to minimizing the expected distance between $\hat{f}^{-1}(f(\mathbf{x}))$ and $\eta(\mathbf{x})$.

Table 1 lists four common surrogate functions used in large-margin classifiers. Here $[u]_+ = \max\{u, 0\}$ is a so-called hinge function and $([u]_+)^2 = (\max\{u,0\})^2$ is a squared hinge function which is used for developing the ℓ_2-SVM (Cristianini and Shawe-Taylor, 2000). Note that we typically scale the logit loss to equal 1 at $yf(\mathbf{x}) = 0$. These functions are convex and the upper bounds of $I_{[yf(\mathbf{x})\le 0]}$. Moreover, they are Fisher consistent. In particular, the following result has been established by Friedman et al. (2000) and Lin (2002).

Proposition 1 *Assume that* $0 < \eta(\mathbf{x}) < 1$ *and* $\eta(\mathbf{x}) \neq 1/2$. *Then, the minimizers of* $\mathbb{E}(\exp[-Yf(X)/2]|X = \mathbf{x})$ *and* $\mathbb{E}(\log[1+\exp(-Yf(X))]|X = \mathbf{x})$ *are both* $\hat{f}(\mathbf{x}) = \log\frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}$, *the minimizer of* $\mathbb{E}\big([1 - Yf(X)]_+|X = \mathbf{x}\big)$ *is* $\hat{f}(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$, *and the minimizer of* $\mathbb{E}\big(([1 - Yf(X)]_+)^2|X = \mathbf{x}\big)$ *is* $\hat{f}(\mathbf{x}) = 2\eta(\mathbf{x}) - 1$.

When the exponential or logit loss function is used, $\hat{f}^{-1}(f(\mathbf{x}))$ exists. It is clear that $\eta(\mathbf{x}) = \hat{f}^{-1}(\hat{f}(\mathbf{x}))$. For any $f(\mathbf{x}) \in \mathcal{F}$, we denote the inverse function by $\tilde{\eta}(\mathbf{x})$, which is

$$\tilde{\eta}(\mathbf{x}) = \hat{f}^{-1}(f(\mathbf{x})) = \frac{1}{1+\exp(-f(\mathbf{x}))}.$$

Unfortunately, the minimization of the hinge loss (which is the basis of the SVM) does not yield a class probability estimate (Lin et al., 2002).

2.2 The Regularization Approach

Given a surrogate loss function ϕ, a large-margin classifier typically solves the following optimization problem:

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}\phi(y_i f(\mathbf{x}_i)) + \gamma J(h),$$

where $f(\mathbf{x}) = \alpha + h(\mathbf{x})$, $J(h)$ is a regularization term to penalize model complexity and γ is the degree of penalization.

Suppose that $f = \alpha + h \in (\{1\} + \mathcal{H}_K)$ where $\mathcal{H}_K$ is a reproducing kernel Hilbert space (RKHS) (Wahba, 1990) induced by a reproducing kernel $K(\cdot,\cdot) : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$. Finding $f(\mathbf{x})$ is then formulated as a regularization problem of the form

$$\min_{f\in\mathcal{H}_K}\left\{\frac{1}{n}\sum_{i=1}^{n}\phi(y_i f(\mathbf{x}_i)) + \frac{\gamma}{2}\|h\|^2_{\mathcal{H}_K}\right\}, \tag{1}$$

where $\|h\|^2_{\mathcal{H}_K}$ is the RKHS norm. By the representer theorem, the solution of (1) is of the form

$$f(\mathbf{x}_i) = \alpha + \sum_{j=1}^{n}\beta_j K(\mathbf{x}_i,\mathbf{x}_j) = \alpha + \mathbf{k}_i'\beta, \tag{2}$$

where $\beta = (\beta_1,\ldots,\beta_n)'$ and $\mathbf{k}_i = (K(\mathbf{x}_i,\mathbf{x}_1),\ldots,K(\mathbf{x}_i,\mathbf{x}_n))'$. Noticing that $\|h\|^2_{\mathcal{H}_K} = \sum_{i,j=1}^{n} K(\mathbf{x}_i,\mathbf{x}_j)\beta_i\beta_j$ and substituting (2) into (1), we obtain the minimization problem with respect to α and β as

$$\min_{\alpha,\beta}\left\{\frac{1}{n}\sum_{i=1}^{n}\phi(y_i(\alpha+\mathbf{k}_i'\beta)) + \frac{\gamma}{2}\beta'\mathbf{K}\beta\right\},$$

where $\mathbf{K} = [\mathbf{k}_1,\ldots,\mathbf{k}_n]$ is the $n\times n$ kernel matrix. Since $\mathbf{K}$ is symmetric and positive semidefinite, the term $\beta'\mathbf{K}\beta$ is in fact an empirical RKHS norm on the training data.

In particular, the conventional SVM defines the surrogate $\phi(\cdot)$ as the hinge loss and solves the following optimization problem:

$$\min_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}[1-y_i(\alpha+\mathbf{k}_i'\beta)]_+ + \frac{\gamma}{2}\beta'\mathbf{K}\beta. \tag{3}$$

In this paper, we are especially interested in *universal kernels*, namely, kernels whose induced RKHS is dense in the space of continuous functions over $\mathcal{X}$ (Steinwart, 2001). The Gaussian RBF kernel is such an example.

2.3 Methods for Class Probability Estimation of SVMs

Let $\hat{f}(\mathbf{x})$ be the solution of the SVM problem in (3). In an attempt to address the problem of class probability estimation for SVMs, Sollich (2002) proposed a class probability estimate

$$\hat{\eta}(\mathbf{x}) = \begin{cases} \frac{1}{1+\exp(-2\hat{f}(\mathbf{x}))} & \text{if } |\hat{f}(\mathbf{x})| \le 1, \\ \frac{1}{1+\exp[-(\hat{f}(\mathbf{x})+\mathrm{sign}(\hat{f}(\mathbf{x})))]} & \text{otherwise.} \end{cases}$$

This class probability was also used in the derivation of a so-called complete SVM studied by Mallick et al. (2005).

Another proposal for obtaining class probabilities from SVM outputs was developed by Platt (1999), who employed a post-processing procedure based on the parametric formula

$$\hat{\eta}(\mathbf{x}) = \frac{1}{1+\exp(A\hat{f}(\mathbf{x})+B)},$$

where the parameters A and B are estimated via the minimization of the empirical cross-entropy error over the training data set.

Wang et al. (2008) proposed a nonparametric form obtained from training a sequence of weighted classifiers:

$$\min_{f\in\mathcal{F}} \frac{1}{n}\left\{(1-\pi_j)\sum_{y_i=1}[1-y_i f(\mathbf{x}_i)]_+ + \pi_j \sum_{y_i=-1}[1-y_i f(\mathbf{x}_i)]_+\right\} + \gamma J(h) \tag{4}$$

for $j = 1,\ldots,m+1$ such that $0 = \pi_1 < \cdots < \pi_{m+1} = 1$. Let $\hat{f}_{\pi_j}(\mathbf{x})$ be the solution of (4). The estimated class probability is then $\hat{\eta}(\mathbf{x}) = \frac{1}{2}(\pi_* + \pi^*)$ where $\pi_* = \min\{\pi_j : \text{sign}(\hat{f}_{\pi_j}(\mathbf{x})) = -1\}$ and $\pi^* = \max\{\pi_j : \text{sign}(\hat{f}_{\pi_j}(\mathbf{x})) = 1\}$.

Additional contributions are due to Steinwart (2003) and Bartlett and Tewari (2007). These authors showed that the class probability can be asymptotically estimated by replacing the hinge loss with various differentiable losses.

3. Coherence Functions

In this section we present a smooth and Fisher-consistent majorization loss, which bridges the hinge loss and the logit loss. We will see that one limit of this loss is equal to the hinge loss. Thus, it is applicable to the asymptotical estimate of the class probability for the conventional SVM as well as the construction of margin-based classifiers, which will be presented in Section 4 and Section 5.

3.1 Definition

Under the $0-1$ loss the misclassification costs are specified to be one, but it is natural to set the misclassification costs to be a positive constant $u > 0$. The empirical generalization error on the training data is given in this case by

$$\frac{1}{n}\sum_{i=1}^{n} uI_{[y_i f(\mathbf{x}_i)\le 0]},$$

where $u > 0$ is a constant that represents the misclassification cost. In this setting we can extend the hinge loss as

$$H_u(yf(\mathbf{x})) = [u - yf(\mathbf{x})]_+.$$

It is clear that $H_u(yf(\mathbf{x})) \ge uI_{[yf(\mathbf{x})\le 0]}$. This implies that $H_u(yf(\mathbf{x}))$ is a majorization of $uI_{[yf(\mathbf{x})\le 0]}$.

We apply the maximum entropy principle to develop a smooth surrogate of the hinge loss $[u-z]_+$. In particular, noting that $[u-z]_+ = \max\{u-z,\, 0\}$, we maximize $w(u-z)$ with respect to $w \in (0,1)$ under the entropy constraint; that is,

$$\max_{w\in(0,1)} \Big\{F = w(u-z) - \rho\big[w\log w + (1-w)\log(1-w)\big]\Big\},$$

where $-[w\log w + (1-w)\log(1-w)]$ is the entropy and $\rho > 0$, a Lagrange multiplier, plays the role of temperature in thermodynamics.

The maximum of F is

$$V_{\rho,u}(z) = \rho \log\left[1+\exp\frac{u-z}{\rho}\right] \tag{5}$$

at $w = \exp((u-z)/\rho)/[1+\exp((u-z)/\rho)]$. We refer to functions of this form as *coherence functions* because their properties (detailed in the next subsection) are similar to statistical mechanical properties of deterministic annealing (Rose et al., 1990).

We also consider a scaled variant of $V_{\rho,u}(z)$:

$$C_{\rho,u}(z) = \frac{u}{\log[1+\exp(u/\rho)]}\log\left[1+\exp\frac{u-z}{\rho}\right],\ \rho>0,\ u>0, \tag{6}$$

which has the property that $C_{\rho,u}(z) = u$ when $z = 0$. Recall that u as a misclassification cost should be specified as a positive value. However, both $C_{\rho,0}(z)$ and $V_{\rho,0}(z)$ are well defined mathematically. Since $C_{\rho,0}(z) = 0$ is a trivial case, we always assume that $u > 0$ for $C_{\rho,u}(z)$ here and later. In the binary classification problem, z is defined as $yf(\mathbf{x})$. In the special case that $u = 1$, $C_{\rho,1}(yf(\mathbf{x}))$ can be regarded as a smooth alternative to the SVM hinge loss $[1-yf(\mathbf{x})]_+$. We refer to $C_{\rho,u}(yf(\mathbf{x}))$ as $\mathcal{C}$-losses.

It is worth noting that $V_{1,0}(z)$ is the logistic function and $V_{\rho,0}(z)$ has been proposed by Zhang and Oles (2001) for binary logistic regression. We keep in mind that $u \geq 0$ for $V_{\rho,u}(z)$ through this paper.

3.2 Properties

It is obvious that $C_{\rho,u}(z)$ and $V_{\rho,u}$ are infinitely smooth with respect to z. Moreover, the first-order and second-order derivatives of $C_{\rho,u}(z)$ with respect to z are given as

$$C'_{\rho,u}(z) = -\frac{u}{\rho\log[1+\exp(u/\rho)]}\frac{\exp\frac{u-z}{\rho}}{1+\exp\frac{u-z}{\rho}},$$
$$C''_{\rho,u}(z) = \frac{u}{\rho^2\log[1+\exp(u/\rho)]}\frac{\exp\frac{u-z}{\rho}}{(1+\exp\frac{u-z}{\rho})^2}.$$

Since $C''_{\rho,u}(z) > 0$ for any $z \in \mathbb{R}$, $C_{\rho,u}(z)$ as well as $V_{\rho,u}(z)$ are strictly convex in z, for fixed $\rho > 0$ and $u > 0$.

We now investigate relationships among the coherence functions and hinge losses. First, we have the following properties.

Proposition 2 *Let $V_{\rho,u}(z)$ and $C_{\rho,u}(z)$ be defined by (5) and (6). Then,*

(i) $u\times I_{[z\leq 0]} \leq [u-z]_+ \leq V_{\rho,u}(z) \leq \rho\log 2+[u-z]_+$;

(ii) $\frac{1}{2}(u-z) \leq V_{\rho,u}(z) - \rho\log 2$;

(iii) $\lim_{\rho\to 0} V_{\rho,u}(z) = [u-z]_+$ *and* $\lim_{\rho\to\infty} V_{\rho,u}(z) - \rho\log 2 = \frac{1}{2}(u-z)$;

(iv) $u\times I_{[z\leq 0]} \leq C_{\rho,u}(z) \leq V_{\rho,u}(z)$;

(v) $\lim_{\rho\to 0} C_{\rho,u}(z) = [u-z]_+$ *and* $\lim_{\rho\to\infty} C_{\rho,u}(z) = u$, *for* $u > 0$.

As a special case of $u = 1$, we have $C_{\rho,1}(z) \geq I_{[z\leq 0]}$. Moreover, $C_{\rho,1}(z)$ approaches $(1-z)_+$ as $\rho \to 0$. Thus, $C_{\rho,1}(z)$ is a majorization of $I_{[z\leq 0]}$.

As we mentioned earlier, $V_{\rho,0}(z)$ are used to devise logistic regression models. We can see from Proposition 2 that $V_{\rho,0}(z) \geq [-z]_+$, which implies that a logistic regression model is possibly no longer a large-margin classifier. Interestingly, however, we consider a variant of $V_{\rho,u}(z)$ as

$$L_{\rho,u}(z) = \frac{1}{\log(1+\exp(u/\rho))} \log\left[1+\exp((u-z)/\rho)\right],\ \rho > 0,\ u \geq 0,$$

which always satisfies that $L_{\rho,u}(z) \geq I_{[z\leq 0]}$ and $L_{\rho,u}(0) = 1$, for any $u \geq 0$. Thus, the $L_{\rho,u}(z)$ for $\rho > 0$ and $u \geq 0$ are majorizations of $I_{[z\leq 0]}$. In particular, $L_{\rho,1}(z) = C_{\rho,1}(u)$ and $L_{1,0}(z)$ is the logit function.

In order to explore the relationship of $C_{\rho,u}(z)$ with $(u-z)_+$, we now consider some properties of $L_{\rho,u}(z)$ when regarding it respectively as a function of ρ and of u.

Proposition 3 *Assume $\rho > 0$ and $u \geq 0$. Then,*

(i) *$L_{\rho,u}(z)$ is a decreasing function in ρ if $z < 0$, and it is an increasing function in ρ if $z \geq 0$;*

(ii) *$L_{\rho,u}(z)$ is a decreasing function in u if $z < 0$, and it is an increasing function in u if $z \geq 0$.*

Results similar to those in Proposition 3-(i) also apply to $C_{\rho,u}(z)$ because of $C_{\rho,u}(z) = uL_{\rho,u}(z)$. Then, according to Proposition 2-(v), we have that $u = \lim_{\rho\to+\infty} C_{\rho,u}(z) \leq C_{\rho,u}(z) \leq \lim_{\rho\to 0} C_{\rho,u}(z) = (u-z)_+$ if $z < 0$ and $(u-z)_+ = \lim_{\rho\to 0} C_{\rho,u}(z) \leq C_{\rho,u}(z) \leq \lim_{\rho\to+\infty} C_{\rho,u}(z) = u$ if $z \geq 0$. It follows from Proposition 3-(ii) that $C_{\rho,1}(z) = L_{\rho,1}(z) \leq L_{\rho,0}(z)$ if $z < 0$ and $C_{\rho,1}(z) = L_{\rho,1}(z) \geq L_{\rho,0}(z)$ if $z \geq 0$. In addition, it is easily seen that $(1-z)_+ \geq ((1-z)_+)^2$ if $z \geq 0$ and $(1-z)_+ \leq ((1-z)_+)^2$ otherwise. We now obtain the following proposition:

Proposition 4 *Assume $\rho > 0$. Then, $C_{\rho,1}(z) \leq \min\left\{L_{\rho,0}(z),\ [1-z]_+,\ ([1-z]_+)^2\right\}$ if $z < 0$, and $C_{\rho,1}(z) \geq \max\left\{L_{\rho,0}(z),\ [1-z]_+,\ ([1-z]_+)^2\right\}$ if $z \geq 0$.*

This proposition is depicted in Figure 1. Owing to the relationships of the C-loss $C_{\rho,1}(yf(x))$ with the hinge and logit losses, it is potentially useful in devising new large-margin classifiers.

We now turn to the derivatives of $C_{\rho,u}(z)$ and $(u-z)_+$. It is immediately verified that $-1 \leq C'_{\rho,u}(z) \leq 0$. Moreover, we have

$$\lim_{\rho\to 0} C'_{\rho,u}(z) = \lim_{\rho\to 0} V'_{\rho,u}(z) = \begin{cases} 0 & z > u, \\ -\frac{1}{2} & z = u, \\ -1 & z < u. \end{cases}$$

Note that $(u-z)'_+ = -1$ if $z < u$ and $(u-z)'_+ = 0$ if $z > u$. Furthermore, $\partial(u-z)_+|_{z=u} = [-1,0]$ where $\partial(u-z)_+|_{z=u}$ denotes the subdifferential of $(u-z)_+$ at $z = u$. Hence,

Proposition 5 *For a fixed $u > 0$, we have that $\lim_{\rho\to 0} C'_{\rho,u}(z) \in \partial(u-z)_+$.*

This proposition again establishes a connection of the hinge loss with the limit of $C_{\rho,u}(z)$ at $\rho = 0$. Furthermore, we obtain from Propositions 2 and 5 that $\partial(u-z)_+ = \partial\lim_{\rho\to 0} C_{\rho,u}(z) \ni \lim_{\rho\to 0} \partial C_{\rho,u}(z)$.

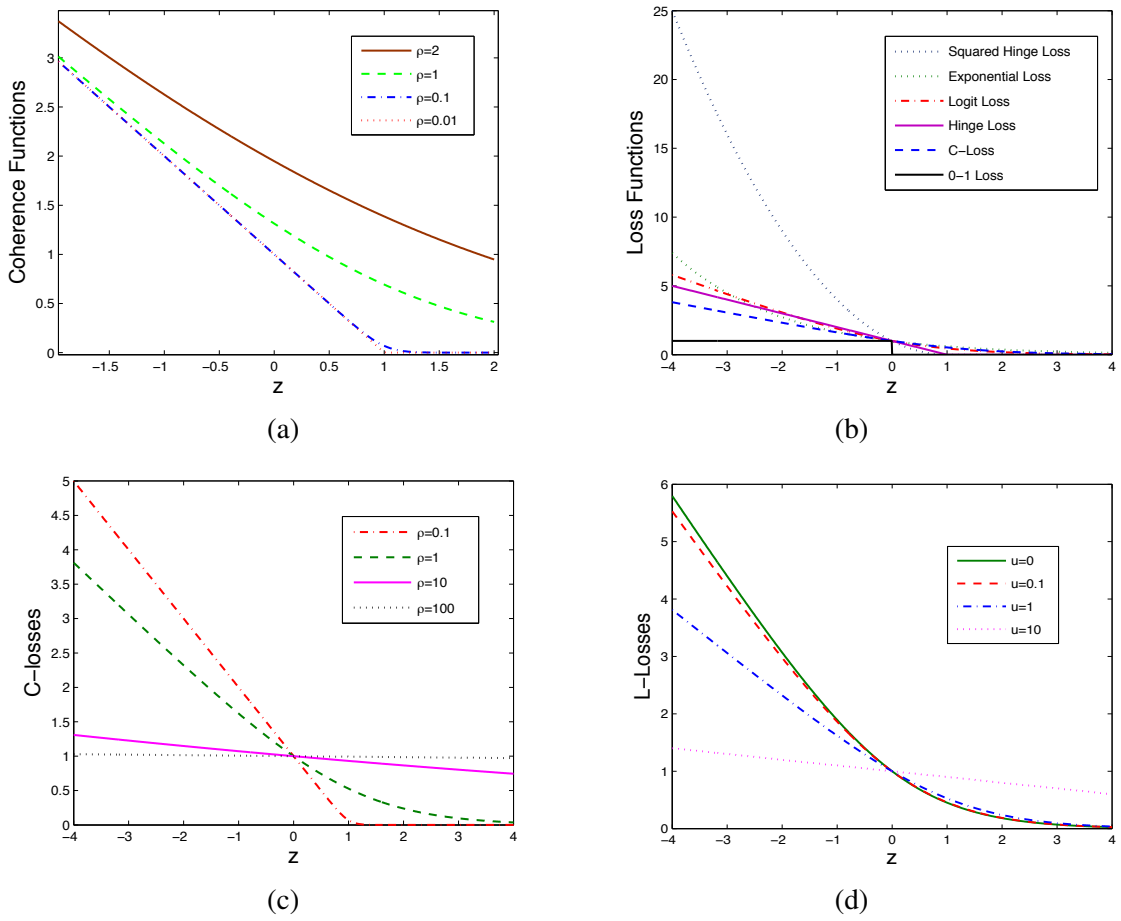


Figure 1: These functions are regarded as a function of $z = yf(\mathbf{x})$. (a) Coherence functions $V_{\rho,1}(z)$ with $\rho = 0.01$, $\rho = 0.1$, $\rho = 1$ and $\rho = 2$. (b) A variety of majorization loss functions, *C-loss*: $C_{1,1}(z)$; *Logit loss*: $L_{1,0}(z)$; *Exponential loss*: $\exp(-z/2)$; *Hinge loss*: $[1-z]_+$; *Squared Hinge Loss*: $([1-z]_+)^2$. (c) $C_{\rho,1}(z)$ (or $L_{\rho,1}(z)$) with $\rho = 0.1$, $\rho = 1$, $\rho = 10$ and $\rho = 100$ (see Proposition 3-(i)). (d) $L_{1,u}(z)$ with $u = 0$, $u = 0.1$, $u = 1$ and $u = 10$ (see Proposition 3-(ii)).

3.3 Consistency in Classification Methods

We now apply the coherence function to the development of classification methods. Recall that $C'_{\rho,u}(0)$ exists and is negative. Thus, the C-loss $C_{\rho,u}(yf(x))$ is Fisher-consistent (or classification calibrated) (Bartlett et al., 2006). In particular, we have the following theorem.

Theorem 6 *Assume* $0 < \eta < 1$ *and* $\eta \neq \frac{1}{2}$. *Consider the optimization problem*

$$\min_{f \in \mathbb{R}} R(f,\eta) := V_{\rho,u}(f)\eta + V_{\rho,u}(-f)(1-\eta)$$

for fixed $\rho > 0$ *and* $u \geq 0$*. Then, the minimizer is unique and is given by*

$$f_*(\eta) = \rho \log \frac{(2\eta-1)\exp(\frac{u}{\rho}) + \sqrt{(1-2\eta)^2\exp(\frac{2u}{\rho}) + 4\eta(1-\eta)}}{2(1-\eta)}. \tag{7}$$

Moreover, we have $f_* > 0$ *if and only if* $\eta > 1/2$*. Additionally, the inverse function* $f_*^{-1}(f)$ *exists and it is given by*

$$\tilde{\eta}(f) := f_*^{-1}(f) = \frac{1+\exp(\frac{f-u}{\rho})}{1+\exp(-\frac{u+f}{\rho}) + 1+\exp(\frac{f-u}{\rho})}, \text{ for } f \in \mathbb{R}. \tag{8}$$

The minimizer $f_*(\mathbf{x})$ of $R(f(\mathbf{x})) := \mathbb{E}(V_{\rho,u}(Yf(X))|X = \mathbf{x})$ and its inverse $\tilde{\eta}(\mathbf{x})$ are immediately obtained by replacing f with $f(\mathbf{x})$ in (7) and (8). Since for $u > 0$ the minimizers of $\mathbb{E}(C_{\rho,u}(Yf(X))|X = \mathbf{x})$ and $\mathbb{E}(V_{\rho,u}(Yf(X))|X = \mathbf{x})$ are the same, this theorem shows that $C_\rho(yf(\mathbf{x}), u)$ is also Fisher-consistent. We see from Theorem 6 that the explicit expressions of $f_*(\mathbf{x})$ and its inverse $\tilde{\eta}(\mathbf{x})$ exist. In the special case that $u = 0$, we have $f_*(\mathbf{x}) = \rho \log \frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}$ and $\tilde{\eta}(\mathbf{x}) = \frac{1}{1+\exp(-f(\mathbf{x})/\rho)}$. Furthermore, when $\rho = 1$, as expected, we recover logistic regression. In other words, the result is identical with that in Proposition 1 for logistic regression.

We further consider properties of $f_*(\eta)$. In particular, we have the following proposition.

Proposition 7 *Let* $f_*(\eta)$ *be defined by (7). Then,*

(i) $\text{sign}(f_*(\eta)) = \text{sign}(\eta - 1/2)$.

(ii) $\lim_{\rho\to 0} f_*(\eta) = u \times \text{sign}(\eta - 1/2)$.

(iii) $f_*'(\eta) = \frac{df_*(\eta)}{d\eta} \geq \frac{\rho}{\eta(1-\eta)}$ *with equality if and only if* $u = 0$.

Proposition 7-(i) shows that the classification rule with $f_*(\mathbf{x})$ is equivalent to the Bayes rule. In the special case that $u = 1$, we have from Proposition 7-(ii) that $\lim_{\rho\to 0} f_*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$. This implies that the current $f_*(\mathbf{x})$ approaches the solution of $\mathbb{E}((1 - Yf(X))_+|X = \mathbf{x})$, which corresponds to the conventional SVM method (see Proposition 1).

We now treat $\tilde{\eta}(f)$ as a function of ρ. The following proposition is easily proven.

Proposition 8 *Let* $\tilde{\eta}(f)$ *be defined by (8). Then, for fixed* $f \in \mathbb{R}$ *and* $u > 0$, $\lim_{\rho\to\infty} \tilde{\eta}(f) = \frac{1}{2}$ *and*

$$\lim_{\rho\to 0} \tilde{\eta}(f) = \begin{cases} 1 & \text{if } f > u, \\ \frac{2}{3} & \text{if } f = u, \\ \frac{1}{2} & \text{if } -u < f < u, \\ \frac{1}{3} & \text{if } f = -u, \\ 0 & \text{if } f < -u. \end{cases}$$

As we discuss in the previous subsection, $V_{\rho,u}(z)$ is obtained when $w = \exp((u-z)/\rho)/(1+\exp((u-z)/\rho))$ by using the maximum entropy principle. Let $z = yf(\mathbf{x})$. We further write w as $w_1(f) = 1/[1+\exp((f-u)/\rho)]$ when $y = 1$ and w as $w_2(f) = 1/[1+\exp(-(f+u)/\rho)]$ when $y = -1$.

We now explore the relationship of $\tilde{\eta}(f)$ with $w_1(f)$ and $w_2(f)$. Interestingly, we first find that

$$\tilde{\eta}(f) = \frac{w_2(f)}{w_1(f)+w_2(f)}.$$

It is easily proven that $w_1(f)+w_2(f) \geq 1$ with equality if and only if $u=0$. We thus have that $\tilde{\eta}(f) \leq w_2(f)$, with equality if and only if $u=0$; that is, the loss becomes logit function $V_{\rho,0}(z)$. Note that $w_2(f)$ represents the probability of the event $\{u+f>0\}$ and $\tilde{\eta}(f)$ represents the probability of the event $\{f>0\}$. Since the event $\{f>0\}$ is a subset of the event $\{u+f>0\}$, we have $\tilde{\eta}(f) \leq w_2(f)$. Furthermore, the statement that $\tilde{\eta}(f) = w_2(f)$ if and only if $u=0$ is equivalent to $\{u+f>0\} = \{f>0\}$ if and only if $u=0$. This implies that only the logit loss induces $\tilde{\eta}(f) = w_2(f)$.

As discussed in Section 2.1, $\tilde{\eta}(\mathbf{x})$ can be regarded as a reasonable estimate of the true class probability $\eta(\mathbf{x})$. Recall that $\triangle R(\eta, f) = R(\eta, f) - R(\eta, f_*(\eta))$ and $\triangle R_f = \mathbb{E}_X[\triangle R(\eta(X), f(X))]$ such that $\triangle R_f$ can be viewed as the expected distance between $\tilde{\eta}(\mathbf{x})$ and $\eta(\mathbf{x})$.

For an arbitrary fixed $f \in \mathbb{R}$, we have

$$\triangle R(\eta, f) = R(\eta, f) - R(\eta, f_*(\eta)) = \eta\rho \log \frac{1+\exp\frac{u-f}{\rho}}{1+\exp\frac{u-f_*(\eta)}{\rho}} + (1-\eta)\rho \log \frac{1+\exp\frac{u+f}{\rho}}{1+\exp\frac{u+f_*(\eta)}{\rho}}.$$

The first-order derivative of $\triangle R(\eta, f)$ with respect to η is

$$\frac{d\triangle R(\eta, f)}{d\eta} = \rho \log \frac{1+\exp\frac{u-f}{\rho}}{1+\exp\frac{u-f_*(\eta)}{\rho}} - \rho \log \frac{1+\exp\frac{u+f}{\rho}}{1+\exp\frac{u+f_*(\eta)}{\rho}}.$$

The Karush-Kuhn-Tucker (KKT) condition for the minimization problem is as follows:

$$\eta \frac{\exp\frac{u-f_*(\eta)}{\rho}}{1+\exp\frac{u-f_*(\eta)}{\rho}} + (1-\eta)\frac{\exp\frac{u+f_*(\eta)}{\rho}}{1+\exp\frac{u+f_*(\eta)}{\rho}} = 0,$$

and the second-order derivative of $\triangle R(\eta, f)$ with respect to η is given by

$$\frac{d^2\triangle R(\eta, f)}{d\eta^2} = \Big(\frac{1}{1+\exp(-\frac{u-f_*(\eta)}{\rho})} + \frac{1}{1+\exp(-\frac{u+f_*(\eta)}{\rho})}\Big) f_*'(\eta) = \big[w_1(f_*(\eta)) + w_2(f_*(\eta))\big] f_*'(\eta).$$

According to Proposition 7-(iii) and using $w_1(f_*(\eta)) + w_2(f_*(\eta)) \geq 1$, we have

$$\frac{d^2\triangle R(\eta, f)}{d\eta^2} \geq \frac{\rho}{\eta(1-\eta)},$$

with equality if and only if $u=0$. This implies $\frac{d^2\triangle R(\eta,f)}{d\eta^2} > 0$. Thus, for a fixed f, $\triangle R(\eta, f)$ is strictly convex in η. Subsequently, we have that $\triangle R(\eta, f) \geq 0$ with equality $\eta = \tilde{\eta}$, or equivalently, $f = f_*$.

Using the Taylor expansion of $\triangle R(\eta, f)$ at $\tilde{\eta} := \tilde{\eta}(f) = f_*^{-1}(f)$, we thus obtain a lower bound for $\triangle R(\eta, f)$; namely,

$$\begin{aligned}\triangle R(\eta, f) &= \triangle R(\tilde{\eta}, f) - \frac{d\triangle R(\tilde{\eta}, f)}{d\eta}(\eta - \tilde{\eta}) + \frac{1}{2}\frac{d^2\triangle R(\bar{\eta}, f)}{d\eta^2}(\eta - \tilde{\eta})^2 \\ &= \frac{1}{2}\frac{d^2\triangle R(\bar{\eta}, f)}{d\eta^2}(\eta - \tilde{\eta})^2 \geq \frac{\rho}{2\bar{\eta}(1-\bar{\eta})}(\eta - \tilde{\eta})^2 \geq 2\rho(\eta - \tilde{\eta})^2,\end{aligned}$$

where $\bar{\eta} \in (\tilde{\eta}, \eta) \subset [0,1]$. In particular, we have that $\triangle R(\eta, 0) \geq 2\rho(\eta - 0.5)^2$. According to Theorem 2.1 and Corollary 3.1 in Zhang (2004), the following theorem is immediately established.

Theorem 9 *Let* $\varepsilon_1 = \inf_{f(\cdot)\in\mathcal{F}} \mathbb{E}_X[\triangle R(\eta(X), f(X))]$, *and let* $f_*(\mathbf{x}) \in \mathcal{F}$ *such that*

$$\mathbb{E}_X[R(\eta(X), f_*(X))] \leq \inf_{f(\cdot)\in\mathcal{F}} \mathbb{E}_X[R(\eta(X), f(X))] + \varepsilon_2$$

for $\varepsilon_2 \geq 0$. *Then for* $\varepsilon = \varepsilon_1 + \varepsilon_2$,

$$\triangle R_{f_*} = \mathbb{E}_X[\triangle R(\eta(X), f_*(X))] \leq \varepsilon$$

and

$$\Psi_{f_*} \leq \hat{\Psi} + \left(2\varepsilon/\rho\right)^{1/2},$$

where $\Psi_{f_*} = \mathbb{E}_P I_{[Yf_*(X)\leq 0]}$, *and* $\hat{\Psi} = \mathbb{E}_P I_{[Y(2\eta(X)-1)\leq 0]}$ *is the optimal Bayes error.*

3.4 Analysis

For notational simplicity, we will use $C_\rho(z)$ for $C_{\rho,1}(z)$. Considering $f(\mathbf{x}) = \alpha + \beta'\mathbf{k}$, we define a regularized optimization problem of the form

$$\min_{\alpha,\,\beta} \Big\{ \frac{1}{n}\sum_{i=1}^{n} C_\rho(y_i f(\mathbf{x}_i)) + \frac{\gamma_n}{2}\beta'\mathbf{K}\beta \Big\}. \tag{9}$$

Here we assume that the regularization parameter γ relies on the number n of training data points, thus we denote it by γ_n.

Since the optimization problem (9) is convex with respect to α and β, the solution exists and is unique. Moreover, since C_ρ is infinitely smooth, we can resort to the Newton-Raphson method to solve (9).

Proposition 10 *Assume that* γ_n *in (9) and* γ *in (3) are same. Then the minimizer of (9) approaches the minimizer of (3) as* $\rho \to 0$.

This proposition is obtained directly from Proposition 5. For a fixed ρ, we are also concerned with the universal consistency of the classifier based on (9) with and without the offset term α.

Theorem 11 *Let* $K(\cdot,\cdot)$ *be a universal kernel on* $\mathcal{X}\times\mathcal{X}$. *Suppose we are given such a positive sequence* $\{\gamma_n\}$ *that* $\gamma_n \to 0$. *If*

$$n\gamma_n^2/\log n \to \infty,$$

then the classifier based on (9) is strongly universally consistent. If

$$n\gamma_n^2 \to \infty,$$

then the classifier based on (9) with $\alpha = 0$ *is universally consistent.*

Finally, we explore the convergence rate of the classification model in (9). It is worth pointing out that Lin (2002) studied the convergence rate of the ℓ_2-SVM based on Theorem 4.1 in Lin (2000). Additionally, Lin (2002) argued that it is harder for the standard SVM to establish a similar convergence rate result due to two main reasons. The first one is that $(1-z)_+$ is not differentiable and the second one is that the target function $\text{sign}(\eta - 1/2)$ is not in the assumed RKHS. We note that Theorem 4.1 in Lin (2000) is elaborated only for the least squares problem. It is also difficult to apply this theorem to our case, although the above two reasons no longer exist in our case. In Section 6.1 we illustrate how our classification model in (9) approaches the corresponding target function given in (7) using the same simulation data set as in Lin (2002). Moreover, under certain conditions, we can have the following theorem about convergence rate.

Theorem 12 *Suppose* $\mathbf{x}$ *takes values in a finite region with density* $p(\mathbf{x})$ *and* $f(\mathbf{x}) = \alpha + h(\mathbf{x})$ *with* α *taking values in a finite open interval. Assume that* h *belongs to a bounded open set of RKHS* $\mathcal{H}_K$ *with positive definite kernel* $K(\cdot,\cdot)$ *and that there exists an* $M_0 > 0$ *such that* $|K(\mathbf{x}_1,\mathbf{x}_2)| < M_0$ *if* $(\mathbf{x}_1,\mathbf{x}_2)$ *are bounded. Let* $\phi(z)$ *be a strictly convex function and twice continuously differentiable. We define*

$$\begin{aligned} f^* &= \operatorname*{argmin}_f \int \phi(yf(\mathbf{x}))dF(\mathbf{x},y), \\ \hat{f}_n &= \operatorname*{argmin}_f \int \phi(yf(\mathbf{x}))dF_n(\mathbf{x},y) + \frac{\gamma}{2}\|h\|^2_{\mathcal{H}_K}. \end{aligned}$$

where $F(\mathbf{x},y)$ *is the distribution of* $(\mathbf{x},y)$ *and* $F_n(\mathbf{x},y)$ *is the empirical distribution of* $(\mathbf{x},y)$. *Then we have*

$$\int |\hat{f}_n(\mathbf{x}) - f^*(\mathbf{x})| p(\mathbf{x})d\mathbf{x} = O(\gamma) + O_p(\frac{1}{\sqrt{n}}).$$

Obviously, the Gaussian RBF kernel satisfies the condition in Theorem 12. Since $C_\rho(z)$ is strictly convex and infinitely smooth, we can directly apply Theorem 12 to the classification model in (9). In particular, let $\hat{f}_n$ be the minimizer of Problem (9) and $f_*(\eta)$ be defined in (7). Under the conditions in Theorem 12, we have

$$\int \left|\hat{f}_n(\mathbf{x}) - f_*(\eta(\mathbf{x}))\right| p(\mathbf{x})d\mathbf{x} = O(\gamma) + O_p(\frac{1}{\sqrt{n}}).$$

4. Class Probability Estimation of SVM Outputs

As discussed earlier, the limit of the coherence function, $V_{\rho,1}(yf(\mathbf{x}))$, at $\rho = 0$ is just the hinge loss. Moreover, Proposition 7 shows that the minimizer of $V_{\rho,1}(f)\eta + V_{\rho,1}(-f)(1-\eta)$ approaches that of $H(f)\eta + H(-f)(1-\eta)$ as $\rho \to 0$. Thus, Theorem 6 provides us with an approach to the estimation of the class probability for the conventional SVM.

In particular, let $\hat{f}(\mathbf{x})$ be the solution of the optimization problem (3) for the conventional SVM. In terms of Theorem 6, we suggest that the estimated class probability $\hat{\eta}(\mathbf{x})$ is defined as

$$\hat{\eta}(\mathbf{x}) = \frac{1 + \exp(\frac{\hat{f}(\mathbf{x})-1}{\rho})}{1 + \exp(-\frac{1+\hat{f}(\mathbf{x})}{\rho}) + 1 + \exp(\frac{\hat{f}(\mathbf{x})-1}{\rho})}. \tag{10}$$

Proposition 7 would seem to motivate setting ρ to a very small value in (10). However, as shown in Proposition 8, the probabilistic outputs degenerate to 0, $1/3$, $1/2$, $2/3$ and 1 in this case. Additionally, the classification function $\hat{f}(\mathbf{x}) = \hat{\alpha} + \sum_{i=1}^{n} \hat{\beta}_i K(\mathbf{x}, \mathbf{x}_i)$ is obtained via fitting a conventional SVM model on the training data. Thus, rather than attempting to specify a fixed value of ρ via a theoretical argument, we instead view it as a hyperparameter to be fit empirically.

In particular, we fit ρ by minimizing the generalized Kullback-Leibler divergence (or cross-entropy error) between $\hat{\eta}(X)$ and $\eta(X)$, which is given by

$$\mathrm{GKL}(\eta, \hat{\eta}) = \mathbb{E}_X \Big[\eta(X) \log \frac{\eta(X)}{\hat{\eta}(X)} + (1-\eta(X)) \log \frac{1-\eta(X)}{1-\hat{\eta}(X)} \Big].$$

Alternatively, we formulate the optimization problem for obtaining ρ as

$$\min_{\rho > 0} \mathrm{EKL}(\hat{\eta}) := -\frac{1}{n} \sum_{i=1}^{n} \Big\{ \frac{1}{2}(y_i + 1) \log \hat{\eta}(\mathbf{x}_i) + \frac{1}{2}(1 - y_i) \log(1 - \hat{\eta}(\mathbf{x}_i)) \Big\}. \tag{11}$$

The problem can be solved by the Newton method. In summary, one first obtains $\hat{f}(\mathbf{x}) = \hat{\alpha} + \sum_{i=1}^{n} \hat{\beta}_i K(\mathbf{x}, \mathbf{x}_i)$ via the conventional SVM model, and estimates ρ via the optimization problem in (11) based on the training data; one then uses the formula in (10) to estimate the class probabilities for the training samples as well as the test samples.

5. $\mathcal{C}$-Learning

Focusing on the relationships of the $\mathcal{C}$-loss $C_\rho(yf(x))$ (i.e., $C_{\rho,1}(yf(x))$) with the hinge and logit losses, we illustrate its application in the construction of large-margin classifiers. Since $C_\rho(yf(\mathbf{x}))$ is smooth, it does not tend to yield a sparse classifier. However, we can employ a sparsification penalty $J(h)$ to achieve sparseness. We use the elastic-net penalty of Zou and Hastie (2005) for the experiments in this section. Additionally, we study two forms of $f(\mathbf{x})$: kernel expansion and feature expansion. Built on these two expansions, sparseness can subserve the selection of support vectors and the selection of features, respectively. The resulting classifiers are called *$\mathcal{C}$-learning*.

5.1 The Kernel Expansion

In the kernel expansion approach, given a reproducing kernel $K(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we define the kernel expansion as $f(\mathbf{x}) = \alpha + \sum_{i=1}^{n} \beta_i K(\mathbf{x}_i, \mathbf{x})$ and solve the following optimization problem:

$$\min_{\alpha,\beta} \frac{1}{n} \sum_{i=1}^{n} C_\rho(y_i f(\mathbf{x}_i)) + \gamma \Big((1-\omega) \frac{1}{2} \beta' \mathbf{K} \beta + \omega \|\beta\|_1 \Big), \tag{12}$$

where $\mathbf{K} = [K(\mathbf{x}_i, \mathbf{x}_j)]$ is the $n \times n$ kernel matrix.

It is worth pointing out that the current penalty is slightly different from the conventional elastic-net penalty, which is $(1-\omega)\frac{1}{2}\beta'\beta + \omega\|\beta\|_1$. In fact, the optimization problem (12) can be viewed equivalently as the optimization problem

$$\min_{\alpha,\beta} \frac{1}{n} \sum_{i=1}^{n} C_\rho(y_i f(\mathbf{x}_i)) + \frac{\gamma}{2} \beta' \mathbf{K} \beta \tag{13}$$

under the ℓ_1 penalty $\|\beta\|_1$. Thus, the method derived from (12) enjoys the generalization ability of the conventional kernel supervised learning method derived from (13) but also the sparsity of the ℓ_1 penalty.

Recently, Friedman et al. (2010) devised a pathwise coordinate descent algorithm for regularized logistic regression problems in which the elastic-net penalty is used. In order to solve the optimization problem in (12), we employ this pathwise coordinate descent algorithm.

Let the current estimates of α and β be $\hat{\alpha}$ and $\hat{\beta}$. We first form a quadratic approximation to $\frac{1}{n}\sum_{i=1}^n C_\rho(y_i f(\mathbf{x}_i))$, which is

$$Q(\alpha,\beta) = \frac{1}{2n\rho}\sum_{i=1}^n q(\mathbf{x}_i)(1-q(\mathbf{x}_i))\big(\alpha+\mathbf{k}_i'\beta - z_i\big)^2 + \text{Const},$$

where

$$\begin{aligned} z_i &= \hat{\alpha}+\mathbf{k}_i'\hat{\beta}+\frac{\rho}{y_i(1-q(\mathbf{x}_i))}, \\ q(\mathbf{x}_i) &= \frac{\exp[(1-y_i(\hat{\alpha}+\mathbf{k}_i'\hat{\beta})/\rho]}{1+\exp[(1-y_i(\hat{\alpha}+\mathbf{k}_i'\hat{\beta}))/\rho]}, \\ \mathbf{k}_i &= (K(\mathbf{x}_1,\mathbf{x}_i),\ldots,K(\mathbf{x}_n,\mathbf{x}_i))'. \end{aligned}$$

We then employ coordinate descent to solve the weighted least-squares problem as follows:

$$\min_{\alpha,\beta} G(\alpha,\beta) := Q(\alpha,\beta) + \gamma\Big((1-\omega)\frac{1}{2}\beta'\mathbf{K}\beta + \omega\|\beta\|_1\Big). \tag{14}$$

Assume that we have estimated $\tilde{\beta}$ for β using $G(\alpha,\beta)$. We now set $\frac{\partial G(\alpha,\tilde{\beta})}{\partial\alpha}=0$ to find the new estimate of α:

$$\tilde{\alpha} = \frac{\sum_{i=1}^n q(\mathbf{x}_i)(1-q(\mathbf{x}_i))\big(z_i-\mathbf{k}_i'\tilde{\beta}\big)}{\sum_{i=1}^n q(\mathbf{x}_i)(1-q(\mathbf{x}_i))}. \tag{15}$$

On the other hand, assume that we have estimated $\tilde{\alpha}$ for α and $\tilde{\beta}_l$ for β_l ($l=1,\ldots,n, l\neq j$). We now optimize β_j. In particular, we only consider the gradient at $\beta_j\neq 0$. If $\beta_j>0$, we have

$$\frac{\partial G(\tilde{\alpha},\tilde{\beta})}{\partial\beta_j} = \frac{1}{n\rho}\sum_{i=1}^n K_{ij}q(\mathbf{x}_i)(1-q(\mathbf{x}_i))\big(\alpha+\mathbf{k}_i'\tilde{\beta}-z_i\big)+\gamma(1-\omega)(K_{jj}\beta_j+\mathbf{k}_j\check{\beta})+\gamma\omega$$

and, hence,

$$\tilde{\beta}_j = \frac{S(t-\gamma(1-\omega)\mathbf{k}_j'\check{\beta},\gamma\omega)}{\frac{1}{n\rho}\sum_{i=1}^n K_{ij}^2 q(\mathbf{x}_i)(1-q(\mathbf{x}_i))+\gamma(1-\omega)K_{jj}}, \tag{16}$$

where $t = \frac{1}{n\rho}\sum_{i=1}^n K_{ij}q(\mathbf{x}_i)(1-q(\mathbf{x}_i))\big(z_i-\tilde{\alpha}-\mathbf{k}_i'\check{\beta}\big)$, $\check{\beta} = (\tilde{\beta}_1,\ldots,\tilde{\beta}_{j-1},0,\tilde{\beta}_{j+1},\ldots,\tilde{\beta}_n)'$, $K_{ij}=K(\mathbf{x}_i,\mathbf{x}_j)$, and $S(\mu,\nu)$ is the soft-thresholding operator:

$$\begin{aligned} S(\mu,\nu) &= \text{sign}(\mu)(|\mu|-\nu)_+ \\ &= \begin{cases} \mu-\nu & \text{if } \mu>0 \text{ and } \mu<|\nu| \\ \mu+\nu & \text{if } \mu<0 \text{ and } \mu<|\nu| \\ 0 & \text{if } \mu>|\nu|. \end{cases} \end{aligned}$$

Algorithm 1 summarizes the coordinate descent algorithm for binary C-learning.

Algorithm 1 The coordinate descent algorithm for binary $\mathcal{C}$-learning

Input: $\mathcal{T} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, γ, ω, ε_m, ε_i, ρ;
Initialize: $\tilde{\alpha} = \alpha_0$, $\tilde{\beta} = \beta_0$
repeat
  Calculate $G(\tilde{\alpha}, \tilde{\beta})$ using (14);
  $\alpha^\star \leftarrow \tilde{\alpha}$;
  $\beta^\star \leftarrow \tilde{\beta}$;
  repeat
    $\bar{\alpha} \leftarrow \tilde{\alpha}$;
    $\bar{\beta} \leftarrow \tilde{\beta}$;
    Calculate $\tilde{\alpha}$ using (15);
    for $j = 1$ **to** n **do**
      Calculate $\tilde{\beta}_j$ using (16);
    end for
  until $\|\tilde{\alpha} - \bar{\alpha}\| + \|\tilde{\beta} - \bar{\beta}\| < \varepsilon_i$
until $\|\tilde{\alpha} - \alpha^\star\| + \|\tilde{\beta} - \beta^\star\| < \varepsilon_m$
Output: $\tilde{\alpha}$, $\tilde{\beta}$, and $f(\mathbf{x}) = \tilde{\alpha} + \sum_{i=1}^n K(\mathbf{x}_i, \mathbf{x})\tilde{\beta}_i$.

5.2 The Linear Feature Expansion

In the linear feature expansion approach, we let $f(\mathbf{x}) = a + \mathbf{x}'\mathbf{b}$, and pose the following optimization problem:

$$\min_{a,\mathbf{b}} \frac{1}{n}\sum_{i=1}^n C_\rho(y_i f(\mathbf{x}_i)) + \gamma J_\omega(\mathbf{b}), \tag{17}$$

where for $\omega \in [0,1]$

$$J_\omega(\mathbf{b}) = (1-\omega)\frac{1}{2}\|\mathbf{b}\|_2^2 + \omega\|\mathbf{b}\|_1 = \sum_{j=1}^d \Big[\frac{1}{2}(1-\omega)b_j^2 + \omega|b_j|\Big].$$

The elastic-net penalty maintains the sparsity of the ℓ_1 penalty, but the number of variables to be selected is no longer bounded by n. Moreover, this penalty tends to generate similar coefficients for highly-correlated variables. We also use a coordinate descent algorithm to solve the optimization problem (17). The algorithm is similar to that for the kernel expansion and the details are omitted here.

6. Experimental Results

In Section 6.1 we illustrate convergence analysis of our classification method based on the $\mathcal{C}$-loss. In Section 6.2 we report the results of experimental evaluations of our method for class probability estimation of the conventional SVM given in Section 4. In Section 6.3 we present results for the $\mathcal{C}$-learning method given in Section 5.

6.1 Simulation Analysis for Convergence of $\mathcal{C}-$Learning

In Section 3.4 we have presented a theoretical analysis for convergence rate of the classification model defined in (9). We now conduct an empirical analysis to illustration how our model in (9)

approaches the target function $f_*(\eta)$ given in (7) and how the class probability estimate $\tilde{\eta}(\hat{f}_n(\mathbf{x}))$ in (8) approaches the underlying class probability $\eta(\mathbf{x})$.

For the purpose of visualization, we employ the same simulation data set as in Lin (2002). In particular, we take n equidistant points on the interval $[0,1]$; that is, $x_i = (i-1)/n$ for $i = 1,\ldots,n$. Let $\eta(x) = \Pr(Y = 1|X = x) = 1 - |1-2x|$. Then the target function $f_*(\eta(x))$ for our model is computed by (7). We randomly generate y_i to be 1 or -1 with probability $\eta(x_i)$ and $1-\eta(x_i)$, and form a training data set $\{(x_i,y_i); i = 1\ldots,n\}$. Following the setting in Lin (2002), we implement our simulations on RKHS $\mathcal{H}^m([0,1])$ (i.e., the Sobolev Hilbert space with order m of univariate functions on domain $[0,1]$) and four data sets with size $n = 33$, 65, 129 and 257. The parameters γ and ρ are chosen to minimize the generalized KL (GKL) divergence by the grid search. The implementation is based on the $\mathcal{C}$-learning algorithm in Section 5 where the hyperparameter ω is approximately set as 0.

Figures 2 and 3 respectively depict the solutions $\hat{f}_n$ to the regularization problem in (9) and the class probability estimates $\tilde{\eta}(\hat{f}_n)$ given in (8), when the sample size is $n = 33$, 65, 129, and 257. We can see that the solution $\hat{f}_n$ is close the target function $f_*(\eta)$ and that the class probability estimate $\tilde{\eta}(\hat{f}_n)$ is close the underlying class probability η, as n increases. Thus, our simulation example shows that our method based on $\mathcal{C}$-loss not only can approach its corresponding target function $f_*(\eta)$ but also can estimate the underlying class probability $\eta(\mathbf{x})$. It should be pointed out that the similar experimental results can be found on the Gaussian RBF kernel, so here we do not include the results with the Gaussian RBF kernel.

6.2 Simulation for Class Probability Estimation of SVM Outputs

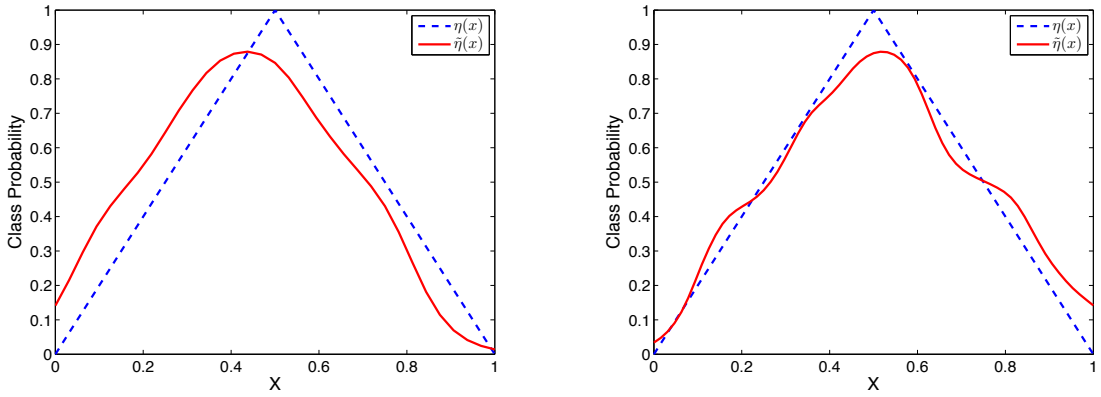
We validate our estimation method for the class probability of SVM outputs ("Ours for SVM"), comparing it with several alternatives: Platt's method (Platt, 1999), Sollich's method (Sollich, 2002), and the method of Wang et al. (2008) (WSL's). Since penalized (or regularized) logistic regression (PLR) and $\mathcal{C}$-learning can directly calculate class probability, we also implement them. Especially, the class probability of $\mathcal{C}$-learning outputs is based on (8) where we set $\rho = 1$ and $u = 1$ since $\mathcal{C}$-learning itself employs the same setting.

We conducted our analysis over two simulation data sets which were used by Wang et al. (2008). The first simulation data set, $\{(x_{i1},x_{i2};y_i)\}_{i=1}^{1000}$, was generated as follows. The $\{(x_{i1},x_{i2})\}_{i=1}^{1000}$ were uniformly sampled from a unit disk $\{(x_1,x_2) : x_1^2 + x_2^2 \le 1\}$. Next, we set $y_i = 1$ if $x_{i1} \ge 0$ and $y_i = -1$ otherwise, $i = 1,\ldots,1000$. Finally, we randomly chose 20% of the samples and flipped their labels. Thus, the true class probability $\eta(Y_i = 1|x_{i1},x_{i2})$ was either 0.8 or 0.2.

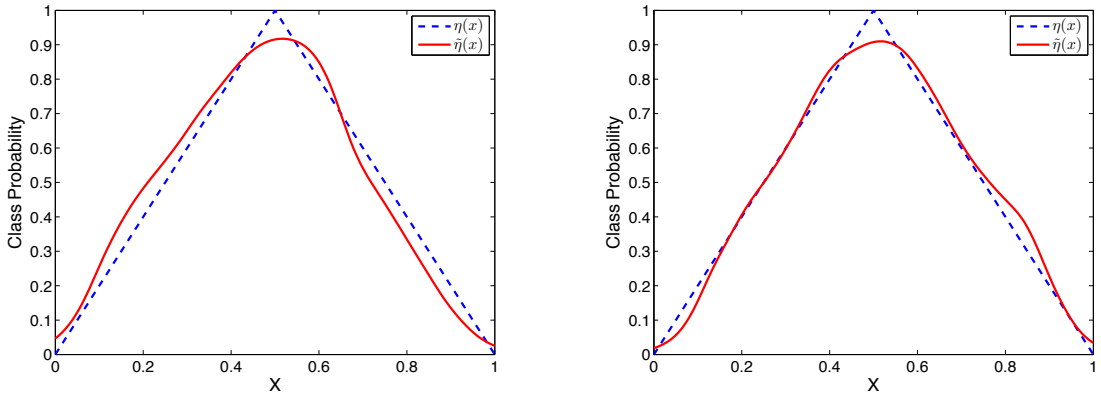
The second data set, $\{(x_{i1},x_{i2};y_i)\}_{i=1}^{1000}$, was generated as follows. First, we randomly assigned 1 or -1 to y_i for $i = 1,\ldots,1000$ with equal probability. Next, we generated x_{i1} from the uniform distribution over $[0,2\pi]$, and set $x_{i2} = y_i(\sin(x_{i1}) + \varepsilon_i)$ where $\varepsilon_i \sim N(\varepsilon_i|1,0.01)$. For the data, the true class probability of $Y = 1$ was given by

$$\eta(Y = 1|x_1,x_2) = \frac{N(x_2|\sin(x_1)+1,0.01)}{N(x_2|\sin(x_1)+1,0.01)+N(x_2|-\sin(x_1)-1,0.01)}.$$

The simulation followed the same setting as that in Wang et al. (2008). That is, we randomly selected 100 samples for training and the remaining 900 samples for test. We did 100 replications for each data set. The values of generalized Kullback-Leibler loss (GKL) and classification error rate (CER) on the test sets were averaged over these 100 simulation replications. Additionally, we



(a) $n = 33$, $\gamma = 5.9186 \times 10^{-5}$, $\rho = 0.5695$ (b) $n = 65$, $\gamma = 7.5120 \times 10^{-6}$, $\rho = 0.5695$

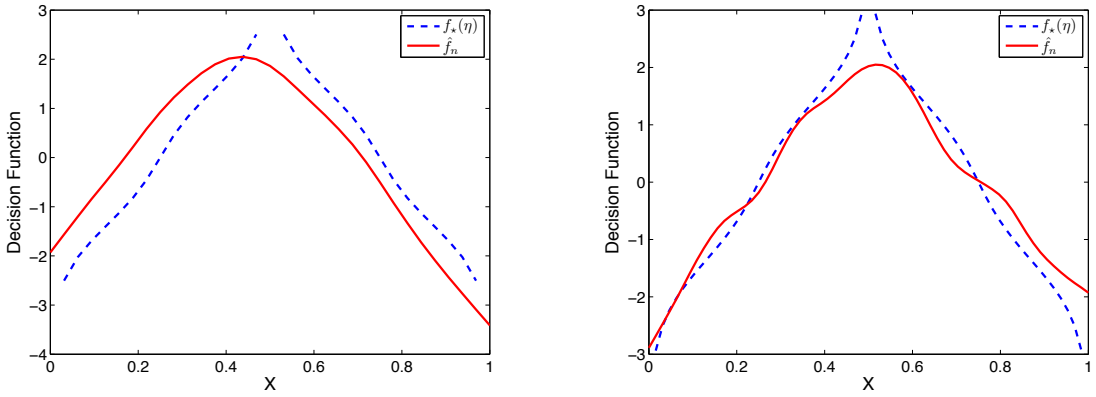


(c) $n = 129$, $\gamma = 1.5141 \times 10^{-5}$, $\rho = 0.5695$ (d) $n = 257$, $\gamma = 3.7999 \times 10^{-6}$, $\rho = 0.5695$

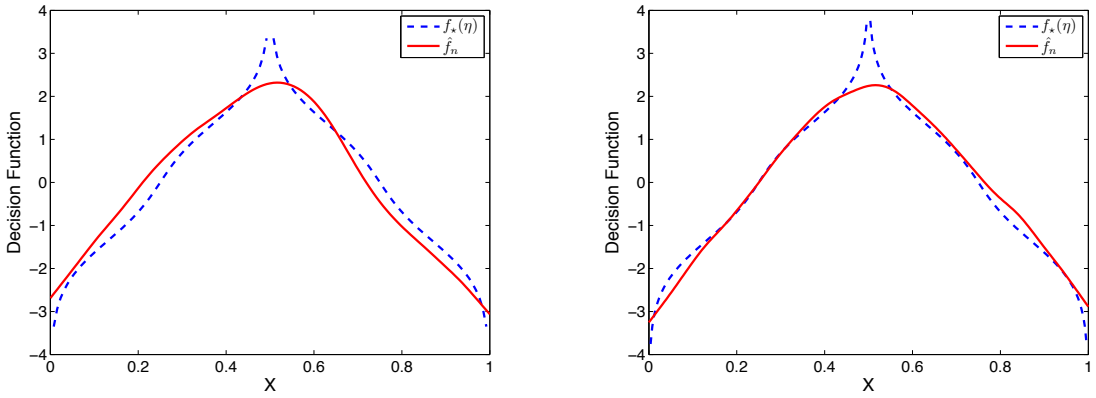
Figure 2: The underlying class probabilities $\eta(x)$ ("blue + dashed line") and estimated class probabilities $\tilde{\eta}(x) = \tilde{\eta}(\hat{f}_n(x))$ ("red + solid line") on RKHS $\mathcal{H}^m([0,1])$ and the simulation data sets with the size $n = 33$, 65, 129, and 257. Here the values of parameters γ and ρ in each data set are obtained by minimizing the GKL divergence.

employed a Gaussian RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$ where the parameter σ was set as the median distance between the positive and negative classes. We reported GKL and CER as well as the corresponding standard deviations in Tables 2 and 3 in which the results with the PLR method, the tuned Platt method and the WSL method are directly cited from Wang et al. (2008).

Note that the results with PLR were averaged only over 66 nondegenerate replications (Wang et al., 2008). Based on GKL and CER, the performance of $\mathcal{C}$-learning is the best in these two simulations. With regard to GKL, our method for SVM outperforms the original and tuned versions of Platt's method as well as the method of Wang et al. (2008). Since our estimation method is based on the $\hat{\eta}(\mathbf{x})$ in (10), the CER with this class probability $\hat{\eta}(\mathbf{x})$ is identical to that with the conventional SVM. This also applies to Sollich's method, thus we did not include the CER of this method. However, Table 3 shows that this does not necessarily hold for Platt's method for SVM



(a) $n = 33$, $\gamma = 5.9186 \times 10^{-5}$, $\rho = 0.5695$ (b) $n = 65$, $\gamma = 7.5120 \times 10^{-6}$, $\rho = 0.5695$



(c) $n = 129$, $\gamma = 1.5141 \times 10^{-5}$, $\rho = 0.5695$ (d) $n = 257$, $\gamma = 3.7999 \times 10^{-6}$, $\rho = 0.5695$

Figure 3: The target decision function $f_*(\eta)$ ("blue + dashed line") and estimated decision functions $\hat{f}_n$ ("red + solid line") on RKHS $\mathcal{H}^m([0,1])$ and the toy data sets with the size $n = 33$, 65, 129, and 257. The parameters are identical to the settings in Figure 2.

probability outputs. In other words, $\hat{\eta}(\mathbf{x}) > 1/2$ is not equivalent to $\hat{f}(\mathbf{x}) > 0$ for Platt's method. In fact, Platt (1999) used the sigmoid-like function to improve the classification accuracy of the conventional SVM. As for the method of Wang et al. (2008) which is built on a sequence of weighted classifiers, the CERs of the method should be different from those of the original SVM. With regard to CER, the performance of PLR is the worst in most cases.

6.3 The Performance Analysis of $\mathcal{C}$-Learning

To evaluate the performance of our $\mathcal{C}$-learning method, we further conducted empirical studies on several benchmark data sets and compared $\mathcal{C}$-learning with two closely related classification methods: the hybrid huberized SVM (HHSVM) of Wang et al. (2007) and the regularized logistic regression model (RLRM) of Friedman et al. (2010), both with the elastic-net penalty. All the three classification methods were implemented in both the feature and kernel expansion settings.

	PLR	Platt's	Tuned Platt	WSL's	Sollich's	Ours for SVM	$\mathcal{C}$-learning
Data 1	0.579 ($\pm$0.0021)	0.582 ($\pm$0.0035)	0.569 ($\pm$0.0015)	0.566 ($\pm$0.0014)	0.566 ($\pm$0.0021)	0.558 ($\pm$0.0015)	0.549 ($\pm$0.0016)
Data 2	0.138 ($\pm$0.0024)	0.163 ($\pm$0.0018)	0.153 ($\pm$0.0013)	0.153 ($\pm$0.0010)	0.155 ($\pm$0.0017)	0.142 ($\pm$0.0016)	0.134 ($\pm$0.0014)

Table 2: Values of GKL over the two simulation test sets (standard deviations are shown in parentheses).

	PLR	Platt's	WSL's	Ours for SVM	$\mathcal{C}$-learning
Data 1	0.258 ($\pm$0.0053)	0.234 ($\pm$0.0026)	0.217 ($\pm$0.0021)	0.219 ($\pm$0.0021)	0.214 ($\pm$0.0015)
Data 2	0.075 ($\pm$0.0018)	0.077 ($\pm$0.0024)	0.069 ($\pm$0.0014)	0.065 ($\pm$0.0015)	0.061 ($\pm$0.0019)

Table 3: Values of CER over the two simulation test sets (standard deviations are shown in parentheses).

In the experiments we used 11 binary classification data sets. Table 4 gives a summary of these benchmark data sets. The seven binary data sets of digits were obtained from the publicly available USPS data set of handwritten digits as follows. The first six data sets were generated from the digit pairs {(1, 7), (2, 3), (2, 7), (3, 8), (4, 7), (6,9)}, and 200 digits were chosen within each class of each data set. The USPS (odd vs. even) data set consisted of the first 80 images per digit in the USPS training set.

The two binary artificial data sets of "g241c" and "g241d" were generated via the setup presented by Chapelle et al. (2006). Each class of these two data sets consisted of 750 samples.

The two binary gene data sets of "colon" and "leukemia" were also used in our experiments. The "colon" data set, consisting of 40 colon tumor samples and 22 normal colon tissue samples with 2,000 dimensions, was obtained by employing an Affymetrix oligonucleotide array to analyze more than 6,500 human genes expressed in sequence tags (Alon et al., 1999). The "leukemia" data set is of the same type as the "colon" cancer data set (Golub et al., 1999), and it was obtained with respect to two variants of leukemia, that is, acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL). It initially contained expression levels of 7129 genes taken over 72 samples (AML, 25 samples, or ALL, 47 samples), and then it was pre-feature selected, leading to a feature space with 3571 dimensions.

In our experiments, each data set was randomly partitioned into two disjoint subsets as the training and test, with the percentage of the training data samples also given in Table 4. Twenty random partitions were chosen for each data set, and the average and standard deviation of their classification error rates over the test data were reported.

Although we can seek an optimum ρ using computationally intensive methods such as cross-validation, the experiments showed that when ρ takes a value in $[0.1, 2]$, our method is always able to obtain promising performance. Here our reported results are based on the setting of $\rho = 1$, due to

Data Set	m	d	k	n/k
USPS (1 vs. 7)	2	256	400	3%
USPS (2 vs. 3)	2	256	400	3%
USPS (2 vs. 7)	2	256	400	3%
USPS (3 vs. 8)	2	256	400	3%
USPS (4 vs. 7)	2	256	400	3%
USPS (6 vs. 9)	2	256	400	3%
USPS (Odd vs. Even)	2	256	800	3%
g241c	2	241	1500	10%
g241d	2	241	1500	10%
colon	2	2000	62	25.8%
leukemia	2	3571	72	27.8%

Table 4: Summary of the benchmark data sets: m—the number of classes; d—the dimension of the input vector; k—the size of the data set; n—the number of the training data.

Data Set	HHSVM	RLRM	$\mathcal{C}$-learning
(1 vs. 7)	2.29±1.17	2.06±1.21	**1.60±0.93**
(2 vs. 3)	**8.13±2.02**	8.29±2.76	8.32±2.73
(2 vs. 7)	5.82±2.59	6.04±2.60	**5.64±2.44**
(3 vs. 8)	12.46±2.90	**10.77±2.72**	11.74±2.83
(4 vs. 7)	7.35±2.89	6.91±2.72	**6.68±3.53**
(6 vs. 9)	2.32±1.65	2.15±1.43	**2.09±1.41**
(Odd vs. Even)	20.94±2.02	19.83±2.82	**19.74±2.81**
g241c	22.30±1.30	21.38±1.12	**21.34±1.11**
g241d	24.32±1.53	**23.81±1.65**	23.85±1.69
colon	14.57±1.86	14.47±2.02	**12.34±1.48**
leukemia	4.06±2.31	4.43±1.65	**3.21±1.08**

Table 5: Classification error rates (%) and standard deviations on the 11 data sets for the feature expansion setting.

the relationship of the $\mathcal{C}$-loss $C(z)$ with the hinge loss $(1-z)_+$ and the logit loss $\log(1+\exp(-z))$ (see our analysis in Section 3 and Figure 1).

As for the parameters γ and ω, they were selected by cross-validation for all the classification methods. In the kernel expansion, the RBF kernel $K(\mathbf{x}_i,\mathbf{x}_j)=\exp(-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma^2)$ was employed, and σ was set to the mean Euclidean distance among the input samples. For $\mathcal{C}$-learning, the other parameters were set as follows: $\varepsilon_m=\varepsilon_i=10^{-5}$.

Tables 5 and 6 show the test results corresponding to the linear feature expansion and RBF kernel expansion, respectively. From the tables, we can see that for the overall performance of $\mathcal{C}$-learning is slightly better than the two competing methods in the feature and kernel settings generally.

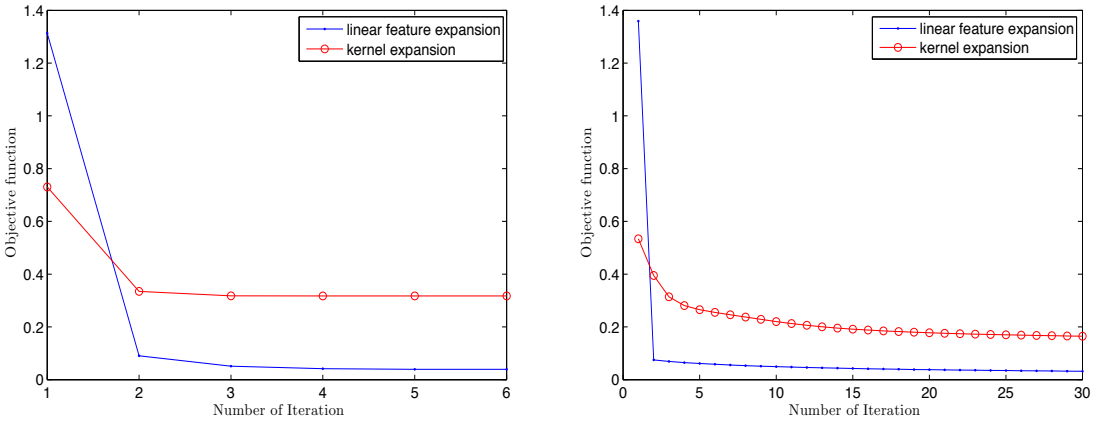
Figure 4 reveals that the values of the objective functions for the linear feature and RBF kernel versions in the outer and inner iterations tend to be significantly reduced as the number of iterations in the coordinate descent procedure increases. Although we report only the change of the values of

Data Set	HHSVM	RLRM	$\mathcal{C}$-learning
(1 vs. 7)	1.73±1.64	1.39±0.64	**1.37±0.65**
(2 vs. 3)	8.55±3.36	8.45±3.38	**8.00±3.32**
(2 vs. 7)	5.09±2.10	4.02±1.81	**3.90±1.79**
(3 vs. 8)	12.09±3.78	10.58±3.50	**10.36±3.52**
(4 vs. 7)	6.74±3.39	6.92±3.37	**6.55±3.28**
(6 vs. 9)	2.12±0.91	1.74±1.04	**1.65±0.99**
(Odd vs. Even)	28.38±10.51	26.92±6.52	**26.29±6.45**
g241c	**21.38±1.45**	21.55±1.42	21.62±1.35
g241d	25.89±2.15	22.34±1.27	**20.37±1.20**
colon	14.26±2.66	14.79±2.80	**13.94±2.44**
leukemia	2.77±0.97	2.74±0.96	**2.55±0.92**

Table 6: Classification error rates (%) and standard deviations on the 11 data sets for the RBF kernel setting.

the objective function for the data set USPS (1 vs. 7) similar results were found on all other data sets. This shows that the coordinate descent algorithm is very efficient.

We also conducted a systematic study of sparseness from the elastic-net penalty. Indeed, the elastic-net penalty does give rise to sparse solutions for our $\mathcal{C}$-learning methods. Moreover, we found that similar to other methods the sparseness of the solution is dependent on the parameters γ and ω that were set to different values for different data sets using cross validation.



(a) (b)

Figure 4: Change of the value of the objective function for the $\mathcal{C}$-learning as the number of iterations in the coordinate descent procedure increases in the linear feature and RBF kernel cases on the data set USPS (1 vs. 7): (a) the values of the objective function (12) in the outer iteration; (b) the objective function values $G(\alpha, \beta)$ for the fature and RBF kernel cases in the inner iteration.

7. Conclusions

In this paper we have studied a family of coherence functions and considered the relationship between coherence functions and hinge functions. In particular, we have established some important properties of these functions, which lead us to a feasible approach for class probability estimation in the conventional SVM. Moreover, we have proposed large-margin classification methods using the $\mathcal{C}$-loss function and the elastic-net penalty, and developed pathwise coordinate descent algorithms for parameter estimation. We have theoretically established the Fisher-consistency of our classification methods and empirically tested the classification performance on several benchmark data sets. Our approach establishes an interesting link between SVMs and logistic regression models due to the relationship of the $\mathcal{C}$-loss with the hinge and logit losses.

Acknowledgments

The authors would like to thank the Action Editor and three anonymous referees for their constructive comments and suggestions on the original version of this paper. This work has supported in part by the Natural Science Foundations of China (No. 61070239), the US ARL and the US ARO under contract/grant number W911NF-11-1-0391.

Appendix A. The Proof of Proposition 2

First, we have

$$\rho\log 2+[u-z]_+-V_{\rho,u}(z)=\rho\log\frac{2\exp\frac{[u-z]_+}{\rho}}{1+\exp\frac{u-z}{\rho}}\geq 0.$$

Second, note that

$$\begin{aligned}\rho\log 2+\frac{u-z}{2}-V_{\rho,u}(z)&=\rho\log\frac{2\exp\frac{1}{2}\frac{u-z}{\rho}}{1+\exp\frac{u-z}{\rho}}\\&\leq\rho\log\frac{\exp\frac{u-z}{\rho}}{1+\exp\frac{u-z}{\rho}}\leq 0,\end{aligned}$$

where we use the fact that $\exp(\cdot)$ is convex.

Third, it immediately follows from Proposition (i) that $\lim_{\rho\to 0}V_{\rho,u}(z)=[u-z]_+$. Moreover, it is easily obtained that

$$\begin{aligned}\lim_{\rho\to\infty}V_{\rho,u}(z)-\rho\log 2&=\lim_{\rho\to\infty}\frac{\log\frac{1+\exp\frac{u-z}{\rho}}{2}}{\frac{1}{\rho}}=\lim_{\alpha\to 0}\frac{\log\frac{1+\exp\alpha(u-z)}{2}}{\alpha}\\&=\lim_{\alpha\to 0}\frac{\frac{1}{2}[u-z]\exp[\alpha(u-z)]}{\frac{1+\exp\alpha(u-z)}{2}}=\frac{1}{2}(u-z).\end{aligned}$$

Since $\log(1+a)\geq\log(a)$ for $a>0$, we have

$$\frac{u}{\log[1+\exp(u/\rho)]}\log\left[1+\exp\frac{u-z}{\rho}\right]\leq\frac{u}{u/\rho}\log\left[1+\exp\frac{u-z}{\rho}\right]=\rho\log\left[1+\exp\frac{u-z}{\rho}\right].$$

We now consider that

$$\lim_{\rho\to\infty} C_{\rho,u}(z) = u \lim_{\rho\to\infty} \frac{\log\left[1+\exp\frac{u-z}{\rho}\right]}{\log[1+\exp(u/\rho)]} = u.$$

Finally, since

$$\lim_{\alpha\to\infty} \frac{\log[1+\exp(u\alpha)]}{\alpha u} = \lim_{\alpha\to\infty} \frac{\exp(u\alpha)]}{1+\exp(u\alpha)} = 1 \text{ for } u > 0,$$

we obtain $\lim_{\rho\to 0} C_{\rho,u}(z) = [u-z]_+$.

Appendix B. The Proof of Proposition 3

Before we prove Proposition 3, we establish the following lemma.

Lemma 13 *Assume that $x > 0$, then $f_1(x) = \frac{x}{1+x}\frac{\log x}{\log(1+x)}$ and $f_2(x) = \frac{x}{1+x}\frac{1}{\log(1+x)}$ are increasing and decreasing, respectively.*

Proof The first derivatives of $f_1(x)$ and $f_2(x)$ are

$$f_1'(x) = \frac{1}{(1+x)^2\log^2(1+x)}\Big[\log x\log(1+x)+\log(1+x)+x\log(1+x)-x\log x\Big]$$
$$f_2'(x) = \frac{1}{(1+x)^2\log^2(1+x)}[\log(1+x)-x] \le 0.$$

This implies that $f_2(x)$ is decreasing. If $\log x \ge 0$, we have $x\log(1+x)-x\log x \ge 0$. Otherwise, if $\log x < 0$, we have $\log x[\log(1+x)-x] \ge 0$. This implies that $f_1'(x) \ge 0$ is always satisfied. Thus, $f_1(x)$ is increasing. ■

Let $\alpha = 1/\rho$ and use $h_1(\alpha)$ for $L_{\rho,u}(z)$ to view it as a function of α. We now compute the derivative of $h_1(\alpha)$ w.r.t. α:

$$\begin{aligned} h_1'(\alpha) &= \frac{\log[1+\exp(\alpha(u-z))]}{\log[1+\exp(u\alpha)]}\times \\ &\quad \Big[\frac{\exp(\alpha(u-z))}{1+\exp(\alpha(u-z))}\frac{u-z}{\log[1+\exp(\alpha(u-z))]} - \frac{\exp(\alpha u)}{1+\exp(\alpha u)}\frac{u}{\log[1+\exp(\alpha u)]}\Big] \\ &= \frac{\log[1+\exp(\alpha(u-z))]}{\alpha\log[1+\exp(u\alpha)]}\times \\ &\quad \Big[\frac{\exp(\alpha(u-z))}{1+\exp(\alpha(u-z))}\frac{\log\exp(\alpha(u-z))}{\log[1+\exp(\alpha(u-z))]} - \frac{\exp(\alpha u)}{1+\exp(\alpha u)}\frac{\log\exp(\alpha u)}{\log[1+\exp(\alpha u)]}\Big]. \end{aligned}$$

When $z < 0$, we have $\exp(\alpha(u-z)) > \exp(\alpha u)$. It then follows from Lemma 13 that $h_1'(\alpha) \ge 0$. When $z \ge 0$, we have $h_1'(\alpha) \le 0$ due to $\exp(\alpha(u-z)) \le \exp(\alpha u)$. The proof of (i) is completed.

To prove part (ii), we regard $L_{\rho,u}(z)$ as a function of u and denote it with $h_2(u)$. The first derivative $h_2'(u)$ is given by

$$\begin{aligned} h_2'(u) &= \frac{\alpha\log[1+\exp(\alpha(u-z))]}{\log[1+\exp(u\alpha)]}\times \\ &\quad \Big[\frac{\exp(\alpha(u-z))}{1+\exp(\alpha(u-z))}\frac{1}{\log[1+\exp(\alpha(u-z))]} - \frac{\exp(\alpha u)}{1+\exp(\alpha u)}\frac{1}{\log[1+\exp(\alpha u)]}\Big]. \end{aligned}$$

Using Lemma 13, we immediately obtain part (ii).

Appendix C. The Proof of Theorem 6

We write the objective function as

$$\begin{aligned} L(f) &= V_{\rho,u}(f)\eta + V_{\rho,u}(-f)(1-\eta) \\ &= \rho \log\left[1+\exp\frac{u-f}{\rho}\right]\eta + \rho\log\left[1+\exp\frac{u+f}{\rho}\right](1-\eta). \end{aligned}$$

The first-order and second-order derivatives of L w.r.t. f are given by

$$\begin{aligned} \frac{dL}{df} &= -\eta\frac{\exp\frac{u-f}{\rho}}{1+\exp\frac{u-f}{\rho}} + (1-\eta)\frac{\exp\frac{u+f}{\rho}}{1+\exp\frac{u+f}{\rho}}, \\ \frac{d^2L}{df^2} &= \frac{\eta}{\rho}\frac{\exp\frac{u-f}{\rho}}{1+\exp\frac{u-f}{\rho}}\frac{1}{1+\exp\frac{u-f}{\rho}} + \frac{1-\eta}{\rho}\frac{\exp\frac{u+f}{\rho}}{1+\exp\frac{u+f}{\rho}}\frac{1}{1+\exp\frac{u+f}{\rho}}. \end{aligned}$$

Since $\frac{d^2L}{df^2} > 0$, the minimum of L is unique. Moreover, letting $\frac{dL}{df} = 0$ yields (7).

Appendix D. The Proof of Proposition 7

First, if $\eta > 1/2$, we have $4\eta(1-\eta) > 4(1-\eta)^2$ and $(2\eta-1)\exp(u/\rho) > 0$. This implies $f_* > 0$. When $\eta < 1/2$, we have $(2\eta-1)\exp(u/\rho) > 0$. In this case, since

$$(1-2\eta)^2\exp(2u/\rho) + 4\eta(1-\eta) < (1-2\eta)^2\exp(2u/\rho) + 4(1-\eta)^2 + 4(1-\eta)(1-2\eta)\exp(u/\rho),$$

we obtain $f_* < 0$.

Second, letting $\alpha = 1/\rho$, we express f_* as

$$\begin{aligned} f_* &= \frac{1}{\alpha}\log\frac{(2\eta-1)\exp(u\alpha) + \sqrt{(1-2\eta)^2\exp(2u\alpha) + 4\eta(1-\eta)}}{2(1-\eta)} \\ &= \frac{1}{\alpha}\log\frac{\frac{(2\eta-1)}{|2\eta-1|} + \sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}}}{2(1-\eta)\exp(-u\alpha)/|2\eta-1|} \\ &= \frac{1}{\alpha}\log\left[\frac{(2\eta-1)}{|2\eta-1|} + \sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}}\right] - \frac{1}{\alpha}\log\left[\frac{2(1-\eta)}{|2\eta-1|}\right] + u. \end{aligned}$$

Thus, if $\eta > 1/2$, it is clear that $\lim_{\alpha\to\infty} f_* = u$. In the case that $\eta < 1/2$, we have

$$\begin{aligned} \lim_{\alpha\to\infty} f_* &= u - u\lim_{\alpha\to\infty}\frac{1}{-1+\sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}}}\frac{1}{\sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}}}\frac{4\eta(1-\eta)}{(1-2\eta)^2}\exp(-2u\alpha) \\ &= u - \frac{4\eta(1-\eta)u}{(1-2\eta)^2}\lim_{\alpha\to\infty}\frac{\exp(-2u\alpha)}{-1+\sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}}} \\ &= u - 2u\lim_{\alpha\to\infty}\sqrt{1+\frac{4\eta(1-\eta)}{(1-2\eta)^2\exp(2u\alpha)}} \\ &= -u. \end{aligned}$$

Here we use l'Hôpital's rule in calculating limits.

Third, let $\alpha = \exp(u/\rho)$. It is then immediately calculated that

$$f'_*(\eta) = 2\rho \frac{\alpha + \frac{(1-2\eta)(1-\alpha^2)}{\sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}}}{(2\eta-1)\alpha + \sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}} + \frac{\rho}{1-\eta}.$$

Consider that

$$\begin{aligned} A &= \frac{2\alpha + \frac{2(1-2\eta)(1-\alpha^2)}{\sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}}}{(2\eta-1)\alpha + \sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}} - \frac{1}{\eta} \\ &= \frac{\alpha - \frac{2\eta+(1-2\eta)\alpha^2}{\sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}}}{\eta(2\eta-1)\alpha + \eta\sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}}. \end{aligned}$$

It suffices for $f'_*(\eta) \geq \frac{\rho}{\eta(1-\eta)}$ to show $A \geq 0$. Note that

$$\frac{(2\eta+(1-2\eta)\alpha^2)^2}{(1-2\eta)^2\alpha^2+4\eta(1-\eta)} - \alpha^2 = \frac{4\eta^2(1-\alpha^2)}{(1-2\eta)^2\alpha^2+4\eta(1-\eta)} \leq 0$$

due to $\alpha \geq 1$, with equality when and only when $\alpha = 1$ or, equivalently, $u = 0$. Accordingly, we have $\alpha - \frac{2\eta+(1-2\eta)\alpha^2}{\sqrt{(1-2\eta)^2\alpha^2+4\eta(1-\eta)}} \geq 0$.

Appendix E. The Proof of Theorem 11

In order to prove the theorem, we define

$$\delta_\gamma := \sup\{t : \gamma t^2 \leq 2V_\rho(0)\} = \sqrt{2/\gamma}$$

for $\gamma > 0$ and let $V_\rho^{(\gamma)}(yf)$ be the coherence function $V_\rho(yf)$ restricted to $\mathcal{Y} \times [-\delta_\gamma k_{\max}, \delta_\gamma k_{\max}]$, where $k_{\max} = \max_{\mathbf{x}\in\mathcal{X}} K(\mathbf{x},\mathbf{x})$. For the Gaussian RBF kernel, we have $k_{\max} = 1$.

It is clear that

$$\|V_\rho^{(\gamma)}\|_\infty := \sup\left\{V_\rho^{(\gamma)}(yf), (y,f) \in \mathcal{Y}\times[-\delta_\gamma k_{\max}, \delta_\gamma k_{\max}]\right\} = \rho \log\left(1 + \exp\frac{u + k_{\max}\sqrt{2/\gamma}}{\rho}\right).$$

Considering that

$$\lim_{\gamma\to 0} \frac{\|V_\rho^{(\gamma)}\|_\infty}{k_{\max}\sqrt{2/\gamma}} = \lim_{\alpha\to\infty} \frac{\exp\frac{u+\alpha}{\rho}}{1+\exp\frac{u+\alpha}{\rho}} = 1,$$

we have $\lim_{\gamma\to 0} \|V_\rho^{(\gamma)}\|_\infty / \sqrt{1/\gamma} = \sqrt{2}k_{\max}$. Hence, we have $\|V_\rho^{(\gamma)}\|_\infty \sim \sqrt{1/\gamma}$.

On the other hand, since

$$V_\rho^{(\gamma)}(yf) = V_\rho^{(\gamma)}(yf_1) - \frac{\partial V_\rho^{(\gamma)}(yf_2)}{\partial f}(f - f_1),$$

where $f_2 \in [f, f_1] \subseteq [-\delta_\gamma k_{\max}, \delta_\gamma k_{\max}]$, we have

$$\begin{aligned}
|V_\rho^{(\gamma)}|_1 &:= \sup\left\{ \frac{|V_\rho^{(\gamma)}(yf) - V_\rho^{(\gamma)|}(yf_1)}{|f - f_1|},\ y \in \mathcal{Y}, f, f_1 \in [-\delta_\gamma k_{\max}, \delta_\gamma k_{\max}], f \neq f_1 \right\} \\
&= \sup\left\{ \left| \frac{\partial V_\rho^{(\gamma)}(yf_2)}{\partial f} \right|,\ y \in \mathcal{Y}, f_2 \in [-\delta_\gamma k_{\max}, \delta_\gamma k_{\max}] \right\} \\
&= \frac{\exp\frac{u + k_{\max}\sqrt{2/\gamma}}{\rho}}{1 + \exp\frac{u + k_{\max}\sqrt{2/\gamma}}{\rho}}.
\end{aligned}$$

In this case, we have $\lim_{\gamma \to 0} |V_\rho^{(\gamma)}|_1 = 1$, which implies that $|V_\rho^{(\gamma)}|_1 \sim 1$.

We now immediately conclude Theorem 11 from Corollary 3.19 and Theorem 3.20 of Steinwart (2005).

Appendix F. The Proof of Theorem 12

Note that the condition for function in the theorem implies that $\phi''(z) > 0$ in $\mathbb{R}$, then it follows that $\phi''(z) > \delta$ for some positive δ in a compact region (and certainly also holds in any bounded region). We also denote

$$\bar{f}_n = \operatorname*{argmin}_f \int \phi(yf(\mathbf{x}))dF_n(\mathbf{x}, y)$$

and

$$\begin{aligned}
L(f) &= \int \phi(yf(\mathbf{x}))dF(\mathbf{x}, y), \\
L_{n1}(f) &= \int \phi(yf(\mathbf{x}))dF_n(\mathbf{x}, y) + \frac{\gamma}{2}\|h\|^2_{\mathcal{H}_K}, \\
L_{n2}(f) &= \int \phi(yf(\mathbf{x}))dF_n(\mathbf{x}, y).
\end{aligned}$$

We have $f^*, \bar{f}_n, \hat{f}_n$ are all unique, because the corresponding objective functions are strictly convex.

Taking the derivative of the functional $L(f)$ w.r.t. f yields

$$\int yv\phi'(yf^*)dF(\mathbf{x}, y) = 0 \quad \text{for any } v \in \mathcal{H}_K. \tag{18}$$

Differentiating the functional $L_{n2}(f)$ w.r.t. f, we have

$$\int yv\phi'(y\bar{f}_n)dF_n(\mathbf{x}, y) = 0 \quad \text{for any } v \in \mathcal{H}_K. \tag{19}$$

It follows from the derivative of the functional $L_{n1}(f)$ w.r.t. h that

$$\int yv\phi'(y\hat{f}_n)dF_n(\mathbf{x}, y) + \gamma\langle \hat{h}_n, v\rangle = 0 \quad \text{for any } v \in \mathcal{H}_K \tag{20}$$

with $\bar{f}_n = \bar{h}_n + \bar{\alpha}_n$ and $\hat{f}_n = \hat{\alpha}_n + \hat{h}_n$. Since $\{\hat{h}_n\}$ is uniformly bounded (the condition of the theorem), and from the property $|\hat{h}_n(\mathbf{x})| \leq \|\hat{h}_n\| K(\mathbf{x}, \mathbf{x})^{1/2}$, we have $\sup_{n,\mathbf{x}} |\hat{f}_n(\mathbf{x})| < M_1$ for some $M_1 > 0$. Similarly, $\sup_{n,\mathbf{x}} |\bar{f}_n(\mathbf{x})| < M_2$ for some $M_2 > 0$.

The left-hand side of (20) is a linear functional, so it has a representer which is of the form $G_1(\hat{h}_n, \hat{\alpha}_n, \gamma) = \psi(\hat{\alpha} + \hat{h}_n) + \gamma \hat{h}_n$. Hence, the Equation (20) can be written as

$$\langle \psi(\hat{\alpha} + \hat{h}_n) + \gamma \hat{h}_n, v \rangle = 0$$

for any v. This implies $G_1(\hat{h}_n, \hat{\alpha}_n, \gamma) = 0$.

Differentiating $L_{n1}(f)$ w.r.t. α, we get $\int y\phi'(y\hat{f}_n(\mathbf{x}))dF_n(\mathbf{x}, y) = 0$ which is of the form $G_2(\hat{h}_n, \hat{\alpha}_n, \gamma) = 0$. Note that the second-order derivative of $L_{n1}(f)$ w.r.t. h and α (simply denoted as D^2) is bounded and bounded away from zero, because $|y_i \hat{f}_n(\mathbf{x}_i)|$ is uniformly bounded and $\phi''(z)$ satisfies the conditions in the theorem. Based on the implicit function theorem, $(\hat{h}_n, \hat{\alpha}_n)$ can be represented as $(\hat{h}_n, \hat{\alpha}_n)^T = \zeta(\gamma)$ for some function $\zeta(\cdot)$ (here we omit the other arguments), and $\zeta'(\gamma) = -(D^2)^{-1}(\hat{h}_n, 0)^T$. As a result, $|\zeta'(\gamma)| \leq M_3$ for some $M_3 > 0$. Then we have

$$\sup_n \{\|\hat{h}_n - \bar{h}_n\| + |\hat{\alpha}_n - \bar{\alpha}_n|\} \to 0 \text{ if } \gamma \to 0.$$

We need to estimate $\hat{f}_n - f^*$ which is represented as the sum of $\bar{f}_n - f^*$ and $\hat{f}_n - \bar{f}_n$. Expanding the function $\phi'(y\hat{f}_n)$ as

$$\phi'(y\hat{f}_n) = \phi'(y\bar{f}_n) + y\phi''(y\bar{f}_n)(\hat{f}_n - \bar{f}_n) + o(\hat{f}_n - \bar{f}_n)$$

and substituting them in Equation (20), we obtain (also keep in mind that (19) holds)

$$\int v\Big[\phi''(y\bar{f}_n) + o(1)\Big](\hat{f}_n - \bar{f}_n)dF_n(\mathbf{x}, y) = -\gamma\langle \hat{h}_n, v \rangle,$$

which is just

$$\frac{1}{n}\sum_{i=1}^{n}[v\phi''(y_i\bar{f}_n(\mathbf{x}_i)) + o(1)](\hat{f}_n - \bar{f}_n) = -\gamma\langle \hat{h}_n, v \rangle.$$

Note that we can chose v to be bounded, say, $\|v\| \leq 1$. From the property of $\phi''(z)$ and uniformly boundedness of $\{y_i\bar{f}_n(\mathbf{x}_i)\}$, we find that $v\phi''(y_i\bar{f}_n(\mathbf{x}_i)) + o(1)$ is uniformly bounded away from 0 if γ is sufficiently small. Hence we get $\sup_{n,\mathbf{x}} |\hat{f}_n(\mathbf{x}) - \bar{f}_n(\mathbf{x})| = O(\gamma)$, which implies $\int |\hat{f}_n(\mathbf{x}) - \bar{f}_n(\mathbf{x})| p(\mathbf{x})d\mathbf{x} = O(\gamma)$.

The next step is to estimate $\int |\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x})| p(\mathbf{x})d\mathbf{x}$. Note that

$$\mathbb{E}(vy\phi'((y\bar{f}_n(\mathbf{x})) = \int vy\phi'(y\bar{f}_n(\mathbf{x}))dF(\mathbf{x}, y).$$

Based on the central limit theorem, we have

$$0 = \int vy\phi'(y\bar{f}_n(\mathbf{x})))dF_n(\mathbf{x}, y) = \int vy\phi'(y\bar{f}_n(\mathbf{x})))dF(\mathbf{x}, y) + O_p(\frac{1}{\sqrt{n}}).$$

Together with Equation (18), we have

$$\int vy[\phi'(y\bar{f}_n(\mathbf{x})) - \phi'(yf^*(\mathbf{x}))]dF(\mathbf{x}, y) = O_p(\frac{1}{\sqrt{n}}),$$

$$\Big|\int v\phi''(yf^*(\mathbf{x}))(\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x}))dF(\mathbf{x}, y)\Big| = O_p(\frac{1}{\sqrt{n}}). \tag{21}$$

If $\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x}) \neq 0$, we substitute $v(\mathbf{x}) = \frac{\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x})}{|\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x})|}$ into the Equation (21). From the boundedness of $f^*(\mathbf{x})$ and the property stated at the beginning of the proof, we conclude

$$\int \left|\bar{f}_n(\mathbf{x}) - f^*(\mathbf{x})\right| p(\mathbf{x}) d\mathbf{x} = O_p(\frac{1}{\sqrt{n}}).$$

Summarily, we have

$$\int |\hat{f}_n(\mathbf{x}) - f^*(\mathbf{x})| p(\mathbf{x}) d\mathbf{x} = O(\gamma) + O_p(\frac{1}{\sqrt{n}}).$$

References

U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences (PNAS)*, 96 (12):6745–6750, 1999.

P. Bartlett and A. Tewari. Sparseness vs estimating conditional probabilities: some asymptotic results. *Journal of Machine Learning Research*, 8:775–790, 2007.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley and Sons, New York, 1994.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, U.K., 2000.

Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 21: 256–285, 1995.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–374, 2000.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classitcation of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286: 531–537, 1999.

T. Hastie, R. Tishirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, 2001.

Y. Lin. Tensor product space ANOVA models. *The Annals of Statistics*, 28(3):734–755, 2000.

Y. Lin. Support vector machines and the Bayes rule in classification. *Data Mining and Knowledge Discovery*, 6:259–275, 2002.

Y. Lin, G. Wahba, H. Zhang, and Y. Lee. Statistical properties and adaptive tuning of support vector machines. *Machine Learning*, 48:115–136, 2002.

B. K. Mallick, D. Ghosh, and M. Ghosh. Bayesian classification of tumours by using gene expression data. *Journal of the Royal Statistical Society Series B*, 67:219–234, 2005.

J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74, Cambridge, MA, 1999. MIT Press.

K. Rose, E. Gurewitz, and G. C. Fox. Statistical mechanics and phase transitions in clustering. *Physics Review Letter*, 65:945–948, 1990.

X. Shen, G. C. Tseng, X. Zhang, and W. H. Wong. On ψ-learning. *Journal of the American Statistical Association*, 98:724–734, 2003.

P. Sollich. Bayesian methods for support vector machines: evidence and predictive class probabilities. *Machine Learning*, 46:21–52, 2002.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67 – 93, 2001.

I. Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 4: 1071 – 1105, 2003.

I. Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128 – 142, 2005.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

G. Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

J. Wang, X. Shen, and Y. Liu. Probability estimation for large-margin classifiers. *Biometrika*, 95 (1):149–167, 2008.

L. Wang, J. Zhu, and H. Zou. Hybrid huberized support vector machines for microarray classification. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 983–990, 2007.

T. Zhang. Statistical behavior and consistency of classifications based on convex risk minimization. *Annals of Statistics*, 32:56–85, 2004.

T. Zhang and F. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, 4:5–31, 2001.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.

Journal of Machine Learning Research 13 (2012) 2735-2772 Submitted 8/11; Revised 1/12; Published 9/12

Linear Regression With Random Projections

Odalric-Ambrym Maillard ODALRIC.MAILLARD@INRIA.FR
Rémi Munos REMI.MUNOS@INRIA.FR
INRIA Lille Nord Europe
SequeL Project
40 avenue Halley
59650 Villeneuve d'Ascq, France

Editor: Sanjoy Dasgupta

Abstract

We investigate a method for regression that makes use of a randomly generated subspace $\mathcal{G}_P \subset \mathcal{F}$ (of finite dimension P) of a given large (possibly infinite) dimensional function space $\mathcal{F}$, for example, $L_2([0,1]^d;\mathbb{R})$. $\mathcal{G}_P$ is defined as the span of P random features that are linear combinations of a basis functions of $\mathcal{F}$ weighted by random Gaussian i.i.d. coefficients. We show practical motivation for the use of this approach, detail the link that this random projections method share with RKHS and Gaussian objects theory and prove, both in deterministic and random design, approximation error bounds when searching for the best regression function in $\mathcal{G}_P$ rather than in $\mathcal{F}$, and derive excess risk bounds for a specific regression algorithm (least squares regression in $\mathcal{G}_P$). This paper stresses the motivation to study such methods, thus the analysis developed is kept simple for explanations purpose and leaves room for future developments.

Keywords: regression, random matrices, dimension reduction

1. Introduction

We consider a standard regression problem. Thus let us introduce $\mathcal{X}$ an input space, and $\mathcal{Y} = \mathbb{R}$ the real line. We denote by $\mathcal{P}$ an unknown probability distribution over the product space $\mathcal{Z} = \mathcal{X} \times \mathbb{R}$ and by $\mathcal{P}_\mathcal{X}$ its first marginal, that is, $d\mathcal{P}_\mathcal{X}(x) = \int_\mathbb{R} \mathcal{P}(x,dy)$. In order for this quantity to be well defined we assume that $\mathcal{X}$ is a Polish space (i.e., metric, complete, separable), see Dudley (1989, Th. 10.2.2). Finally, let $L_{2,\mathcal{P}_\mathcal{X}}(\mathcal{X};\mathbb{R})$ be the space of real-valued functions on $\mathcal{X}$ that are squared integrable with respect to (w.r.t.) $\mathcal{P}_\mathcal{X}$, equipped with the quadratic norm

$$\|f\|_{\mathcal{P}_\mathcal{X}} \stackrel{\text{def}}{=} \sqrt{\int_\mathcal{X} f^2(x) dP_\mathcal{X}(x)}\,.$$

In this paper, we consider that $\mathcal{P}$ has some structure corresponding to a model of regression with random design; there exists a (unknown) function $f^\star : \mathcal{X} \to \mathbb{R}$ such that if $(x_n, y_n)_{n\leqslant N} \in \mathcal{X} \times \mathbb{R}$ are independently and identically distributed (i.i.d.) according to $\mathcal{P}$, then one can write

$$y_n = f^\star(x_n) + \eta_n\,,$$

where η_n is a centered noise, independent from $\mathcal{P}_\mathcal{X}$, introduced for notational convenience. In terms of random variables, we will often simply write $Y = f^\star(X) + \eta$ where $(X,Y) \sim \mathcal{P}$.

Let $\mathcal{F} \subset L_{2,\mathcal{P}_X}(\mathcal{X};\mathbb{R})$ be some given class of functions. The goal of the statistician is to build, from the observations only, a regression function $\widehat{f} \in \mathcal{F}$ that is closed to the so-called target function $f^\star$, in the sense that it has a low excess risk $R(f) - R(f^\star)$, where the risk of any $f \in L_{2,\mathcal{P}_X}(\mathcal{X};\mathbb{R})$ is defined as

$$R(f) \stackrel{\text{def}}{=} \int_{\mathcal{X}\times\mathbb{R}} (y - f(x))^2 d\mathcal{P}(x,y)\,.$$

Similarly, we introduce the empirical risk of a function f to be

$$R_N(f) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} [y_n - f(x_n)]^2\,,$$

and we define the empirical norm of f as $\|f\|_N \stackrel{\text{def}}{=} \sqrt{\frac{1}{N}\sum_{n=1}^{N} f(x_n)^2}$.

Function spaces and penalization. In this paper, we consider that $\mathcal{F}$ is an infinite dimensional space that is generated by the span over a denumerable family of functions $\{\varphi_i\}_{i\geqslant 1}$ of $L_{2,\mathcal{P}_X}(\mathcal{X};\mathbb{R})$: We call the $\{\varphi_i\}_{i\geqslant 1}$ the *initial features* and thus refer to $\mathcal{F}$ as to the initial feature space:

$$\mathcal{F} \stackrel{\text{def}}{=} \Big\{ f_\alpha(x) \stackrel{\text{def}}{=} \sum_{i\geqslant 1} \alpha_i \varphi_i(x), \|\alpha\| < \infty \Big\}.$$

Examples of initial features include Fourier basis, multi-resolution basis such as wavelets, and also less explicit features coming from a preliminary dictionary learning process.

In the sequel, for the sake of simplicity we focus our attention to the case when the target function $f^\star = f_{\alpha^\star}$ belongs to the space $\mathcal{F}$, in which case the excess risk of a function f can be written as $R(f) - R(f^\star) = \|f - f^\star\|_{\mathcal{P}_X}$. Since $\mathcal{F}$ is an infinite dimensional space, empirical risk minimization in $\mathcal{F}$ defined by $\operatorname{argmin}_{f\in\mathcal{F}} R_N(f)$ is certainly subject to overfitting. Traditional methods to circumvent this problem consider penalization techniques, that is, one searches for a function that satisfies

$$\widehat{f} = \arg\min_{f\in\mathcal{F}} R_N(f) + \text{pen}(f),$$

where typical examples of penalization include $\text{pen}(f) = \lambda \|f\|_p^p$ for $p = 1$ or 2, where λ is a parameter and usual choices for the norm are ℓ_2 (ridge-regression: Tikhonov 1963) and ℓ_1 (LASSO: Tibshirani 1994).

Motivation. In this paper we follow a complementary approach introduced in Maillard and Munos (2009) for finite dimensional space, called Compressed Least Squares Regression, and extended in Maillard and Munos (2010), which considers generating *randomly* a space $\mathcal{G}_P \in \mathcal{F}$ of finite dimension P and then returning an empirical estimate in $\mathcal{G}_P$. The empirical risk minimizer in $\mathcal{G}_P$, that is, $\arg\min_{g\in\mathcal{G}_P} R_N(g)$ is a natural candidate, but other choices of estimates are possible, based on traditional literature on regression when $P < N$ (penalization, projection, PAC-Bayesian estimates...). The generation of the space $\mathcal{G}_P$ makes use of random matrices, that have already demonstrated their benefit in different settings (see for instance Zhao and Zhang 2009 about spectral clustering or Dasgupta and Freund 2008 about manifold learning).

Our goal is first to give some intuition about this method by providing approximation error and simple excess risk bounds (which may not be the tightest possible ones as explained in Section 4.2)

for the proposed method, and also by providing links to other standards approaches, in order to encourage research in that direction, which, as showed in the next section, has already been used in several applications.

Outline of the paper. In Section 2, we quickly present the method and give practical motivation for investigating this approach. In Section 3, we give a short overview of Gaussian objects theory (Section 3.1), which enables us to show how to relate the choice of the initial features $\{\varphi_i\}_{i\geqslant 1}$ to the construction of standard function spaces via Gaussian objects (Section 3.2), and we finally state a useful version of the Johnson-Lindenstrauss Lemma for our setting (Section 3.3).

In Section 4, we describe a typical algorithm (Section 4.1), and then provide some quick survey of classical results in regression while discussing the validity of their assumptions in our setting (Section 4.2). Then our main results are stated in Section 4.3, where we provide bounds on approximation error of the random space $\mathcal{G}_P$ in the framework of regression with deterministic and random designs, and in Section 4.4, where we derive excess risk bounds for a specific estimate.

Section 5 provides some discussion about existing results and finally appendix A contains the proofs of our results.

2. Summary Of The Random Projection Method

From now on, we assume that the set of features $\{\varphi_i\}_{i\geqslant 1}$ are continuous and satisfy the assumption that,

$$\sup_{x\in\mathcal{X}} \|\varphi(x)\|^2 < \infty, \text{ where } \varphi(x) \stackrel{\text{def}}{=} (\varphi_i(x))_{i\geqslant 1} \in \ell_2 \text{ and } \|\varphi(x)\|^2 \stackrel{\text{def}}{=} \sum_{i\geqslant 1} \varphi_i(x)^2.$$

Let us introduce a set of P *random features* $(\psi_p)_{1\leqslant p\leqslant P}$ defined as linear combinations of the initial features $\{\varphi_i\}_{1\geqslant 1}$ weighted by random coefficients:

$$\psi_p(x) \stackrel{\text{def}}{=} \sum_{i\geqslant 1} A_{p,i}\varphi_i(x), \text{ for } 1 \leqslant p \leqslant P, \tag{1}$$

where the (infinitely many) coefficients $A_{p,i}$ are drawn i.i.d. from a centered distribution (e.g., Gaussian) with variance $1/P$. Then let us define $\mathcal{G}_P$ to be the (random) vector space spanned by those features, that is,

$$\mathcal{G}_P \stackrel{\text{def}}{=} \Big\{ g_\beta(x) \stackrel{\text{def}}{=} \sum_{p=1}^{P} \beta_p \psi_p(x), \beta \in \mathbb{R}^P \Big\}.$$

In the sequel, $\mathcal{P}_\mathcal{G}$ will refer to the law of the Gaussian variables, $\mathcal{P}_\eta$ to the law of the observation noise and $\mathcal{P}_\mathcal{Y}$ to the law of the observations. Remember also that $\mathcal{P}_\mathcal{X}$ refers to the law of the inputs.

One may naturally wish to build an estimate $g_{\widehat{\beta}}$ in the linear space $\mathcal{G}_P$. For instance in the case of deterministic design, if we consider the ordinary least squares estimate, that is, $\widehat{\beta} = \arg\min_{\beta\in\mathbb{R}^P} R_N(g_\beta)$, then we can derive the following result (see Section 4.4 for a similar result with random design):

Theorem 1 (Deterministic design) *Assuming that the random variable Y is such that $|Y| \leqslant B$, then for all $P \geqslant 1$, for all $\delta \in (0,1)$ there exists an event of $\mathcal{P}_\mathcal{Y} \times \mathcal{P}_\mathcal{G}$-probability larger than $1-\delta$ such*

that on this event, the excess risk of the least squares estimate $g_{\widehat{\beta}}$ is bounded as

$$\|g_{\widehat{\beta}} - f^\star\|_N^2 \leqslant \frac{12\log(8N/\delta)}{P}\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\varphi(x_n)\|^2 + \kappa B^2 \frac{P+\log(2/\delta)}{N}\,, \tag{2}$$

for some numerical constant $\kappa > 0$.

Example: Let us consider as an example the features $\{\varphi_i\}_{i\geqslant 1}$ to be a set of functions defined by rescaling and translation of a mother one-dimensional hat function (illustrated in Figure 1, middle column) and defined precisely in paragraph 3.2.2. Then in this case we can show that

$$\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\varphi(x_n)\|^2 \leqslant \frac{1}{2}\|f^\star\|_{H^1}^2\,,$$

where $H^1 = H^1([0,1])$ is the Sobolev space of order 1. Thus we deduce that the excess risk is bounded as $\|g_{\widehat{\beta}} - f^\star\|_N^2 = O(\frac{B\|f^\star\|_{H^1}\log(N/\delta)}{\sqrt{N}})$ for P of the order $\sqrt{N}$.

Similarly, the analysis given in paragraph 3.2.1 below shows that when the features $\{\varphi_i\}_{i\geqslant 1}$ are wavelets rescaled by a factor $\sigma_i = \sigma_{j,l} = 2^{-js}$ for some real number $s > 1/2$, where j,l are the scale and position index corresponding to the ith element of the family, and that the mother wavelet enables to generate the Besov space $\mathcal{B}_{s,2,2}([0,1])$ (see paragraph 3.2.1), then for some constant c, it holds that

$$\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\varphi(x_n)\|^2 \leqslant \frac{c}{1-2^{-2s+1}}\|f^\star\|_{s,2,2}^2\,.$$

Thus the excess risk in this case is bounded as $\|g_{\widehat{\beta}} - f^\star\|_N^2 = O(\frac{B\|f^\star\|_{s,2,2}\log(N/\delta)}{\sqrt{N}})$.

2.1 Comments

The second term in the bound (2) is a usual estimation error term in regression, while the first term comes from the additional approximation error of the space $\mathcal{G}_P$ w.r.t. $\mathcal{F}$. It involves the norm of the parameter $\alpha^\star$, and also the norm $\|\varphi(x)\|$ at the sample points.

The nice aspects of this result:

- The weak dependency of this bound with the dimension of the initial space $\mathcal{F}$. This appears implicitly in the terms $\|\alpha^\star\|^2$ and $\frac{1}{N}\sum_{n=1}^N \|\varphi(x_n)\|^2$, and we will show that for a large class of function spaces, these terms can be bounded by a function of the norm of $f^\star$ only.

- The result does not require any specific smoothness assumptions on the initial features $\{\varphi_i\}_{i\geqslant 1}$; by optimizing over P, we get a rate of order $N^{-1/2}$ that corresponds to the *minimax* rates under such assumptions up to logarithmic factors.

- Because the choice of the subspace $\mathcal{G}_P$ within which we perform the least-squares estimate is random, we avoid (with high probability) degenerated situations where the target function $f^\star$ cannot be well approximated with functions in $\mathcal{G}_P$. Indeed, in methods that consider a given deterministic finite-dimensional subspace $\mathcal{G}$ of the big space $\mathcal{F}$ (such as linear approximation using a predefined set of wavelets), it is often possible to find a target function $f^\star$ such that

$\inf_{g\in\mathcal{G}_P} \|f^\star - g\|_N$ is large. On the other hand when we use the random projection method, the random choice of $\mathcal{G}_P$ implies that for any $f^\star \in \mathcal{F}$, the approximation error $\inf_{g\in\mathcal{G}_P} \|f^\star - g\|_N$ can be controlled (by the first term of the bound (2)) in high probability. See section 5.2 for an illustration of this property. Thus the results we obtain is able to compete with non-linear approximation (Barron et al., 2008) or kernel ridge regression (Caponnetto and De Vito, 2007).

- In terms of numerical complexity, this approach is more efficient than non-linear regression and kernel ridge regression. Indeed, once the random space has been generated, we simply solve a least squares estimate in a low-dimensional space. The computation of the Gram matrix involves performing random projections (which can be computed efficiently for several choices of the random coefficients $A_{p,i}$, see Liberty et al. 2008; Ailon and Chazelle 2006; Sarlos 2006 and many other references therein). Numerical aspects of the algorithms are described in Section 5.4.

Possible improvements. As mentioned previously we do not make specific assumptions about the initial features $\{\varphi_i\}_{i\geqslant 1}$. However, considering smoothness assumptions on the features would enable to derive a better approximation error term (first term of the bound (2)); typically with a Sobolev assumption or order s, we would get a term of order P^{-2s} instead of P^{-1}. For simplicity of the presentation, we do not consider such assumptions here and report the general results only.

The $\log(N)$ factor may be seen as unwanted and one would like to remove it. However, this term comes from a variant of the Johnson-Lindenstrauss lemma combined with a union bound, and it seems difficult to remove it, unless the dimension of $\mathcal{F}$ is small (e.g., we can then use covers) but this case is not interesting for our purpose.

Possible extensions of the random projection method. It seems natural to consider other constructions than the use of i.i.d. Gaussian random coefficients. For instance we may consider Gaussian variables with variance σ_i^2/P different for each i instead of homeoscedastic variables, which is actually equivalent to considering the features $\{\varphi'_i\}_{i\geqslant 1}$ with $\varphi'_i = \sigma_i\varphi_i$ instead.

Although in the paper we develop results using Gaussian random variables, such method will essentially work similarly for matrices with sub-Gaussian entries as well.

A more important modification of the method would be to consider, like for data-driven penalization techniques, a data-dependent construction of the random space $\mathcal{G}_P$, that is, using a data-driven distribution for the random variable $A_{p,i}$ instead of a Gaussian distribution. However the analysis developed in this paper *will not* work for such modification, due to the fact we longer have independent variables, and thus a different analysis is required.

Illustration. In order to illustrate the method, we show in figure 1 three examples of initial features $\{\varphi_i\}$ (top row) and random features $\{\psi_p\}$ (bottom row). The first family of features is the basis of wavelet Haar functions. The second one consists of multi-resolution hat functions (see paragraph 3.2.2) and the last one shows multi-resolution Gaussian functions. For example, in the case of multi-resolution hat functions (middle column), the corresponding random features are Brownian motions. The linear regression with random projections approach described here simply consists in performing least-squares regression using the set of randomly generated features $\{\psi_p\}_{1\leqslant p\leqslant P}$ (e.g., Brownian motions).

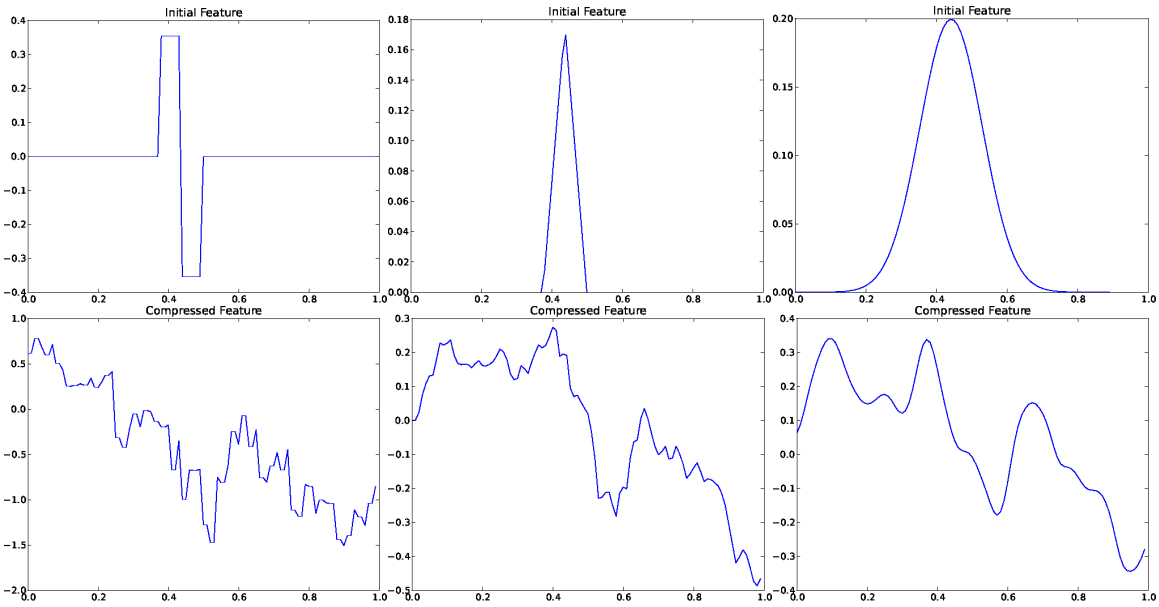


Figure 1: Three representative of initial features φ (top row) and a sample of a corresponding random feature ψ (bottom row). The initial set of features are (respectively) Haar functions (left), multi-resolution hat functions (middle) and multi-resolution Gaussian functions (right).

2.2 Motivation From Practice

We conclude this introduction with some additional motivation to study such objects coming from practical applications. Let us remind that the use of random projections is well-known in many domains and applications, with different names according to the corresponding field, and that the corresponding random objects are widely studied and used. Our contribution is to provide an analysis of this method in a regression setting.

For instance, in Sutton and Whitehead (1993) the authors mentioned such constructions under the name *random representation* as a tool for performing value function approximation in practical implementations of reinforcement learning algorithms, and provided experiments demonstrating the benefit of such methods. They also pointed out that such representations were already used in 1962 in Rosenblatt's perceptron as a preprocessing layer. See also Sutton (1996) for other comments concerning the practical benefit of "random collapsing" methods.

Another example is in image processing, when the initial features are chosen to be a wavelet (rescaled) system, in which case the corresponding random features $\{\psi_p\}_{1\leqslant p\leqslant P}$ are special cases of *random wavelet series*, objects that are well studied in signal processing and mathematical physics (see Aubry and Jaffard 2002; Durand 2008 for a study of the law of the spectrum of singularities of these series).

Noise model and texture generation. The construction of Gaussian objects (see paragraph 3.2.1) is highly flexible and enables to do automatic noise-texture generation easily, as explained in Deguy

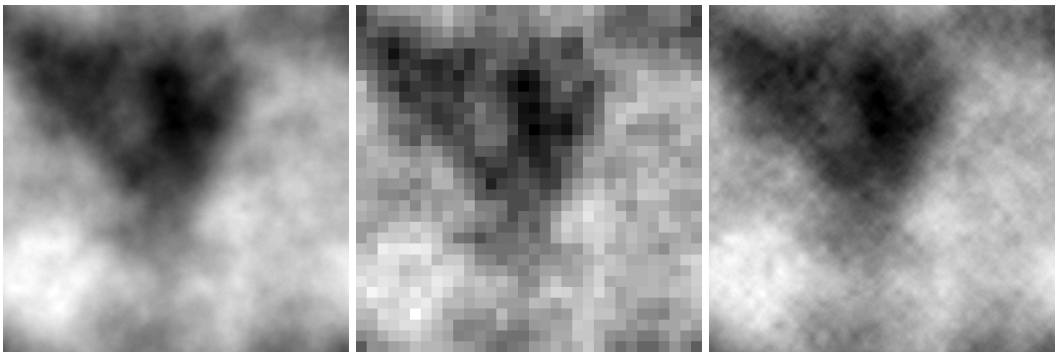


Figure 2: Example of an initial large texture (left), subsampled (middle), and possible recovery using regression with random projections (right)

and Benassi (2001). In their paper, the authors show that with the appropriate choice of the wavelet functions and when using rescaling coefficients of the form $\sigma_{j,l} = 2^{-js}$ with scale index j an position index l (see paragraph 3.2.1), where s is not a constant but is now a function of j and l, we can generate fractional Brownian motions, multi-scale fractional Brownian motions, and more generally what is called intermittent locally self-similar Gaussian processes.

In particular, for image texture generation they introduce a class of functions called *morphlets* that enables to perform approximations of intermittent locally self-similar Gaussian processes. These approximations are both numerically very efficient and have visually imperceptible differences to the targeted images, which make them very suitable for texture generation. The authors also allow other distributions than the Gaussian for the random variables ξ (which thus does not fit the theory presented here), and use this additional flexibility to produce an impressive texture generator.

Figure 2 illustrates an example performed on some simple texture model[1] where an image of size 512×512 is generated (two-dimensional Brownian sheet with Hurst index $H = 1.1$) (left) and then subsampled at 32×32 (middle), which provides the data samples for generating a regression function (right) using random features (generated from the symlets as initial features, in the simplest model when s is constant).

3. Gaussian Objects

We now describe some tools of Gaussian object theory that would be useful in later analysis of the method. Each random feature ψ_p built from Equation (1), when the coefficients are Gaussian, qualifies as a Gaussian object. It is thus natural to study some important features of Gaussian objects.

1. The authors wish to thank Pierre Chainais for performing experimental study of random projection methods applied to image processing, and for providing us with interesting pointers to related works.

3.1 Reminder of Gaussian Objects Theory

In all this section, $\mathcal{S}$ will refer to a vector space, $\mathcal{S}'$ to its topological dual, and $(\cdot,\cdot)$ to its duality product. The reader mostly interested in application of the random projection method may skip this section and directly go to Subsection 3.2 that provides examples of function spaces together with explicit construction of the abstract objects considered here.

Definition 2 (Gaussian objects) *A random variable $W \in \mathcal{S}$ is called a Gaussian object if for all $\nu \in \mathcal{S}'$, (ν, W) is a Gaussian (real-valued) variable. We further call any $a \in \mathcal{S}$ to be an* expectation *of W if*

$$\forall \nu \in \mathcal{S}', \mathbb{E}(\nu, W) = (\nu, a) < \infty,$$

and any $K : \mathcal{S}' \to \mathcal{S}$ to be a covariance operator *of W if*

$$\forall \nu, \nu' \in \mathcal{S}', Cov((\nu, W)(\nu', W)) = (\nu, K\nu') < \infty,$$

where Cov refers to the correlation between two real-valued random variables.

Whenever there exist such a and K, we say that W follows the law $\mathcal{N}(a,K)$. Moreover, W is called a centered Gaussian object if $a = 0$.

Kernel space. We only provide a brief introduction to this notion and refer the interested reader to Lifshits (1995) or Janson (1997) for refinements.

Let $I' : \mathcal{S}' \to L^2(\mathcal{S}, \mathcal{N}(0,K))$ be the canonical injection from the space of continuous linear functionals $\mathcal{S}'$ to the space of measurable linear functionals

$$L_2(\mathcal{S}; \mathbb{R}, \mathcal{N}(0,K)) = \left\{ z : \mathcal{S} \to \mathbb{R}, \mathbb{E}_{W \sim \mathcal{N}(0,K)} |z(W)|^2 < \infty \right\},$$

endowed with inner product $\langle z_1, z_2 \rangle = \mathbb{E}(z_1(W) z_2(W))$, that is, for any $\nu \in \mathcal{S}'$, I' is defined by $I'(\nu) = (\nu, \cdot)$. It belongs to $L_2(\mathcal{S}; \mathbb{R}, \mathcal{N}(0,K))$ since by definition of K we have $(\nu, K\nu) = \mathbb{E}(\nu, W)^2 < \infty$.

Then note that the space defined by $\mathcal{S}'_{\mathcal{N}} \stackrel{\text{def}}{=} \overline{I'(\mathcal{S}')}$, that is, the closure of the image of $\mathcal{S}'$ by I' in the sense of $L_2(\mathcal{S}; \mathbb{R}, \mathcal{N}(0,K))$, is a Hilbert space with inner product inherited from $L_2(\mathcal{S}; \mathbb{R}, \mathcal{N}(0,K))$.

Now under the assumption that I' is continuous (see Section 4.1 for practical conditions ensuring that this is the case), we can define the adjoint $I : \mathcal{S}'_{\mathcal{N}} \to \mathcal{S}$ of I', by duality. Indeed for any $\mu \in \mathcal{S}'$ and $z \in I'(\mathcal{S}')$, we have by definition that

$$(\mu, Iz) = \left\langle I'\mu, z \right\rangle_{\mathcal{S}'_{\mathcal{N}}} = \mathbb{E}_W((\mu, W) z(W)),$$

from which we deduce by continuity that $Iz = \mathbb{E}_W(W z(W))$. For the sake of clarity, this specifies for instance in the case when $\mathcal{S} = L_2(\mathcal{X}; \mathbb{R})$, for all $x \in \mathcal{X}$ as

$$(Iz)(x) = \mathbb{E}_W(W(x) z(W)).$$

Now that the two injection mappings I, I' have been defined, we are ready to provide the formal (though slightly abstract) definition for our main object of interest:

Definition 3 (Kernel space) *Provided that the mapping I' is continuous, then we define the* **kernel space** *of a centered Gaussian object W as $\mathcal{K} \stackrel{\text{def}}{=} I(\overline{I'(\mathcal{S}')}) \subset \mathcal{S}$.*

A more practical way of dealing with kernels is given by the two following lemmas that we use extensively in Section 3.2. First, the kernel space can be built alternatively based on a separable Hilbert space $\mathcal{H}$ as follows (Lifshits, 1995):

Lemma 4 (Construction of the Kernel space.) *Let $J : \mathcal{H} \to \mathcal{S}$ be an injective linear mapping such that $K = JJ'$, where J' is the adjoint operator of J. Then the kernel space of $\mathcal{N}(0,K)$ is $\mathcal{K} = J(\mathcal{H})$, endowed with inner product $\langle Jh_1, Jh_2 \rangle_H \stackrel{\text{def}}{=} \langle h_1, h_2 \rangle_{\mathcal{H}}$.*

We then conclude this section with the following Lemma from Lifshits (1995) that enables to define the expansion of a Gaussian object W.

Lemma 5 (Expansion of a Gaussian object) *Let $\{\varphi_i\}_{i\geqslant 1}$ be an orthonormal basis of $\mathcal{K}$ for the inner product $\langle \cdot, \cdot \rangle_{\mathcal{K}}$ and $\{\xi_i \stackrel{i.i.d.}{\sim} \mathcal{N}(0,1)\}_{i\geqslant 1}$. Then $\sum_{i=1}^{\infty} \xi_i \varphi_i$ is a Gaussian object following the law $\mathcal{N}(0,K)$. It is called an* **expansion** *for $\mathcal{N}(0,K)$.*

Note that from Lemma 4, one can build an orthonormal basis $\{\varphi_i\}_{i\geqslant 1}$ by defining, for all $i \geqslant 1$, $\varphi_i = Jh_i$ where $\{h_i\}_{i\geqslant 1}$ is an orthonormal basis of $\mathcal{H}$ and J satisfies conditions of Lemma 4.

3.2 Interpretation of Some Function Spaces with Gaussian Objects Theory

In this section, we precise the link between Gaussian objects theory and reproducing kernel Hilbert spaces (RKHS) in order to provide more intuition about such objects. Indeed in many cases, the kernel space of a Gaussian object is a RKHS. Note, however, that in general, depending on the Gaussian object we consider, the former space may also be a more general space for instance when the Hilbert assumption is dropped (see Canu et al. 2002 about RKS). Therefore, there is no one-to-one correspondence between RKHS and kernel spaces of Gaussian objects and it is worth explaining when the two notions coincide. More importantly, this section shows various examples of classical function spaces, related to the construction of the space $\mathcal{G}_P$ for different choices of initial features $\{\varphi_i\}_{i\geqslant 1}$, and that can be useful for applications.

3.2.1 GAUSSIAN OBJECTS WITH A SUPPORTING HILBERT SPACE

In this subsection only, we make the assumption that $\mathcal{S} = \mathcal{H}$ is a Hilbert space and we introduce $\{e_i\}_{i\geqslant 1}$ an orthonormal basis of $\mathcal{H}$. Let us now consider $\xi_i \sim \mathcal{N}(0,1)$ i.i.d., and positive coefficients $\sigma_i \geqslant 0$ such that $\sum_i \sigma_i^2 < \infty$. Since $\sum_i \sigma_i^2 < \infty$, the Gaussian object $W = \sum_i \xi_i \sigma_i e_i$ is well defined and our goal is to identify the kernel of the law of W.

To this aim we first identify the function I'. Since $\mathcal{S}$ is a Hilbert space, then its dual is $\mathcal{S}' = \mathcal{S}$, thus we consider $f = \sum_i c_i e_i \in \mathcal{S}'$ for some $c \in \ell_2$. For such an f, we deduce by the previous section that the injection mapping is given by $(I'f)(g) = \sum_i c_i (g, e_i)$, and that we also have

$$\|I'f\|^2_{\mathcal{S}'_{\mathcal{N}}} = \mathbb{E}\big((I'f, W)^2\big) = \mathbb{E}\Big(\big(\sum_{i\geqslant 1} \sigma_i \xi_i c_i\big)^2\Big) = \sum_{i\geqslant 1} \sigma_i^2 c_i^2 .$$

Now, since $\|f\|_{\mathcal{S}} = \|c\|_{\ell_2}$, the continuity of I' is insured by the assumption that $\sum_i \sigma_i^2 < \infty$, and thus I is defined as in the previous section. Therefore, a function in the space $\mathcal{K}$ corresponding to f is of the form $\sum_i \sigma_i c_i e_i$, and one can easily check that the kernel space of the law of W is thus given by

$$\mathcal{K} = \Big\{ f_c = \sum_{i\geqslant 1} c_i e_i \ ; \sum_{i\geqslant 1} \Big(\frac{c_i}{\sigma_i}\Big)^2 < \infty \Big\},$$

endowed with inner product $(f_c, f_d)_{\mathcal{K}} = \sum_{i\geqslant 1} \frac{c_i d_i}{\sigma_i^2}$.

Reproducing Kernel Hilbert Spaces (RKHS). Note that if we now introduce the functions $\{\varphi_i\}_{i\geqslant 1}$ defined by $\varphi_i \stackrel{\text{def}}{=} \sigma_i e_i \in \mathcal{H}$, then we get

$$\mathcal{K} = \Big\{ f_\alpha = \sum_{i\geqslant 1} \alpha_i \varphi_i \; ; \|\alpha\|_{l_2} < \infty \Big\},$$

endowed with inner product $(f_\alpha, f_\beta)_{\mathcal{K}} = \langle \alpha, \beta \rangle_{l_2}$. For instance, if we consider that $\mathcal{H} \subset L_{2,\mu}(\mathcal{X};\mathbb{R})$ for some reference measure μ, and that $\{e_i\}_{i\geqslant 1}$ are orthonormal w.r.t. $L_{2,\mu}(\mathcal{X};\mathbb{R})$, then $\mathcal{K}$ appears to be a RKHS that can be made fully explicit; its kernel is defined by $k(x,y) = \sum_{i=1}^\infty \sigma_i^2 e_i(x) e_i(y)$, and $\{\sigma_i\}_{i\geqslant 1}$ and $\{e_i\}_{i\geqslant 1}$ are trivially the eigenvalues and eigenfunctions of the integral operator $T_k : L_{2,\mu}(\mathcal{X}) \to L_{2,\mu}(\mathcal{X})$ defined by $(T_k(f))(x) = \int_{\mathcal{X}} k(x,y) f(y) d\mu(y)$.

Wavelet basis and Besov spaces. In this paragraph, we now apply the previous construction to the case when the $\{e_i\}_{i\geqslant 1}$ are chosen to be a wavelet basis of functions defined on $\mathcal{X} = [0,1]$ with reference measure μ being the Lebesgue measure. Let e denote the mother wavelet function, and let us write $e_{j,l}$ the ith element of the basis, with $j \in \mathbb{N}$ a scale index and $l \in \{0, \dots, 2^j - 1\}$ a position index, where we re-index all families indexed by i with the indice j,l. Let us define the coefficients $\{\sigma_i\}_{i\geqslant 1}$ to be exponentially decreasing with the scale index:

$$\sigma_{j,l} \stackrel{\text{def}}{=} 2^{-js} \text{ for all } j \geqslant 0 \text{ and } l \in \{0, \dots, 2^j - 1\},$$

where we introduced some positive real number s.

Now assume that for some $q \in \mathbb{N} \setminus \{0\}$ such that $q > s$, the mother wavelet function e belongs to $\mathcal{C}^q(\mathcal{X})$, the set of q-times continuously differentiable functions on $\mathcal{X}$, and admits q vanishing moments. The reason to consider such case is that the (homogeneous) Besov space $\mathcal{B}_{s,2,2}([0,1]^d)$ then admits the following known characterization (independent of the choice of the wavelets, see Frazier and Jawerth 1985; Bourdaud 1995):

$$\mathcal{B}_{s,2,2}(\mathcal{X};\mu) = \Big\{ f \in L_{2,\mu}(\mathcal{X}) ; \; \|f\|_{s,2,2}^2 \stackrel{\text{def}}{=} \sum_{j=1}^\infty \Big[2^{2js} \sum_{l=0}^{2^j-1} |\langle f, e_{j,l} \rangle|^2 \Big] < \infty \Big\}.$$

On the other hand, with the notations above, where in particular $\varphi_{j,l} = \sigma_{j,l} \varepsilon_{j,l}$, we deduce that the kernel space of the Gaussian object $W = \sum_{j,l} \xi_{j,l} \varphi_{j,l}$ (that we call a Scrambled wavelet), is simply the space

$$\mathcal{K} = \Big\{ f_\alpha = \sum_{j,l} \alpha_{j,l} \varphi_{j,l} \; ; \sum_{j,l} \alpha_{j,l}^2 < \infty \Big\},$$

and a straightforward computation shows that $\|\alpha\|_{l_2}^2 = \|f_\alpha\|_{s,2,2}^2$, so that $\mathcal{K} = \mathcal{B}_{s,2,2}(\mathcal{X};\mu)$. Moreover, assuming that the mother wavelet is bounded by a constant λ and has compact support $[0,1]$, then we have the property that is useful in view of our main Theorem

$$\sup_{x\in\mathcal{X}} \|\varphi(x)\|^2 \leqslant \frac{\lambda^2}{1 - 2^{-2s+1}}.$$

Note that a similar construction applies to the case when the orthonormal basis $\{e_i\}_{i\geqslant 1}$ is chosen to be a Fourier basis of functions, and the coeficients $\{\sigma_i\}_{i\geqslant 1}$ are chosen to be of the form $\sigma_i = i^{-s}$.

3.2.2 GAUSSIAN OBJECTS DEFINED BY A CARLEMAN EXPANSION

We now no longer assume that the supporting space $\mathcal{S}$ is a Hilbert space. In this case, it is still possible to generate a Gaussian object with kernel space being a RKHS by resorting to Carleman operators.

A Carleman operator is a linear injective mapping $J : \mathcal{H} \mapsto \mathcal{S}$ (where $\mathcal{H}$ is a Hilbert space) such that $J(h)(t) = \int \Gamma_t(s)h(s)ds$ where $(\Gamma_t)_t$ is a collection of functions of $\mathcal{H}$. As shown for instance in Canu et al. (2002); Saitoh (1988), there is a bijection between Carleman operators and the set of RKHSs. In particular, $J(\mathcal{H})$ is a RKHS.

A Gaussian object admitting $J(\mathcal{H})$ as a kernel space can be built as follows. By application of Lemma 5, we have that $\mathcal{K} = J(\mathcal{H})$ endowed with the inner product $\langle Jh_1, Jh_2\rangle_{\mathcal{K}} \stackrel{\text{def}}{=} \langle h_1, h_2\rangle_{\mathcal{H}}$ is the kernel space of $\mathcal{N}(0, JJ')$. Now, if we consider an orthonormal basis $\{e_i\}_{i\geqslant 1}$ of $\mathcal{H}$, an application of Lemma 5 shows that the functions $\{\varphi_i\}_{i\geqslant 1}$ defined by $\varphi_i \stackrel{\text{def}}{=} J(e_i)$ form an orthonormal basis of $J(\mathcal{H})$ and are such that the object $W = \sum_{i\geqslant 1} \xi_i \varphi$ is first a well-defined Gaussian object and then an expansion for the law $\mathcal{N}(0, JJ')$. We call this expansion a Carleman expansion. Note that this expansion is bottom-up whereas the Mercer expansion of a kernel via the spectral Theorem is top-down, see, for example, Zaanen (1960).

Cameron-Martin space. We apply as an example this construction to the case of the Brownian motion and the Cameron-Martin space.

Let $\mathcal{S} = \mathcal{C}([0,1])$ be the space of continuous real-valued functions of the unit interval. Then $\mathcal{S}'$ is the set of signed measures and we can define the dual product by $(\nu, f) = \int_{[0,1]} f d\nu$. It is straightforward to check that the Brownian motion indexed by $[0,1]$ is a Gaussian object $W \in \mathcal{S}$, with $a \equiv 0$ and K defined by $(K\nu)(t) = \int_{[0,1]} \min(s,t)\nu(ds)$.

Kernel space. We consider the Hilbert space $\mathcal{H} = L_2([0,1])$ and define the mapping $J : \mathcal{H} \mapsto \mathcal{S}$ by

$$(Jh)(t) = \int_{[0,t]} h(s)ds;$$

simple computations show that $(J'\nu)(t) = \nu([t,1])$, $K = JJ'$ and that J is a Carleman operator. Therefore, the kernel space $\mathcal{K}$ is equal to $J(L_2([0,1]))$, or more explicitly

$$\mathcal{K} = \left\{k \in H^1([0,1]); k(0) = 0\right\},$$

where $H^1([0,1])$ is the Sobolev space of order 1.

Expansion of the Brownian motion. We build a Carleman expansion for the Brownian motion thanks to the Haar basis of $L^2([0,1])$, whose image by J defines an orthonormal basis of $\mathcal{K}$; the Haar basis $(e_0, \{e_{j,l}\}_{j,l\in\mathbb{N}})$ is defined in a wavelet-way via a mother function $e(x) = \mathbb{I}_{[0,1/2[} - \mathbb{I}_{[1/2,1[}$ and father function $e_0(x) = \mathbb{I}_{[0,1]}(x)$ with functions $\{e_{j,l}\}_{j,l\in\mathbb{N}}$ defined for any scale $j \geqslant 1$ and translation index $0 \leqslant l \leqslant 2^j - 1$ by

$$e_{j,l}(x) \stackrel{\text{def}}{=} 2^{j/2} e(2^j x - l).$$

An orthonormal basis of the kernel space of the Brownian motion W and an expansion of W is thus obtained by

$$W = \sum_{j,l\geqslant 1} \xi_{j,l}\varphi_{j,l} + \xi_0\varphi_0,$$

$$\text{with } \varphi_{j,l}(x) = Je_{j,l}(x) = 2^{-j/2}\Lambda(2^j x - l) \text{ and } \varphi_0(x) = Je_0(x) = x,$$

where $\Lambda(x) = x\mathbb{I}_{[0,1/2[} + (1-x)\mathbb{I}_{[1/2,1[}$ is the mother hat function.

Bounded energy. Note that the rescaling factor inside $\varphi_{j,l}$ naturally appears as $2^{-j/2}$, and not as $2^{j/2}$ as usually defined in wavelet-like transformations. Note also that since the support of the mother function Λ is $[0,1]$, and also $\|\Lambda\|_\infty \leqslant 1/2$, then for any $x \in [0,1]^d$, for all j there exists at most one $l = l(x)$ such that $\varphi_{j,l}(x) \neq 0$, and we have the property that

$$\|\varphi(x)\|^2 = \sum_{j\geqslant 1} \varphi_{j,l(x)}(x)^2 \leqslant \sum_{j\geqslant 1} (2^{-j/2}\|\Lambda\|_\infty)^2 \leqslant \frac{1}{2}.$$

Remark 6 *This construction can be extended to the dimension $d > 1$ in at least two ways. Consider the space $\mathcal{S} = \mathcal{C}([0,1]^d)$, and the Hilbert space $\mathcal{H} = L_2([0,1]^d)$. Then if we define J to be the volume integral $(Jh)(t) = \int_{[0,t]} h(s)ds$ where $[0,t] \subset [0,1]^d$, this corresponds to the covariance operator defined by $(K\nu)(t) = \int_{[0,1]^d} \Pi_{i=1}^d \min(s_i,t_i)\nu(ds)$, that is, to the Brownian sheet defined by tensorization of the Brownian motion. The corresponding kernel space in this case is thus $\mathcal{K} = J(L^2([0,1]^d))$, endowed with the norm $\|f\|_\mathcal{K} = \|\frac{\partial^d f}{\partial x_1 \ldots \partial x_d}\|_{L^2([0,1]^d)}$. It corresponds to the* Cameron-Martin space *(Janson, 1997) of functions having a d-th order crossed (weak) derivative $\frac{\partial^d f}{\partial x_1 \ldots \partial x_d}$ that belongs to $L^2([0,1]^d)$, vanishing on the "left" boundary (edges containing 0) of the unit d-dimensional cube. A second possible extension that is not detailed here would be to consider the isotropic Brownian sheet.*

3.3 A Johnson-Lindenstrauss Lemma for Gaussian Objects

In this section, we derive a version of the Johnson-Lindenstrauss' lemma that applies to the case of Gaussian objects.

The original Johnson-Lindenstrauss' lemma can be stated as follows; its proof directly uses concentration inequalities (Cramer's large deviation Theorem from 1938) and may be found, for example, in Achlioptas (2003).

Lemma 7 *Let A be a $P \times F$ matrix of i.i.d. Gaussian $\mathcal{N}(0,1/P)$ entries. Then for any vector α in $\mathbb{R}^F$, the random (with respect to the choice of the matrix A) variable $\|A\alpha\|^2$ concentrates around its expectation $\|\alpha\|^2$ when P is large: for $\varepsilon \in (0,1)$, we have*

$$\mathbb{P}\Big(\|A\alpha\|^2 \geqslant (1+\varepsilon)\|\alpha\|^2\Big) \leqslant e^{-P(\varepsilon^2/4-\varepsilon^3/6)},$$
$$\text{and } \mathbb{P}\Big(\|A\alpha\|^2 \leqslant (1-\varepsilon)\|\alpha\|^2\Big) \leqslant e^{-P(\varepsilon^2/4-\varepsilon^3/6)}.$$

Remark 8 *Note the Gaussianity is not mandatory here, and this is also true for other distributions, such as:*

- *Rademacher distributions, that is, which takes values $\pm 1/\sqrt{P}$ with equal probability $1/2$,*
- *Distribution taking values $\pm\sqrt{3/P}$ with probability $1/6$ and 0 with probability $2/3$.*

What is very important is the scaling factor $1/P$ appearing in the variance of $\mathcal{N}(0,1/P)$.

This Lemma together with the measurability properties of Gaussian objects enable us to derive the following statement.

Lemma 9 *Let $\{x_n\}_{n\leqslant N}$ be N (deterministic) points of $\mathcal{X}$. Let $A : \ell_2(\mathbb{R}) \mapsto \mathbb{R}^P$ be the operator defined with i.i.d. Gaussian $\mathcal{N}(0,1/P)$ variables $(A_{i,p})_{i\geqslant 1, p\leqslant P}$, such that for all $\alpha \in \ell_2(\mathbb{R})$, then*

$$(A\alpha)_p = \sum_{i\geqslant 1} \alpha_i A_{i,p}\,.$$

Let us also define $\psi_p = \sum_{i\geqslant 1} A_{i,p}\varphi_i$, $f_\alpha = \sum_{i\geqslant 1}\alpha_i\varphi_i$ and $g_\beta = \sum_{p=1}^{P}\beta_p\psi_p$.

Then, A is well-defined and for all $P \geqslant 1$, for all $\varepsilon \in (0,1)$, with probability larger than $1 - 4Ne^{-P(\varepsilon^2/4-\varepsilon^3/6)}$ w.r.t. the Gaussian random variables,

$$\|f_\alpha - g_{A\alpha}\|_N^2 \leqslant \varepsilon^2\|\alpha\|^2\frac{1}{N}\sum_{n=1}^{N}\|\varphi(x_n)\|^2\,,$$

where we recall that by assumption, for any x, $\varphi(x) \stackrel{\text{def}}{=} (\varphi_i(x))_{i\geqslant 1}$ is in ℓ_2.

This result is natural in view of concentration inequalities, since for all $x \in \mathcal{X}$, the expectation satisfies $\mathbb{E}_{\mathcal{P}_\mathcal{G}}(g_{A\alpha}(x)) = f_\alpha(x)$ and the variance $\mathbb{V}_{\mathcal{P}_\mathcal{G}}(g_{A\alpha}(x)) = \frac{1}{P}(f_\alpha^2(x) + \|\alpha\|^2\|\varphi(x)\|^2)$. See Appendix A.1 for the full proof.

Note also that a natural idea in order to derive generalization bounds would be to derive a similar result uniformly over $\mathcal{X}$ instead of a union bound over the samples. However, while such extension would be possible for finite dimensional spaces $\mathcal{F}$ (by resorting to covers) these kind of results are not possible in the general case, since $\mathcal{F}$ is typically big.

More intuition. Let us now provide some more intuition about when such a result is interesting. In interesting situations described in Section 4 we consider a number of projections P lower than the number of data samples N, typically P is of order $\sqrt{N}$. Thus, it may seem counter-intuitive that we can approximate—at a set of N points—a function f_α that lies in a high (possibly infinite) dimensional space $\mathcal{F}$ by a function $g_{A\alpha}$ in a space $\mathcal{G}$ of dimension $P < N$.

Of course in general this is not possible. To illustrate this case, let us consider that there is no noise, assume that all points $(x_n)_{n\leqslant N}$ belong to the unit sphere, and that φ is the identity of $\mathcal{X} = \mathbb{R}^D$. Thus a target function f is specified by some $\alpha \in \mathbb{R}^D$ (where D is assumed to be large, that is, $D > N$) and the response values are $y_n = f_\alpha(x_n) = \alpha^T x_n$. Write $\widehat{y} \in \mathbb{R}^D$ the estimate $g_{A\alpha}$ at the points, that is, such that $\widehat{y}_n = g_{A\alpha}(x_n)$. In that case, the bound of Lemma 9 provides an average quadratic estimation error $\frac{1}{N}\|y-\widehat{y}\|^2$ of order $\dfrac{\log(N/\delta)}{P}||\alpha||^2$, with probability $1-\delta$.

On the other hand the zero-value regressor has an estimation error of

$$\frac{1}{N}\|y\|^2 = \frac{1}{N}\sum_{n=1}^{N}(\alpha^T x_n)^2 = \alpha^T S\alpha\,,\ \text{where } S \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} x_n x_n^T \in \mathbb{R}^{D\times D}\,.$$

This shows that the result of Lemma 9 is essentially interesting when $\dfrac{\alpha^T S\alpha}{\|\alpha\|^2} \gg \dfrac{\log(N/\delta)}{P}$, which may not happen in certain cases: Indeed if we specifically choose $x_n = e_n \in \mathbb{R}^D$, for $n \leqslant N \leqslant D$, where $(e_1,\dots,e_D)$ denotes the Euclidean basis of $\mathbb{R}^D$, then for such a choice, we have

$$\frac{\alpha^T S\alpha}{||\alpha||^2} = \frac{\sum_{d=1}^{N}\alpha_d^2}{N\sum_{d=1}^{D}\alpha_d^2} \leqslant \frac{1}{N} \leqslant \frac{\log(N/\delta)}{P}\,,$$

which means that the random projection method fails to recover a better solution than a trivial one. The reason why it fails is that in that case the points $\{x_n\}_{n\leqslant N}$ lie in a subspace of $\mathbb{R}^D$ of *high-dimension N*, that is, such that the information at any set of points does not help us to predict the value at any other point. Essentially, what Lemma 9 tells us is that the random projection method will work when the points $\{x_n\}_{n\leqslant N}$ lie in a vector subspace of smaller dimension $d_0 < N$ and that the d_0 corresponding coefficients of α contain most information about α (i.e., the other $D - d_0$ coordinates are small). Let us illustrate this case by considering the case where $x_n = e_{1+(n \bmod d_0)}$ for all $n \leqslant N$. In that case, we have (for N multiple of d_0),

$$\frac{\alpha^T S \alpha}{||\alpha||^2} = \frac{\sum_{d=1}^{d_0} \alpha_d^2}{d_0 \sum_{d=1}^{D} \alpha_d^2},$$

which is larger than $\frac{\log(N/\delta)}{P}$ whenever the components $\{\alpha_d\}_{d>d_0}$ decrease fast and P is large enough, in which case, the random projection method will work well.

Now introducing features, the condition says that the number of relevant features should be relatively small, in the sense that the parameter should mostly contain information at the corresponding coordinates, which is the case in many functional spaces, such as the Sobolev and Besov spaces (for which $D = \infty$) described in Section 2 and Section 3.2.1, paragraph "Wavelet basis and Besov spaces", for which $\|\alpha\|$ equals the norm of the function f_α in the corresponding space. Thus a "smooth" function f_α (in the sense of having a low functional norm) has a low norm of the parameter $\|\alpha\|$, and is thus well approximated with a small number of wavelets coefficients. Therefore, Lemma 9 is interesting and the random projection method will work in such cases (i.e., the additional projection error is controlled by a term of order $\|\alpha\|^2 \frac{\log(N/\delta)}{P}$).

4. Regression With Random Subspaces

In this section, we describe the construction of the random subspace $\mathcal{G}_P \subset \mathcal{F}$ defined as the span of the random features $\{\psi_p\}_{p\leqslant P}$ generated from the initial features $\{\varphi_i\}_{i\geqslant 1}$. This method was originally described in Maillard and Munos (2009) for the case when $\mathcal{F}$ is of finite dimension, and we extend it here to the non-obvious case of infinite dimensional spaces $\mathcal{F}$, which relies on the fact that the randomly generated features $\{\psi_p\}_{p\leqslant P}$ are well-defined Gaussian objects.

The next subsection is devoted to the analysis of the approximation power of the random features space. We first give a survey of existing results on regression together with the standard hypothesis under which they hold in section 4.2, then we describe in section 4.4 an algorithm that builds the proposed regression function and provide excess risk bounds for this algorithm.

4.1 Construction of Random Subspaces

Assumption on initial features. In this paper we assume that the set of features $\{\varphi_i\}_{i\geqslant 1}$ are continuous and satisfy the assumption that,

$$\sup_{x\in\mathcal{X}} \|\varphi(x)\|^2 < \infty, \text{ where } \|\varphi(x)\|^2 \stackrel{\text{def}}{=} \sum_{i\geqslant 1} \varphi_i(x)^2. \tag{3}$$

Note that all examples in Section 3 satisfy this condition.

Random features. The random subspace $\mathcal{G}_P$ is generated by building a set of P *random features* $\{\psi_p\}_{1\leqslant p\leqslant P}$ defined as linear combinations of the initial features $\{\varphi_i\}_{1\geqslant 1}$ weighted by random coefficients:

$$\psi_p(x) \stackrel{\text{def}}{=} \sum_{i\geqslant 1} A_{p,i}\varphi_i(x), \text{ for } 1 \leqslant p \leqslant P,$$

where the (infinitely many) coefficients $A_{p,i}$ are drawn i.i.d. from a centered distribution with variance $1/P$. Here we explicitly choose a Gaussian distribution $\mathcal{N}(0,1/P)$. Such a definition of the features ψ_p as an infinite sum of random variable is not obvious (this is an expansion of a Gaussian object) and we refer to the Section 3 for elements of theory about Gaussian objects and Lemma 5 for the expansion of a Gaussian object. It is shown that under Assumption (3), the random features are well defined. Actually, they are random samples of a centered Gaussian process indexed by the space $\mathcal{X}$ with covariance structure given by $\frac{1}{P}\langle\varphi(x),\varphi(x')\rangle$, where we use the notation $\langle u,v\rangle \stackrel{\text{def}}{=} \sum_i u_i v_i$ for two square-summable sequences u and v. Indeed, $\mathbb{E}_{A_p}[\psi_p(x)] = 0$, and

$$\mathrm{Cov}_{A_p}(\psi_p(x),\psi_p(x')) = \mathbb{E}_{A_p}[\psi_p(x)\psi_p(x')] = \frac{1}{P}\sum_{i\geqslant 1}\varphi_i(x)\varphi_i(x') = \frac{1}{P}\left\langle\varphi(x),\varphi(x')\right\rangle.$$

The continuity of each of the initial features $\{\varphi_i\}_{i\geqslant 1}$ guarantees that there exists a continuous version of the process ψ_p that is thus a Gaussian process.

Random subspace. We finally define $\mathcal{G}_P \subset \mathcal{F}$ to be the (random) vector space spanned by those features, that is,

$$\mathcal{G}_P \stackrel{\text{def}}{=} \left\{ g_\beta(x) \stackrel{\text{def}}{=} \sum_{p=1}^{P} \beta_p\psi_p(x), \beta\in\mathbb{R}^P \right\}.$$

We now want to compute a high probability bound on the excess risk of an estimator built using the random space $\mathcal{G}_P$. To this aim, we first quickly review known results in regression and see what kind of estimator can be considered and what results can be applied. Then we compute a high probability bound on the approximation error of the considered random space w.r.t. to initial space $\mathcal{F}$. Finally, we combine both bounds in order to derive a bound on the excess risk of the proposed estimate.

4.2 Reminder of Results on Regression

Short review of existing results. For the sake of completeness, we now review other existing results in regression that may or may not apply to our setting. Indeed it seems natural to apply existing results for regression to the space $\mathcal{G}_P$. For that purpose, we focus on the randomness coming from the data points only, and not from the Gaussian entries. We will thus consider in this subsection only a space $\mathcal{G}$ that is the span over a *deterministic* set of P functions $\{\psi_p\}_{p\leqslant P}$, and we will write, for a convex subset $\Theta\subset\mathbb{R}^P$,

$$\mathcal{G}_\Theta \stackrel{\text{def}}{=} \left\{ g_\theta\in\mathcal{G}; \theta\in\Theta \right\}.$$

Similarly, we write $g^\star \stackrel{\text{def}}{=} \operatorname*{argmin}_{g\in\mathcal{G}} R(g)$ and $g^\star_\Theta \stackrel{\text{def}}{=} \operatorname*{argmin}_{g\in\mathcal{G}_\Theta} R(g)$. Examples of well studied estimates are:

- $\widehat{g}^{ols} \stackrel{\text{def}}{=} \text{argmin}_{g\in\mathcal{G}} R_N(g)$, the ordinary least-squares (ols) estimate.
- $\widehat{g}^{erm} \stackrel{\text{def}}{=} \text{argmin}_{g\in\mathcal{G}_\Theta} R_N(g)$ the empirical risk minimizer (erm) that coincides with the ols when $\Theta = \mathbb{R}^P$.
- $\widehat{g}^{ridge} \stackrel{\text{def}}{=} \text{argmin}_{g\in\mathcal{G}} R_N(g) + \lambda\|\theta\|$, $\widehat{g}^{lasso} \stackrel{\text{def}}{=} \text{argmin}_{g\in\mathcal{G}} R_N(g) + \lambda\|\theta\|_1$.

We also introduce for convenience g_B, the truncation at level $\pm B$ of some $g \in \mathcal{G}$, defined by $g_B(x) \stackrel{\text{def}}{=} T_B[g(x)]$, where $T_B(u) \stackrel{\text{def}}{=} \begin{cases} u & \text{if } |u| \leqslant B, \\ B\,\text{sign}(u) & \text{otherwise.} \end{cases}$

There are at least 9 different theorems that one may want to apply in our setting. Since those theorems hold under some assumptions, we list them now. Unfortunately, as we will see, these assumptions are usually slightly too strong to apply in our setting, and thus we will need to build our own analysis instead.

Assumptions Let us list the following assumptions.

- Noise assumptions: (for some constants B, B_1, σ, ξ)
  (N_1) $|Y| \leqslant B_1$,
  (N_2) $\sup_{x\in\mathcal{X}} \mathbb{E}(Y|X=x) \leqslant B$,
  (N_3) $\sup_{x\in\mathcal{X}} \mathbb{V}(Y|X=x) \leqslant \sigma^2$,
  (N_4) $\forall k \geqslant 3$ $\sup_{x\in\mathcal{X}} \mathbb{E}(|Y|^k|X=x) \leqslant \sigma^2 k!\xi^{k-2}$.
- Moment assumptions: (for some constants σ, a, M)
  (M_1) $\sup_{x\in\mathcal{X}} \mathbb{E}([Y - g^\star_\Theta(X)]^2|X=x) \leqslant \sigma^2$,
  (M_2) $\sup_{x\in\mathcal{X}} \mathbb{E}(\exp[a|Y - g^\star_\Theta(X)|]|X=x) \leqslant M$,
  (M_3) $\exists g_0 \in \mathcal{G}_\Theta$ $\sup_{x\in\mathcal{X}} \mathbb{E}(\exp[a|Y - g_0(X)|]|X=x) \leqslant M$.
- Function space assumptions for $\mathcal{G}$: (for some constant D)
  (G_1) $\sup_{g_1,g_2\in\mathcal{G}_\Theta} \|g_1 - g_2\|_\infty \leqslant D$,
  (G_2) $\exists g_0 \in \mathcal{G}_\Theta$, known, such that $\|g_0 - g^\star_\Theta\|_\infty \leqslant D$.
- Dictionary assumptions:
  (D_1) $L = \max_{1\leqslant p\leqslant P} \|\psi_p\|_\infty < \infty$,
  (D_2) $L = \sup_{x\in\mathcal{X}} \|\psi(x)\|_2 < \infty$,
  (D_3) $\text{esssup}\|\psi(X)\|_2 \leqslant L$,
  (D_4) $L = \inf_{\{\psi'_p\}_{p\leqslant P}} \sup_{\theta\in\mathbb{R}^d-\{0\}} \frac{\|\sum_{p=1}^P \theta_p\psi'_p\|_\infty}{\|\theta\|_\infty} < \infty$ where the infimum is over all orthonormal basis of $\mathcal{G}$ w.r.t. to $L_{2,\mathcal{P}_\mathcal{X}}(\mathcal{X};\mathbb{R})$.
- Orthogonality assumptions:
  (O_1) $\{\psi_p\}_{p\leqslant P}$ is an orthonormal basis of $\mathcal{G}$ w.r.t. to $L_{2,\mathcal{P}_\mathcal{X}}(\mathcal{X};\mathbb{R})$,
  (O_2) $\det(\Psi) > 0$, where $\Psi = \mathbb{E}(\psi(X)\psi(X)^T)$ is the Gram matrix.
- Parameter space assumptions:
  (P_1) $\sup_{\theta\in\Theta} \|\theta\|_\infty < \infty$,
  (P_2) $\|\theta^\star\|_1 \leqslant S$ where $\theta^\star$ is such that $g_{\theta^\star} = g^\star_\Theta$ and S is known,
  (P_3) $\sup_{\theta\in\Theta} \|\theta\|_2 \leqslant 1$.

Theorem 10 (Györfi et al. 2002) *Let $\Theta = \mathbb{R}^P$. Under assumption (N_2) and (N_3), the truncated estimator $\widehat{g}_L = T_L(\widehat{g}^{ols})$ satisfies*

$$\mathbb{E}R(\widehat{g}_L) - R(f^{(reg)}) \leqslant 8[R(g^*) - R(f^{(reg)})] + \kappa \frac{(\sigma^2 \vee B^2) P \log(N)}{N},$$

where κ is some numerical constant and $f^{(reg)}(x) \stackrel{\text{def}}{=} \mathbb{E}(Y|X=x)$.

Theorem 11 (Catoni 2004) *Let $\Theta \subset \mathbb{R}^P$. Under assumption (M_3), (G_1) and (O_2), there exists constants $C_1, C_2 > 0$ (depending only on a, M and D) such that with probability $1-\delta$, provided that*

$$\left\{ g \in \mathcal{G} \,;\, R_N(g) \leqslant R_N(\widehat{g}^{ols}) + C_1 \frac{P}{N} \right\} \subset \mathcal{G}_\Theta,$$

then the ordinary least squares estimate satisfies

$$R(\widehat{g}^{ols}) - R(g^\star_\Theta) \leqslant C_2 \frac{P + \log(\delta^{-1}) + \log(\frac{det\widehat{\Psi}}{det\Psi})}{N},$$

where $\widehat{\Psi} = \frac{1}{N}\sum_{i=1}^N \psi(X_i)\psi(X_i)^T$ is the empirical Gram matrix.

Theorem 12 (Audibert and Catoni 2010 from Alquier 2008) *Let $\Theta = \mathbb{R}^P$. Under assumption (N_1) and (G_2), there exists a randomized estimate $\widehat{g}$ that only depends on g_0, L, C, such that for all $\delta > 0$, with probability larger than $1-\delta$ w.r.t. all sources of randomness,*

$$R(\widehat{g}) - R(g^\star) \leqslant \kappa(B_1^2 + D^2) \frac{P \log(3\nu_{\min}^{-1}) + \log(\log(N)\delta^{-1})}{N},$$

where κ does not depend on P and N, and $\nu_{\min}$ is the smallest eigenvalue of Ψ.

Theorem 13 (Koltchinskii 2006) *Let $\Theta \subset \mathbb{R}^P$. Under assumption (N_1), (D_3) and (P_3), $\widehat{g}^{erm}$ satisfies, for any $\delta > 0$ with probability higher than $1-\delta$,*

$$R(\widehat{g}^{erm}) - R(g^\star_\Theta) \leqslant \kappa(B_1 + L)^2 \frac{rank(\Psi) + \log(\delta^{-1})}{N},$$

where κ is some constant.

Theorem 14 (Birgé and Massart 1998) *Let $\Theta \subset \mathbb{R}^P$. Under assumption (M_3), (G_1) and (D_4), for all $\delta > 0$ with probability higher than $1-\delta$,*

$$R(\widehat{g}^{erm}) - R(g^\star_\Theta) \leqslant \kappa(a^{-2} + D^2) \frac{P \log(2 + (L^2/N) \wedge (N/P)) + \log(\delta^{-1})}{N},$$

where κ is some constant depending only on M.

Theorem 15 (Tsybakov 2003) *Let $\Theta = \mathbb{R}^P$. Under assumption (N_2), (N_3) and (O_1), the projection estimate $\widehat{g}^{proj}$ satisfies*

$$\mathbb{E}(R(\widehat{g}^{proj})) - R(g^\star) \leqslant \frac{(\sigma^2 + B^2)P}{N}.$$

Theorem 16 (Caponnetto and De Vito 2007) *Under assumption (M_2) and (D_2), for all $\delta > 0$ for $\lambda = PL^2 \log^2(\delta^{-1})/N \leqslant \nu_{\min}$, with probability higher than $1-\delta$,*

$$R(\widehat{g}^{ridge}) - R(g^\star_\Theta) \leqslant \kappa(a^{-2} + \frac{\lambda L^2 \|\theta^\star\|^2 \log^2(\delta^{-1})}{\nu_{\min}}) \frac{P \log^2(\delta^{-1})}{N},$$

where κ is some constant depending only on M.

Theorem 17 (Alquier and Lounici 2011) *Let $\Theta = \mathbb{R}^P$ and define for all $\alpha \in (0,1)$ the prior $\pi_\alpha(J) = \frac{\alpha^{|J|}}{\sum_{i=0}^N \alpha^i} \binom{P}{|J|}^{-1}$ for all $J \subset 2^P$. Under assumption $(N_2), (N_3)$, (N_4), (D_1) and (P_2), by setting $\lambda = \frac{N}{2C}$ where*

$$C \stackrel{\text{def}}{=} \max\{64\sigma^2 + (2B + L(2S + \frac{1}{N}))^2, 64[\xi + 2B + L(2S + \frac{1}{N})]L(2S + \frac{1}{N})\},$$

the randomized aggregate estimator $\widehat{g}$ defined in Alquier and Lounici (2011) based on prior π_α satisfies, for any $\delta > 0$ with probability higher than $1-\delta$,

$$R(\widehat{g}) - R(g^\star_\Theta) \leqslant C \frac{S^\star \log(\frac{(S+c)eNP}{\alpha S^\star}) + \log(2\delta^{-1}/(1-\alpha))}{N} + \frac{3L^2}{N^2},$$

where $S^\star = \|\theta^\star\|_0$.

Theorem 18 (Audibert and Catoni 2010) *Let $\Theta \subset \mathbb{R}^P$. Under assumption (M_1), (G_1) and (P_1) so that one can define the uniform probability distribution over Θ, there exists a random estimator $\widehat{g}$ (drawn according to a Gibbs distribution $\widehat{\pi}$) that satisfies, with probability higher than $1-\delta$ w.r.t. all source of randomness,*

$$R(\widehat{g}) - R(g^\star_\Theta) \leqslant (2\sigma + D)^2 \frac{16.6P + 12.5\log(2\delta^{-1})}{N}.$$

Note that Theorem 10 and Theorem 15 provide a result in expectation only, which is not enough for our purpose, since we need high probability bounds on the excess risk in order to be able to handle the randomness of the space $\mathcal{G}_P$.

Assumptions satisfied by the random space $\mathcal{G}_P$

We now discuss the assumptions that are satisfied in our setting where $\mathcal{G}$ is a random space $\mathcal{G}_P$ built from the random features $\{\psi_p\}_{p \leqslant P}$, in terms of assumptions on the underlying initial space $\mathcal{F}$.

- The noise assumptions (N) do not concern $\mathcal{G}$.
- The moment assumptions (M) are not restrictive. By combining similar assumptions on $\mathcal{F}$, the results on approximation error of Section 4.3 can be shown to hold (with different constants).
- Assumptions (P) are generally too strong. For (P_1), the reason is that there is no high probability link between $\|A\alpha\|_\infty$ and $\|\alpha\|$ for usual norms. Now even if $\alpha^\star$ is sparse or has low l_1-norm, this does not imply this is the case for $\beta^\star = \text{argmin}_{\beta \in \mathbb{R}^P} R(g_\beta)$ or $A\alpha^\star$ in general, thus (P_2) cannot be assumed either. Finally (P_3) may be assumed in some cases: Let us assume that we know that $\|\alpha^\star\|_2 \leqslant 1$. Then $\|A\alpha^\star\|_2 \leqslant 1+\varepsilon$ with high probability, thus it is enough to consider the space $\mathcal{G}_P(\Theta)$ with parameter space $\Theta = \{\beta; \|\beta\|_2 \leqslant (1+\varepsilon)\}$, and thus $A\alpha^\star \in \Theta$ with high probability.

- Assumptions (G) are strong assumptions. The reason is that it is difficult to relate the vector coefficient $\beta^\star$ or even $A\alpha^\star$ to the vector coefficient $\alpha^\star$ of $f^\star = f_{\alpha^\star}$ in l_∞ norm. Thus even if we know some f_0 close to $f^\star$ in ℓ_∞-norm, this does not imply that we can build a function g_0 close to $g^\star = g_{\beta^\star}$.

- Assumptions (D) will not be valid a.s. w.r.t. the law of the Gaussian variables. The assumptions (D_1) and (D_4) are difficult to satisfy since they concern $\|.\|_\infty$. For assumption (D_2) and (D_3), we have the property that for each x, $\|\psi(x)\|_2^2$ is close to $\|\varphi(x)\|_2^2$ with high probability. However, we need here a uniform result over $x \in \mathcal{X}$ which seems difficult to get since the space $\mathcal{F}$ is actually big (not of finite dimension).

- Assumptions (O), which are typically strong assumptions for specific features φ appear to be almost satisfied. The reason is due to the covariance structure of the random features. Indeed whatever the distribution $\mathcal{P}_\mathcal{X}$ (independent of $\mathcal{P}_\mathcal{G}$), we have that $\langle \psi_p, \psi_q \rangle$ concentrates around

$$\mathbb{E}_{\mathcal{P}_\mathcal{G}} \langle \psi_p, \psi_q \rangle = \frac{1}{P} \| \sum_{i \geqslant 1} \varphi_i \|_{\mathcal{P}_\mathcal{X}}^2 \delta_{p,q} \,,$$

  where $\delta_{p,q}$ is the Kronecker symbol between p and q. Thus the orthogonality assumption is satisfied with high probability. Note that the knowledge of $\mathcal{P}_\mathcal{X}$ is still needed in order to rescale the features and obtain orthonormality. Similar argument shows that (O_2) is also valid.

As a consequence, only Theorems 10 and 15 would apply safely, but unfortunately these Theorems do not give results in high probability.

In the next two sections, we derive similar results but in high probability with assumptions that correspond to our setting. We provide a hand-made Theorem that makes use of the technique introduced in Györfi et al. (2002) and that can be applied without too restrictive assumptions, although not being optimal in terms of constant and logarithmic factors.

4.3 Approximation Power of Random Spaces

We assume from now on that we are in the case when $f^\star = f_{\alpha^\star} \in \mathcal{F}$.

Theorem 19 (Approximation error with deterministic design) *For all $P \geqslant 1$, for all $\delta \in (0,1)$ there exists an event of $\mathcal{P}_\mathcal{G}$-probability higher than $1-\delta$ such that on this event,*

$$\inf_{g \in \mathcal{G}_P} \|f^\star - g\|_N^2 \leqslant 12 \frac{\log(4N/\delta)}{P} \|\alpha^\star\|^2 \frac{1}{N} \sum_{n=1}^N \|\varphi(x_n)\|^2 \,.$$

Theorem 20 (Approximation error with random design) *Under assumption (N_2), then for all $P \geqslant 1$, for all $\delta \in (0,1)$, the following bound holds with $\mathcal{P}_\mathcal{G}$-probability higher than $1-\delta$:*

$$\inf_{g \in \mathcal{G}_P} \|f^\star - T_B(g)\|_{\mathcal{P}_\mathcal{X}}^2 \leqslant 25 \frac{\|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2}{P} \Big(1 + \frac{1}{2} \log \big(\frac{P \log(8P/\gamma^2\delta)}{18 \gamma^2 \delta} \big) \Big) \,,$$

where $\gamma \stackrel{\text{def}}{=} \frac{1}{B} \|\alpha^\star\| \sup_x \|\varphi(x)\|$ and T_B is the truncation operator at level B.

The result is not trivial because of the randomness of the space $\mathcal{G}_P$. Thus in order to keep the explanation simple, the proof (detailed in the Appendix) makes use of Hoeffding's Lemma only, which relies on the bounded assumption of the features (which can be seen either as a nice assumption, since it is simple and easy to check, or as a too strong assumption for some cases). Note that this result can be further refined by making use, for instance, of moment assumptions on the feature space instead.

4.4 Excess Risk of Random Spaces

In this section, we analyze the excess risk of the random projection method. Thus for a proposed random estimate $\widehat{g}$, we are interested in bounding $R(\widehat{g}) - R(f^\star)$ in high probability with respect to any source of randomness.

4.4.1 Regression Algorithm

From now on we consider the estimate $\widehat{g}$ to be the least-squares estimate $g_{\widehat{\beta}} \in \mathcal{G}_P$ that is the function in $\mathcal{G}_P$ with minimal empirical error, that is,

$$g_{\widehat{\beta}} \stackrel{\text{def}}{=} \arg\min_{g_\beta \in \mathcal{G}_P} R_N(g_\beta), \tag{4}$$

and that is the solution of a least-squares regression problem, that is, $\widehat{\beta} = \Psi^\dagger Y \in \mathbb{R}^P$ with matrix-wise notations, where $Y \in \mathbb{R}^N$ is here the vector of observations (not to be confused with the random variable Y that shares the same notation), Ψ is the $N \times P$-matrix composed of the elements: $\Psi_{n,p} \stackrel{\text{def}}{=} \psi_p(x_n)$, and $\Psi^\dagger$ is the Moore-Penrose pseudo-inverse[2] of Ψ. The final prediction function $\widehat{g}(x)$ is the truncation (at level $\pm B$) of $g_{\widehat{\beta}}$, that is, $\widehat{g}(x) \stackrel{\text{def}}{=} T_B[g_{\widehat{\beta}}(x)]$.

In the next subsection, we provide excess risk bounds w.r.t. $f^\star$ in $\mathcal{G}_P$.

4.4.2 Regression with Deterministic Design

Theorem 21 *Under assumption (N_1), then for all $P \geqslant 1$, for all $\delta \in (0,1)$ there exists an event of $\mathcal{P}_\mathcal{Y} \times \mathcal{P}_\mathcal{G}$-probability higher than $1-\delta$ such that on this event, the excess risk of the estimator $g_{\widehat{\beta}}$ is bounded as*

$$\|f^\star - g_{\widehat{\beta}}\|_N^2 \leqslant \frac{12\log(8N/\delta)}{P}\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\psi(x_n)\|^2 + \kappa B_1^2 \frac{P + \log(2/\delta)}{N},$$

for some numerical constant $\kappa > 0$.

Note that from this theorem, we deduce (without further assumptions on the features $\{\psi_i\}_{i\geqslant 1}$) that for instance for the choice $P = \frac{\sqrt{N}}{\log(N/\delta)}$ then

$$\|f^\star - g_{\widehat{\beta}}\|_N^2 \leqslant \kappa' \Big[\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\psi(x_n)\|^2 \sqrt{\frac{\log(N/\delta)}{N}} + \frac{\log(1/\delta)}{N} \Big],$$

for some positive constant κ'. Note also that whenever an upper-bound on the square terms $\|\alpha^\star\|^2 \frac{1}{N}\sum_{n=1}^N \|\psi(x_n)\|^2$ is known, this can be used in the definition of P in order to improve this bound.

2. In the full rank case when $N \geqslant P$, $\Psi^\dagger = (\Psi^T\Psi)^{-1}\Psi^T$.

4.4.3 REGRESSION WITH RANDOM DESIGN

In the regression problem with random design, the analysis of the excess risk of a given method is not straightforward, since the assumptions to apply standard techniques may not be satisfied without further knowledge on the structure of the features. In a general case, we can use the techniques introduced in Györfi et al. (2002), which yields to the following (not optimal) result:

Theorem 22 *Under assumption* (N_1) *and* (N_2)*, provided that* $N\log(N) \geqslant \frac{4}{P}$ *(thus whenever* $\min(N,P) \geqslant 2$*), then with* $\mathcal{P}_{\mathcal{G}} \times \mathcal{P}$*-probability at least* $1-\delta$*,*

$$R(T_B(g_{\widehat{\beta}})) - R(f^\star) \leqslant \kappa\Big[\frac{\log(12N/\delta)}{P}\|\alpha^\star\|^2 \sup_{x\in\mathcal{X}}\|\varphi(x)\|^2 + \max\{B_1^2, B^2\}\frac{P + P\log(N) + \log(3/\delta)}{N}\Big],$$

for some positive constant κ*.*

Let us now provide some intuition about the proof of this result. We first start by explaining what does not work. A natural idea in order to derive this result would be to consider the following decomposition:

$$R(T_B(g_{\widehat{\beta}})) - R(f^\star) \leqslant [R(T_B(g_B^\star)) - R(f^\star)] + [R(T_B(g_{\widehat{\beta}})) - R(T_B(g_B^\star))],$$

where $g_B^\star \in \operatorname*{argmin}_{g\in\mathcal{G}} R(T_B(g)) - R(f^\star)$.

Indeed the first term is controlled on an event $\Omega_{\mathcal{G}}$ of high $\mathcal{P}_{\mathcal{G}}$-probability by Theorem 20, and since $R(g_{\widehat{\beta}}) - R(g_B^\star) \leqslant R(g_{\widehat{\beta}}) - R(g^\star)$, the second term is controlled for each fixed $\omega_{\mathcal{G}} \in \Omega_{\mathcal{G}}$ with high $\mathcal{P}$-probability by standard Theorems for regression, provided that we can relate $R(T_B(g_{\widehat{\beta}})) - R(T_B(g_B^\star))$ to $R(g_{\widehat{\beta}}) - R(g_B^\star)$. Thus by doing the same careful analysis of the events involved, this should lead to the desired result.

However, the difficulty lies first in ensuring that the conditions of application of standard Theorems are satisfied with high $\mathcal{P}_{\mathcal{G}}$-probability and then in relating the excess risk of the truncated function to that of the non-truncated ones, since it is not true in general that $R(T_B(g_{\widehat{\beta}})) - R(T_B(g_B^\star)) \leqslant R(g_{\widehat{\beta}}) - R(g_B^\star)$. Thus we resort to a different decomposition in order to derive our results. The sketch of proof of Theorem 22 actually consists in applying the following lemma.

Lemma 23 *The following decomposition holds for all* $C > 0$

$$\|T_B(g_{\widehat{\beta}}) - f^\star\|_{\mathcal{P}_X}^2 \leqslant C\|f^\star - g_{\tilde{\beta}}\|_N^2 + C\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2 + \sup_{g\in\mathcal{G}}\left(\|f^\star - T_B(g)\|_{\mathcal{P}_X}^2 - C\|f^\star - T_B(g)\|_N^2\right),$$

where $g_{\tilde{\beta}} = \Pi_{\|.\|_N}(f^\star, \mathcal{G})$ *and* $g_{\widehat{\beta}} = \Pi_{\|.\|_N}(Y, \mathcal{G})$ *are the projections of the target function* $f^\star$ *and observation* Y *onto the random linear space* $\mathcal{G}$ *with respect to the empirical norm* $\|.\|_N$*.*

We then call the first term $\|f^\star - g_{\tilde{\beta}}\|_N^2$ an approximation error term, the second $\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2$ a noise error term and the third one $\sup_{g\in\mathcal{G}}\left(\|f^\star - T_B(g)\|_{\mathcal{P}_X}^2 - C\|f^\star - T_B(g)\|_N^2\right)$ an estimation of the error term.

In order to prove Theorem 22, we then control each of these terms: We apply Lemma 19 to the first term, Lemma 24 below to the second term and finally Theorem 11.2 of Györfi et al. (2002) to the last term with $C = 8$, and the result follows by gathering all the bounds.

Let us now explain the contribution to each of the three terms in details.

Approximation error term The first term, $\|f^\star - g_{\tilde{\beta}}\|_N^2$, is an approximation error term in empirical norm, it contains the number of projections as well as the norm of the target function. This term plays the role of the approximation term that exists for regression with penalization by a factor $\lambda\|f\|^2$. This term is controlled by application of Theorem 19 conditionally on the random samples, and then w.r.t. all source of randomness by independence of the Gaussian random variables with the random samples.

Noise error term The second term, $\|g_{\tilde{\beta}} - g_{\hat{\beta}}\|_N^2$, is an error term due to the observation noise η. This term classically decreases at speed $\frac{D\sigma^2}{N}$ where σ^2 is the variance of the noise and D is related to the log entropy of the space of function $\mathcal{G}$ considered. Without any more assumption, we only know that this is a linear space of dimension P, so this term finally behaves like $\frac{P\sigma^2}{N}$, but note that this dependency with P may be improved depending on the knowledge about the functions ψ (for instance, if $\mathcal{G}$ is included in a Sobolev space of order s, we would have $P^{1/2s}$ instead of P).

Lemma 24 *Under assumption (N_1), then for each realization of the Gaussian variables, with $\mathcal{P}$-probability higher than $1-\delta$, the following holds true:*

$$\|g_{\tilde{\beta}} - g_{\hat{\beta}}\|_N^2 \leqslant 6B_1^2 \frac{1616P + 200\log(6/\delta) + \log(3/\delta)}{N}.$$

Note that we may consider different assumptions on the noise term. Here we considered only that the noise is upper-bounded as $\|\eta\|_\infty \leqslant B_1$, but another possible assumption is that the noise has finite variance σ^2 or that the tail of the distribution of the noise behaves nicely, for example, that $\|\eta\|_{\psi_\alpha} \leqslant B$, where ψ_α is the Orlicz norm or order α, with $\alpha = 1$ or 2.

Estimation error term The third term, $\sup_{g\in\mathcal{G}_P}(\|f^\star - T_B(g)\|_{\mathcal{P}_X}^2 - C\|f^\star - T_B(g)\|_N^2)$, is an estimation of the error term due to finiteness of the data. This term also depends on the log entropy of the space of functions, thus the same remark applies to the dependency with P as for the noise error term. We bound the third term by applying Theorem 11.2 of Györfi et al. (2002) to the class of functions $\mathcal{G}^0 = \{f^\star - T_B(g), g \in \mathcal{G}_P\}$, for fixed random Gaussian variables. Note that for all $f \in \mathcal{G}^0$, $\|f\|_\infty \leqslant 2B$. The precise result of Györfi et al. (2002) is the following :

Theorem 25 *Let $\mathcal{F}$ be a class of functions $f : \mathbb{R}^d \to \mathbb{R}$ bounded in absolute value by B. Let $\varepsilon > 0$. Then*

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \|f\|_{\mathcal{P}_X} - 2\|f\|_N > \varepsilon\right) \leqslant 3\mathbb{E}\left(\mathcal{N}\left(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{F}, \|.\|_{2N}\right)\right)\exp\left(-\frac{N\varepsilon^2}{288B^2}\right).$$

We now have the following lemma whose proof is given in the Appendix:

Lemma 26 *Assuming that $N\log(N) \geqslant \frac{4}{P}$, then for each realization of the Gaussian variables, with $\mathcal{P}$-probability higher than $1-\delta$, the following holds true:*

$$\sup_{g\in\mathcal{G}_P} \|f^\star - T_B(g)\|_{\mathcal{P}_X}^2 - 8\|f^\star - T_B(g)\|_N^2 \leqslant (24B)^2 \frac{4\log(3/\delta) + 2P\log(N)}{N}.$$

5. Discussion

In this section, we now provide more insights about the main results of this paper by reminding some closely related existing works, showing some numerical illustration of the method and discussing some numerical issues.

5.1 Non-linear Approximation

In the work of Barron et al. (2008), the authors provide excess risk bounds for greedy algorithms (i.e., in a non-linear approximation setting). The precise result they derive in their Theorem 3.1 is reported now, using the notations of section 4.2:

Theorem 27 (Barron et al. 2008) *Consider spaces $\{\mathcal{G}_P\}_{P\geqslant 1}$ generated respectively by the span of features $\{e_p\}_{p\leqslant P}$ with increasing dimension P (thus $\Theta=\mathbb{R}^P$ for each P). For each $\mathcal{G}_P$ we compute a corresponding greedy empirical estimate $\widehat{g}_P\in\mathcal{G}_P$ provided by some algorithm (see Barron et al., 2008), then we define $\widehat{P}=\operatorname{argmin}\|y-T_{B_1}\widehat{f}_P\|_N^2+\kappa\frac{P\log(N)}{N}$ for some constant κ, and finally define $\widehat{g}=T_{B_1}(\widehat{g}_{\widehat{P}})$, and fix some P_0.*

Under assumption (N_1), there exists κ_0 depending only on B_1 and a where $P_0=\llcorner N^a\lrcorner$ such that if $\kappa\geqslant\kappa_0$, then for all $P>0$ and for all functions g_θ in $\mathcal{G}_{P_0}$, the estimator $\widehat{g}$ satisfies

$$\mathbb{E}R(\widehat{g})-R(f^{(reg)})\leqslant 2[R(g_\theta)-R(f^{(reg)})]+8\frac{\|\theta\|_1^2}{P}+C\frac{P\log N}{N},$$

where the constant C only depends on κ, B_1 and a.

The bound is thus similar to that of Theorem 22 in Section 4.4. One difference is that this bound contains the l_1 norm of the coefficients θ^* while the ℓ_2 norm of the coefficients $\alpha^\star$ appears in our setting. We leave as an open question to understand whether this difference is a consequence of the non-linear aspect of their approximation or if it results from the different assumptions made about the approximation spaces, in terms of rate of decrease of the coefficients.

The main difference is actually about the tractability of the proposed estimator, since the result of Theorem 27 relies on greedy estimation that is computationally heavy while on the other hand, random projection is cheap (see Subsection 5.4).

5.2 Adaptivity

Randomization enables to define approximation spaces such that the approximation error, either in expectation or in high probability on the choice of the random space, is controlled, whatever the measure $\mathcal{P}$ that is used to assess the performance. This is specially interesting in the regression setting where $\mathcal{P}$ is unknown. As mentioned in the introduction, because the choice of the subspace $\mathcal{G}_P$ within which we perform the least-squares estimate is *random*, we avoid (with high probability) degenerated situations where the target function $f^\star$ cannot be well approximated with functions in $\mathcal{G}_P$. Indeed, in methods that consider a given *deterministic* finite-dimensional subspace $\mathcal{G}$ of the big space $\mathcal{F}$ (such as linear approximation using a predefined set of wavelets), it is often possible to find a target function $f^\star$ such that $\inf_{g\in\mathcal{G}_P}\|f^\star-g\|_N$ is large, whereas using the random projection method, the random choice of $\mathcal{G}_P$ implies that for any $f^\star\in\mathcal{F}$, the approximation error $\inf_{g\in\mathcal{G}_P}\|f^\star-g\|_N$ can be controlled (by the first term of the bound (2)) in high probability. We now illustrate this property on a simple example.

Example Let us consider a very peaky (a spot) distribution $\mathcal{P}$. Regular linear approximation, say with wavelets (see, e.g., DeVore, 1997), will most probably miss the specific characteristics of $f^\star$ at the spot, since the first wavelets have large support. On the contrary, the random features $\{\psi_p\}_{p\leqslant P}$ that are functions that contain (random combinations of) all wavelets, will be able to detect correlations between the data and some high frequency wavelets, and thus discover relevant features of $f^\star$ at the spot. This is illustrated in the numerical experiment below.

Here $\mathcal{P}$ is a very peaky Gaussian distribution and $f^\star$ is a 1-dimensional periodic function. We consider as initial features $\{\varphi_i\}_{i\geqslant 1}$ the set of hat functions defined in Section 3.2.2. Figure 3 shows the target function $f^\star$, the distribution $\mathcal{P}$, and the data $(x_n,y_n)_{1\leqslant n\leqslant 100}$ (left plots). The middle plots represents the least-squares estimate $\widehat{g}$ using $P=40$ scrambled objects $\{\psi_p\}_{1\leqslant p\leqslant 40}$ that are here Brownian motions. The right plots shows the least-squares estimate using the initial features $\{\varphi_i\}_{1\leqslant i\leqslant 40}$. The top figures represent a high level view of the whole domain $[0,1]$. No method is able to learn $f^\star$ on the whole space (this is normal since the available data are only generated from a peaky distribution). The bottom figures shows a zoom $[0.45,0.51]$ around the data. Least-squares regression using scrambled objects is able to learn the structure of $f^\star$ in terms of the measure $\mathcal{P}$, while least-squares regression with the initial features completely fails.

5.3 Other Related Work

In Rahimi and Recht (2008, 2007), the authors consider, for a given parameterized function $\Phi : \mathcal{X}\times\Theta\to\mathbb{R}$ bounded by 1, and a probability measure μ over Θ, the space $\mathcal{F}$ of functions $f(x)=\int_\Theta \alpha(\theta)\Phi(x,\theta)d\theta$ such that $\|f\|_\mu = \sup_\theta|\frac{\alpha(\theta)}{\mu(\theta)}|<\infty$. They show that this is a dense subset of the RKHS with kernel $k(x,y)=\int_\Theta\mu(\theta)\Phi(x,\theta)\Phi(y,\theta)d\theta$, and that if $f\in\mathcal{F}$, then with high probability over $\{\theta_p\}_{p\leqslant P}\overset{i.i.d}{\sim}\mu$, there exist coefficients $\{c_p\}_{p\leqslant P}$ such that $\widehat{f}(x)=\sum_{p=1}^P c_p\Phi(x,\theta_p)$ satisfies $\|\widehat{f}-f\|_2^2\leqslant O(\frac{\|f\|_\mu}{\sqrt{P}})$. The method is analogous to the construction of the empirical estimates $g_{A\alpha}\in\mathcal{G}_P$ of function $f_\alpha\in\mathcal{K}$ in our setting. Indeed we may formally identify $\Phi(x,\theta_p)$ with $\psi_p(x)=\sum_i A_{p,i}\varphi_i(x)$, θ_p with the sequence $(A_{p,i})_i$, and the distribution μ with the distribution of this infinite sequence. However, in our setting we do not require the condition $\sup_{x,\theta}\Phi(x,\theta)\leqslant 1$ to hold and the fact that Θ is a set of infinite sequences makes the identification tedious without the Gaussian random functions theory used here. Anyway, we believe that this link provides a better mutual understanding of both approaches (i.e., Rahimi and Recht 2008 and this paper).

5.4 Tractability

In practice, in order to build the least-squares estimate, one needs to compute the values of the random features $\{\psi_p\}_{1\leqslant p\leqslant P}$ at the data points $\{x_n\}_{1\leqslant n\leqslant N}$, that is, the matrix $\Psi=(\psi_p(x_n))_{p\leqslant P,n\leqslant N}$. Moreover, due to finite memory and precision of computers, numerical implementations can only handle a finite number F of initial features $\{\varphi_i\}_{1\leqslant i\leqslant F}$.

Approximation error Using a finite F introduces an additional approximation (squared) error term in the final excess risk bounds. This additional error that is due to the numerical approximation is of order $O(F^{-\frac{2s}{d}})$ for a wavelet basis adapted to $H^s([0,1]^d)$ and can be made arbitrarily small, for example, $o(N^{-1/2})$, whenever the depth of the wavelet dyadic-tree is bigger than $\frac{\log N}{d}$. Our main concern is thus about efficient computation.

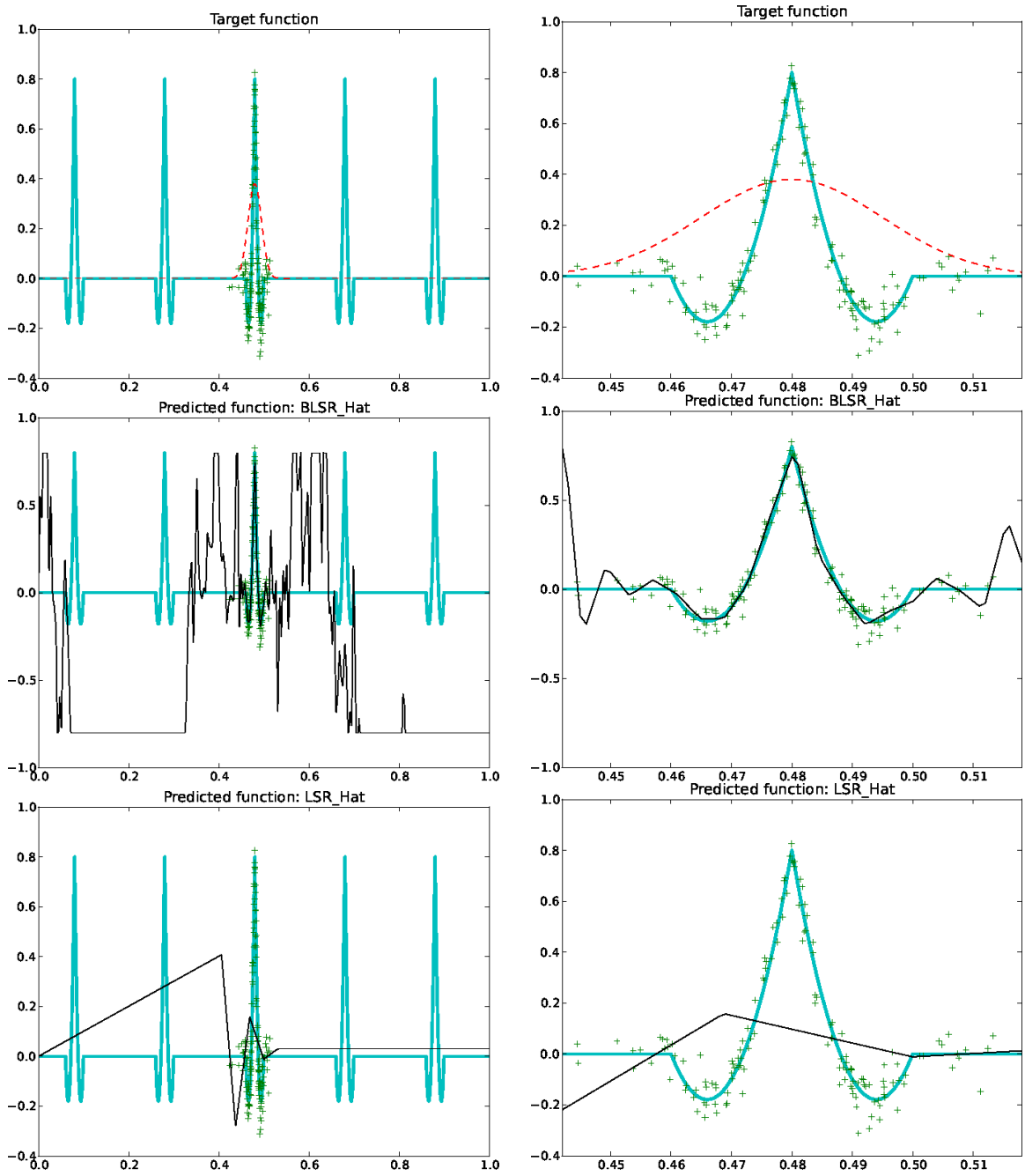


Figure 3: Least squares estimates of $f^\star$, using $N = 100$ data generated from a peaky distribution $\mathcal{P}$ (dashed line in top plots), using 40 Brownian motions $\{\psi_p\}$ (middle plots) and using 40 hat functions $\{\varphi_i\}$ (bottom plots). The target function $f^\star$ is plotted with thick line while the two estimates are plotted with thin line. The right column shows a zoom around the data.

Numerical complexity In Maillard and Munos (2009) it was mentioned that the computation of Ψ, which makes use of the random matrix $A = (A_{p,i})_{p \leqslant P, i \leqslant F}$, has a complexity $O(FPN)$.

In the multi-resolution schemes described now, provided that the mother function has compact support (such as the hat functions), we can significantly speed up the computation of the matrix Ψ by resorting to a *tree-based lazy expansion*, that is, where the expansion of the random features $\{\psi_p\}_{p \leqslant P}$ is built only when needed for the evaluation at the points $\{x_n\}_n$. Note that in the specific case of wavelets, we can even think to combine random projection with tools like fast wavelet transform which would be even faster (which we do not do here for simplicity and generality purpose).

Example: Consider the example of the scrambled wavelets. In dimension 1, using a wavelet dyadic-tree of depth H (i.e., $F = 2^{H+1}$), the numerical cost for computing Ψ is $O(HPN)$ (using one tree per random feature). Now, in dimension d the classical extension of one-dimensional wavelets uses a family of $2^d - 1$ wavelets, thus requires $2^d - 1$ trees each one having 2^{dH} nodes. While the resulting number of initial features F is of order $2^{d(H+1)}$, thanks to the lazy evaluation (notice that one never computes all the initial features), one needs to expand at most one path of length H per training point, and the resulting complexity to compute Ψ is $O(2^d HPN)$. Thus the method is linear with N and reduces the amount of computation by an exponential factor (from 2^{dH} to $2^d H$).

Note that one may alternatively use the so-called sparse-grids instead of wavelet trees, which have been introduced by Griebel and Zenger (see Zenger, 1990; Bungartz and Griebel, 2004). The main result is that one can reduce significantly the total number of features to $F = O(2^H H^d)$ (while preserving a good approximation for sufficiently smooth functions). Similar lazy evaluation techniques can be applied to sparse-grids.

Thus, using $P = O(\sqrt{N})$ random features, we deduce that the complexity of building the matrix Ψ is at most $O(2^d N^{3/2} \log N)$. Then in order to solve the least squares system, one has to compute $\Psi^T \Psi$, that has cost at most $O(P^2 N)$, and then solve the system by inversion, which has numerical cost $O(P^{2.376})$ by Coppersmith and Winograd (1987). Thus, with $P = O(\sqrt{N})$, the overall cost of the algorithm is at most $O(2^d N^{3/2} \log N + N^2)$, without using any fancy computations designed for random matrices, and the numerical complexity to make a new prediction is at most $O(2^d N^{1/2} \log(N))$.

Acknowledgments

The authors want to thank *Pierre Chainais* and *Olivier Degris* for interesting pointers to the literature in image processing and applied reinforcement learning.
This work was supported by the French National Research Agency through the EXPLO-RA project No ANR-08-COSI-004 and the CompLACS project by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No 270327.

Appendix A. Technical Details

In this technical section we gathered the proofs of the important Lemmas and of the main Theorems 19, 20, 21 and 22.

A.1 Proof of Lemma 9

Proof *Step 1.* First, we derive a result similar to Lemma 7 that holds for dot products, by polarisation of the Euclidean norm. The precise statement for our purpose is the following one.

Lemma 28 *Let A be a $P \times F$ matrix of i.i.d. elements drawn from one of the previously defined distributions. Let $(u_n)_{1\leqslant n\leqslant N}$ and v be $N+1$ vectors of $\mathbb{R}^F$.*

Then for any $\varepsilon \in (0,1)$, with probability at least $1-4Ne^{-P(\varepsilon^2/4-\varepsilon^3/6)}$, simultaneoulsy for all $n \leqslant N$,

$$|Au_n \cdot Av - u_n \cdot v| \leqslant \varepsilon \|u_n\| \, \|v\| \, .$$

We apply Lemma 7 to any couple of vectors $u+w$ and $u-w$, where u and w are vectors of norm 1. By polarisation, we have that

$$\begin{aligned} 4Au \cdot Aw &= \|Au+Aw\|^2 - \|Au-Aw\|^2 \\ &\leqslant (1+\varepsilon)\|u+w\|^2 - (1-\varepsilon)\|u-w\|^2 \\ &= 4u \cdot w + \varepsilon(\|u+w\|^2 + \|u-w\|^2) \\ &= 4u \cdot w + 2\varepsilon(\|u\|^2 + \|w\|^2) = 4u \cdot w + 4\varepsilon \, , \end{aligned}$$

fails with probability $2e^{-P(\varepsilon^2/4-\varepsilon^3/6)}$ (we applied the previous lemma twice at line 2).

Thus for each $n \leqslant N$, we have with same probability:

$$Au_n \cdot Av \leqslant u_n \cdot v + \varepsilon \|u_n\| \, \|v\| .$$

Now the symmetric inequality holds with the same probability, and using a union bound for considering all $(u_n)_{n\leqslant N}$, we have that

$$|Au_n \cdot Av - u_n \cdot v| \leqslant \varepsilon \|u_n\| \, \|v\| ,$$

holds for all $n \leqslant N$, with probability $1-4Ne^{-P(\varepsilon^2/4-\varepsilon^3/6)}$.

Step 2. We now extend this Lemma to the case of infinite sequences. This is made possible thanks to the measurability properties of Gaussian Objects. Indeed, for any given F, Lemma 28 applies to the two truncated sequences $\overline{\alpha}_F = (\alpha_1, \ldots, \alpha_F)$ and $\overline{\varphi}_F(x_n) = (\varphi_1(x_n), \ldots, \varphi_F(x_n))$; this gives that for all n simultaneoulsy,

$$|\sum_{i=1}^{F} \alpha_i \varphi_i(x_n) - \frac{1}{P} \sum_{p=1}^{P} \big(\sum_{i=1}^{F} \xi_{i,p} \alpha_i\big) \big(\sum_{i=1}^{F} \xi_{i,p} \varphi_i(x_n)\big)| \leqslant \varepsilon \, \|\overline{\alpha}_F\| \, \|\overline{\varphi}_F(x_n)\| \, ,$$

happens with probability higher than $1-4Ne^{-P(\varepsilon^2/4-\varepsilon^3/6)}$, where we introduced $\xi_{i,p} \stackrel{\text{def}}{=} \sqrt{P} A_{i,p} \sim \mathcal{N}(0,1)$ in order to avoid confusion with the section on Gaussian objects. Now by the assumption that $\alpha \in \ell_2(\mathbb{R})$ and $\varphi(x) \in \ell_2(\mathbb{R})$ for all x, then the Gaussian objects $\sum_{i=1}^{\infty} \xi_{i,p} \alpha_i$ and $\sum_{i=1}^{\infty} \xi_{i,p} \varphi_i(x_n)$ are well-defined square integrable random variables. Thus, taking the limit of the above inequality when F tends to ∞ yields that with same probability, for all $n \leqslant N$

$$|f_\alpha(x_n) - g_{A\alpha}(x_n)| \leqslant \varepsilon \, \|\alpha\| \, \|\varphi(x_n)\| \, .$$

■

A.2 Proof of Lemma 24

Proof We can bound the noise term $\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2$ using a simple Chernoff bound together with a chaining argument. Indeed, by definition of $g_{\tilde{\beta}}$ and $g_{\widehat{\beta}}$, if we introduce the noise vector η defined by $\eta = Y - f$, we have

$$\begin{aligned}
\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2 &= \langle g_{\tilde{\beta}} - g_{\widehat{\beta}}, \eta \rangle_N \\
&= \frac{1}{N}\sum_{i=1}^N \eta_i (g_{\tilde{\beta}} - g_{\widehat{\beta}})(X_i) \\
&\leqslant \Big(\sup_{g \in \mathcal{G}} \frac{\frac{1}{N}\sum_{i=1}^N \eta_i g(X_i)}{\|g\|_N} \Big) \|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N \\
&\leqslant \Big(\sup_{g \in \mathcal{G}} \frac{\frac{1}{N}\sum_{i=1}^N \eta_i g(X_i)}{\|g\|_N} \Big)^2 .
\end{aligned}$$

Thus, we focus on the set $\mathcal{G}^1 = \{g \in \mathcal{G}; \|g\|_N = 1\}$. Note that since $\mathcal{G}^1$ is a sphere in a space of dimension P, its ε-packing number in empirical norm is bounded above by $\mathcal{M}(\varepsilon, \mathcal{G}^1, \|.\|_N) \leqslant \mathcal{N}(\varepsilon/2, \mathcal{G}^1, \|.\|_N) \leqslant \mathcal{N}(\varepsilon/2, \{g \in \mathcal{G}; \|g\|_N \leqslant 1\}, \|.\|_N) \leqslant (\frac{4}{\varepsilon} + 1)^P \leqslant \max(\frac{5}{\varepsilon}, 5)^P$, where $\mathcal{N}$ refers here to the covering number.

We now introduce for convenience the following notation, for fixed Gaussian random variables and data points $(X_i)_{i=1..n}$:

$$\rho(t) \stackrel{\text{def}}{=} \mathbb{P}_Y\Big(\exists g \in \mathcal{G} \frac{\frac{1}{N}\sum_{i=1}^N \eta_i g(X_i)}{\|g\|_N} > t\Big) \quad = \quad \mathbb{P}_Y\Big(\exists g \in \mathcal{G}^1 \frac{1}{N}\sum_{i=1}^N \eta_i g(X_i) > t\Big).$$

For $j = 0 \ldots \infty$, let us consider ε_j-packings C_j of $\mathcal{G}^1$ for the empirical norm $\|.\|_N$, with $C_0 = g_0$, such that C_{j+1} is a refinement of C_j and $\varepsilon_j \leqslant \varepsilon_{j-1}$. Then for a given $g \in \mathcal{G}^1$, we define $g_j = \Pi(g, C_j)$ the projection of g into C_j, for the norm $\|g\|_N$. Thus, $g - g_0 = (g - g_J) + \sum_{j=1}^J (g_j - g_{j-1})$. Note that since by definition of $\mathcal{G}^1$ we have $\|g - g_0\|_N \leqslant 2$, we need to consider $\varepsilon_0 \geqslant 2$.

Thus if we now introduce real numbers γ and $(\gamma_j)_{j \geqslant 1}$ such that $\sum_{j=1}^J \gamma_j \leqslant \gamma$, then we have

$$\begin{aligned}
\rho(\gamma t_1 + t_2 + t_3) &\leqslant \mathbb{P}\Big(\exists g \in \mathcal{G}^1 \frac{1}{N}\sum_{i=1}^N \eta_i (g - g_0)(X_i) > \gamma t_1 + t_2\Big) + \exp(-\frac{t_3^2 N}{2B_1^2}) \\
&\leqslant \mathbb{P}\Big(g \in \exists \mathcal{G}^1 \frac{1}{N}\sum_{i=1}^N \eta_i (g - g_J)(X_i) + \\
&\qquad \sum_{j=1}^J \frac{1}{N}\sum_{i=1}^N \eta_i (g_j - g_{j-1})(X_i) \geqslant \sum_{j=1}^J \gamma_j t_1 + t_2\Big) + \exp(-\frac{t_3^2 N}{2B_1^2}),
\end{aligned}$$

where we applied Hoeffding's inequality in the first line. We further have:

$$\begin{aligned}\rho(\gamma t_1+t_2+t_3) &\leqslant \sum_{j=1}^{J}\mathbb{P}\Big(\exists g\in\mathcal{G}^1\frac{1}{N}\sum_{i=1}^{N}\eta_i(g_j-g_{j-1})(X_i)>t_1\gamma_j\Big)\\ &\quad+\exp(-\frac{t_2^2N}{2B_1^2\varepsilon_J^2})+\exp(-t_3^2N2B_1^2)\\ &\leqslant \mathbb{E}\sum_{j=1}^{J}\mathcal{M}_j\mathcal{M}_{j-1}\mathbb{P}\Big(\frac{1}{N}\sum_{i=1}^{N}\eta_i(g_j-g_{j-1})(X_i)>t_1\gamma_j\Big)\\ &\quad+\exp(-\frac{t_2^2N}{2B_1^2\varepsilon_J^2})+\exp(-\frac{t_3^2N}{2B_1^2}),\end{aligned}$$

where we introduced for convenience the notation $\mathcal{M}_j \stackrel{\text{def}}{=} \mathcal{M}(\varepsilon_j,\mathcal{G}^1,\|.\|_N)$. Now, note that since $\varepsilon_j\leqslant\varepsilon_{j-1}$, then $\mathcal{M}_{j-1}\leqslant\mathcal{M}_j$. Note also that $\|g_j-g_{j-1}\|_N\leqslant\eta_j$ since C_j is a refinement of C_{j-1}. Finally, we can bound the packing number by $\mathcal{M}_j\leqslant N_j=\max(\frac{5}{\varepsilon_j},5)^P$ where P is the dimension of $\mathcal{G}$. Thus we deduce that:

$$\rho(\gamma t_1+t_2+t_3) \leqslant \sum_{j=1}^{J}N_j^2\exp(-\frac{t_1^2N\gamma_j^2}{2B_1^2\varepsilon_j^2})+\exp(-\frac{t^2N}{2B_1^2\varepsilon_J^2})+\exp(-\frac{t_3^2N}{2B_1^2}).$$

Now, we define $\gamma_j=\frac{2\varepsilon_jB_1}{t_1}\sqrt{\frac{2\log(N_j)}{N}}$, $t_2=B_1\varepsilon_J\sqrt{\frac{2\log(1/\delta_2)}{N}}$ and $t_3=B_1\sqrt{\frac{2\log(1/\delta_3)}{N}}$, for some $\delta_2,\delta_3\in(0,1]$. Thus, we get:

$$\rho(\eta t_1+t_2+t_3) \leqslant \sum_{j=1}^{J}\frac{1}{N_j^2}+\delta_2+\delta_3.$$

Thus, it remains to define ε_j. Since $N_j=\max(\frac{5}{\varepsilon_j},5)^P$, we define the covering radius ε_j to be $\varepsilon_j=2^{-j}5\delta_1^{1/2P}(2^{2P}-1)^{1/2P}$ for some $\delta_1\in(0,1]$, which entails that $\sum_{j=1}^{J}\frac{1}{N_j^2}\leqslant\delta_1$. Now since $\varepsilon_j\to0$ when $j\to\infty$, we can make the sum goes to infinity. We deduce that:

$$\rho(\gamma t_1+B_1\sqrt{\frac{2\log(1/\delta_3)}{N}})\leqslant\delta_1+\delta_2+\delta_3.$$

Now, in order to bound the term $\gamma t_1 + t_2 + t_3$,we look at the following term:

$$\begin{aligned}
\gamma t_1 &= 2\sum_{j=1}^{\infty} \varepsilon_j B_1 \sqrt{\frac{2\log(N_j)}{N}} \\
&\leqslant \frac{20B_1}{\sqrt{N}} \sum_{j=1}^{\infty} 2^{-j} \sqrt{2jP\log(2) + \log(1/\delta_1) - \log(2^{2P}-1)} \\
&\leqslant \frac{20B_1}{\sqrt{N}} \sum_{j=1}^{\infty} 2^{-j} \sqrt{2(j-1)P\log(2) + \log(2/\delta_1)} \\
&\leqslant \frac{20B_1}{\sqrt{N}} \Big(\sum_{j=1}^{\infty} 2^{-j} \sqrt{2(j-1)P\log(2)} + \sqrt{\log(2/\delta_1)} \Big) \\
&\leqslant \frac{20B_1}{\sqrt{N}} \Big((1+\sqrt{2}) \sqrt{2P\log(2)} + \sqrt{\log(2/\delta_1)} \Big).
\end{aligned}$$

where we use the fact that $\sum_{j=1}^{\infty} 2^{-j} \leqslant 1$, and that $\sum_{j=1}^{\infty} 2^{-j}\sqrt{(j-1)} \leqslant 1+\sqrt{2}$.

Using the inequalities $\sqrt{a}+\sqrt{b}+\sqrt{c} \leqslant \sqrt{3(a+b+c)}$, we thus deduce the following bound:

$$\begin{aligned}
\gamma t_1 + t_2 + t_3 &\leqslant \frac{B_1}{\sqrt{N}} \Big(20(1+\sqrt{2})\sqrt{2P\log(2)} + 20\sqrt{\log(2/\delta_1)} + \sqrt{2\log(1/\delta_3)} \Big) \\
&\leqslant \frac{\sqrt{6}B_1}{\sqrt{N}} \sqrt{400\log(2)(1+\sqrt{2})^2 P + 200\log(2/\delta_1) + \log(1/\delta_3)}.
\end{aligned}$$

Thus, by setting $\delta_1 = \delta_2 = \delta_3 = \delta/3$, we deduce that with $\mathcal{P}$-probability higher than $1-\delta$,

$$\sup_{g \in \mathcal{G}_P} \frac{\frac{1}{N}\sum_{i=1}^N \varepsilon_i g(X_i)}{\|g\|_N} \leqslant \frac{B_1\sqrt{6}}{\sqrt{N}} \sqrt{400\log(2)(1+\sqrt{2})^2 P + 200\log(6/\delta) + \log(3/\delta)}\,.$$

■

A.3 Proof of Lemma 26

Proof Indeed, let us introduce the space of functions $\mathcal{G}^0 = \{f^\star - T_B(g), g \in \mathcal{G}_P\}$. Then we have for $g \in \mathcal{G}^0$, $\|g\|_N \leqslant \|g\|_\infty \leqslant 2B$. Thus Theorem 11.2 of Györfi et al. (2002) gives the following bound:

$$\mathbb{P}\big(\sup_{g\in\mathcal{G}_P} \|f^\star - T_B(g)\|_{\mathcal{P}_X} - 2\|f^\star - T_B(g)\|_N > \varepsilon\big) \leqslant 3\mathbb{E}\big(\mathcal{N}\big(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{G}^0, \|.\|_{2N}\big)\big) \exp\Big(-\frac{N\varepsilon^2}{288(2B)^2}\Big).$$

Then, since $\mathcal{G}^0 = f^\star + T_B(\mathcal{G}_P)$, we bound the entropy number by:

$$\mathcal{N}\big(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{G}^0, \|.\|_{2N}\big) \leqslant \mathcal{N}\big(\frac{\sqrt{2}}{24}\varepsilon, T_B(\mathcal{G}_P), \|.\|_{2N}\big) \leqslant \big(\frac{2(2B).24}{\sqrt{2}\varepsilon} + 1\big)^P.$$

Thus we deduce that if $\varepsilon \geqslant \frac{24.4B}{\sqrt{2}}u$, then with probability higher than $1-\delta$ w.r.t $\mathbb{P}$, for fixed random Gaussian variables,

$$\sup_{g\in\mathcal{G}_P} \|f^\star - T_B(g)\|_{\mathcal{P}_X} - 2\|f^\star - T_B(g)\|_N \leqslant \varepsilon = 24B\sqrt{\log(3/\delta) + P\log\big(\frac{1}{u}+1\big)}\sqrt{\frac{2}{N}}.$$

Thus,we consider $u = \frac{1}{N-1}$, and deduce that, provided that $N\log(N) \geqslant \frac{4}{P}$, then with probability higher than $1-\delta$ w.r.t $\mathbb{P}$, for fixed random Gaussian variables (i.e., conditionally on them),

$$\sup_{g\in\mathcal{G}_P} \|f^\star - T_B(g)\|_{\mathcal{P}_X} - 2\|f^\star - T_B(g)\|_N \leqslant 24B\sqrt{\frac{2\log(3/\delta)+P\log(N)}{N}}.$$

Thus, we deduce that on this event, for all $g \in \mathcal{G}_P$

$$\begin{aligned}\|f^\star - T_B(g)\|^2_{\mathcal{P}_X} &\leqslant (2\|f^\star - T_B(g)\|_N + 24B\sqrt{\frac{2\log(3/\delta)+P\log(N)}{N}})^2 \\ &\leqslant 8\|f^\star - T_B(g)\|^2_N + (24B)^2\frac{4\log(3/\delta)+2P\log(N)}{N}.\end{aligned}$$

This gives the following upper bound, that holds with probability higher than $1-\delta$:

$$\sup_{g\in\mathcal{G}_P} \|f^\star - T_B(g)\|^2_{\mathcal{P}_X} - 8\|f^\star - T_B(g)\|^2_N \leqslant (24B)^2\frac{4\log(3/\delta)+2P\log(N)}{N}.$$

■

A.4 Proof of Theorem 19

Proof Since by assumption $f^\star = f_{\alpha^\star}$ for some $\alpha^\star$, we have by direct application of Lemma 9

$$\inf_{g\in\mathcal{G}} \|f^\star - g\|^2_N \leqslant \|f_{\alpha^\star} - g_{A\alpha^\star}\|^2_N.$$

Now let us define for some $N \geqslant 1$ the quantity $\varepsilon = \varepsilon_N(\delta)$ that appears in Lemma 9, such that

$$\frac{\log(4N/\delta)}{P} = \frac{\varepsilon^2}{4} - \frac{\varepsilon^3}{6}.$$

Thus, since $\varepsilon \in (0,1)$, this means in particular that we have

$$\frac{\varepsilon^2}{3} \leqslant 4\frac{\log(4N/\delta)}{P} \leqslant \varepsilon^2.$$

■

A.5 Proof of Theorem 20

Proof By assumption, we consider that $f^\star \in \mathcal{F}$. Thus there exists a sequence $\alpha^\star \in \mathbb{R}^{\mathbb{N}}$ such that one can write:

$$f^\star = f_{\alpha^\star} = \sum_{i\geqslant 1} \alpha^\star_i \varphi_i,$$

Thus we consider in the sequel one such $\alpha^\star$. This enables to derive the following upper bound:

$$\inf_{g\in\mathcal{G}} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} \leqslant \|f_{\alpha^*} - T_L(g_{A\alpha^*})\|^2_{\mathcal{P}_X}.$$

where we applied the gaussian operator A to the sequence $\alpha^\star$.

Step 1. Applying Johnson-Lindenstrauss' Lemma. Let us introduce m ghost samples $(X'_j)_{j\leqslant m}$ i.i.d. according to $\mathcal{P}_\mathcal{X}$, and thus consider the following associated norm

$$\|f_{\alpha^*} - T_L(g_{A\alpha^*})\|_m^2 = \frac{1}{m}\sum_{j=1}^m (f_{\alpha^*} - T_L(g_{A\alpha^*}))^2(X'_j)\,.$$

We now make explicit the probability spaces corresponding to the different sources of randomness. Consider the probability space defined over the product sample space $\Omega_\mathcal{X}\times\Omega_\mathcal{G}$, where $\Omega_\mathcal{X}$ consists of all the possible realizations of J states $X'_1,\ldots,X'_m$ drawn i.i.d. from $\mathcal{P}_\mathcal{X}$, and $\Omega_\mathcal{G}$ is the set of all possible realizations of the random elements $(A_{p,i})_{1\leqslant p\leqslant P, i\geqslant 1}$ (which define the random feature space $\mathcal{G}_P$).

Let us fix some $\omega_\mathcal{G}\in\Omega_\mathcal{G}$ (which defines the random subspace $\mathcal{G}_P(\omega_\mathcal{G})$). Since for all j, we have that $(f_{\alpha^*} - T_L(g_{A\alpha^*}))^2(X'_j)\in[0,4L^2]$ $\mathcal{P}_\mathcal{X}$-a.s., then Hoeffding's inequality applies; we deduce that there exists an event $\Omega_\mathcal{X}(\omega_\mathcal{G})$ of $\mathcal{P}_\mathcal{X}$-probability higher than $1-\delta_\mathcal{X}$ such that on this event

$$\|f_{\alpha^*} - T_L(g_{A\alpha^*})\|_{\mathcal{P}_\mathcal{X}}^2 \leqslant \|f_{\alpha^*} - T_L(g_{A\alpha^*})\|_m^2 + (2L)^2\sqrt{\frac{\log(1/\delta)}{2m}}.$$

Now by independence between the Gaussian random variables and the sample, the same inequality is valid on the event

$$\Omega_1 = \{\omega_\mathcal{X}\times\omega_\mathcal{G}; \omega_\mathcal{G}\in\Omega_\mathcal{G}, \omega_\mathcal{X}\in\Omega_\mathcal{X}(\omega_\mathcal{G})\}\,,$$

and this event has $\mathcal{P}_\mathcal{X}\times\mathcal{P}_\mathcal{G}$-probability higher than $1-\delta_\mathcal{X}$.

In order to bound the first term of the right hand side of this inequality, we first notice that since $\|f_{\alpha^*}\|_\infty\leqslant L$, then

$$\|f_{\alpha^*} - T_L(g_{A\alpha^*})\|_m^2 \quad\leqslant\quad \|f_{\alpha^*} - g_{A\alpha^*}\|_m^2\,,$$

then for some fixed $\omega_\mathcal{X}\in\Omega_\mathcal{X}$, that last term is bounded by $\varepsilon^2\|\alpha^\star\|^2\sup_x\|\varphi(x)\|^2$ on an event $\Omega_\mathcal{G}(\omega_\mathcal{X})$ of $\mathcal{P}_\mathcal{G}$-probability higher than $1-4me^{-P(\varepsilon^2/4-\varepsilon^3/6)}$ by application of Lemma 9.

Thus still by independence, the same inequality is valid on the event

$$\Omega_2 = \{(\omega_\mathcal{X},\omega_\mathcal{G}); \omega_\mathcal{X}\in\Omega_\mathcal{X}, \omega_\mathcal{G}\in\Omega_\mathcal{G}(\omega_\mathcal{X})\}\,,$$

and this event has $\mathcal{P}_\mathcal{X}\times\mathcal{P}_\mathcal{G}$-probability higher than $1-4me^{-P(\varepsilon^2/4-\varepsilon^3/6)}$.

Thus, we deduce, by a union bound that for all $\varepsilon\in(0,1)$ and $m\geqslant 1$ there exists an event $\Omega_1\cap\Omega_2$ of $\mathcal{P}_\mathcal{X}\times\mathcal{P}_\mathcal{G}$-probability higher than $1-\delta_\mathcal{X}-4me^{-P(\varepsilon^2/4-\varepsilon^3/6)}$ such that on this event,

$$\inf_{g\in\mathcal{G}}\|f^\star - T_L(g)\|_{\mathcal{P}_\mathcal{X}}^2 \leqslant \varepsilon^2\|\alpha^\star\|^2\sup_x\|\varphi(x)\|^2 + (2L)^2\sqrt{\frac{\log(1/\delta)}{2m}}\,.$$

Finally in order to get a bound in high $\mathcal{P}_\mathcal{G}$-probability only, we introduce for any $\omega_\mathcal{G}\in\Omega_\mathcal{G}$ the event $\Omega'_\mathcal{X}(\omega_\mathcal{G})\stackrel{\text{def}}{=}\{\omega_\mathcal{X}\in\Omega_\mathcal{X}; (\omega_\mathcal{X},\omega_\mathcal{G})\in\Omega_1\times\Omega_2\}$ and then define for all $\lambda>0$ the event

$$\Lambda\stackrel{\text{def}}{=}\{\omega_\mathcal{G}\in\Omega_\mathcal{G}; \mathbb{P}_\mathcal{X}(\Omega'_\mathcal{X}(\omega_\mathcal{G}))\geqslant 1-\lambda\}\,.$$

Using this notation, we deduce that for all $\omega_{\mathcal{G}} \in \Lambda$, the following bound holds

$$\begin{aligned}\inf_{g\in\mathcal{G}_P(\omega_{\mathcal{G}})} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} &\leqslant \int_{\Omega'_X(\omega_{\mathcal{G}})} \inf_{g\in\mathcal{G}_P(\omega_{\mathcal{G}})} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} d\omega_X \\ &\quad + \int_{\Omega'_X(\omega_{\mathcal{G}})^c} \inf_{g\in\mathcal{G}_P(\omega_{\mathcal{G}})} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} d\omega_X \\ &\leqslant \varepsilon^2 \|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2 + (2L)^2 \sqrt{\frac{\log(1/\delta)}{2m}} + (2L)^2\lambda .\end{aligned}$$

Moreover, since $\mathbb{P}_{\mathcal{X}\times\mathcal{G}}(\Omega_1 \cap \Omega_2) \geqslant 1 - \delta_{\mathcal{X}} - 4me^{-P(\varepsilon^2/4-\varepsilon^3/6)}$ and on the other side

$$\begin{aligned}\mathbb{P}_{\mathcal{X}\times\mathcal{G}}(\Omega_1 \cap \Omega_2) &= \int_{\Omega_{\mathcal{G}}} \mathbb{P}_{\mathcal{X}}(\Omega'_{\mathcal{X}}(\omega_{\mathcal{G}})) d\omega_{\mathcal{G}} \\ &\leqslant \int_{\Omega_{\mathcal{G}}} \mathbb{I}_{\mathbb{P}_{\mathcal{X}}(\Omega'_{\mathcal{X}}(\omega_{\mathcal{G}}))\geqslant 1-\lambda} d\omega_{\mathcal{G}} + (1-\lambda) \int_{\Omega_{\mathcal{G}}} \mathbb{I}_{\mathbb{P}_{\mathcal{X}}(\Omega'_{\mathcal{X}}(\omega_{\mathcal{G}}))<1-\lambda} d\omega_{\mathcal{G}} \\ &\leqslant \mathbb{P}_{\mathcal{G}}(\Lambda) + (1-\lambda)(1 - \mathbb{P}_{\mathcal{G}}(\Lambda)) ,\end{aligned}$$

then we deduce that $\mathbb{P}_{\mathcal{G}}(\Lambda)) \geqslant 1 - \frac{\delta_{\mathcal{X}} + 4me^{-P(\varepsilon^2/4-\varepsilon^3/6)}}{\lambda}$.

Step 2. Tuning the parameters ε. Now let us introduce $\delta_{\mathcal{G}}$ and define for some $m \geqslant 1$ the quantity $\varepsilon = \varepsilon_m(\delta_{\mathcal{G}})$ such that

$$\frac{\log(4m/\delta\mathcal{G})}{P} = \frac{\varepsilon^2}{4} - \frac{\varepsilon^3}{6} .$$

Thus, since $\varepsilon \in (0,1)$, this means in particular that we have

$$\frac{\varepsilon^2}{3} \leqslant 4\frac{\log(4m/\delta_{\mathcal{G}})}{P} \leqslant \varepsilon^2 .$$

Now by rewriting the bound using $\delta = \frac{\delta_{\mathcal{X}} + \delta_{\mathcal{G}}}{\lambda}$, we deduce that for all δ, for all m and λ, there exists an event of $\mathcal{P}_{\mathcal{G}}$-probability higher than $1-\delta$ such that

$$\inf_{g\in\mathcal{G}} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} \leqslant 12\frac{\log(\frac{8m}{\lambda\delta})}{P} \|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2 + (2L)^2 \Big(\sqrt{\frac{\log(\frac{2}{\lambda\delta})}{2m}} + \lambda \Big) .$$

Step 3. Optimizing over λ *and* m. Now, it remains to optimize the free parameter m and λ in this last bound; the optimal value for m is given by

$$m_{opt} = \frac{P^2 L^4 \log(\frac{2}{\lambda\delta})}{72\|\alpha\|^4 \sup_x \|\varphi(x)\|^4} ,$$

and the corresponding bound is thus

$$\inf_{g\in\mathcal{G}} \|f^\star - T_L(g)\|^2_{\mathcal{P}_X} \leqslant 24\frac{\|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2}{P} \Big(1 + \log\big(\frac{PL^2\sqrt{\log(2/\lambda\delta)/\lambda\delta}}{3\|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2} \big) \Big) + (2L)^2\lambda .$$

Now one can take $\lambda \stackrel{\text{def}}{=} \frac{\|\alpha^\star\|^2 \sup_x \|\varphi(x)\|^2}{(2L)^2 P}$ and deduce the final bound. ■

A.6 Proof of Theorem 21

Proof We make use of the following decomposition:

$$\|f^\star - g_{\widehat{\beta}}\|_N^2 \leqslant \|f^\star - g_{\tilde{\beta}}\|_N^2 + \|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2 ,$$

and introduce the sets $\Omega_{\mathcal{G}}$ that consists of all possible realizations of the random elements $(A_{p,i})_{1\leqslant p\leqslant P, i\geqslant 1}$, and $\Omega_{\mathcal{Y}}$ that corresponds to the observation variables Y.

High $\mathcal{P}_{\mathcal{Y}} \times \mathcal{P}_{\mathcal{G}}$-probability bound. We again make explicit the probability spaces. For the first term on right hand side, an application of Theorem 19 ensures that there exists an event $\Omega'_{\mathcal{G}} \subset \Omega_{\mathcal{G}}$ of $\mathcal{P}_{\mathcal{G}}$-probability higher than $1-\delta$ such that for all $\omega_{\mathcal{G}} \in \Omega'_{\mathcal{G}}$,

$$\|f^\star - g_{\tilde{\beta}}\|_N^2 \leqslant 12 \frac{\log(4N/\delta)}{P} \|\alpha^\star\|^2 \frac{1}{N} \sum_{n=1}^{N} \|\varphi(x_n)\|^2 .$$

Since no random variable Y appears in this term, this is also true on the event

$$\Omega_1 \stackrel{\text{def}}{=} \{(\omega_{\mathcal{Y}}, \omega_{\mathcal{G}}) \in \Omega_{\mathcal{Y}} \times \Omega_{\mathcal{G}}; \omega_{\mathcal{G}} \in \Omega'_{\mathcal{G}}\} ,$$

and Ω_1 has $\mathcal{P}_{\mathcal{Y}} \times \mathcal{P}_{\mathcal{G}}$-probability higher than $1-\delta$.

For the second term, let us fix some $\omega_{\mathcal{G}} \in \Omega_{\mathcal{G}}$. Then Lemma 24 below shows that there exists an event $\Omega_{\mathcal{Y}}(\omega_{\mathcal{G}}) \subset \Omega_{\mathcal{G}}$ of $\mathcal{P}_{\mathcal{Y}}$-probability higher than $1-\delta'$ such that for all $\omega_{\mathcal{Y}} \in \Omega_{\mathcal{Y}}(\omega_{\mathcal{G}})$,

$$\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2 \leqslant \kappa B^2 \frac{P + \log(1/\delta')}{N} ,$$

for some numerical constant $\kappa > 0$. Thus by independence of the noise term with the Gaussian variables, we deduce that a similar bound holds on the event

$$\Omega_2 \stackrel{\text{def}}{=} \{(\omega_{\mathcal{Y}}, \omega_{\mathcal{G}}) \in \Omega_{\mathcal{Y}} \times \Omega_{\mathcal{G}}; \omega_{\mathcal{Y}} \in \Omega_{\mathcal{Y}}(\omega_{\mathcal{G}})\} ,$$

and that Ω_2 has $\mathcal{P}_{\mathcal{Y}} \times \mathcal{P}_{\mathcal{G}}$-probability higher than $1-\delta'$. Thus, we conclude by a simple union bound in order te get a result in high $\mathcal{P}_{\mathcal{Y}} \times \mathcal{P}_{\mathcal{G}}$-probability. ■

A.7 Proof of Theorem 22

Proof

Similarly to the proof of Theorem 20, we introduce the sets $\Omega_{\mathcal{X}}, \Omega_\eta$ and $\Omega_{\mathcal{G}}$ that consist of all possible realizations of the input, noise and Gaussian random variables. We then define $\Omega \stackrel{\text{def}}{=} \Omega_{\mathcal{X}} \times \Omega_\eta \times \Omega_{\mathcal{G}}$.

Step 1. High $\mathcal{P} \times \mathcal{P}_{\mathcal{G}}$-probability bound. In order to get a high probability bound, we use the decomposition given by Lemma 23. Now let us consider some fixed $\omega_{\mathcal{G}} \in \Omega_{\mathcal{G}}$. One can apply Lemma 24 and Lemma 26 below for the noise and estimation term.

Thus when $N\log(N) \geqslant \frac{4}{P}$, there exists an event $\Omega_1(\omega_{\mathcal{G}})$ of $\mathcal{P}$-probability higher than $1-\delta_1$ and an event $\Omega_2(\omega_{\mathcal{G}})$ of $\mathcal{P}$-probability higher than $1-\delta_2$ such that for all $(\omega_{\mathcal{X}}, \omega_\eta) \in \Omega_1(\omega_{\mathcal{G}})$ we have

$$\|g_{\tilde{\beta}} - g_{\widehat{\beta}}\|_N^2 \leqslant 6B^2 \frac{(1616P + 200\log(6/\delta) + \log(3/\delta))}{N} ,$$

and for all $(\omega_X, \omega_\eta) \in \Omega_2(\omega_G)$ we have

$$\sup_{g \in \mathcal{G}_P} \|f^\star - T_L(g)\|_{\mathcal{P}_X}^2 - 8\|f^\star - T_L(g)\|_N^2 \leqslant (24L)^2 \frac{4\log(3/\delta) + 2P\log(N)}{N}.$$

On the other hand, by application of Theorem 19, for any given (ω_X, ω_η), there exists an event $\Omega_G(\omega_X, \omega_\eta) \subset \Omega_G$ of $\mathcal{P}_G$-probability higher than $1 - \delta_3$ such that on this event

$$\|f^\star - g_{\tilde{\beta}}\|_N^2 \leqslant 12 \frac{\log(4N/\delta_3)}{P} \|\alpha^\star\|^2 \sup_{x \in X} \|\varphi(x)\|^2.$$

Thus by independence of the noise, data points and Gaussian variables, the three previous inequalities are valid respectively on the events

$$\Omega_1 = \{(\omega_X, \omega_\eta, \omega_G) \in \Omega; (\omega_X, \omega_\eta) \in \Omega_1(\omega_G)\},$$
$$\Omega_2 = \{(\omega_X, \omega_\eta, \omega_G) \in \Omega; (\omega_X, \omega_\eta) \in \Omega_2(\omega_G)\},$$
$$\Omega_3 = \{(\omega_X, \omega_\eta, \omega_G) \in \Omega; \omega_G \in \Omega_G(\omega_X)\}.$$

Moreover Ω_1 has $\mathcal{P} \times \mathcal{P}_G$-probability higher than $1 - \delta_1$, Ω_2 has $\mathcal{P} \times \mathcal{P}_G$-probability higher than $1 - \delta_2$, and Ω_1 has $\mathcal{P} \times \mathcal{P}_G$-probability higher than $1 - \delta_3$. We thus conclude by a simple union bound, and then by some cosmetic simplifications introducing some constant κ. ■

References

Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal Of Computer And System Sciences*, 66(4):671–687, June 2003.

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings Of The 38th Annual ACM Symposium On Theory Of Computing*, STOC '06, pages 557–563, New York, NY, USA, 2006. ACM. ISBN 1-59593-134-1.

Pierre Alquier. PAC-Bayesian bounds for randomized empirical risk minimizers. *Mathematical Methods Of Statistics*, 17(4):279–304, dec 2008.

Pierre Alquier and Karim Lounici. PAC-Bayesian Bounds for Sparse Regression Estimation with Exponential Weights. *Electronic Journal Of Statistics*, 5:127–145, 2011.

Jean-Marie Aubry and Stéphane Jaffard. Random wavelet series. *Communications In Mathematical Physics*, 227:483–514, 2002.

Jean-Yves Audibert and Olivier Catoni. Robust linear regression through PAC-Bayesian truncation. arXiv, 2010.

Andrew Barron, Albert Cohen, Wolfgang Dahmen, and Ronald DeVore. Approximation and learning by greedy algorithms. *Annals Of Statistics*, 36:1:64–94, 2008.

Lucien Birgé and Pascal Massart. Minimum contrast estimators on sieves: Exponential bounds and rates of convergence. *Bernoulli*, 4(3):329–375, sep 1998.

Gerard Bourdaud. Ondelettes et espaces de besov. *Rev. Mat. Iberoamericana*, 11:3:477–512, 1995.

Hans-Joachim Bungartz and Michaël Griebel. Sparse grids. In Arieh Iserles, editor, *Acta Numerica*, volume 13. University of Cambridge, 2004.

Stphane Canu, Xavier Mary, and Alain Rakotomamonjy. Functional learning through kernel. In *Advances In Learning Theory: Methods, Models And Applications NATO Science Series III: Computer And Systems Sciences*, pages 89–110. IOS Press, 2002.

Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations Of Computational Mathematics*, 7:331–368, jul 2007. ISSN 1615-3375.

Olivier Catoni. *Statistical Learning Theory And Stochastic Optimization*. Springer-Verlag, 2004.

Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings Of The 19th Annual ACM Symposium On Theory Of Computing*, STOC '87, pages 1–6, New York, NY, USA, 1987. ACM.

Sanjoy Dasgupta and Yoav Freund. Random projection trees and low dimensional manifolds. In *Proceedings Of The 40th Annual ACM Symposium On Theory Of Computing*, STOC '08, pages 537–546, New York, NY, USA, 2008. ACM.

Sébastien Deguy and Albert Benassi. A flexible noise model for designing maps. In *Proceedings Of The Vision Modeling And Visualization Conference 2001*, VMV '01, pages 299–308. Aka GmbH, 2001. ISBN 3-89838-028-9.

Ronald DeVore. *Nonlinear Approximation*. Acta Numerica, 1997.

Richard M. Dudley. *Real Analysis And Probability*. Wadsworth, Belmont, Calif, 1989.

Arnaud Durand. Random wavelet series based on a tree-indexed Markov chain. *Communications In Mathematical Physics*, 283:451–477, 2008.

Michaël Frazier and Björn Jawerth. Decomposition of Besov spaces. *Indiana University Mathematics Journal*, (34), 1985.

László Györfi, Michaël Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-free Theory Of Nonparametric Regression*. Springer-Verlag, New York, 2002.

Svante Janson. *Gaussian Hilbert Spaces*. Cambridge Univerity Press, Cambridge, UK, 1997.

Vladimir Koltchinskii. Local rademacher complexities and oracle inequalities in risk minimization. *The Annals of Statistics*, 34(6):2593–2656, 2006.

Edo Liberty, Nir Ailon, and Amit Singer. Dense fast random projections and lean Walsh transforms. In *Proceedings Of The 11th International Workshop, APPROX 2008, And 12th International Workshop, RANDOM 2008 On Approximation, Randomization And Combinatorial Optimization: Algorithms And Techniques*, APPROX '08 / RANDOM '08, pages 512–522, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85362-6.

Mikhail A. Lifshits. *Gaussian Random Functions*. Kluwer Academic Publishers, Dordrecht, Boston, 1995.

Odalric-Ambrym Maillard and Rémi Munos. Compressed least-squares regression. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Chris K. I. Williams, and Aron Culotta, editors, *Proceedings Of The 23rd Conference On Advances In Neural Information Processing Systems*, NIPS '09, pages 1213–1221, Vancouver, British Columbia, Canada, dec 2009.

Odalric-Ambrym Maillard and Rémi Munos. Scrambled objects for least-squares regression. In John D. Lafferty, Chris K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, editors, *Proceedings Of The 24th Conference On Advances In Neural Information Processing Systems*, NIPS '10, pages 1549–1557, Vancouver, British Columbia, Canada, dec 2010.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *Proceedings Of The 21st Conference On Advances In Neural Information Processing Systems*, NIPS '07, Vancouver, British Columbia, Canada, dec 2007. MIT Press.

Ali Rahimi and Benjamin Recht. Uniform approximation of functions with random bases. *Proceedings Of The 46th Annual Allerton Conference*, 2008.

Saburou Saitoh. *Theory Of Reproducing Kernels And Its Applications*. Longman Scientific & Technical, Harlow, UK, 1988.

Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Proceedings Of The 47th Annual IEEE Symposium On Foundations Of Computer Science.*, FOCS '06, pages 143–152, 2006.

Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances In Neural Information Processing Systems 8*, pages 1038–1044. MIT Press, 1996.

Richard S. Sutton and Steven D. Whitehead. Online learning with random representations. In *In Proceedings Of The 10th International Conference On Machine Learning*, ICML '93, pages 314–321. Morgan Kaufmann, 1993.

Robert Tibshirani. Regression shrinkage and selection via the Lasso. *Journal Of The Royal Statistical Society, Series B*, 58:267–288, 1994.

Andrei N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math Dokl 4*, pages 1035–1038, 1963.

Alexandre B. Tsybakov. Optimal rates of aggregation. In *Proceedings Of The 16th Annual Conference On Learning Theory*, pages 303–313, 2003.

Adriaan Zaanen. *Linear Analysis*. North Holland Publishing, 1960.

Christoph Zenger. Sparse grids. In W. Hackbusch, editor, *Parallel Algorithms For Partial Differential Equations, Proceedings Of The Sixth GAMM-Seminar*, volume 31 of Notes on Num. Fluid Mechanics, Kiel, 1990. Vieweg-Verlag.

Bin Zhao and Changshui Zhang. Compressed spectral clustering. In *Proceedings Of The 2009 IEEE International Conference On Data Mining Workshops*, ICDMW '09, pages 344–349, Washington, DC, USA, 2009. IEEE Computer Society.

Journal of Machine Learning Research 13 (2012) 2773-2812 Submitted 7/11; Revised 4/12; Published 9/12

Multi-task Regression using Minimal Penalties

Matthieu Solnon MATTHIEU.SOLNON@ENS.FR
ENS; Sierra Project-team
Département d'Informatique de l'École Normale Supérieure
(CNRS/ENS/INRIA UMR 8548)
23, avenue d'Italie, CS 81321
75214 Paris Cedex 13, France

Sylvain Arlot SYLVAIN.ARLOT@ENS.FR
CNRS; Sierra Project-team
Département d'Informatique de l'École Normale Supérieure
(CNRS/ENS/INRIA UMR 8548)
23, avenue d'Italie, CS 81321
75214 Paris Cedex 13, France

Francis Bach FRANCIS.BACH@ENS.FR
INRIA; Sierra Project-team
Département d'Informatique de l'École Normale Supérieure
(CNRS/ENS/INRIA UMR 8548)
23, avenue d'Italie, CS 81321
75214 Paris Cedex 13, France

Editor: Tong Zhang

Abstract

In this paper we study the kernel multiple ridge regression framework, which we refer to as multi-task regression, using penalization techniques. The theoretical analysis of this problem shows that the key element appearing for an optimal calibration is the covariance matrix of the noise between the different tasks. We present a new algorithm to estimate this covariance matrix, based on the concept of minimal penalty, which was previously used in the single-task regression framework to estimate the variance of the noise. We show, in a non-asymptotic setting and under mild assumptions on the target function, that this estimator converges towards the covariance matrix. Then plugging this estimator into the corresponding ideal penalty leads to an oracle inequality. We illustrate the behavior of our algorithm on synthetic examples.

Keywords: multi-task, oracle inequality, learning theory

1. Introduction

A classical paradigm in statistics is that increasing the sample size (that is, the number of observations) improves the performance of the estimators. However, in some cases it may be impossible to increase the sample size, for instance because of experimental limitations. Hopefully, in many situations practicioners can find many related and similar problems, and might use these problems as if more observations were available for the initial problem. The techniques using this heuristic are called "multi-task" techniques. In this paper we study the kernel ridge regression procedure in a multi-task framework.

One-dimensional kernel ridge regression, which we refer to as "single-task" regression, has been widely studied. As we briefly review in Section 3 one has, given n data points $(X_i, Y_i)_{i=1}^n$, to estimate a function f, often the conditional expectation $f(X_i) = \mathbb{E}[Y_i|X_i]$, by minimizing the quadratic risk of the estimator regularized by a certain norm. A practically important task is to calibrate a regularization parameter, that is, to estimate the regularization parameter directly from data. For kernel ridge regression (a.k.a. smoothing splines), many methods have been proposed based on different principles, for example, Bayesian criteria through a Gaussian process interpretation (see, e.g., Rasmussen and Williams, 2006) or generalized cross-validation (see, e.g., Wahba, 1990). In this paper, we focus on the concept of minimal penalty, which was first introduced by Birgé and Massart (2007) and Arlot and Massart (2009) for model selection, then extended to linear estimators such as kernel ridge regression by Arlot and Bach (2011).

In this article we consider $p \geq 2$ different (but related) regression tasks, a framework we refer to as "multi-task" regression. This setting has already been studied in different papers. Some empirically show that it can lead to performance improvement (Thrun and O'Sullivan, 1996; Caruana, 1997; Bakker and Heskes, 2003). Liang et al. (2010) also obtained a theoretical criterion (unfortunately non observable) which tells when this phenomenon asymptotically occurs. Several different paths have been followed to deal with this setting. Some consider a setting where $p \gg n$, and formulate a sparsity assumption which enables to use the group Lasso, assuming all the different functions have a small set of common active covariates (see for instance Obozinski et al., 2011; Lounici et al., 2010). We exclude this setting from our analysis, because of the Hilbertian nature of our problem, and thus will not consider the similarity between the tasks in terms of sparsity, but rather in terms of an Euclidean similarity. Another theoretical approach has also been taken (see for example, Brown and Zidek (1980), Evgeniou et al. (2005) or Ando and Zhang (2005) on semi-supervised learning), the authors often defining a theoretical framework where the multi-task problem can easily be expressed, and where sometimes solutions can be computed. The main remaining theoretical problem is the calibration of a matricial parameter M (typically of size p), which characterizes the relationship between the tasks and extends the regularization parameter from single-task regression. Because of the high dimensional nature of the problem (i.e., the small number of training observations) usual techniques, like cross-validation, are not likely to succeed. Argyriou et al. (2008) have a similar approach to ours, but solve this problem by adding a convex constraint to the matrix, which will be discussed at the end of Section 5.

Through a penalization technique we show in Section 2 that the only element we have to estimate is the correlation matrix Σ of the noise between the tasks. We give here a new algorithm to estimate Σ, and show that the estimation is sharp enough to derive an oracle inequality for the estimation of the task similarity matrix M, both with high probability and in expectation. Finally we give some simulation experiment results and show that our technique correctly deals with the multi-task settings with a low sample-size.

1.1 Notations

We now introduce some notations, which will be used throughout the article.

- The integer n is the sample size, the integer p is the number of tasks.
- For any $n \times p$ matrix Y, we define

$$y = \mathrm{vec}(Y) := (Y_{1,1}, \ldots, Y_{n,1}, Y_{1,2}, \ldots, Y_{n,2}, \ldots, Y_{1,p}, \ldots, Y_{n,p}) \in \mathbb{R}^{np},$$

that is, the vector in which the columns $Y^j := (Y_{i,j})_{1\le i\le n}$ are stacked.

- $\mathcal{M}_n(\mathbb{R})$ is the set of all matrices of size n.
- $\mathcal{S}_p(\mathbb{R})$ is the set of symmetric matrices of size p.
- $\mathcal{S}_p^+(\mathbb{R})$ is the set of symmetric positive-semidefinite matrices of size p.
- $\mathcal{S}_p^{++}(\mathbb{R})$ is the set of symmetric positive-definite matrices of size p.
- $\preceq$ denotes the partial ordering on $\mathcal{S}_p(\mathbb{R})$ defined by: $A \preceq B$ if and only if $B-A \in \mathcal{S}_p^+(\mathbb{R})$.
- $\mathbf{1}$ is the vector of size p whose components are all equal to 1.
- $\|\cdot\|_2$ is the usual Euclidean norm on $\mathbb{R}^k$ for any $k \in \mathbb{N}$: $\forall u \in \mathbb{R}^k$, $\|u\|_2^2 := \sum_{i=1}^k u_i^2$.

2. Multi-task Regression: Problem Set-up

We consider p kernel ridge regression tasks. Treating them simultaneously and sharing their common structure (e.g., being close in some metric space) will help in reducing the overall prediction error.

2.1 Multi-task with a Fixed Kernel

Let $\mathcal{X}$ be some set and $\mathcal{F}$ a set of real-valued functions over $\mathcal{X}$. We suppose $\mathcal{F}$ has a reproducing kernel Hilbert space (RKHS) structure (Aronszajn, 1950), with kernel k and feature map $\Phi : \mathcal{X} \to \mathcal{F}$. We observe $\mathcal{D}_n = (X_i, Y_i^1, \ldots, Y_i^p)_{i=1}^n \in (\mathcal{X} \times \mathbb{R}^p)^n$, which gives us the positive semidefinite kernel matrix $K = (k(X_i, X_\ell))_{1\le i,\ell\le n} \in \mathcal{S}_n^+(\mathbb{R})$. For each task $j \in \{1,\ldots,p\}$, $\mathcal{D}_n^j = (X_i, y_i^j)_{i=1}^n$ is a sample with distribution $\mathcal{P}_j$, for which a simple regression problem has to be solved. In this paper we consider for simplicity that the different tasks have the same design $(X_i)_{i=1}^n$. When the designs of the different tasks are different the analysis is carried out similarly by defining $X_i = (X_i^1, \ldots, X_i^p)$, but the notations would be more complicated.

We now define the model. We assume $(f^1, \ldots, f^p) \in \mathcal{F}^p$, Σ is a symmetric positive-definite matrix of size p such that the vectors $(\varepsilon_i^j)_{j=1}^p$ are i.i.d. with normal distribution $\mathcal{N}(0,\Sigma)$, with mean zero and covariance matrix Σ, and

$$\forall i \in \{1,\ldots,n\}, \forall j \in \{1,\ldots,p\},\ y_i^j = f^j(X_i) + \varepsilon_i^j \ . \tag{1}$$

This means that, while the observations are independent, the outputs of the different tasks can be correlated, with correlation matrix Σ between the tasks. We now place ourselves in the fixed-design setting, that is, $(X_i)_{i=1}^n$ is deterministic and the goal is to estimate $\left(f^1(X_i), \ldots, f^p(X_i)\right)_{i=1}^n$. Let us introduce some notation:

- $\mu_{\min} = \mu_{\min}(\Sigma)$ (resp. $\mu_{\max}$) denotes the smallest (resp. largest) eigenvalue of Σ.
- $c(\Sigma) := \mu_{\max}/\mu_{\min}$ is the condition number of Σ.

To obtain compact equations, we will use the following definition:

Definition 1 *We denote by F the $n \times p$ matrix $(f^j(X_i))_{1\leq i\leq n, 1\leq j\leq p}$ and introduce the vector $f := \mathrm{vec}(F) = (f^1(X_1),\dots,f^1(X_n),\dots,f^p(X_1),\dots,f^p(X_n)) \in \mathbb{R}^{np}$, obtained by stacking the columns of F. Similarly we define $Y := (y_i^j) \in \mathcal{M}_{n\times p}(\mathbb{R})$, $y := \mathrm{vec}(Y)$, $E := (\varepsilon_i^j) \in \mathcal{M}_{n\times p}(\mathbb{R})$ and $\varepsilon := \mathrm{vec}(E)$.*

In order to estimate f, we use a regularization procedure, which extends the classical ridge regression of the single-task setting. Let M be a $p \times p$ matrix, symmetric and positive-definite. Generalizing the work of Evgeniou et al. (2005), we estimate $(f^1,\dots,f^p) \in \mathcal{F}^p$ by

$$\widehat{f}_M \in \operatorname*{argmin}_{g\in\mathcal{F}^p} \left\{ \frac{1}{np}\sum_{i=1}^{n}\sum_{j=1}^{p}(y_i^j - g^j(X_i))^2 + \sum_{j=1}^{p}\sum_{\ell=1}^{p} M_{j,l}\langle g^j, g^\ell\rangle_{\mathcal{F}} \right\} \quad . \tag{2}$$

Although M could have a general unconstrained form we may restrict M to certain forms, for either computational or statistical reasons.

Remark 2 *Requiring that $M \succeq 0$ implies that Equation* (2) *is a convex optimization problem, which can be solved through the resolution of a linear system, as explained later. Moreover it allows an RKHS interpretation, which will also be explained later.*

Example 3 *The case where the p tasks are treated independently can be considered in this setting: taking $M = M_{\mathrm{ind}}(\lambda) := \frac{1}{p}\mathrm{Diag}(\lambda_1,\dots,\lambda_p)$ for any $\lambda \in \mathbb{R}^p$ leads to the criterion*

$$\frac{1}{p}\sum_{j=1}^{p}\left[\frac{1}{n}\sum_{i=1}^{n}(y_i^j - g^j(X_i))^2 + \lambda_j\|g^j\|_{\mathcal{F}}^2\right] \quad , \tag{3}$$

that is, the sum of the single-task criteria described in Section 3. Hence, minimizing Equation (3) *over $\lambda \in \mathbb{R}^p$ amounts to solve* independently *p single task problems.*

Example 4 *As done by Evgeniou et al. (2005), for every $\lambda,\mu \in (0,+\infty)^2$, define*

$$M_{\mathrm{similar}}(\lambda,\mu) := (\lambda + p\mu)I_p - \mu 1 1^\top = \begin{pmatrix} \lambda+(p-1)\mu & & -\mu \\ & \ddots & \\ -\mu & & \lambda+(p-1)\mu \end{pmatrix} \quad .$$

Taking $M = M_{\mathrm{similar}}(\lambda,\mu)$ in Equation (2) *leads to the criterion*

$$\frac{1}{np}\sum_{i=1}^{n}\sum_{j=1}^{p}(y_i^j - g^j(X_i))^2 + \lambda\sum_{j=1}^{p}\left\|g^j\right\|_{\mathcal{F}}^2 + \frac{\mu}{2}\sum_{j=1}^{p}\sum_{k=1}^{p}\left\|g^j - g^k\right\|_{\mathcal{F}}^2 \quad . \tag{4}$$

Minimizing Equation (4) *enforces a regularization on both the norms of the functions g^j and the norms of the differences $g^j - g^k$. Thus, matrices of the form $M_{\mathrm{similar}}(\lambda,\mu)$ are useful when the functions g^j are assumed to be similar in $\mathcal{F}$. One of the main contributions of the paper is to go beyond this case and learn from data a more general similarity matrix M between tasks.*

Example 5 *We extend Example 4 to the case where the p tasks consist of two groups of close tasks. Let I be a subset of $\{1,\dots,p\}$, of cardinality $1 \leq k \leq p-1$. Let us denote by I^c the complementary*

of I in $\{1,\dots,p\}$, 1_I the vector v with components $v_i = 1_{i\in I}$, and $\mathrm{Diag}(I)$ *the diagonal matrix d with components $d_{i,i} = 1_{i\in I}$. We then define*

$$M_I(\lambda,\mu,\nu) := \lambda I_p + \mu\,\mathrm{Diag}(I) + \nu\,\mathrm{Diag}(I^c) - \frac{\mu}{k}1_I 1_I^\top - \frac{\nu}{p-k}1_{I^c}1_{I^c}^\top \ .$$

This matrix leads to the following criterion, which enforces a regularization on both the norms of the functions g^j and the norms of the differences $g^j - g^k$ inside the groups I and I^c:

$$\frac{1}{np}\sum_{i=1}^{n}\sum_{j=1}^{p}(y_i^j - g^j(X_i))^2 + \lambda\sum_{j=1}^{p}\left\|g^j\right\|_{\mathcal{F}}^2 + \frac{\mu}{2k}\sum_{j\in I}\sum_{k\in I}\left\|g^j - g^k\right\|_{\mathcal{F}}^2 + \frac{\nu}{2(p-k)}\sum_{j\in I^c}\sum_{k\in I^c}\left\|g^j - g^k\right\|_{\mathcal{F}}^2 \ . \tag{5}$$

As shown in Section 6, we can estimate the set I from data (see Jacob et al., 2008 for a more general formulation).

Remark 6 *Since I_p and $11^\top$ can be diagonalized simultaneously, minimizing Equation* (4) *and Equation* (5) *is quite easy: it only demands optimization over two independent parameters, which can be done with the procedure of Arlot and Bach (2011).*

Remark 7 *As stated below (Proposition 8), M acts as a scalar product between the tasks. Selecting a general matrix M is thus a way to express a similarity between tasks.*

Following Evgeniou et al. (2005), we define the vector-space $\mathcal{G}$ of real-valued functions over $\mathcal{X}\times\{1,\dots,p\}$ by

$$\mathcal{G} := \left\{g : \mathcal{X}\times\{1,\dots,p\} \to \mathbb{R} \,/\, \forall j \in \{1,\dots,p\}\ , g(\cdot,j)\in\mathcal{F}\right\} \ .$$

We now define a bilinear symmetric form over $\mathcal{G}$,

$$\forall g,h\in\mathcal{G} \ , \quad \langle g,h\rangle_{\mathcal{G}} := \sum_{j=1}^{p}\sum_{l=1}^{p} M_{j,l}\langle g(\cdot,j), h(\cdot,l)\rangle_{\mathcal{F}},$$

which is a scalar product as soon as M is positive semi-definite (see proof in Appendix A) and leads to a RKHS (see proof in Appendix B):

Proposition 8 *With the preceding notations $\langle\cdot,\cdot\rangle_{\mathcal{G}}$ is a scalar product on $\mathcal{G}$.*

Corollary 9 *$(\mathcal{G},\langle\cdot,\cdot\rangle_{\mathcal{G}})$ is a RKHS.*

In order to write down the kernel matrix in compact form, we introduce the following notations.

Definition 10 (Kronecker Product) *Let $A\in\mathcal{M}_{m,n}(\mathbb{R})$, $B\in\mathcal{M}_{p,q}(\mathbb{R})$. We define the Kronecker product $A\otimes B$ as being the $(mp)\times(nq)$ matrix built with $p\times q$ blocks, the block of index (i,j) being $A_{i,j}\cdot B$:*

$$A\otimes B = \begin{pmatrix} A_{1,1}B & \dots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{m,1}B & \dots & A_{m,n}B \end{pmatrix} \ .$$

The Kronecker product is a widely used tool to deal with matrices and tensor products. Some of its classical properties are given in Section E; see also Horn and Johnson (1991).

Proposition 11 *The kernel matrix associated with the design* $\widetilde{X} := (X_i, j)_{i,j} \in \mathcal{X} \times \{1,\dots,p\}$ *and the RKHS* $(\mathcal{G}, \langle\cdot,\cdot\rangle_{\mathcal{G}})$ *is* $\widetilde{K}_M := M^{-1} \otimes K$.

Proposition 11 is proved in Appendix C. We can then apply the representer's theorem (Schölkopf and Smola, 2002) to the minimization problem (2) and deduce that $\widehat{f}_M = A_M y$ with

$$A_M = A_{M,K} := \widetilde{K}_M(\widetilde{K}_M + npI_{np})^{-1} = (M^{-1} \otimes K)\left((M^{-1} \otimes K) + npI_{np}\right)^{-1} \quad .$$

2.2 Optimal Choice of the Kernel

Now when working in multi-task regression, a set $\mathcal{M} \subset \mathcal{S}_p^{++}(\mathbb{R})$ of matrices M is given, and the goal is to select the "best" one, that is, minimizing over M the quadratic risk $n^{-1}\|\widehat{f}_M - f\|_2^2$. For instance, the single-task framework corresponds to $p = 1$ and $\mathcal{M} = (0, +\infty)$. The multi-task case is far richer. The oracle risk is defined as

$$\inf_{M \in \mathcal{M}} \left\{ \left\| \widehat{f}_M - f \right\|_2^2 \right\} \quad . \tag{6}$$

The ideal choice, called the oracle, is any matrix

$$M^\star \in \operatorname*{argmin}_{M \in \mathcal{M}} \left\{ \left\| \widehat{f}_M - f \right\|_2^2 \right\} \quad .$$

Nothing here ensures the oracle exists. However in some special cases (see for instance Example 12) the infimum of $\|\widehat{f}_M - f\|^2$ over the set $\{\widehat{f}_M,\ M \in \mathcal{M}\}$ may be attained by a function $f^* \in \mathcal{F}^p$— which we will call "oracle" by a slight abuse of notation—while the former problem does not have a solution.

From now on we always suppose that the infimum of $\{\|\widehat{f}_M - f\|^2\}$ over $\mathcal{M}$ is attained by some function $f^\star \in \mathcal{F}^p$. However the oracle $M^\star$ is not an estimator, since it depends on f.

Example 12 (Partial computation of the oracle in a simple setting) *It is possible in certain simple settings to exactly compute the oracle (or, at least, some part of it). Consider for instance the set-up where the p functions are taken to be equal (that is,* $f^1 = \cdots = f^p$*). In this setting it is natural to use the set*

$$\mathcal{M}_{\text{similar}} := \left\{ M_{\text{similar}}(\lambda,\mu) = (\lambda + p\mu)I_p - \frac{\mu}{p} 11^\top \,/\, (\lambda,\mu) \in (0,+\infty)^2 \right\} \quad .$$

Using the estimator $\widehat{f}_M = A_M y$ *we can then compute the quadratic risk using the bias-variance decomposition given in Equation* (36):

$$\mathbb{E}\left[\left\| \widehat{f}_M - f \right\|_2^2 \right] = \|(A_M - I_{np})f\|_2^2 + \operatorname{tr}(A_M^\top A_M \cdot (\Sigma \otimes I_n)) \quad .$$

Computations (*reported in Appendix D*) *show that, with the change of variables* $\widetilde{\mu} = \lambda + p\mu$*, the bias does not depend on* $\widetilde{\mu}$ *and the variance is a decreasing function of* $\widetilde{\mu}$*. Thus the oracle is obtained when* $\widetilde{\mu} = +\infty$*, leading to a situation where the oracle functions* $f^{1,\star},\dots,f^{p,\star}$ *verify* $f^{1,\star} = \cdots = f^{p,\star}$*. It is also noticeable that, if one assumes the maximal eigenvalue of* Σ *stays bounded with respect to p, the variance is of order* $O(p^{-1})$ *while the bias is bounded with respect to p.*

As explained by Arlot and Bach (2011), we choose

$$\widehat{M} \in \operatorname*{argmin}_{M \in \mathcal{M}} \{\mathrm{crit}(M)\} \quad \text{with} \quad \mathrm{crit}(M) = \frac{1}{np} \left\| y - \widehat{f}_M \right\|_2^2 + \mathrm{pen}(M) \ ,$$

where the penalty term $\mathrm{pen}(M)$ has to be chosen appropriately.

Remark 13 *Our model* (1) *does not constrain the functions $f^1, \ldots, f^p$. Our way to express the similarities between the tasks (that is, between the f^j) is via the set $\mathcal{M}$, which represents the a priori knowledge the statistician has about the problem. Our goal is to build an estimator whose risk is the closest possible to the oracle risk. Of course using an inappropriate set $\mathcal{M}$ (with respect to the target functions $f^1, \ldots, f^p$) may lead to bad overall performances. Explicit multi-task settings are given in Examples 3, 4 and 5 and through simulations in Section 6.*

The unbiased risk estimation principle (introduced by Akaike, 1970) requires

$$\mathbb{E}\left[\mathrm{crit}(M)\right] \approx \mathbb{E}\left[\frac{1}{np} \left\| \widehat{f}_M - f \right\|_2^2\right] \ ,$$

which leads to the (deterministic) *ideal penalty*

$$\mathrm{pen}_{\mathrm{id}}(M) := \mathbb{E}\left[\frac{1}{np} \|\widehat{f}_M - f\|_2^2\right] - \mathbb{E}\left[\frac{1}{np} \left\| y - \widehat{f}_M \right\|_2^2\right] \ .$$

Since $\widehat{f}_M = A_M y$ and $y = f + \varepsilon$, we can write

$$\left\| \widehat{f}_M - y \right\|_2^2 = \left\| \widehat{f}_M - f \right\|_2^2 + \|\varepsilon\|_2^2 - 2\langle \varepsilon, A_M \varepsilon \rangle + 2\langle \varepsilon, (I_{np} - A_M) f \rangle \ .$$

Since ε is centered and M is deterministic, we get, up to an additive factor independent of M,

$$\mathrm{pen}_{\mathrm{id}}(M) = \frac{2\mathbb{E}\left[\langle \varepsilon, A_M \varepsilon \rangle\right]}{np} \ ,$$

that is, as the covariance matrix of ε is $\Sigma \otimes I_n$,

$$\mathrm{pen}_{\mathrm{id}}(M) = \frac{2\,\mathrm{tr}\left(A_M \cdot (\Sigma \otimes I_n)\right)}{np} \ . \tag{7}$$

In order to approach this penalty as precisely as possible, we have to sharply estimate Σ. In the single-task case, such a problem reduces to estimating the variance σ^2 of the noise and was tackled by Arlot and Bach (2011). Since our approach for estimating Σ heavily relies on these results, they are summarized in the next section.

Note that estimating Σ is a mean towards estimating M. The technique we develop later for this purpose is not purely a multi-task technique, and may also be used in a different context.

3. Single Task Framework: Estimating a Single Variance

This section recalls some of the main results from Arlot and Bach (2011) which can be considered as solving a special case of Section 2, with $p = 1$, $\Sigma = \sigma^2 > 0$ and $\mathcal{M} = [0, +\infty]$. Writing $M = \lambda$ with $\lambda \in [0, +\infty]$, the regularization matrix is

$$\forall \lambda \in (0, +\infty), \quad A_\lambda = A_{\lambda,K} = K(K + n\lambda I_n)^{-1} \ ,$$

$A_0 = I_n$ and $A_{+\infty} = 0$; the ideal penalty becomes

$$\mathrm{pen}_{\mathrm{id}}(\lambda) = \frac{2\sigma^2 \operatorname{tr}(A_\lambda)}{n} \ .$$

By analogy with the case where A_λ is an orthogonal projection matrix, $\mathrm{df}(\lambda) := \operatorname{tr}(A_\lambda)$ is called the effective degree of freedom, first introduced by Mallows (1973); see also the work by Zhang (2005). The ideal penalty however depends on σ^2; in order to have a fully data-driven penalty we have to replace σ^2 by an estimator $\widehat{\sigma}^2$ inside $\mathrm{pen}_{\mathrm{id}}(\lambda)$. For every $\lambda \in [0, +\infty]$, define

$$\mathrm{pen}_{\min}(\lambda) = \mathrm{pen}_{\min}(\lambda, K) := \frac{(2\operatorname{tr}(A_{\lambda,K}) - \operatorname{tr}(A_{\lambda,K}^\top A_{\lambda,K}))}{n} \ .$$

We shall see now that it is a *minimal penalty* in the following sense. If for every $C > 0$

$$\widehat{\lambda}_0(C) \in \operatorname*{argmin}_{\lambda \in [0,+\infty]} \left\{ \frac{1}{n} \left\| A_{\lambda,K} Y - Y \right\|_2^2 + C\,\mathrm{pen}_{\min}(\lambda, K) \right\} \ ,$$

then—up to concentration inequalities—$\widehat{\lambda}_0(C)$ acts as a mimimizer of

$$g_C(\lambda) = \mathbb{E}\left[\frac{1}{n} \|A_\lambda Y - Y\|_2^2 + C\,\mathrm{pen}_{\min}(\lambda) \right] - \sigma^2 = \frac{1}{n} \|(A_\lambda - I_n) f\|_2^2 + (C - \sigma^2)\,\mathrm{pen}_{\min}(\lambda) \ .$$

The former theoretical arguments show that

- if $C < \sigma^2$, $g_C(\lambda)$ decreases with $\mathrm{df}(\lambda)$ so that $\mathrm{df}(\widehat{\lambda}_0(C))$ is huge: the procedure overfits;
- if $C > \sigma^2$, $g_C(\lambda)$ increases with $\mathrm{df}(\lambda)$ when $\mathrm{df}(\lambda)$ is large enough so that $\mathrm{df}(\widehat{\lambda}_0(C))$ is much smaller than when $C < \sigma^2$.

The following algorithm was introduced by Arlot and Bach (2011) and uses this fact to estimate σ^2.

Algorithm 14 **Input:** $Y \in \mathbb{R}^n$, $K \in \mathcal{S}_n^{++}(\mathbb{R})$

1. *For every* $C > 0$, *compute*

$$\widehat{\lambda}_0(C) \in \operatorname*{argmin}_{\lambda \in [0,+\infty]} \left\{ \frac{1}{n} \left\| A_{\lambda,K} Y - Y \right\|_2^2 + C\,\mathrm{pen}_{\min}(\lambda, K) \right\} \ .$$

2. **Output:** $\widehat{C}$ *such that* $\mathrm{df}(\widehat{\lambda}_0(\widehat{C})) \in [n/10, n/3]$.

An efficient algorithm for the first step of Algorithm 14 is detailed by Arlot and Massart (2009), and we discuss the way we implemented Algorithm 14 in Section 6. The output $\widehat{C}$ of Algorithm 14 is a provably consistent estimator of σ^2, as stated in the following theorem.

Theorem 15 (Corollary of Theorem 1 of Arlot and Bach, 2011) *Let* $\beta = 150$. *Suppose* $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_n)$ *with* $\sigma^2 > 0$, *and that* $\lambda_0 \in (0, +\infty)$ *and* $d_n \geq 1$ *exist such that*

$$\mathrm{df}(\lambda_0) \leq \sqrt{n} \text{ and } \frac{1}{n} \left\| (A_{\lambda_0} - I_n) F \right\|_2^2 \leq d_n \sigma^2 \sqrt{\frac{\ln n}{n}} \ . \tag{8}$$

Then for every $\delta \geq 2$, *some constant* $n_0(\delta)$ *and an event* Ω *exist such that* $\mathbb{P}(\Omega) \geq 1 - n^{-\delta}$ *and if* $n \geq n_0(\delta)$, *on* Ω,

$$\left(1 - \beta(2+\delta)\sqrt{\frac{\ln n}{n}}\right)\sigma^2 \leq \widehat{C} \leq \left(1 + \beta(2+\delta) d_n \sqrt{\frac{\ln(n)}{n}}\right)\sigma^2 \ . \tag{9}$$

Remark 16 *The values* $n/10$ *and* $n/3$ *in Algorithm 14 have no particular meaning and can be replaced by* n/k, n/k', *with* $k > k' > 2$. *Only* β *depends on* k *and* k'. *Also the bounds required in Assumption* (8) *only impact the right hand side of Equation* (9) *and are chosen to match the left hand side. See Proposition 10 of Arlot and Bach (2011) for more details.*

4. Estimation of the Noise Covariance Matrix Σ

Thanks to the results developped by Arlot and Bach (2011) (recapitulated in Section 3), we know how to estimate a variance for any one-dimensional problem. In order to estimate Σ, which has $p(p+1)/2$ parameters, we can use several one-dimensional problems. Projecting Y onto some direction $z \in \mathbb{R}^p$ yields

$$Y_z := Y \cdot z = F \cdot z + E \cdot z = F_z + \varepsilon_z \ , \tag{10}$$

with $\varepsilon_z \sim \mathcal{N}(0, \sigma_z^2 I_n)$ and $\sigma_z^2 := \mathrm{Var}[\varepsilon \cdot z] = z^\top \Sigma z$. Therefore, we will estimate σ_z^2 for $z \in \mathcal{Z}$ a well chosen set, and use these estimators to build back an estimation of Σ.

We now explain how to estimate Σ using those one-dimensional projections.

Definition 17 *Let* $a(z)$ *be the output* $\widehat{C}$ *of Algorithm 14 applied to problem* (10), *that is, with inputs* $Y_z \in \mathbb{R}^n$ *and* $K \in \mathcal{S}_n^{++}(\mathbb{R})$.

The idea is to apply Algorithm 14 to the elements z of a carefully chosen set $\mathcal{Z}$. Noting e_i the i-th vector of the canonical basis of $\mathbb{R}^p$, we introduce $\mathcal{Z} = \{e_i,\ i \in \{1, \ldots, p\}\} \cup \{e_i + e_j,\ 1 \leq i < j \leq p\}$. We can see that $a(e_i)$ estimates $\Sigma_{i,i}$, while $a(e_i + e_j)$ estimates $\Sigma_{i,i} + \Sigma_{j,j} + 2\Sigma_{i,j}$. Henceforth, $\Sigma_{i,j}$ can be estimated by $(a(e_i + e_j) - a(e_i) - a(e_j))/2$. This leads to the definition of the following map J, which builds a symmetric matrix using the latter construction.

Definition 18 *Let* $J : \mathbb{R}^{\frac{p(p+1)}{2}} \to \mathcal{S}_p(\mathbb{R})$ *be defined by*

$$J(a_1, \ldots, a_p, a_{1,2}, \ldots, a_{1,p}, \ldots, a_{p-1,p})_{i,i} = a_i \text{ if } 1 \leq i \leq p \ ,$$
$$J(a_1, \ldots, a_p, a_{1,2}, \ldots, a_{1,p}, \ldots, a_{p-1,p})_{i,j} = \frac{a_{i,j} - a_i - a_j}{2} \text{ if } 1 \leq i < j \leq p \ .$$

This map is bijective, and for all $B \in \mathcal{S}_p(\mathbb{R})$

$$J^{-1}(B) = (B_{1,1}, \ldots, B_{p,p}, B_{1,1} + B_{2,2} + 2B_{1,2}, \ldots, B_{p-1,p-1} + B_{p,p} + 2B_{p-1,p}) \ .$$

This leads us to defining the following estimator of Σ:

$$\widehat{\Sigma} := J\left(a(e_1), \ldots, a(e_p), a(e_1+e_2), \ldots, a(e_1+e_p), \ldots, a(e_{p-1}+e_p)\right) \ . \tag{11}$$

Remark 19 *If a diagonalization basis $(e'_1, \ldots, e'_p)$ (whose basis matrix is P) of Σ is known, or if Σ is diagonal, then a simplified version of the algorithm defined by Equation* (11) *is*

$$\widehat{\Sigma}_{\text{simplified}} = P^\top \operatorname{Diag}(a(e'_1), \ldots, a(e'_p)) P \ . \tag{12}$$

This algorithm has a smaller computational cost and leads to better theoretical bounds (*see Remark 24 and Section 5.2*).

Let us recall that $\forall \lambda \in (0, +\infty)$, $A_\lambda = A_{\lambda,K} = K(K + n\lambda I_n)^{-1}$. Following Arlot and Bach (2011) we make the following assumption from now on:

$$\left.\begin{array}{l} \forall j \in \{1, \ldots, p\}, \exists \lambda_{0,j} \in (0, +\infty), \\ \qquad \operatorname{df}(\lambda_{0,j}) \leq \sqrt{n} \quad \text{and} \quad \dfrac{1}{n} \left\| (A_{\lambda_{0,j}} - I_n) F_{e_j} \right\|_2^2 \leq \Sigma_{j,j} \sqrt{\dfrac{\ln n}{n}} \end{array}\right\} \tag{13}$$

We can now state the first main result of the paper.

Theorem 20 *Let $\widehat{\Sigma}$ be defined by Equation* (11), $\alpha = 2$ *and assume* (13) *holds. For every $\delta \geq 2$, a constant $n_0(\delta)$, an absolute constant $L_1 > 0$ and an event Ω exist such that $\mathbb{P}(\Omega) \geq 1 - p(p+1)/2 \times n^{-\delta}$ and if $n \geq n_0(\delta)$, on Ω,*

$$(1-\eta)\Sigma \preceq \widehat{\Sigma} \preceq (1+\eta)\Sigma \tag{14}$$

$$\text{where} \qquad \eta := L_1(2+\delta) p \sqrt{\frac{\ln(n)}{n}} c(\Sigma)^2 \ .$$

Theorem 20 is proved in Section E. It shows $\widehat{\Sigma}$ estimates Σ with a "multiplicative" error controlled with large probability, in a non-asymptotic setting. The multiplicative nature of the error is crucial for deriving the oracle inequality stated in Section 5, since it allows to show the ideal penalty defined in Equation (7) is precisely estimated when Σ is replaced by $\widehat{\Sigma}$.

An important feature of Theorem 20 is that it holds under very mild assumptions on the mean f of the data (see Remark 22). Therefore, it shows $\widehat{\Sigma}$ is able to estimate a covariance matrix *without prior knowledge on the regression function*, which, to the best of our knowledge, has never been obtained in multi-task regression.

Remark 21 (Scaling of (n,p) for consistency) *A sufficient condition for ensuring $\widehat{\Sigma}$ is a consistent estimator of Σ is*

$$p c(\Sigma)^2 \sqrt{\frac{\ln(n)}{n}} \longrightarrow 0 \ ,$$

which enforces a scaling between n, p and $c(\Sigma)$. Nevertheless, this condition is probably not necessary since the simulation experiments of Section 6 show that Σ can be well estimated (*at least for estimator selection purposes*) *in a setting where $\eta \gg 1$.*

Remark 22 (On assumption (13)) *Assumption* (13) *is a single-task assumption* (*made independently for each task*). *The upper bound* $\sqrt{\ln(n)/n}$ *can be multiplied by any factor* $1 \leq d_n \ll \sqrt{n/\ln(n)}$ (*as in Theorem 15*), *at the price of multiplying* η *by* d_n *in the upper bound of Equation* (14). *More generally the bounds on the degree of freedom and the bias in* (13) *only influence the upper bound of Equation* (14). *The rates are chosen here to match the lower bound, see Proposition 10 of Arlot and Bach (2011) for more details*.

Assumption (13) *is rather classical in model selection, see Arlot and Bach (2011) for instance*. *In particular,* (*a weakened version of*) (13) *holds if the bias* $n^{-1}\|(A_\lambda - I_n)F_{e_i}\|_2^2$ *is bounded by* $C_1 \operatorname{tr}(A_\lambda)^{-C_2}$, *for some* $C_1, C_2 > 0$.

Remark 23 (Choice of the set $\mathcal{Z}$**)** *Other choices could have been made for* $\mathcal{Z}$, *however ours seems easier in terms of computation, since* $|\mathcal{Z}| = p(p+1)/2$. *Choosing a larger set* $\mathcal{Z}$ *leads to theoretical difficulties in the reconstruction of* $\widehat{\Sigma}$, *while taking other basis vectors leads to more complex computations*. *We can also note that increasing* $|\mathcal{Z}|$ *decreases the probability in Theorem 20, since it comes from an union bound over the one-dimensional estimations*.

Remark 24 *When* $\widehat{\Sigma} = \widehat{\Sigma}_{\text{simplified}}$ *as defined by Equation* (12), *that is, when a diagonalization basis of* Σ *is known, Theorem 20 still holds on a set of larger probability* $1 - \kappa p n^{-\delta}$ *with a reduced error* $\eta = L_1(\alpha+\delta)\sqrt{\ln(n)/n}$. *Then, a consistent estimation of* Σ *is possible whenever* $p = O(n^\delta)$ *for some* $\delta \geq 0$.

5. Oracle Inequality

This section aims at proving "oracle inequalities", as usually done in a model selection setting: given a set of models or of estimators, the goal is to upper bound the risk of the selected estimator by the oracle risk (defined by Equation (6)), up to an additive term and a multiplicative factor. We show two oracle inequalities (Theorems 26 and 29) that correspond to two possible definitions of $\widehat{\Sigma}$.

Note that "oracle inequality" sometimes has a different meaning in the literature (see for instance Lounici et al., 2011) when the risk of the proposed estimator is controlled by the risk of an estimator using information coming from the true parameter (that is, available only if provided by an oracle).

5.1 A General Result for Discrete Matrix Sets $\mathcal{M}$

We first show that the estimator introduced in Equation (11) is precise enough to derive an oracle inequality when plugged in the penalty defined in Equation (7) in the case where $\mathcal{M}$ is finite.

Definition 25 *Let* $\widehat{\Sigma}$ *be the estimator of* Σ *defined by Equation* (11). *We define*

$$\widehat{M} \in \operatorname*{argmin}_{M \in \mathcal{M}} \left\{ \left\| \widehat{f}_M - y \right\|_2^2 + 2 \operatorname{tr}\left(A_M \cdot (\widehat{\Sigma} \otimes I_n) \right) \right\} \quad .$$

We assume now the following holds true:

$$\exists (C, \alpha_{\mathcal{M}}) \in (0, +\infty)^2, \quad \operatorname{card}(\mathcal{M}) < C n^{\alpha_{\mathcal{M}}} \quad . \tag{15}$$

Theorem 26 *Let $\alpha = \max(\alpha_{\mathcal{M}}, 2)$, $\delta \geq 2$ and assume* (13) *and* (15) *hold true. Absolute constants $L_2, \kappa' > 0$, a constant $n_1(\delta)$ and an event $\widetilde{\Omega}$ exist such that $\mathbb{P}(\widetilde{\Omega}) \geq 1 - \kappa' p(p+C) n^{-\delta}$ and the following holds as soon as $n \geq n_1(\delta)$. First, on $\widetilde{\Omega}$,*

$$\frac{1}{np}\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^2 \leq \left(1 + \frac{1}{\ln(n)}\right)^2 \inf_{M \in \mathcal{M}} \left\{ \frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2 \right\} + L_2 c(\Sigma)^4 \operatorname{tr}(\Sigma)(\alpha+\delta)^2 \frac{p^3 \ln(n)^3}{np} \quad . \tag{16}$$

Second, an absolute constant L_3 exists such that

$$\begin{aligned} \mathbb{E}\left[\frac{1}{np}\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^2\right] &\leq \left(1 + \frac{1}{\ln(n)}\right)^2 \mathbb{E}\left[\inf_{M \in \mathcal{M}} \left\{ \frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2 \right\}\right] \\ &\quad + L_2 c(\Sigma)^4 \operatorname{tr}(\Sigma)(\alpha+\delta)^2 \frac{p^3 \ln(n)^3}{np} + L_3 \frac{\sqrt{p(p+C)}}{n^{\delta/2}} \left(|||\Sigma||| + \frac{\|f\|_2^2}{np} \right) \quad . \end{aligned} \tag{17}$$

Theorem 26 is proved in Section F.

Remark 27 *If $\widehat{\Sigma} = \widehat{\Sigma}_{\text{simplified}}$ is defined by Equation* (12) *the result still holds on a set of larger probability $1 - \kappa' p(1+C) n^{-\delta}$ with a reduced error, similar to the one in Theorem 29.*

5.2 A Result for a Continuous Set of Jointly Diagonalizable Matrices

We now show a similar result when matrices in $\mathcal{M}$ can be jointly diagonalized. It turns out a faster algorithm can be used instead of Equation (11) with a reduced error and a larger probability event in the oracle inequality. Note that we no longer assume $\mathcal{M}$ is finite, so it can be parametrized by continuous parameters.

Suppose now the following holds, which means the matrices of $\mathcal{M}$ are jointly diagonalizable:

$$\exists P \in O_p(\mathbb{R}), \quad \mathcal{M} \subseteq \left\{ P^\top \operatorname{Diag}(d_1, \ldots, d_p) P, (d_i)_{i=1}^p \in (0, +\infty)^p \right\} \quad . \tag{18}$$

Let P be the matrix defined in Assumption (18), $\widetilde{\Sigma} = P \Sigma P^\top$ and recall that $A_\lambda = K(K + n\lambda I_n)^{-1}$. Computations detailed in Appendix D show that the ideal penalty introduced in Equation (7) can be written as

$$\begin{aligned} &\forall M = P^\top \operatorname{Diag}(d_1, \ldots, d_p) P \in \mathcal{M}, \\ &\qquad \operatorname{pen}_{\text{id}}(M) = \frac{2 \operatorname{tr}\left(A_M \cdot (\Sigma \otimes I_n)\right)}{np} = \frac{2}{np}\left(\sum_{j=1}^p \operatorname{tr}(A_{pd_j}) \widetilde{\Sigma}_{j,j} \right) \quad . \end{aligned} \tag{19}$$

Equation (19) shows that under Assumption (18), we do not need to estimate the entire matrix Σ in order to have a good penalization procedure, but only to estimate the variance of the noise in p directions.

Definition 28 *Let $(e_1, \ldots, e_p)$ be the canonical basis of $\mathbb{R}^p$, $(u_1, \ldots, u_p)$ be the orthogonal basis defined by $\forall j \in \{1, \ldots, p\}$, $u_j = P^\top e_j$. We then define*

$$\widehat{\Sigma}_{HM} = P \operatorname{Diag}(a(u_1), \ldots, a(u_p)) P^\top \quad ,$$

*where for every $j \in \{1, \ldots, p\}$, $a(u_j)$ denotes the output of Algorithm 14 applied to Problem (**Pu**$_\mathbf{j}$), and*

$$\widehat{M}_{HM} \in \operatorname*{argmin}_{M \in \mathcal{M}} \left\{ \left\|\widehat{f}_M - y\right\|_2^2 + 2 \operatorname{tr}\left(A_M \cdot (\widehat{\Sigma}_{HM} \otimes I_n)\right) \right\} \quad . \tag{20}$$

Theorem 29 *Let $\alpha = 2$, $\delta \geq 2$ and assume* (13) *and* (18) *hold true. Absolute constants $L_2 > 0$, and κ'', a constant $n_1(\delta)$ and an event $\widetilde{\Omega}$ exist such that $\mathbb{P}(\widetilde{\Omega}) \geq 1 - \kappa'' pn^{-\delta}$ and the following holds as soon as $n \geq n_1(\delta)$. First, on $\widetilde{\Omega}$,*

$$\frac{1}{np}\left\|\widehat{f}_{\widehat{M}_{HM}} - f\right\|_2^2 \leq \left(1 + \frac{1}{\ln(n)}\right)^2 \inf_{M \in \mathcal{M}} \left\{\frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2\right\} + L_2 \operatorname{tr}(\Sigma)(2+\delta)^2 \frac{\ln(n)^3}{n} \quad . \tag{21}$$

Second, an absolute constant L_4 exists such that

$$\begin{aligned}\mathbb{E}\left[\frac{1}{np}\left\|\widehat{f}_{\widehat{M}_{HM}} - f\right\|_2^2\right] \leq \left(1 + \frac{1}{\ln(n)}\right)^2 \mathbb{E}\left[\inf_{M \in \mathcal{M}} \left\{\frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2\right\}\right] \\ + L_4 \operatorname{tr}(\Sigma)(2+\delta)^2 \frac{\ln(n)^3}{n} + \frac{p}{n^{\delta/2}} \frac{\|f\|_2^2}{np} \quad .\end{aligned} \tag{22}$$

Theorem 29 is proved in Section F.

5.3 Comments on Theorems 26 and 29

Remark 30 *Taking $p = 1$ (hence $c(\Sigma) = 1$ and $\operatorname{tr}(\Sigma) = \sigma^2$), we recover Theorem 3 of Arlot and Bach (2011) as a corollary of Theorem 26.*

Remark 31 (Scaling of (n, p)) *When assumption* (15) *holds, Equation* (16) *implies the asymptotic optimality of the estimator $\widehat{f}_{\widehat{M}}$ when*

$$c(\Sigma)^4 \frac{\operatorname{tr}\Sigma}{p} \times \frac{p^3 \left(\ln(n)\right)^3}{n} \ll \inf_{M \in \mathcal{M}} \left\{\frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2\right\} \quad .$$

In particular, only (n, p) such that $p^3 \ll n/(\ln(n))^3$ are admissible. When assumption (18) *holds, the scalings required to ensure optimality in Equation* (21) *are more favorable:*

$$\operatorname{tr}\Sigma \times \frac{\left(\ln(n)\right)^3}{n} \ll \inf_{M \in \mathcal{M}} \left\{\frac{1}{np}\left\|\widehat{f}_M - f\right\|_2^2\right\} \quad .$$

It is to be noted that p still influences the left hand side via $\operatorname{tr}\Sigma$.

Remark 32 *Theorems 26 and 29 are non asymptotic oracle inequalities, with a multiplicative term of the form $1 + o(1)$. This allows us to claim that our selection procedure is nearly optimal, since our estimator is close (with regard to the empirical quadratic norm) to the oracle one. Furthermore the term $1 + (\ln(n))^{-1}$ in front of the infima in Equations* (16), (21), (17) *and* (22) *can be further diminished, but this yields a greater remainder term as a consequence.*

Remark 33 (On assumption (18)**)** *Assumption* (18) *actually means all matrices in $\mathcal{M}$ can be diagonalized in a unique orthogonal basis, and thus can be parametrized by their eigenvalues as in Examples 3, 4 and 5.*

In that case the optimization problem is quite easy to solve, as detailed in Remark 36. If not, solving (20) *may turn out to be a hard problem, and our theoretical results do not cover this setting. However, it is always possible to discretize the set $\mathcal{M}$ or, in practice, to use gradient descent.*

Compared to the setting of Theorem 26, assumption (18) *allows a simpler estimator for the penalty* (19)*, with an increased probability and a reduced error in the oracle inequality.*

The main theoretical limitation comes from the fact that the probabilistic concentration tools used apply to discrete sets $\mathcal{M}$ (through union bounds). The structure of kernel ridge regression allows us to have a uniform control over a continuous set for the single-task estimators at the "cost" of n pointwise controls, which can then be extended to the multi-task setting via (18)*. We conjecture Theorem 29 still holds without* (18) *as long as $\mathcal{M}$ is not "too large", which could be proved similarly up to some uniform concentration inequalities.*

Note also that if $\mathcal{M}_1, \ldots, \mathcal{M}_K$ all satisfy (18) *(with different matrices P_k), then Theorem 29 still holds for $\mathcal{M} = \bigcup_{k=1}^{K} \mathcal{M}_k$ with the penalty defined by Equation* (20) *with $P = P_k$ when $M \in \mathcal{M}_k$, and $\mathbb{P}(\widetilde{\Omega}) \geq 1 - 9Kp^2n^{-\delta}$, by applying the union bound in the proof.*

Remark 34 (Relationship with the trace norm) *Our approach relies on the minimization of Equation* (2) *with respect to f. Argyriou et al. (2008) has shown that if we also minimize Equation* (2) *with respect to the matrix M subject to the constraint $\operatorname{tr} M^{-1} = 1$, then we obtain an equivalent regularization by the nuclear norm (a.k.a. trace norm), which implies the prior knowledge that our p prediction functions may be obtained as the linear combination of $r \ll p$ basis functions. This situation corresponds to cases where the matrix M^{-1} is singular.*

Note that the link between our framework and trace norm (i.e., nuclear norm) regularization is the same than between multiple kernel learning and the single task framework of Arlot and Bach (2011). In the multi-task case, the trace-norm regularization, though efficient computationally, does not lead to an oracle inequality, while our criterion is an unbiased estimate of the generalization error, which turns out to be non-convex in the matrix M. While DC programming techniques (see, e.g., Gasso et al., 2009, and references therein) could be brought to bear to find local optima, the goal of the present work is to study the theoretical properties of our estimators, assuming we can minimize the cost function (e.g., in special cases, where we consider spectral variants, or by brute force enumeration).

6. Simulation Experiments

In all the experiments presented in this section, we consider the framework of Section 2 with $\mathcal{X} = \mathbb{R}^d$, $d = 4$, and the kernel defined by $\forall x, y \in \mathcal{X}$, $k(x,y) = \prod_{j=1}^{d} e^{-|x_j - y_j|}$. The design points $X_1, \ldots, X_n \in \mathbb{R}^d$ are drawn (repeatedly and independently for each sample) independently from the multivariate standard Gaussian distribution. For every $j \in \{1, \ldots, p\}$, $f^j(\cdot) = \sum_{i=1}^{m} \alpha_i^j k(\cdot, z_i)$ where $m = 4$ and $z_1, \ldots, z_m \in \mathbb{R}^d$ are drawn (once for all experiments except in Experiment D) independently from the multivariate standard Gaussian distribution, independently from the design $(X_i)_{1 \leq i \leq n}$. Thus, the expectations that will be considered are taken conditionally to the z_i. The coefficients $(\alpha_i^j)_{1 \leq i \leq m, 1 \leq j \leq p}$ differ according to the setting. Matlab code is available online.[1]

6.1 Experiments

Five experimental settings are considered:

A⌋ Various numbers of tasks: $n = 10$ and $\forall i, j, \alpha_i^j = 1$, that is, $\forall j, f^j = f_A := \sum_{i=1}^{m} k(\cdot, z_i)$. The number of tasks is varying: $p \in \{2k \,/\, k = 1, \ldots, 25\}$. The covariance matrix is $\Sigma = 10 \cdot I_p$.

1. Matlab code can be found at `http://www.di.ens.fr/~solnon/multitask_minpen_en.html`.

B⌋ Various sample sizes: $p = 5, \forall j, f^j = f_A$ and $\Sigma = \Sigma_B$ has been drawn (once for all) from the Whishart $W(I_5, 10, 5)$ distribution; the condition number of Σ_B is $c(\Sigma_B) \approx 22.05$. The only varying parameter is $n \in \{50k \,/\, k = 1, \ldots, 20\}$.

C⌋ Various noise levels: $n = 100$, $p = 5$ and $\forall j, f^j = f_A$. The varying parameter is $\Sigma = \Sigma_{C,t} := 5t \cdot I_5$ with $t \in \{0.2k \,/\, k = 1, \ldots, 50\}$. We also ran the experiments for $t = 0.01$ and $t = 100$.

D⌋ Clustering of two groups of functions: $p = 10$, $n = 100$, $\Sigma = \Sigma_E$ has been drawn (once for all) from the Whishart $W(I_{10}, 20, 10)$ distribution; the condition number of Σ_E is $c(\Sigma_E) \approx 24.95$. We pick the function $f_D := \sum_{i=1}^m \alpha_i k(\cdot, z_i)$ by drawing $(\alpha_1, \ldots, \alpha_m)$ and $(z_1, \ldots, z_m)$ from standard multivariate normal distribution (independently in each replication) and finally $f^1 = \cdots = f^5 = f_D$, $f^6 = \cdots = f^{10} = -f_D$.

E⌋ Comparison to cross-validation parameter selection: $p = 5, \Sigma = 10 \cdot I_5, \forall j, f^j = f_A$. The sample size is taken in $\{10, 50, 100, 250\}$.

6.2 Collections of Matrices

Two different sets of matrices $\mathcal{M}$ are considered in the Experiments A–C, following Examples 3 and 4:

$$\mathcal{M}_{\text{similar}} := \left\{ M_{\text{similar}}(\lambda, \mu) = (\lambda + p\mu) I_p - \frac{\mu}{p} 11^\top \,/\, (\lambda, \mu) \in (0, +\infty)^2 \right\}$$
$$\text{and} \quad \mathcal{M}_{\text{ind}} := \{ M_{\text{ind}}(\lambda) = \text{Diag}(\lambda_1, \ldots, \lambda_p) \,/\, \lambda \in (0, +\infty)^p \} \quad .$$

In Experiment D, we also use two different sets of matrices, following Example 5:

$$\mathcal{M}_{\text{clus}} := \bigcup_{I \subset \{1,\ldots,p\}, I \notin \{\{1,\ldots,p\}, \emptyset\}} \left\{ M_I(\lambda, \mu, \mu) \,/\, (\lambda, \mu) \in (0, +\infty)^2 \right\} \cup \mathcal{M}_{\text{similar}}$$
$$\text{and} \quad \mathcal{M}_{\text{interval}} := \bigcup_{1 \le k \le p-1} \left\{ M_I(\lambda, \mu, \mu) \,/\, (\lambda, \mu) \in (0, +\infty)^2, I = \{1, \ldots, k\} \right\} \cup \mathcal{M}_{\text{similar}} \quad .$$

Remark 35 *The set $\mathcal{M}_{\text{clus}}$ contains $2^p - 1$ models, a case we will denote by "clustering". The other set, $\mathcal{M}_{\text{interval}}$, only has p models, and is adapted to the structure of the Experiment D. We call this setting "segmentation into intervals".*

6.3 Estimators

In Experiments A–C, we consider four estimators obtained by combining two collections $\mathcal{M}$ of matrices with two formulas for Σ which are plugged into the penalty (7) (that is, either Σ known or estimated by $\widehat{\Sigma}$):

$$\forall \alpha \in \{\text{similar}, \text{ind}\}, \forall S \in \left\{ \Sigma, \widehat{\Sigma}_{\text{HM}} \right\}, \quad \widehat{f}_{\alpha,S} := \widehat{f}_{\widehat{M}_{\alpha,S}} = A_{\widehat{M}_{\alpha,S}} y$$
$$\text{where} \quad \widehat{M}_{\alpha,S} \in \operatorname*{argmin}_{M \in \mathcal{M}_\alpha} \left\{ \frac{1}{np} \left\| y - \widehat{f}_M \right\|_2^2 + \frac{2}{np} \operatorname{tr}\left(A_M \cdot (S \otimes I_n) \right) \right\}$$

and $\widehat{\Sigma}_{\text{HM}}$ is defined in Section 5.2. As detailed in Examples 3–4, $\widehat{f}_{\text{ind},\widehat{\Sigma}_{\text{HM}}}$ and $\widehat{f}_{\text{ind},\Sigma}$ are concatenations of single-task estimators, whereas $\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}}$ and $\widehat{f}_{\text{similar},\Sigma}$ should take advantage of a setting

where the functions f^j are close in $\mathcal{F}$ thanks to the regularization term $\sum_{j,k} \|f^j - f^k\|_{\mathcal{F}}^2$. In Experiment D we consider the following three estimators, that depend on the choice of the collection $\mathcal{M}$:

$$\forall \beta \in \{\text{clus}, \text{interval}, \text{ind}\}\,, \quad \widehat{f}_\beta := \widehat{f}_{\widehat{M}_\beta} = A_{\widehat{M}_\beta} y$$
$$\text{where} \quad \widehat{M}_\beta \in \operatorname*{argmin}_{M \in \mathcal{M}_\beta} \left\{ \frac{1}{np} \left\| y - \widehat{f}_M \right\|_2^2 + \frac{2}{np} \operatorname{tr}\left(A_M \cdot (\widehat{\Sigma} \otimes I_n) \right) \right\}$$

and $\widehat{\Sigma}$ is defined by Equation (11).

In Experiment E we consider the estimator $\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}}$. As explained in the following remark the parameters of the former estimator are chosen by optimizing (20), in practice by choosing a grid. We also consider the estimator $\widehat{f}_{\text{similar,CV}}$ where the parameters are selected by performing 5-fold cross-validation on the mentionned grid.

Remark 36 (Optimization of (20)**)** *Thanks to Assumption* (18) *the optimization problem* (20) *can be solved easily. It suffices to diagonalize in a common basis the elements of $\mathcal{M}$ and the problem splits into several multi-task problems, each with one real parameter. The optimization was then done by using a grid on the real parameters, chosen such that the degree of freedom takes all integer values from* 0 *to n.*

Remark 37 (Finding the jump in Algorithm 14) *Algorithm 14 raises the question of how to detect the jump of* $\mathrm{df}(\lambda)$*, which happens around $C = \sigma^2$. We chose to select an estimator $\widehat{C}$ of σ^2 corresponding to the smallest index such that* $\mathrm{df}(\widehat{\lambda}_0(\widehat{C})) < n/2$*. Another approach is to choose the index corresponding to the largest instantaneous jump of* $\mathrm{df}(\widehat{\lambda}_0(C))$ (*which is piece-wise constant and non-increasing*)*. This approach has a major drawback, because it sometimes selects a jump far away from the "real" jump around σ^2, when the real jump consists of several small jumps. Both approaches gave similar results in terms of prediction error, and we chose the first one because of its direct link to the theoretical criterion given in Theorem 15.*

6.4 Results

In each experiment, $N = 1000$ independent samples $y \in \mathbb{R}^{np}$ have been generated. Expectations are estimated thanks to empirical means over the N samples. Error bars correspond to the classical Gaussian 95% confidence interval (that is, empirical standard-deviation over the N samples multiplied by $1.96/\sqrt{N}$). The results of Experiments A–C are reported in Figures 2–8. The results of Experiments C–E are reported in Tables 1–3. The p-values correspond to the classical Gaussian difference test, where the hypotheses tested are of the shape $\mathbb{H}_0 = \{q > 1\}$ against the hypotheses $\mathbb{H}_1 = \{q \leq 1\}$, where the different quantities q are detailed in Tables 2–3.

6.5 Comments

As expected, multi-task learning significantly helps when all f^j are equal, as soon as p is large enough (Figure 1), especially for small n (Figure 6) and large noise-levels (Figure 8 and Table 1). Increasing the number of tasks rapidly reduces the quadratic error with multi-task estimators (Figure 2) contrary to what happens with single-task estimators (Figure 3).

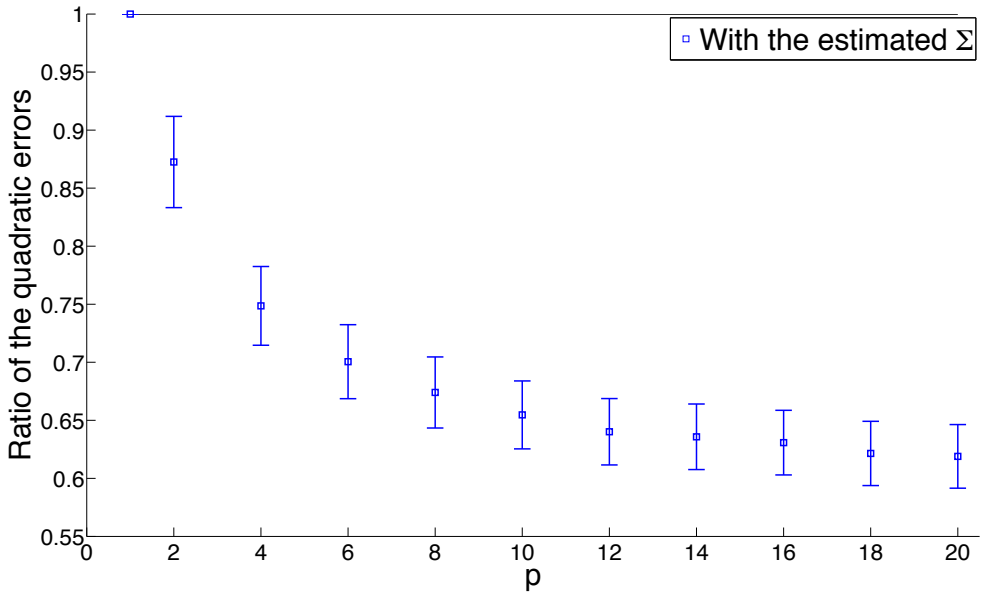


Figure 1: Increasing the number of tasks p (Experiment A), improvement of multi-task compared to single-task: $\mathbb{E}[\|\widehat{f}_{\text{similar},\widehat{\Sigma}} - f\|^2 / \|\widehat{f}_{\text{ind},\widehat{\Sigma}} - f\|^2]$.

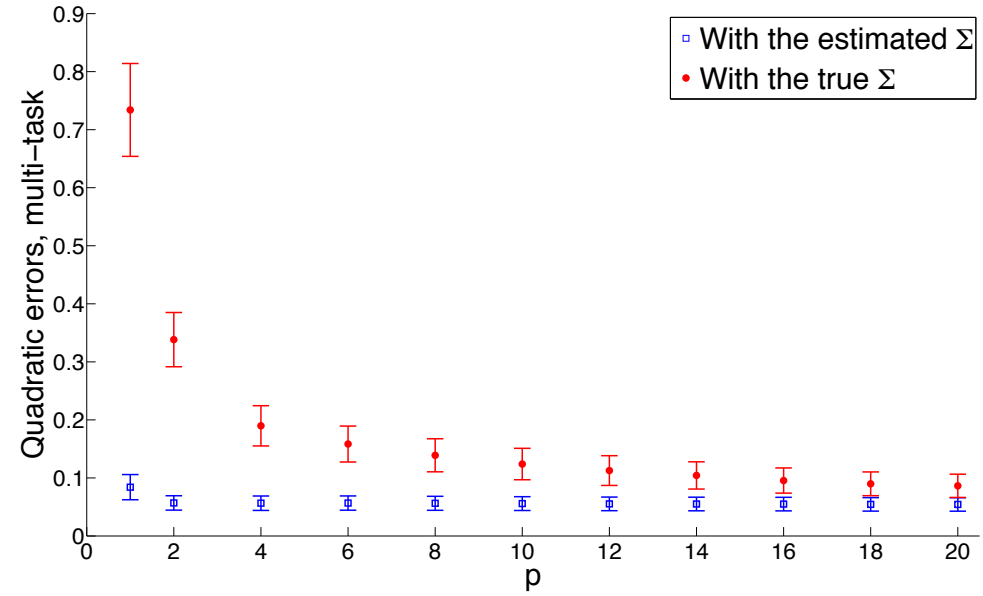


Figure 2: Increasing the number of tasks p (Experiment A), quadratic errors of multi-task estimators $(np)^{-1}\mathbb{E}[\|\widehat{f}_{\text{similar},S} - f\|^2]$. Blue: $S = \widehat{\Sigma}$. Red: $S = \Sigma$.

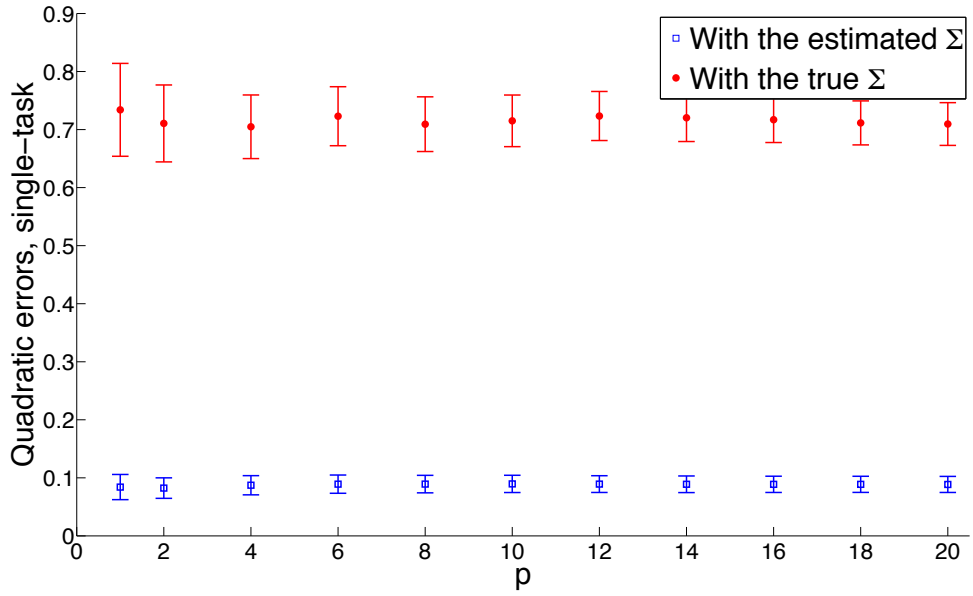


Figure 3: Increasing the number of tasks p (Experiment A), quadratic errors of single-task estimators $(np)^{-1}\mathbb{E}[\|\widehat{f}_{\mathrm{ind},S} - f\|^2]$. Blue: $S = \widehat{\Sigma}$. Red: $S = \Sigma$.

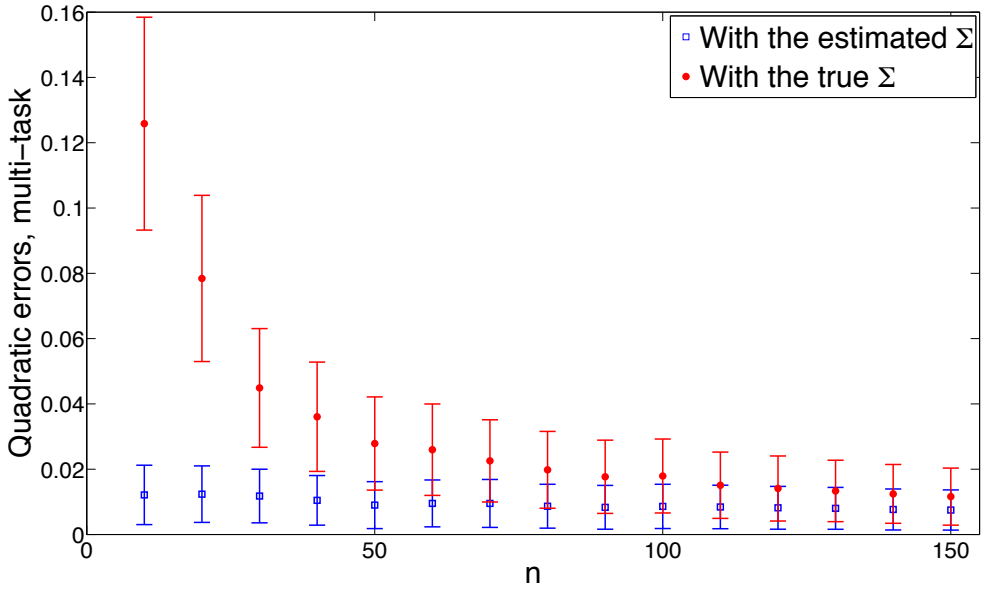


Figure 4: Increasing the sample size n (Experiment B), quadratic errors of multi-task estimators $(np)^{-1}\mathbb{E}[\|\widehat{f}_{\mathrm{similar},S} - f\|^2]$. Blue: $S = \widehat{\Sigma}$. Red: $S = \Sigma$.

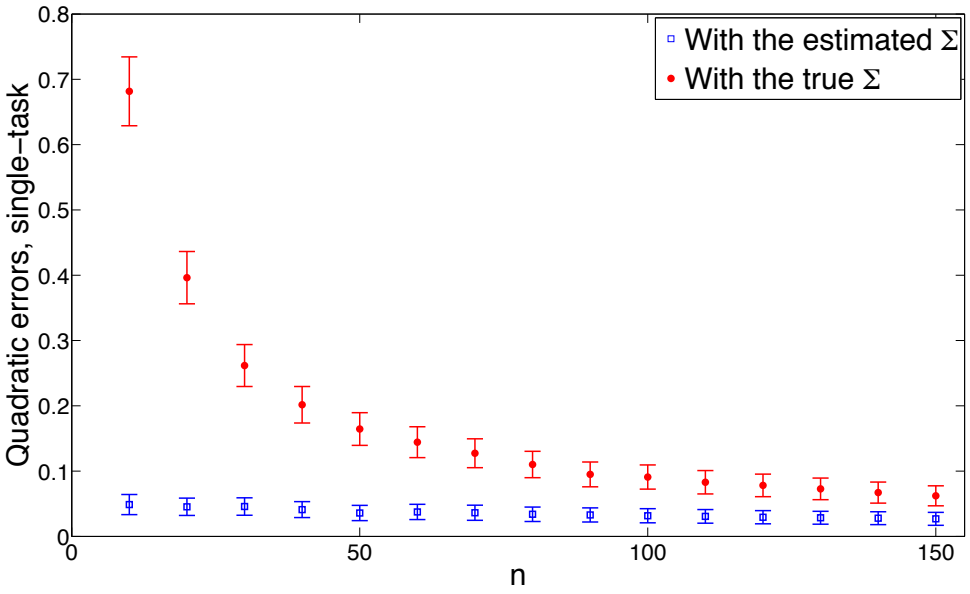


Figure 5: Increasing the sample size n (Experiment B), quadratic errors of single-task estimators $(np)^{-1}\mathbb{E}[\|\widehat{f}_{\mathrm{ind},S} - f\|^2]$. Blue: $S = \widehat{\Sigma}$. Red: $S = \Sigma$.

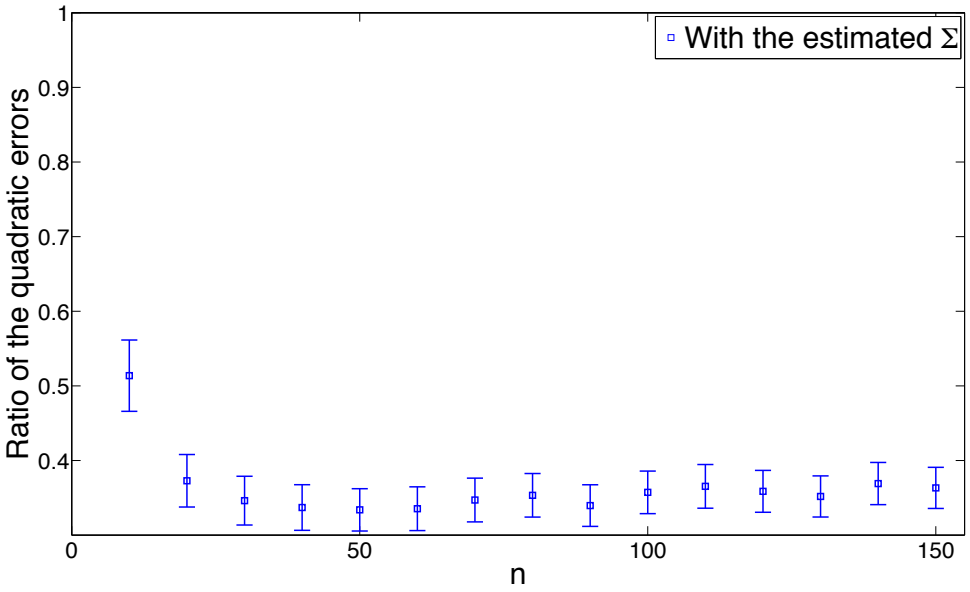


Figure 6: Increasing the sample size n (Experiment B), improvement of multi-task compared to single-task: $\mathbb{E}[\|\widehat{f}_{\mathrm{similar},\widehat{\Sigma}} - f\|^2 / \|\widehat{f}_{\mathrm{ind},\widehat{\Sigma}} - f\|^2]$.

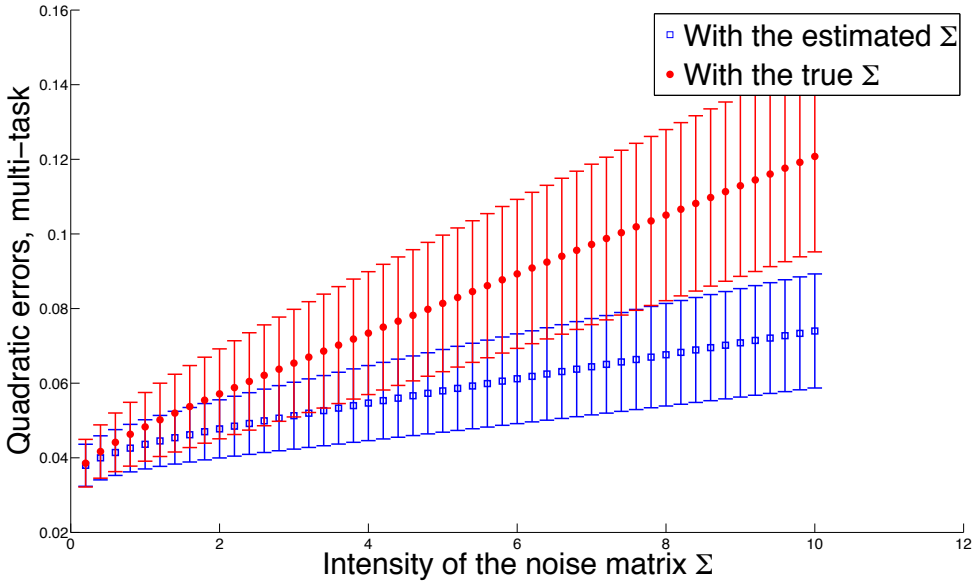


Figure 7: Increasing the signal-to-noise ratio (Experiment C), quadratic errors of multi-task estimators $(np)^{-1}\mathbb{E}[\|\widehat{f}_{\text{similar},S} - f\|^2]$. Blue: $S = \widehat{\Sigma}$. Red: $S = \Sigma$.

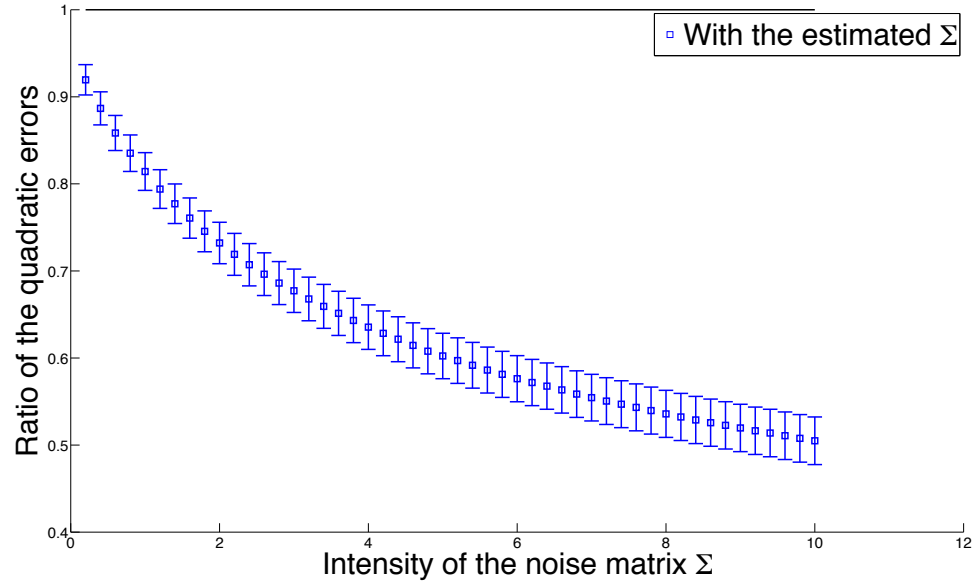


Figure 8: Increasing the signal-to-noise ratio (Experiment C), improvement of multi-task compared to single-task: $\mathbb{E}[\|\widehat{f}_{\text{similar},\widehat{\Sigma}} - f\|^2 / \|\widehat{f}_{\text{ind},\widehat{\Sigma}} - f\|^2]$.

t	0.01	100
$\mathbb{E}[\|\widehat{f}_{\text{similar},\widehat{\Sigma}} - f\|^2 / \|\widehat{f}_{\text{ind},\widehat{\Sigma}} - f\|^2]$	1.80 ± 0.02	0.300 ± 0.003
$\mathbb{E}[\|\widehat{f}_{\text{similar},\widehat{\Sigma}} - f\|^2]$	$(2.27 \pm 0.38) \times 10^{-2}$	0.357 ± 0.048
$\mathbb{E}[\|\widehat{f}_{\text{similar},\Sigma} - f\|^2]$	$(1.20 \pm 0.28) \times 10^{-2}$	0.823 ± 0.080
$\mathbb{E}[\|\widehat{f}_{\text{ind},\widehat{\Sigma}} - f\|^2]$	$(1.26 \pm 0.26) \times 10^{-2}$	1.51 ± 0.07
$\mathbb{E}[\|\widehat{f}_{\text{ind},\Sigma} - f\|^2]$	$(1.20 \pm 0.24) \times 10^{-2}$	4.47 ± 0.13

Table 1: Results of Experiment C for the extreme values of t.

q	$\mathbb{E}[q]$	Std$[q]$	p-value for $\mathbb{H}_0 = \{q > 1\}$
$\|\widehat{f}_{\text{clus}} - f\|^2 / \|\widehat{f}_{\text{ind}} - f\|^2$	0.668	0.294	$< 10^{-15}$
$\|\widehat{f}_{\text{interval}} - f\|^2 / \|\widehat{f}_{\text{ind}} - f\|^2$	0.660	0.270	$< 10^{-15}$
$\|\widehat{f}_{\text{interval}} - f\|^2 / \|\widehat{f}_{\text{clus}} - f\|^2$	1.00	0.165	0.50

Table 2: Clustering and segmentation (Experiment D).

q	n	$\mathbb{E}[q]$	Std$[q]$	p-value for $\mathbb{H}_0 = \{q > 1\}$
$\|\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}} - f\|^2 / \|\widehat{f}_{\text{similar,CV}} - f\|^2$	10	0.35	0.46	$< 10^{-15}$
$\|\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}} - f\|^2 / \|\widehat{f}_{\text{similar,CV}} - f\|^2$	50	0.56	0.42	$< 10^{-15}$
$\|\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}} - f\|^2 / \|\widehat{f}_{\text{similar,CV}} - f\|^2$	100	0.71	0.34	$< 10^{-15}$
$\|\widehat{f}_{\text{similar},\widehat{\Sigma}_{\text{HM}}} - f\|^2 / \|\widehat{f}_{\text{similar,CV}} - f\|^2$	250	0.87	0.19	$< 10^{-15}$

Table 3: Comparison of our method to 5-fold cross-validation (Experiment E).

A noticeable phenomenon also occurs in Figure 2 and even more in Figure 3: the estimator $\widehat{f}_{\text{ind},\Sigma}$ (that is, obtained knowing the true covariance matrix Σ) is less efficient than $\widehat{f}_{\text{ind},\widehat{\Sigma}}$ where the covariance matrix is estimated. It corresponds to the combination of two facts: (i) multiplying the ideal penalty by a small factor $1 < C_n < 1 + o(1)$ is known to often improve performances in practice when the sample size is small (see Section 6.3.2 of Arlot, 2009), and (ii) minimal penalty algorithms like Algorithm 14 are conjectured to overpenalize slightly when n is small or the noise-level is large (Lerasle, 2011) (as confirmed by Figure 7). Interestingly, this phenomenon is stronger for single-task estimators (differences are smaller in Figure 2) and disappears when n is large enough (Figure 5), which is consistent with the heuristic motivating multi-task learning: "increasing the number of tasks p amounts to increase the sample size".

Figures 4 and 5 show that our procedure works well with small n, and that increasing n does not seem to significantly improve the performance of our estimators, except in the single-task setting with Σ known, where the over-penalization phenomenon discussed above disappears.

Table 2 shows that using the multitask procedure improves the estimation accuracy, both in the clustering setting and in the segmentation setting. The last line of Table 2 does not show that the clustering setting improves over the "segmentation into intervals" one, which was awaited if a model close to the oracle is selected in both cases.

Table 3 finally shows that our parameter tuning procedure outperforms 5-fold cross-validation.

7. Conclusion and Future Work

This paper shows that taking into account the unknown similarity between p regression tasks can be done optimally (Theorem 26). The crucial point is to estimate the $p \times p$ covariance matrix Σ of the noise (covariance between tasks), in order to learn the task similarity matrix M. Our main contributions are twofold. First, an estimator of Σ is defined in Section 4, where non-asymptotic bounds on its error are provided under mild assumptions on the mean of the sample (Theorem 20). Second, we show an oracle inequality (Theorem 26), more particularly with a simplified estimation of Σ and increased performances when the matrices of $\mathcal{M}$ are jointly diagonalizable (which often corresponds to cases where we have a prior knowledge of what the relations between the tasks would be). We do plan to expand our results to larger sets $\mathcal{M}$, which may require new concentration inequalities and new optimization algorithms.

Simulation experiments show that our algorithm works with reasonable sample sizes, and that our multi-task estimator often performs much better than its single-task counterpart. Up to the best of our knowledge, a theoretical proof of this point remains an open problem that we intend to investigate in a future work.

Acknowledgments

This paper was supported by grants from the Agence Nationale de la Recherche (DETECT project, reference ANR-09-JCJC-0027-01) and from the European Research Council (SIERRA Project ERC-239993).

We give in Appendix the proofs of the different results stated in Sections 2, 4 and 5. The proofs of our main results are contained in Sections E and F.

Appendix A. Proof of Proposition 8

Proof It is sufficient to show that $\langle \cdot, \cdot \rangle_{\mathcal{G}}$ is positive-definite on $\mathcal{G}$. Take $g \in \mathcal{G}$ and $S = (S_{i,j})_{1 \leq i \leq j \leq p}$ the symmetric postive-definite matrix of size p verifying $S^2 = M$, and denote $T = S^{-1} = (T_{i,j})_{1 \leq i,j \leq p}$.

Let f be the element of $\mathcal{G}$ defined by $\forall i \in \{1 \dots p\},\ g(\cdot,i) = \sum_{k=1}^{n} T_{i,k} f(\cdot,k)$. We then have:

$$\begin{aligned}
\langle g,g\rangle_{\mathcal{G}} &= \sum_{i=1}^{p}\sum_{j=1}^{p} M_{i,j}\langle g(\cdot,i), g(\cdot,j)\rangle_{\mathcal{F}} \\
&= \sum_{i=1}^{p}\sum_{j=1}^{p}\sum_{k=1}^{p}\sum_{l=1}^{p} M_{i,j}T_{i,k}T_{j,l}\langle f(\cdot,k), f(\cdot,l)\rangle_{\mathcal{F}} \\
&= \sum_{j=1}^{p}\sum_{k=1}^{p}\sum_{l=1}^{p} T_{l,j}\langle f(\cdot,k), f(\cdot,l)\rangle_{\mathcal{F}} \sum_{i=1}^{p} M_{j,i}T_{i,k} \\
&= \sum_{j=1}^{p}\sum_{k=1}^{p}\sum_{l=1}^{p} T_{l,j}\langle f(\cdot,k), f(\cdot,l)\rangle_{\mathcal{F}} (M\cdot T)_{j,k} \\
&= \sum_{k=1}^{p}\sum_{l=1}^{p} T_{l,j}\langle f(\cdot,k), f(\cdot,l)\rangle_{\mathcal{F}} \sum_{j=1}^{p} T_{l,j}(M\cdot T)_{j,k} \\
&= \sum_{k=1}^{p}\sum_{l=1}^{p} \langle f(\cdot,k), f(\cdot,l)\rangle_{\mathcal{F}} (T\cdot M\cdot T)_{k,l} \\
&= \sum_{k=1}^{p} \|f(\cdot,k)\|_{\mathcal{F}}^2 .
\end{aligned}$$

This shows that $\langle g,g\rangle_{\mathcal{G}} \geq 0$ and that $\langle g,g\rangle_{\mathcal{G}} = 0 \Rightarrow f = 0 \Rightarrow g = 0$. ■

Appendix B. Proof of Corollary 9

Proof If $(x,j) \in \mathcal{X} \times \{1,\dots,p\}$, the application $(f^1,\dots,f^p) \mapsto f^j(x)$ is clearly continuous. We now show that $(\mathcal{G}, \langle\cdot,\cdot\rangle_{\mathcal{G}})$ is complete. If $(g_n)_{n\in\mathbb{N}}$ is a Cauchy sequence of $\mathcal{G}$ and if we define, as in Section A, the functions f_n by $\forall n \in \mathbb{N},\ \forall i \in \{1 \dots p\},\ g_n(\cdot,i) = \sum_{k=1}^{p} T_{i,k} f_n(\cdot,k)$. The same computations show that $(f_n(\cdot,i))_{n\in\mathbb{N}}$ are Cauchy sequences of $\mathcal{F}$, and thus converge. So the sequence $(f_n)_{n\in\mathbb{N}}$ converges in $\mathcal{G}$, and $(g_n)_{n\in\mathbb{N}}$ does likewise. ■

Appendix C. Proof of Proposition 11

Proof We define

$$\widetilde{\Phi}(x,j) = M^{-1} \cdot \begin{pmatrix} \delta_{1,j}\Phi(x) \\ \vdots \\ \delta_{p,j}\Phi(x) \end{pmatrix} ,$$

with $\delta_{i,j} = \mathbf{1}_{i=j}$ being the Kronecker symbol, that is, $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. We now show that $\widetilde{\Phi}$ is the feature function of the RKHS. For $g \in \mathcal{G}$ and $(x,l) \in \mathcal{X} \times \{1,\dots,p\}$, we have:

$$\begin{aligned}
\langle g, \widetilde{\Phi}(x,l)\rangle_{\mathcal{G}} &= \sum_{j=1}^{p}\sum_{i=1}^{p} M_{j,i}\langle g(\cdot,j), \widetilde{\Phi}(x,l)^i\rangle_{\mathcal{F}} \\
&= \sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{m=1}^{p} M_{j,i}M^{-1}_{i,m}\delta_{m,l}\langle g(\cdot,j), \Phi(x)\rangle_{\mathcal{F}} \\
&= \sum_{j=1}^{p}\sum_{m=1}^{p} \left(M\cdot M^{-1}\right)_{j,m}\delta_{m,l}\, g(x,j) \\
&= \sum_{j=1}^{p} \delta_{j,l}\, g(x,j) = g(x,l) \quad .
\end{aligned}$$

Thus we can write:

$$\begin{aligned}
\widetilde{k}((x,i),(y,j)) &= \langle \widetilde{\Phi}(x,i), \widetilde{\Phi}(y,j)\rangle_{\mathcal{G}} \\
&= \sum_{h=1}^{p}\sum_{h'=1}^{p} M_{h,h'}\langle M^{-1}_{h,i}\Phi(x), M^{-1}_{h',j}\Phi(y)\rangle_{\mathcal{F}} \\
&= \sum_{h=1}^{p}\sum_{h'=1}^{p} M_{h,h'}M^{-1}_{h,i}M^{-1}_{h',j}K(x,y) \\
&= \sum_{h=1}^{p} M^{-1}_{h,i}\left(M\cdot M^{-1}\right)_{h,j}K(x,y) \\
&= \sum_{h=1}^{p} M^{-1}_{h,i}\delta_{h,j}K(x,y) = M^{-1}_{i,j}K(x,y) \quad .
\end{aligned}$$

■

Appendix D. Computation of the Quadratic Risk in Example 12

We consider here that $f^1 = \cdots = f^p$. We use the set $\mathcal{M}_{\text{similar}}$:

$$\mathcal{M}_{\text{similar}} := \left\{ M_{\text{similar}}(\lambda,\mu) = (\lambda + p\mu)I_p - \frac{\mu}{p}11^\top \,/\, (\lambda,\mu) \in (0,+\infty)^2 \right\}$$

Using the estimator $\widehat{f}_M = A_M y$ we can then compute the quadratic risk using the bias-variance decomposition given in Equation (36):

$$\mathbb{E}\left[\left\|\widehat{f}_M - f\right\|_2^2\right] = \|(A_M - I_{np})f\|_2^2 + \operatorname{tr}(A_M^\top A_M \cdot (\Sigma \otimes I_n)) \quad .$$

Les us denote by $(e_1,\ldots,e_p)$ the canonical basis of $\mathbb{R}^p$. The eigenspaces of $p^{-1}11^\top$ are:

- span $\{e_1 + \cdots + e_p\}$ corresponding to eigenvalue p,

- span $\{e_2 - e_1, \dots, e_p - e_1\}$ corresponding to eigenvalue 0.

Thus, with $\widetilde{\mu} = \lambda + p\mu$ we can diagonalize in an orthonormal basis any matrix $M_{\lambda,\mu} \in \mathcal{M}$ as $M = P^\top D_{\lambda,\widetilde{\mu}} P$, with $D = D_{\lambda,\widetilde{\mu}} = \mathrm{Diag}\{\lambda, \widetilde{\mu}, \dots, \widetilde{\mu}\}$. Les us also diagonalise in an orthonormal basis K: $K = Q^\top \Delta Q, \Delta = \mathrm{Diag}\{\mu_1, \dots, \mu_n\}$. Thus we can write (see Properties 38 and 39 for basic properties of the Kronecker product):

$$A_M = A_{M_{\lambda,\mu}} = (P^\top \otimes Q^\top) \left[(D^{-1} \otimes \Delta) \left((D^{-1} \otimes \Delta) + npI_{np} \right)^{-1} \right] (P \otimes Q) \ .$$

We can then note that $(D^{-1} \otimes \Delta) \left((D^{-1} \otimes \Delta) + npI_{np} \right)^{-1}$ is a diagonal matrix, whose diagonal entry of index $(j-1)n+i$ ($i \in \{1, \dots, n\}$, $j \in \{1, \dots, p\}$) is

$$\begin{cases} \frac{\mu_i}{\mu_i + np\lambda} \text{ if } j = 1 \ , \\ \frac{\mu_i}{\mu_i + np\widetilde{\mu}} \text{ if } j > 1 \ . \end{cases}$$

We can now compute both bias and variance.

Bias: We can first remark that $(P^\top \otimes Q^\top) = (P \otimes Q)^\top$ is an orthogonal matrix and that $P \times 1 = (1, 0, \dots, 0)^\top$. Thus, as in this setting $f^1 = \cdots = f^p$, we have $f = 1 \otimes (f^1(X_1), \dots, f^1(X_n))^\top$ and $(P^\top \otimes Q^\top) f = (1, 0, \dots, 0)^\top \otimes Q(f^1(X_1), \dots, f^1(X_n))^\top$. To keep notations simple we note $Q(f^1(X_1), \dots, f^1(X_n))^\top := (g_1, \dots, g_n)^\top$. Thus

$$\begin{aligned} \|(A_M - I_{np}) f\|_2^2 &= \|(P \otimes Q)^\top \left[(D^{-1} \otimes K) \left((D^{-1} \otimes K) + npI_{np} \right)^{-1} - I_{np} \right] (P \otimes Q) f\|_2^2 \\ &= \| \left[(D^{-1} \otimes \Delta) \left((D^{-1} \otimes \Delta) + npI_{np} \right)^{-1} - I_{np} \right] \\ &\qquad \times (1, 0, \dots, 0)^\top \otimes (g_1, \dots, g_n)^\top \|_2^2 \ . \end{aligned}$$

As only the first n terms of $(P \otimes Q) f$ are non-zero we can finally write

$$\|(A_M - I_{np}) f\|_2^2 = \sum_{i=1}^{n} \left(\frac{np\lambda}{\mu_i + np\lambda} \right)^2 g_i^2 \ .$$

Variance: First note that

$$(P \otimes Q)(\Sigma \otimes I_n)(P \otimes Q)^\top = (P \Sigma P^\top \otimes I_n) \ .$$

We can also note that $\widetilde{\Sigma} := P \Sigma P^\top$ is a symmetric positive definite matrix, with positive diagonal coefficients. Thus we can finally write

$$\begin{aligned} \mathrm{tr}(A_M^\top A_M \cdot (\Sigma \otimes I_n)) &= \mathrm{tr} \Bigg(P \otimes Q)^\top \left[(D^{-1} \otimes \Delta) \left((D^{-1} \otimes \Delta) + npI_{np} \right)^{-1} \right]^2 \\ &\qquad \times (P \otimes Q)(\Sigma \otimes I_n) \Bigg) \\ &= \mathrm{tr} \Bigg(\left[(D^{-1} \otimes \Delta) \left((D^{-1} \otimes \Delta) + npI_{np} \right)^{-1} \right]^2 \\ &\qquad \times (P \otimes Q)(\Sigma \otimes I_n)(P \otimes Q)^\top \Bigg) \\ &= \sum_{i=1}^{n} \left[\left(\frac{\mu_i}{\mu_i + np\lambda} \right)^2 \widetilde{\Sigma}_{1,1} + \left(\frac{\mu_i}{\mu_i + np\widetilde{\mu}} \right)^2 \sum_{j=2}^{p} \widetilde{\Sigma}_{j,j} \right] \ . \end{aligned}$$

As noted at the end of Example 12 this leads to an oracle which has all its p functions equal.

D.1 Proof of Equation (19) in Section 5.2

Let $M \in \mathcal{S}_p^{++}(\mathbb{R})$, $P \in O_p(\mathbb{R})$ such that $M = P^\top \operatorname{Diag}(d_1,\dots,d_p)P$ and $\widetilde{\Sigma} = P\Sigma P^\top$. We recall that $A_\lambda = K(K+n\lambda I_n)^{-1}$. The computations detailed above also show that the ideal penalty introduced in Equation (7) can be written as

$$\operatorname{pen}_{\mathrm{id}}(M) = \frac{2\operatorname{tr}\left(A_M \cdot (\Sigma \otimes I_n)\right)}{np} = \frac{2}{np}\left(\sum_{j=1}^{p} \operatorname{tr}(A_{pd_j})\widetilde{\Sigma}_{j,j}\right) \ .$$

Appendix E. Proof of Theorem 20

Theorem 20 is proved in this section, after stating some classical linear algebra results (Section E.1).

E.1 Some Useful Tools

We now give two properties of the Kronecker product, and then introduce a useful norm on $\mathcal{S}_p(\mathbb{R})$, upon which we give several properties. Those are the tools needed to prove Theorem 20.

Property 38 *The Kronecker product is bilinear, associative and for every matrices A,B,C,D such that the dimensions fit, $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$.*

Property 39 *Let $A \in \mathcal{M}_n(\mathbb{R})$, $B \in \mathcal{M}_B(\mathbb{R})$, $(A\otimes B)^\top = (A^\top \otimes B^\top)$.*

Definition 40 *We now introduce the norm $|||\cdot|||$ on $\mathcal{S}_p(\mathbb{R})$, which is the modulus of the eigenvalue of largest magnitude, and can be defined by*

$$|||S||| := \sup_{z\in\mathbb{R}^p, \|z\|_2=1} \left|z^\top S z\right| \ .$$

This norm has several interesting properties, some of which we will use are stated below.

Property 41 *The norm $|||\cdot|||$ is a matricial norm: $\forall (A,B) \in \mathcal{S}_p(\mathbb{R})^2,\ |||AB||| \le |||A|||\,|||B|||$.*

We will use the following result, which is a consequence of the preceding Property.

$$\forall S \in \mathcal{S}_p(\mathbb{R}),\ \forall T \in \mathcal{S}_p^{++}(\mathbb{R}),\ |||T^{-\frac{1}{2}} S T^{-\frac{1}{2}}||| \le |||S|||\,|||T^{-1}||| \ .$$

We also have:

Proposition 42

$$\forall \Sigma \in \mathcal{S}_p(\mathbb{R}),\ |||\Sigma \otimes I_n||| = |||\Sigma||| \ .$$

Proof We can diagonalize Σ in an orthonormal basis: $\exists U \in O_n(\mathbb{R}),\ \exists D = \operatorname{Diag}(\mu_1,\dots,\mu_p),\ \Sigma = U^\top D U$. We then have, using the properties of the Kronecker product:

$$\begin{aligned}\Sigma \otimes I_n &= (U^\top \otimes I_n)(D\otimes I_n)(U\otimes I_n) \\ &= (U\otimes I_n)^\top (D\otimes I_n)(U\otimes I_n) \ .\end{aligned}$$

We just have to notice that $U \otimes I_n \in O_{np}(\mathbb{R})$ and that:

$$D \otimes I_n = \mathrm{Diag}(\underbrace{\mu_1, \ldots, \mu_1}_{n \text{ times}}, \ldots, \underbrace{\mu_p, \ldots, \mu_p}_{n \text{ times}}) \ .$$

■

This norm can also be written in other forms:

Property 43 *If $M \in \mathcal{M}_n(\mathbb{R})$, the operator norm $\|M\|_2 := \sup_{t \in \mathbb{R}^n \setminus \{0\}} \left\{ \frac{\|Mt\|_2}{\|t\|_2} \right\}$ is equal to the greatest singular value of M: $\sqrt{\rho(M^\top M)}$. Henceforth, if S is symmetric, we have $|||S||| = \|S\|_2$*

E.2 The Proof

We now give a proof of Theorem 20, using Lemmas 46, 48 and 49, which are stated and proved in Section E.3. The outline of the proof is the following:

1. Apply Theorem 15 to problem (10) for every $z \in \mathcal{Z}$ in order to
2. control $\|s - \zeta\|_\infty$ with a large probability, where $s, \zeta \in \mathbb{R}^{p(p+1)/2}$ are defined by

$$s := (\Sigma_{1,1}, \ldots, \Sigma_{p,p}, \Sigma_{1,1} + \Sigma_{2,2} + 2\Sigma_{1,2}, \ldots, \Sigma_{i,i} + \Sigma_{j,j} + 2\Sigma_{i,j}, \ldots)$$
$$\text{and} \quad \zeta := (a(e_1), \ldots, a(e_p), a(e_1+e_2), \ldots, a(e_1+e_p), a(e_2+e_3), \ldots, a(e_{p-1}+e_p)) \ .$$

3. Deduce that $\widehat{\Sigma} = J(\zeta)$ is close to $\Sigma = J(s)$ by controlling the Lipschitz norm of J.

Proof *1. Apply Theorem 15:* We start by noticing that Assumption (13) actually holds true with all $\lambda_{0,j}$ equal. Indeed, let $(\lambda_{0,j})_{1 \le j \le p}$ be given by Assumption (13) and define $\lambda_0 := \min_{j=1,\ldots,p} \lambda_{0,j}$. Then, $\lambda_0 \in (0, +\infty)$ and $\mathrm{df}(\lambda_0)$ since all $\lambda_{0,j}$ satisfy these two conditions. For the last condition, remark that for every $j \in \{1, \ldots, p\}$, $\lambda_0 \le \lambda_{0,j}$ and $\lambda \mapsto \|(A_\lambda - I)F_{e_j}\|_2^2$ is a nonincreasing function (as noticed in Arlot and Bach, 2011 for instance), so that

$$\frac{1}{n} \left\| (A_{\lambda_0} - I_n) F_{e_j} \right\|_2^2 \le \frac{1}{n} \left\| (A_{\lambda_{0,j}} - I_n) F_{e_j} \right\|_2^2 \le \Sigma_{j,j} \sqrt{\frac{\ln(n)}{n}} \ . \tag{23}$$

In particular, Equation (8) holds with $d_n = 1$ for problem (10) whatever $z \in \{e_1, \ldots, e_p\}$.

Let us now consider the case $z = e_i + e_j$ with $i \ne j \in \{1, \ldots, p\}$. Using Equation (23) and that $F_{e_i+e_j} = F_{e_i} + F_{e_j}$, we have

$$\left\| (B_{\lambda_0} - I_n) F_{e_i+e_j} \right\|_2^2 \le \left\| (B_{\lambda_0} - I_n) F_{e_i} \right\|_2^2 + \left\| (B_{\lambda_0} - I_n) F_{e_j} \right\|_2^2 + 2\langle (B_{\lambda_0} - I_n) F_{e_i}, (B_{\lambda_0} - I_n) F_{e_j} \rangle \ .$$

The last term is bounded as follows:

$$\begin{aligned}
2\langle (B_{\lambda_0} - I_n) F_{e_i}, (B_{\lambda_0} - I_n) F_{e_j} \rangle &\le 2 \|(B_{\lambda_0} - I_n) F_{e_i}\| \cdot \|(B_{\lambda_0} - I_n) F_{e_j}\| \\
&\le 2\sqrt{n \ln(n)} \sqrt{\Sigma_{i,i} \Sigma_{j,j}} \\
&\le \sqrt{n \ln(n)} (\Sigma_{i,i} + \Sigma_{j,j}) \\
&\le (1 + c(\Sigma)) \sqrt{n \ln(n)} (\Sigma_{i,i} + \Sigma_{j,j} + 2\Sigma_{i,j}) \\
&= (1 + c(\Sigma)) \sqrt{n \ln(n)} \sigma_{e_i+e_j}^2 \ ,
\end{aligned}$$

because Lemma 46 shows

$$2(\Sigma_{i,i}+\Sigma_{j,j}) \leq (1+c(\Sigma))(\Sigma_{i,i}+\Sigma_{j,j}+2\Sigma_{i,j}) \ .$$

Therefore, Equation (8) holds with $d_n = 1+c(\Sigma)$ for problem (10) whatever $z \in \mathcal{Z}$.

2. Control $\|s-\zeta\|_\infty$: Let us define

$$\eta_1 := \beta(2+\delta)(1+c(\Sigma))\sqrt{\frac{\ln(n)}{n}} \ .$$

By Theorem 15, for every $z \in \mathcal{Z}$, an event Ω_z of probability greater than $1-n^{-\delta}$ exists on which, if $n \geq n_0(\delta)$,

$$(1-\eta_1)\sigma_z^2 \leq a(z) \leq (1+\eta_1)\sigma_z^2 \ .$$

So, on $\Omega := \bigcap_{z\in Z}\Omega_z$,

$$\|\zeta - s\|_\infty \leq \eta_1 \|s\|_\infty \ , \tag{24}$$

and $\mathbb{P}(\Omega) \geq 1 - p(p+1)/2 \times n^{-\delta}$ by the union bound. Let

$$\|\Sigma\|_\infty := \sup_{i,j} |\Sigma_{i,j}| \quad \text{and} \quad C_1(p) := \sup_{\Sigma \in \mathcal{S}_p(\mathbb{R})} \left\{ \frac{\|\Sigma\|_\infty}{|||\Sigma|||} \right\} \ .$$

Since $\|s\|_\infty \leq 4\|\Sigma\|_\infty$ and $C_1(p) = 1$ by Lemma 48, Equation (24) implies that on Ω,

$$\|\zeta - s\|_\infty \leq 4\eta_1 \|\Sigma\|_\infty \leq 4\eta_1 |||\Sigma||| \ . \tag{25}$$

3. Conclusion of the proof: Let

$$C_2(p) := \sup_{\zeta \in \mathbb{R}^{p(p+1)/2}} \left\{ \frac{|||J(\zeta)|||}{\|\zeta\|_\infty} \right\} \ .$$

By Lemma 49, $C_2(p) \leq \frac{3}{2}p$. By Equation (25), on Ω,

$$|||\widehat{\Sigma} - \Sigma||| = |||J(\zeta) - J(s)||| \leq C_2(p)\|\zeta - s\|_\infty \leq 4\eta_1 C_2(p) |||\Sigma||| \ . \tag{26}$$

Since

$$|||\Sigma^{-\frac{1}{2}}\widehat{\Sigma}\Sigma^{-\frac{1}{2}} - I_p||| = |||\Sigma^{-\frac{1}{2}}(\Sigma - \widehat{\Sigma})\Sigma^{-\frac{1}{2}}||| \leq |||\Sigma^{-1}|||\,|||\Sigma - \widehat{\Sigma}||| \ ,$$

and $|||\Sigma|||\,|||\Sigma^{-1}||| = c(\Sigma)$, Equation (26) implies that on Ω,

$$|||\Sigma^{-\frac{1}{2}}\widehat{\Sigma}\Sigma^{-\frac{1}{2}} - I_p||| \leq 4\eta_1 C_2(p) |||\Sigma|||\,|||\Sigma^{-1}||| = 4\eta_1 C_2(p) c(\Sigma) \leq 6\eta_1 p c(\Sigma) \ .$$

To conclude, Equation (14) holds on Ω with

$$\eta = 6pc(\Sigma)\beta(2+\delta)(1+c(\Sigma))\sqrt{\frac{\ln(n)}{n}} \leq L_1(2+\delta)p\sqrt{\frac{\ln(n)}{n}}c(\Sigma)^2 \tag{27}$$

for some numerical constant L_1. ■

Remark 44 *As stated in Arlot and Bach (2011), we need $\sqrt{n_0(\delta)/\ln(n_0(\delta))} \geq 504$ and $\sqrt{n_0(\delta)}/\ln(n_0(\delta)) \geq 24(290+\delta)$.*

Remark 45 *To ensure that the estimated matrix $\widehat{\Sigma}$ is positive-definite we need that $\eta < 1$, that is,*

$$\sqrt{\frac{n}{\ln(n)}} > 6\beta(2+\delta)pc(\Sigma)\,(1+c(\Sigma)) \ .$$

E.3 Useful Lemmas

Lemma 46 *Let $p \geq 1$, $\Sigma \in \mathcal{S}_p^{++}(\mathbb{R})$ and $c(\Sigma)$ its condition number. Then,*

$$\forall 1 \leq i < j \leq p\,, \quad \Sigma_{i,j} \geq -\frac{c(\Sigma)-1}{c(\Sigma)+1}\frac{\Sigma_{i,i}+\Sigma_{j,j}}{2} \quad , \tag{28}$$

Remark 47 *The proof of Lemma 46 shows the constant $\frac{c(\Sigma)-1}{c(\Sigma)+1}$ cannot be improved without additional assumptions on Σ.*

Proof It suffices to show the result when $p = 2$. Indeed, (28) only involves 2×2 submatrices $\widetilde{\Sigma}(i,j) \in \mathcal{S}_2^{++}(\mathbb{R})$ for which

$$1 \leq c(\widetilde{\Sigma}) \leq c\,(\Sigma) \quad \text{hence} \quad 0 \leq \frac{c(\widetilde{\Sigma})-1}{c(\widetilde{\Sigma})+1} \leq \frac{c(\Sigma)-1}{c(\Sigma)+1} \quad .$$

So, some $\theta \in \mathbb{R}$ exists such that $\Sigma = |||\Sigma||| R_\theta^\top D R_\theta$ where

$$R_\theta := \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \qquad D = \begin{pmatrix} 1 & 0 \\ 0 & \lambda \end{pmatrix} \quad \text{and} \quad \lambda := \frac{1}{c(\Sigma)} \quad .$$

Therefore,

$$\Sigma = |||\Sigma||| \begin{pmatrix} \cos^2(\theta)+\lambda\sin^2(\theta) & \frac{1-\lambda}{2}\sin(2\theta) \\ \frac{1-\lambda}{2}\sin(2\theta) & \lambda\cos^2(\theta)+\sin^2(\theta) \end{pmatrix} \quad .$$

So, Equation (28) is equivalent to

$$\frac{(1-\lambda)\sin(2\theta)}{2} \geq -\frac{1-\lambda}{1+\lambda}\frac{1+\lambda}{2} \quad ,$$

which holds true for every $\theta \in \mathbb{R}$, with equality for $\theta \equiv \pi/2 \ (\text{mod. } \pi)$. ■

Lemma 48 *For every $p \geq 1$, $C_1(p) := \sup_{\Sigma \in \mathcal{S}_p(\mathbb{R})} \frac{\|\Sigma\|_\infty}{|||\Sigma|||} = 1$.*

Proof With $\Sigma = I_p$ we have $\|\Sigma\|_\infty = |||\Sigma||| = 1$, so $C_1(p) \geq 1$.
Let us introduce (i,j) such that $|\Sigma_{i,j}| = \|\Sigma\|_\infty$. We then have, with e_k being the k^{th} vector of the canonical basis of $\mathbb{R}^p$,

$$|\Sigma_{i,j}| = |e_i^\top \Sigma e_j| \leq |e_i^\top \Sigma e_i|^{1/2} |e_j^\top \Sigma e_j|^{1/2} \leq (\|\Sigma\|_2^{1/2})^2 \quad .$$

■

Lemma 49 *For every $p \geq 1$, let $C_2(p) := \sup_{\zeta \in \mathbb{R}^{p(p+1)/2}} \frac{|||J(\zeta)|||}{\|\zeta\|_\infty}$. Then,*

$$\frac{p}{4} \leq C_2(p) \leq \frac{3}{2}p \quad .$$

Proof For the lower bound, we consider

$$\zeta_1 = (\underbrace{1,\dots,1}_{p \text{ times}}, \underbrace{4,\dots,4}_{\frac{p(p-1)}{2} \text{ times}}), \quad \text{then} \quad J(\zeta_1) = \begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix}$$

so that $|||J(\zeta)||| = p$ and $\|\zeta\|_\infty = 4$.

For the upper bound, we have for every $\zeta \in \mathbb{R}^{p(p+1)/2}$ and $z \in \mathbb{R}^p$ such that $\|z\|_2 = 1$

$$z^\top J(\zeta) z = \left| \sum_{1 \leq i,j \leq p} z_i z_j J(\zeta)_{i,j} \right| \leq \sum_{1 \leq i,j \leq p} |z_i| \left|z_j\right| |J(\zeta)| \leq \|J(\zeta)\|_\infty \|z\|_1^2 \ .$$

By definition of J, $\|J(\zeta)\|_\infty \leq 3/2 \|\zeta\|_\infty$. Remarking that $\|z\|_1^2 \leq p \|z\|_2^2$ yields the result. ■

Appendix F. Proof of Theorem 26

The proof of Theorem 26 is similar to the proof of Theorem 3 in Arlot and Bach (2011). We give it here for the sake of completeness. We also show how to adapt its proof to demonstrate Theorem 29. The two main mathematical results used here are Theorem 20 and a gaussian concentration inequality from Arlot and Bach (2011).

F.1 Key Quantities and their Concentration Around their Means

Definition 50 *We introduce, for $S \in \mathcal{S}_p^{++}(\mathbb{R})$,*

$$\widehat{M}_o(S) \in \operatorname*{argmin}_{M \in \mathcal{M}} \left\{ \left\| \widehat{F}_M - Y \right\|_2 + 2 \operatorname{tr}\left(A_M \cdot (S \otimes I_n)\right) \right\} \tag{29}$$

Definition 51 *Let $S \in \mathcal{S}_p(\mathbb{R})$, we note S_+ the symmetric matrix where the eigenvalues of S have been thresholded at 0. That is, if $S = U^\top D U$, with $U \in O_p(\mathbb{R})$ and $D = \operatorname{Diag}(d_1,\dots,d_p)$, then*

$$S_+ := U^\top \operatorname{Diag}\left(\max\{d_1,0\},\dots,\max\{d_n,0\}\right) U \ .$$

Definition 52 *For every $M \in \mathcal{M}$, we define*

$$\begin{aligned}
b(M) &= \|(A_M - I_{np}) f\|_2^2 \ , \\
v_1(M) &= \mathbb{E}\left[\langle \varepsilon, A_M \varepsilon \rangle\right] = \operatorname{tr}(A_M \cdot (\Sigma \otimes I_n)) \ , \\
\delta_1(M) &= \langle \varepsilon, A_M \varepsilon \rangle - \mathbb{E}\left[\langle \varepsilon, A_M \varepsilon \rangle\right] = \langle \varepsilon, A_M \varepsilon \rangle - \operatorname{tr}(A_M \cdot (\Sigma \otimes I_n)) \ , \\
v_2(M) &= \mathbb{E}\left[\|A_M \varepsilon\|_2^2\right] = \operatorname{tr}(A_M^\top A_M \cdot (\Sigma \otimes I_n)) \ , \\
\delta_2(M) &= \|A_M \varepsilon\|_2^2 - \mathbb{E}\left[\|A_M \varepsilon\|_2^2\right] = \|A_M \varepsilon\|_2^2 - \operatorname{tr}(A_M^\top A_M \cdot (\Sigma \otimes I_n)) \ , \\
\delta_3(M) &= 2\langle A_M \varepsilon, (A_M - I_{np}) f \rangle \ , \\
\delta_4(M) &= 2\langle \varepsilon, (I_{np} - A_M) f \rangle \ , \\
\widehat{\Delta}(M) &= -2\delta_1(M) + \delta_4(M) \ .
\end{aligned}$$

Definition 53 *Let $C_A, C_B, C_C, C_D, C_E, C_F$ be fixed nonnegative constants. For every $x \geq 0$ we define the event*

$$\Omega_x = \Omega_x(\mathcal{M}, C_A, C_B, C_C, C_D, C_E, C_F)$$

on which, for every $M \in \mathcal{M}$ and $\theta_1, \theta_2, \theta_3, \theta_4 \in (0,1]$:

$$|\delta_1(M)| \leq \theta_1 \operatorname{tr}\left(A_M^\top A_M \cdot (\Sigma \otimes I_n)\right) + (C_A + C_B \theta_1^{-1}) x |||\Sigma||| \tag{30}$$

$$|\delta_2(M)| \leq \theta_2 \operatorname{tr}\left(A_M^\top A_M \cdot (\Sigma \otimes I_n)\right) + (C_C + C_D \theta_2^{-1}) x |||\Sigma||| \tag{31}$$

$$|\delta_3(M)| \leq \theta_3 \|(I_{np} - A_M) f\|_2^2 + C_E \theta_3^{-1} x |||\Sigma||| \tag{32}$$

$$|\delta_4(M)| \leq \theta_4 \|(I_{np} - A_M) f\|_2^2 + C_F \theta_4^{-1} x |||\Sigma||| \tag{33}$$

Of key interest is the concentration of the empirical processes δ_i, uniformly over $M \in \mathcal{M}$. The following Lemma introduces such a result, when $\mathcal{M}$ contains symmetric matrices parametrized with their eigenvalues (with fixed eigenvectors).

Lemma 54 *Let*

$$C_A = 2,\ C_B = 1,\ C_C = 2,\ C_D = 1,\ C_E = 306.25,\ C_F = 306.25\ .$$

Suppose that (18) *holds. Then* $\mathbb{P}(\Omega_x(\mathcal{M}, C_A, C_B, C_C, C_D, C_E, C_F)) \geq 1 - p e^{1027 + \ln(n)} e^{-x}$. *Suppose that* (15) *holds. Then* $\mathbb{P}(\Omega_x(\mathcal{M}, C_A, C_B, C_C, C_D, C_E, C_F)) \geq 1 - 6p\,\mathrm{card}(\mathcal{M}) e^{-x}$.

.

Proof

First common step. Let $M \in \mathcal{M}$, $P_M \in O_p(\mathbb{R})$ such that $M = P_M^\top D P_M$, with $D = \mathrm{Diag}(d_1, \ldots, d_p)$. We can write:

$$\begin{aligned} A_M = A_{d_1,\ldots,d_p} &= (P_M \otimes I_n)^\top \left[(D^{-1} \otimes K) \left(D^{-1} \otimes K + np I_{np} \right)^{-1} \right] (P_M \otimes I_n) \\ &= Q^\top \widetilde{A}_{d_1,\ldots,d_p} Q\ , \end{aligned}$$

with $Q = P_M \otimes I_n$ and $\widetilde{A}_{d_1,\ldots,d_p} = (D^{-1} \otimes K)(D^{-1} \otimes K + np I_{np})^{-1}$. Remark that $\widetilde{A}_{d_1,\ldots,d_p}$ is block-diagonal, with diagonal blocks being $B_{d_1}, \ldots, B_{d_p}$ using the notations of Section 3. With $\widetilde{\varepsilon} = Q\varepsilon = (\widetilde{\varepsilon_1}^\top, \ldots, \widetilde{\varepsilon_p}^\top)^\top$ and $\widetilde{f} = Qf = (\widetilde{f_1}^\top, \ldots, \widetilde{f_p}^\top)^\top$ we can write

$$\begin{aligned} |\delta_1(M)| &= \langle \widetilde{\varepsilon}, \widetilde{A}_{d_1,\ldots,d_p} \widetilde{\varepsilon} \rangle - \mathbb{E}\left[\langle \widetilde{\varepsilon}, \widetilde{A}_{d_1,\ldots,d_p} \widetilde{\varepsilon} \rangle \right]\ , \\ |\delta_2(M)| &= \left\| \widetilde{A}_{d_1,\ldots,d_p} \widetilde{\varepsilon} \right\|_2^2 - \mathbb{E}\left[\left\| \widetilde{A}_{d_1,\ldots,d_p} \widetilde{\varepsilon} \right\|_2^2 \right]\ , \\ |\delta_3(M)| &= 2 \langle \widetilde{A}_{d_1,\ldots,d_p} \widetilde{\varepsilon}, (\widetilde{A}_{d_1,\ldots,d_p} - I_{np}) \widetilde{f} \rangle\ , \\ |\delta_4(M)| &= 2 \langle \widetilde{\varepsilon}, (I_{np} - \widetilde{A}_{d_1,\ldots,d_p}) \widetilde{f} \rangle\ . \end{aligned}$$

We can see that the quantities δ_i decouple, therefore

$$
\begin{aligned}
|\delta_1(M)| &= \sum_{i=1}^{p} \langle \widetilde{\varepsilon}_i, A_{pd_i} \widetilde{\varepsilon}_i \rangle - \mathbb{E}\left[\langle \widetilde{\varepsilon}_i, A_{pd_i} \widetilde{\varepsilon} \rangle\right] \ , \\
|\delta_2(M)| &= \sum_{i=1}^{p} \|A_{pd_i} \widetilde{\varepsilon}_i\|_2^2 - \mathbb{E}\left[\|A_{pd_i} \widetilde{\varepsilon}_i\|_2^2\right] \ , \\
|\delta_3(M)| &= \sum_{i=1}^{p} 2\langle A_{pd_i} \widetilde{\varepsilon}_i, (A_{pd_i} - I_n) \widetilde{f}_i \rangle \ , \\
|\delta_4(M)| &= \sum_{i=1}^{p} 2\langle \widetilde{\varepsilon}_i, (I_n - A_{pd_i}) \widetilde{f}_i \rangle \ .
\end{aligned}
$$

Supposing (18)**.** Assumption (18) implies that the matrix P used above is the same for all the matrices M of $\mathcal{M}$. Using Lemma 9 of Arlot and Bach (2011), where we have p concentration results on the sets $\widetilde{\Omega}_i$, each of probability at least $1 - e^{1027+\ln(n)} e^{-x}$ we can state that, on the set $\bigcap_{i=1}^{p} \widetilde{\Omega}_i$, we have uniformly on $\mathcal{M}$

$$
\begin{aligned}
|\delta_1(M)| &\leq \sum_{i=1}^{p} \theta_1 \operatorname{Var}[\widetilde{\varepsilon}_i] \operatorname{tr}(A_{pd_i}^\top A_{pd_i}) + (C_A + C_B \theta_1^{-1}) x \operatorname{Var}[\widetilde{\varepsilon}_i] \ , \\
|\delta_2(M)| &\leq \sum_{i=1}^{p} \theta_2 \operatorname{Var}[\widetilde{\varepsilon}_i] \operatorname{tr}(A_{pd_i}^\top A_{pd_i}) + (C_C + C_D \theta_2^{-1}) x \operatorname{Var}[\widetilde{\varepsilon}_i] \ , \\
|\delta_3(M)| &\leq \sum_{i=1}^{p} \theta_3 \left\| (I_n - A_{pd_i}) \widetilde{f}_i \right\|_2^2 + C_E \theta_3^{-1} x \operatorname{Var}[\widetilde{\varepsilon}_i] \ , \\
|\delta_4(M)| &\leq \sum_{i=1}^{p} \theta_4 \left\| (I_n - A_{pd_i}) \widetilde{f}_i \right\|_2^2 + C_F \theta_4^{-1} x \operatorname{Var}[\widetilde{\varepsilon}_i] \ .
\end{aligned}
$$

Supposing (15)**.** We can use Lemma 8 of Arlot and Bach (2011) where we have p concentration results on the sets $\widetilde{\Omega}_{j,M}$, each of probability at least $1 - 6e^{-x}$ we can state that, on the set $\bigcap_{j=1}^{p} \bigcap_{M \in \mathcal{M}} \widetilde{\Omega}_i$, we have uniformly on $\mathcal{M}$ the same inequalities written above.

Final common step. To conclude, it suffices to see that for every $i \in \{1, \ldots, p\}$, $\operatorname{Var}[\widetilde{\varepsilon}_i] \leq |||\Sigma|||$.

■

F.2 Intermediate Result

We first prove a general oracle inequality, under the assumption that the penalty we use (with an estimator of Σ) does not underestimate the ideal penalty (involving Σ) too much.

Proposition 55 *Let $C_A, C_B, C_C, C_D, C_E \geq 0$ be fixed constants, $\gamma > 0$, $\theta_S \in [0, 1/4)$ and $K_S \geq 0$. On $\Omega_{\gamma \ln(n)}(\mathcal{M}, C_A, C_B, C_C, C_D, C_E)$, for every $S \in \mathcal{S}_p^{++}(\mathbb{R})$ such that*

$$\begin{aligned} &\operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot ((S-\Sigma) \otimes I_n)\right) \\ &\geq -\theta_S \operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot (\Sigma \otimes I_n)\right) \inf_{M \in \mathcal{M}} \left\{ \frac{b(M) + v_2(M) + K_S \ln(n) |||\Sigma|||}{v_1(M)} \right\} \end{aligned} \tag{34}$$

and for every $\theta \in (0, (1-4\theta_S)/2)$, we have:

$$\begin{aligned} \frac{1}{np} \left\| \widehat{f}_{\widehat{M}_o(S)} - f \right\|_2^2 \leq \frac{1+2\theta}{1-2\theta-4\theta_S} \inf_{M \in \mathcal{M}} \left\{ \frac{1}{np} \left\| \widehat{F}_M - F \right\|_2^2 + \frac{2 \operatorname{tr}(A_M \cdot ((S-\Sigma)_+ \otimes I_n))}{np} \right\} \\ + \frac{1}{1-2\theta-4\theta_S} \left[(2C_A + 3C_C + 6C_D + 6C_E + \frac{2}{\theta}(C_B + C_F))\gamma + \frac{\theta_S K_S}{4} \right] \frac{\ln(n) |||\Sigma|||}{np} \end{aligned} \tag{35}$$

Proof The proof of Proposition 55 is very similar to the one of Proposition 5 in Arlot and Bach (2011). First, we have

$$\left\| \widehat{f}_M - f \right\|_2^2 = b(M) + v_2(M) + \delta_2(M) + \delta_3(M) \ , \tag{36}$$

$$\left\| \widehat{f}_M - y \right\|_2^2 = \| \widehat{f}_M - f \|_2^2 - 2v_1(M) - 2\delta_1(M) + \delta_4(M) + \|\varepsilon\|_2^2 \ . \tag{37}$$

Combining Equation (29) and (37), we get:

$$\begin{aligned} &\left\| \widehat{f}_{\widehat{M}_o(S)} - f \right\|_2^2 + 2 \operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot ((S-\Sigma)_+ \otimes I_n)\right) + \widehat{\Delta}(\widehat{M}_o(S)) \\ &\leq \inf_{M \in \mathcal{M}} \left\{ \left\| \widehat{f}_M - f \right\|_2^2 + 2 \operatorname{tr}(A_M \cdot ((S-\Sigma) \otimes I_n)) + \widehat{\Delta}(M) \right\} \ . \end{aligned} \tag{38}$$

On the event $\Omega_{\gamma \ln(n)}$, for every $\theta \in (0,1]$ and $M \in \mathcal{M}$, using Equation (30) and (33) with $\theta = \theta_1 = \theta_4$,

$$|\widehat{\Delta}(M)| \leq \theta(b(M) + v_2(M)) + (C_A + \frac{1}{\theta}(C_B + C_F))\gamma \ln(n) |||\Sigma||| \ . \tag{39}$$

Using Equation (31) and (32) with $\theta_2 = \theta_3 = 1/2$ we get that for every $M \in \mathcal{M}$ Equation

$$\left\| \widehat{F}_M - F \right\|_2^2 \geq \frac{1}{2}(b(M) + v_2(M)) - (C_C + 2C_D + 2C_E)\gamma \ln(n) |||\Sigma||| \ ,$$

which is equivalent to

$$b(M) + v_2(M) \leq 2 \left\| \widehat{F}_M - F \right\|_2^2 + 2(C_C + 2C_D + 2C_E)\gamma \ln(n) |||\Sigma||| \ . \tag{40}$$

Combining Equation (39) and (40), we get

$$|\widehat{\Delta}(M)| \leq 2\theta \left\| \widehat{F}_M - F \right\|_2^2 + \left(C_A + (2C_C + 4C_D + 4C_E)\theta + (C_B + C_F)\frac{1}{\theta} \right) \gamma \ln(n) |||\Sigma||| \ .$$

With Equation (38), and with $C_1 = C_A, C_2 = 2C_C + 4C_D + 4C_E$ and $C_3 = C_B + C_F$ we get

$$\begin{aligned}(1-2\theta)\left\|\widehat{f}_{\widehat{M}_o(S)} - f\right\|_2^2 + 2\operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot ((S-\Sigma)_+ \otimes I_n)\right) \leq \\ \inf_{M\in\mathcal{M}}\left\{\left\|\widehat{f}_M - f\right\|_2^2 + 2\operatorname{tr}\left(A_M \cdot ((S-\Sigma)\otimes I_n)\right)\right\} + \left(C_1 + C_2\theta + \frac{C_3}{\theta}\right)\gamma\ln(n)|||\Sigma||| \ .\end{aligned} \tag{41}$$

Using Equation (34) we can state that

$$\operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot ((S-\Sigma)\otimes I_n)\right) \geq \frac{b(\widehat{M}_o(S)) + v_2(\widehat{M}_o(S)) + K_S\ln(n)|||\Sigma|||}{v_1(\widehat{M}_o(S))}\operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot (\Sigma\otimes I_n)\right)$$

so that

$$\operatorname{tr}\left(A_{\widehat{M}_o(S)} \cdot ((S-\Sigma)\otimes I_n)\right) \geq -\theta_S\left((b(\widehat{M}_o(S)) + v_2(\widehat{M}_o(S)) + K_S\ln(n)|||S|||\right) \ ,$$

which then leads to Equation (35) using Equation (40) and (41). ■

F.3 The Proof Itself

We now show Theorem 26 as a consequence of Proposition 55. It actually suffices to show that $\widehat{\Sigma}$ does not underestimate Σ too much, and that the second term in the infimum of Equation (35) is negligible in front of the quadratic error $(np)^{-1}\|\widehat{f}_M - f\|^2$.

Proof On the event Ω introduced in Theorem 20, Equation (14) holds. Let

$$\gamma = pc(\Sigma)\left(1 + c(\Sigma)\right) \ .$$

By Lemma 56 below, we have:

$$\inf_{M\in\mathcal{M}}\left\{\frac{b(M) + v_2(M) + K_S\ln(n)|||\Sigma|||}{v_1(M)}\right\} \geq 2\sqrt{\frac{K_S\ln(n)|||\Sigma|||}{n\operatorname{tr}(\Sigma)}} \ .$$

We supposed Assumption (15) holds. Using elementary algebra it is easy to show that, for every symmetric positive definite matrices A, M and N of size p, $M \succeq N$ implies that $\operatorname{tr}(AM) \geq \operatorname{tr}(AN)$. In order to have $\widehat{M}_o(\widehat{\Sigma})$ satisfying Equation (34), Theorem 20 shows that it suffices to have, for every $\theta_S > 0$,

$$2\theta_S\sqrt{\frac{K_S\ln(n)|||\Sigma|||}{n\operatorname{tr}(\Sigma)}} = 6\beta(2+\delta)\gamma\sqrt{\frac{\ln(n)}{n}} \ ,$$

which leads to the choice

$$K_S = \left(\frac{3\beta(\alpha+\delta)\gamma\operatorname{tr}(\Sigma)}{\theta_S|||\Sigma|||}\right)^2 \ .$$

We now take $\theta_S = \theta = (9\ln(n))^{-1}$. Let Ω be the set given by Theorem 20. Using Equation (35) and requiring that $\ln(n) \geq 6$ we get, on the set $\widetilde{\Omega} = \Omega \cap \Omega_{(\alpha+\delta)\ln(n)}(\mathcal{M}, C_A, C_B, C_C, C_D, C_E, C_F)$ of

probability $1-(p(p+1)/2+6pC)n^{-\delta}$, using that $\alpha \geq 2$:

$$\frac{1}{np}\left\|\widehat{f}_{\widehat{M}}-f\right\|_2 \leq \left(1+\frac{1}{\ln(n)}\right)\inf_{M\in\mathcal{M}}\left\{\frac{1}{np}\left\|\widehat{f}_M-f\right\|_2^2+\frac{2\operatorname{tr}\left(A_M\cdot((\widehat{\Sigma}-\Sigma)_+\otimes I_n)\right)}{np}\right\}$$
$$+\left(1-\frac{2}{3\ln(n)}\right)^{-1}\left[2C_A+3C_C+6C_D+6C_E+\ln(n)\left(18C_B+18C_F+\frac{729\beta^2\gamma^2\operatorname{tr}(\Sigma)^2}{4|||\Sigma|||^2}\right)\right]$$
$$\times(\alpha+\delta)^2\frac{\ln(n)^2|||\Sigma|||}{np} \ .$$

Using Equation (27) and defining

$$\eta_2 := 12\beta(\alpha+\delta)\gamma\sqrt{\frac{\ln(n)}{n}} \ ,$$

we get

$$\frac{1}{np}\left\|\widehat{f}_{\widehat{M}}-f\right\|_2 \leq \left(1+\frac{1}{\ln(n)}\right)\inf_{M\in\mathcal{M}}\left\{\frac{1}{np}\left\|\widehat{f}_M-f\right\|_2^2+\eta_2\frac{\operatorname{tr}(A_M\cdot(\Sigma\otimes I_n))}{np}\right\}$$
$$+\left(1-\frac{2}{3\ln(n)}\right)^{-1}\left[2C_A+3C_C+6C_D+6C_E+\ln(n)\left(18C_B+18C_F+\frac{729\beta^2\gamma^2\operatorname{tr}(\Sigma)^2}{4|||\Sigma|||^2}\right)\right] \quad (42)$$
$$\times(\alpha+\delta)^2\frac{\ln(n)^2|||\Sigma|||}{np} \ .$$

Now, to get a classical oracle inequality, we have to show that $\eta_2 v_1(M)=\eta_2\operatorname{tr}(A_M\cdot(\Sigma\otimes I_n))$ is negligible in front of $\|\widehat{f}_M-f\|^2$. Lemma 56 ensures that:

$$\forall M\in\mathcal{M}\,,\forall x\geq 0\,,\quad 2\sqrt{\frac{x|||\Sigma|||}{n\operatorname{tr}(\Sigma)}}v_1(M)\leq v_2(M)+x|||\Sigma||| \ .$$

With $0<C_n<1$, taking x to be equal to $72\beta^2\ln(n)\gamma^2\operatorname{tr}(\Sigma)/(C_n|||\Sigma|||)$ leads to

$$\eta_2 v_1(M)\leq 2C_n v_2(M)+\frac{72\beta^2\ln(n)\gamma^2\operatorname{tr}(\Sigma)}{C_n} \ . \quad (43)$$

Then, since $v_2(M)\leq v_2(M)+b(M)$ and using also Equation (36), we get

$$v_2(M)\leq\left\|\widehat{f}_M-f\right\|_2^2+|\delta_2(m)|+|\delta_3(M)| \ .$$

On $\widetilde{\Omega}$ we have that for every $\theta\in(0,1)$, using Equation (31) and (32),

$$|\delta_2(M)|+|\delta_3(M)|\leq 2\theta\left(\left\|\widehat{f}_M-f\right\|_2^2-|\delta_2(M)|-|\delta_3(M)|\right)+(C_C+(C_D+C_E)\theta^{-1})(\alpha+\delta)\ln(n)|||\Sigma||| \ ,$$

which leads to

$$|\delta_2(M)|+|\delta_3(M)|\leq\frac{2\theta}{1+2\theta}\left\|\widehat{f}_M-f\right\|_2^2+\frac{C_C+(C_D+C_E)\theta^{-1}}{1+2\theta}(\alpha+\delta)\ln(n)|||\Sigma||| \ .$$

Now, combining this equation with Equation (43), we get

$$\eta_2 v_1(M) \leq \left(1+\frac{4C_n\theta}{1+2\theta}\right)\left\|\widehat{f}_M - f\right\|_2^2 + 2C_n\frac{C_C+(C_D+C_E)\theta^{-1}}{1+2\theta}(\alpha+\delta)\ln(n)|||\Sigma||| + \frac{72\beta^2\ln(n)\gamma^2\operatorname{tr}(\Sigma)}{C_n} \ .$$

Taking $\theta = 1/2$ then leads to

$$\eta_2 v_1(M) \leq (1+C_n)\left\|\widehat{f}_M - f\right\|_2^2 + C_n(C_C+2(C_D+C_E))(\alpha+\delta)\ln(n)|||\Sigma||| + \frac{72\beta^2\ln(n)\gamma\operatorname{tr}(\Sigma)}{C_n} \ .$$

We now take $C_n = 1/\ln(n)$. We now replace the constants $C_A, C_B, C_C, C_D, C_E, C_F$ by their values in Lemma 54 and we get, for some constant L_2,

$$\left(1-\frac{2}{3\ln(n)}\right)^{-1}\left[1851.5+\ln(n)\left(5530.5+\frac{729\beta^2\gamma^2}{4|||\Sigma|||^2}\right)+616.5\left(1+\frac{1}{\ln(n)}\right)\frac{1}{\ln(n)}\right] + \frac{72\beta^2\ln(n)\gamma^2\operatorname{tr}(\Sigma)}{C_n} \leq L_2\ln(n)\gamma^2\frac{\operatorname{tr}(\Sigma)^2}{|||\Sigma|||^2}$$

From this we can deduce Equation (16) by noting that $\gamma \leq 2pc(\Sigma)^2$.

Finally we deduce an oracle inequality in expectation by noting that if $n^{-1}\|f_{\widehat{M}} - f\|^2 \leq R_{n,\delta}$ on $\widetilde{\Omega}$, using Cauchy-Schwarz inequality

$$\begin{aligned}\mathbb{E}\left[\frac{1}{np}\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^2\right] &= \mathbb{E}\left[\frac{\mathbf{1}_{\widetilde{\Omega}}}{np}\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^2\right] + \mathbb{E}\left[\frac{\mathbf{1}_{\widetilde{\Omega}^c}}{np}\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^2\right] \\ &\leq \mathbb{E}\left[R_{n,\delta}\right] + \frac{1}{np}\sqrt{\frac{4p(p+1)+6pC}{n^\delta}}\sqrt{\mathbb{E}\left[\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^4\right]} \ . \end{aligned} \tag{44}$$

We can remark that, since $|||A_M||| \leq 1$,

$$\left\|\widehat{f}_M - f\right\|_2^2 \leq 2\|A_M\varepsilon\|_2^2 + 2\|(I_{np}-A_M)f\|_2^2 \leq 2\|\varepsilon\|_2^2 + 8\|f\|_2^2 \ .$$

So

$$\mathbb{E}\left[\left\|\widehat{f}_{\widehat{M}} - f\right\|_2^4\right] \leq 12\left(np|||\Sigma||| + 4\|f\|_2^2\right)^2 \ ,$$

together with Equation (42) and Equation (44), induces Equation (17), using that for some constant $L_3 > 0$,

$$12\sqrt{\frac{p(p+1)/2+6pC}{n^\delta}}\left(|||\Sigma||| + \frac{4}{np}\|f\|_2^2\right) \leq L_3\frac{\sqrt{p(p+C)}}{n^{\delta/2}}\left(|||\Sigma||| + \frac{1}{np}\|f\|_2^2\right) \ .$$

■

Lemma 56 *Let $n, p \geq 1$ be two integers, $x \geq 0$ and $\Sigma \in \mathcal{S}_p^{++}(\mathbb{R})$. Then,*

$$\inf_{A \in \mathcal{M}_{np}(\mathbb{R}), |||A||| \leq 1} \left\{ \frac{\operatorname{tr}(A^\top A \cdot (\Sigma \otimes I_n)) + x|||\Sigma|||}{\operatorname{tr}(A \cdot (\Sigma \otimes I_n))} \right\} \geq 2\sqrt{\frac{x|||\Sigma|||}{n \operatorname{tr}(\Sigma)}}$$

Proof First note that the bilinear form on $\mathcal{M}_{np}(\mathbb{R})$, $(A,B) \mapsto \operatorname{tr}(A^\top B \cdot (\Sigma \otimes I_n))$ is a scalar product. By Cauchy-Schwarz inequality, for every $A \in \mathcal{M}_{np}(\mathbb{R})$,

$$\operatorname{tr}(A \cdot (\Sigma \otimes I_n))^2 \leq \operatorname{tr}(\Sigma \otimes I_n) \operatorname{tr}(A^\top A \cdot (\Sigma \otimes I_n)) \ .$$

Thus, since $\operatorname{tr}(\Sigma \otimes I_n) = n \operatorname{tr}(\Sigma)$, if $c = \operatorname{tr}(A \cdot (\Sigma \otimes I_n)) > 0$,

$$\operatorname{tr}(A^\top A \cdot (\Sigma \otimes I_n)) \geq \frac{c^2}{n \operatorname{tr}(\Sigma)} \ .$$

Therefore

$$\begin{aligned} \inf_{A \in \mathcal{M}_{np}(\mathbb{R}), |||A||| \leq 1} \left\{ \frac{\operatorname{tr}(A^\top A \cdot (\Sigma \otimes I_n)) + x|||\Sigma|||}{\operatorname{tr}(A \cdot (\Sigma \otimes I_n))} \right\} &\geq \inf_{c>0} \left\{ \frac{c}{n \operatorname{tr}(\Sigma)} + \frac{x|||\Sigma|||}{c} \right\} \\ &\geq 2\sqrt{\frac{x|||\Sigma|||}{n \operatorname{tr}(\Sigma)}} \ . \end{aligned}$$

■

F.4 Proof of Theorem 29

We now prove Theorem 29, first by proving that $\widehat{\Sigma}_{\mathrm{HM}}$ leads to a sharp enough approximation of the penalty.

Lemma 57 *Let $\widehat{\Sigma}_{HM}$ be defined as in Definition 28, $\alpha = 2$, $\kappa > 0$ be the numerical constant defined in Theorem 15 and assume* (13) *and* (18) *hold. For every $\delta \geq 2$, a constant $n_0(\delta)$, an absolute constant $L_1 > 0$ and an event Ω exist such that $\mathbb{P}(\Omega_{HM}) \geq 1 - pn^{-\delta}$ and for every $n \geq n_0(\delta)$, on Ω_{HM}, for every M in $\mathcal{M}$*

$$(1-\eta) \operatorname{tr}(A_M \cdot (\Sigma \otimes I_n)) \leq \operatorname{tr}(A_M \cdot (\widehat{\Sigma}_{HM} \otimes I_n)) \leq (1+\eta) \operatorname{tr}(A_M \cdot (\Sigma \otimes I_n)) \ , \qquad (45)$$

where $\eta := L_1(\alpha+\delta)\sqrt{\dfrac{\ln(n)}{n}}$.

Proof Let P be defined by (18). Let $M \in \mathcal{M}$, and $(d_1, \ldots, d_p) \in (0, +\infty)^p$ such that $M = P^\top \operatorname{Diag}(d_1, \ldots, d_p) P$. Thus, as shown in Section D, we have with $\widetilde{\Sigma} = P \Sigma P^\top$:

$$\operatorname{tr}(A_M \cdot (\Sigma \otimes I_n)) = \sum_{j=1}^{p} \operatorname{tr}(A_{pd_j}) \widetilde{\Sigma}_{j,j} \ .$$

let $\widetilde{\sigma}_j$ be defined as in Definition 28 (and thus $\widehat{\Sigma}_{\mathrm{HM}} = P\mathrm{Diag}(\widetilde{\sigma}_1,\ldots,\widetilde{\sigma}_p)P^\top$), we then have by Theorem 15 that for every $j \in \{1,\ldots,p\}$ an event Ω^j of probability $1-\kappa n^{-\delta}$ exists such that on Ω^j $|\widetilde{\Sigma}_{j,j} - \widetilde{\sigma}_j| \leq \eta\widetilde{\Sigma}_{j,j}$. Since

$$\mathrm{tr}(A_M \cdot (\widehat{\Sigma}_{\mathrm{HM}} \otimes I_n)) = \sum_{j=1}^{p} \mathrm{tr}(A_{pd_j})\widetilde{\sigma}_j \ ,$$

taking $\Omega_{\mathrm{HM}} = \cap_{j=1}^{p}\Omega^j$ suffices to conclude. ■

Proof [of Theorem 26] Adapting the proof of Theorem 26 to Assumption (18) first requires to take $\gamma = 1$ as Lemma 57 allows us. It then suffices to take the set $\widetilde{\Omega} = \Omega_{\mathrm{HM}} \cap \Omega_{(2+\delta)\ln(n)}(\mathcal{M}, C_A, C_B, C_C, C_D, C_E, C_F)$ (thus replacing α by 2) of probability $1-(p(p+1)/2+p)n^{-\delta} \geq 1-p^2n^{-\delta}$—supposing $p \geq 2$—if we require that $2\ln(n) \geq 1027$.

To get to the oracle inequality in expectation we use the same technique than above, but we note that $\sqrt{\mathbb{P}(\widetilde{\Omega}^c)} \leq \widetilde{L}_4 \times p/n^{\delta/2}$. We can finally define the constant L_4 by:

$$L_3\,\mathrm{tr}(\Sigma)(2+\delta)^2\frac{p\ln(n)^3}{np} + \frac{p}{n^{\delta/2}}|||\Sigma||| \leq L_4\gamma^2\,\mathrm{tr}(\Sigma)(\alpha+\delta)^2\frac{p\ln(n)^3}{np} \ .$$

■

References

Hirotogu Akaike. Statistical predictor identification. *Annals of the Institute of Statistical Mathematics*, 22:203–217, 1970.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, December 2005. ISSN 1532-4435.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Sylvain Arlot. Model selection by resampling penalization. *Electron. J. Stat.*, 3:557–624 (electronic), 2009. ISSN 1935-7524. doi: 10.1214/08-EJS196.

Sylvain Arlot and Francis Bach. Data-driven calibration of linear estimators with minimal penalties, July 2011. arXiv:0909.1884v2.

Sylvain Arlot and Pascal Massart. Data-driven calibration of penalties for least-squares regression. *Journal of Machine Learning Research*, 10:245–279 (electronic), 2009.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, May 1950.

Bart Bakker and Tom Heskes. Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, December 2003. ISSN 1532-4435. doi: http://dx.doi.org/10.1162/153244304322765658.

Lucien Birgé and Pascal Massart. Minimal penalties for Gaussian model selection. *Probability Theory and Related Fields*, 138:33–73, 2007.

Philip J. Brown and James V. Zidek. Adaptive multivariate ridge regression. *The Annals of Statistics*, 8(1):pp. 64–74, 1980. ISSN 00905364.

Rich Caruana. Multitask learning. *Machine Learning*, 28:41–75, July 1997. ISSN 0885-6125. doi: 10.1023/A:1007379606734.

Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

Gilles Gasso, Alain Rakotomamonjy, and Stéphane Canu. Recovering sparse signals with non-convex penalties and dc programming. *IEEE Trans. Signal Processing*, 57(12):4686–4698, 2009.

Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991. ISBN 9780521467131.

Laurent Jacob, Francis Bach, and Jean-Philippe Vert. Clustered multi-task learning: A convex formulation. *Computing Research Repository*, pages –1–1, 2008.

Matthieu Lerasle. Optimal model selection in density estimation. *Ann. Inst. H. Poincaré Probab. Statist.*, 2011. ISSN 0246-0203. Accepted. arXiv:0910.1654.

Percy Liang, Francis Bach, Guillaume Bouchard, and Michael I. Jordan. Asymptotically optimal regularization in smooth parametric models. In *Advances in Neural Information Processing Systems*, 2010.

Karim Lounici, Massimiliano Pontil, Alexandre B. Tsybakov, and Sara van de Geer. Oracle inequalities and optimal inference under group sparsity. Technical Report arXiv:1007.1771, Jul 2010. Comments: 37 pages.

Karim Lounici, Massimiliano Pontil, Sarah van de Geer, and Alexandre Tsybakov. Oracle inequalities and optimal inference under group sparsity. *The Annals of Statistics*, 39(4):2164–2204, 2011.

Colin L. Mallows. Some comments on C_P. *Technometrics*, pages 661–675, 1973.

Guillaume Obozinski, Martin J. Wainwright, and Michael I. Jordan. Support union recovery in high-dimensional multivariate regression. *The Annals of Statistics*, 39(1):1–17, 2011.

Carl E. Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA, 12 2002.

Sebastian Thrun and Joseph O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. *Proceedings of the 13th International Conference on Machine Learning*, 1996.

Grace Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990. ISBN 0-89871-244-0.

Tong Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17(9):2077–2098, 2005.

Journal of Machine Learning Research 13 (2012) 2813-2869 Submitted 1/12; Revised 7/12; Published 10/12

A Unified View of Performance Metrics: Translating Threshold Choice into Expected Classification Loss

José Hernández-Orallo JORALLO@DSIC.UPV.ES
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
Camí de Vera s/n, 46020, València, Spain

Peter Flach PETER.FLACH@BRISTOL.AC.UK
Intelligent Systems Laboratory
University of Bristol, United Kingdom
Merchant Venturers Building, Woodland Road
Bristol, BS8 1UB, United Kingdom

Cèsar Ferri CFERRI@DSIC.UPV.ES
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
Camí de Vera s/n, 46020, València, Spain

Editor: Charles Elkan

Abstract

Many performance metrics have been introduced in the literature for the evaluation of classification performance, each of them with different origins and areas of application. These metrics include accuracy, unweighted accuracy, the area under the ROC curve or the ROC convex hull, the mean absolute error and the Brier score or mean squared error (with its decomposition into refinement and calibration). One way of understanding the relations among these metrics is by means of variable operating conditions (in the form of misclassification costs and/or class distributions). Thus, a metric may correspond to some expected loss over different operating conditions. One dimension for the analysis has been the distribution for this range of operating conditions, leading to some important connections in the area of proper scoring rules. We demonstrate in this paper that there is an equally important dimension which has so far received much less attention in the analysis of performance metrics. This dimension is given by the decision rule, which is typically implemented as a *threshold choice method* when using scoring models. In this paper, we explore many old and new threshold choice methods: fixed, score-uniform, score-driven, rate-driven and optimal, among others. By calculating the expected loss obtained with these threshold choice methods for a uniform range of operating conditions we give clear interpretations of the 0-1 loss, the absolute error, the Brier score, the *AUC* and the refinement loss respectively. Our analysis provides a comprehensive view of performance metrics as well as a systematic approach to loss minimisation which can be summarised as follows: given a model, apply the threshold choice methods that correspond with the available information about the operating condition, and compare their expected losses. In order to assist in this procedure we also derive several connections between the aforementioned performance metrics, and we highlight the role of calibration in choosing the threshold choice method.

Keywords: classification performance metrics, cost-sensitive evaluation, operating condition, Brier score, area under the ROC curve (*AUC*), calibration loss, refinement loss

1. Introduction

The choice of a proper performance metric for evaluating classification (Hand, 1997) is an old but still lively debate which has incorporated many different performance metrics along the way. Besides accuracy (*Acc*, or, equivalently, the error rate or 0-1 loss), many other performance metrics have been studied. The most prominent and well-known metrics are the Brier Score (*BS*, also known as Mean Squared Error) (Brier, 1950) and its decomposition in terms of refinement and calibration (Murphy, 1973), the absolute error (*MAE*), the log(arithmic) loss (or cross-entropy) (Good, 1952) and the area under the ROC curve (*AUC*, also known as the Wilcoxon-Mann-Whitney statistic, linearly related to the Gini coefficient and to the Kendall's tau distance to a perfect model) (Swets et al., 2000; Fawcett, 2006). There are also many graphical representations and tools for model evaluation, such as ROC curves (Swets et al., 2000; Fawcett, 2006), ROC isometrics (Flach, 2003), cost curves (Drummond and Holte, 2000, 2006), DET curves (Martin et al., 1997), lift charts (Piatetsky-Shapiro and Masand, 1999), and calibration maps (Cohen and Goldszmidt, 2004). A survey of graphical methods for classification predictive performance evaluation can be found in the work of Prati et al. (2011).

Many classification models can be regarded as functions which output a score for each example and class. This score represents a probability estimate of each example to be in one of the classes (or may just represent an unscaled magnitude which is monotonically related with a probability estimate). A score can then be converted into a class label using a decision rule. One of the reasons for evaluation being so multi-faceted is that models may be learnt in one context (misclassification costs, class distribution, etc.) but *deployed* in a different context. A context is usually described by a set of parameters, known as *operating condition*. When we have a clear operating condition at deployment time, there are effective tools such as ROC analysis (Swets et al., 2000; Fawcett, 2006) to establish which model is best and what its expected loss will be. However, the question is more difficult in the general case when we do not have information about the operating condition where the model will be applied. In this case, we want our models to perform well in a wide range of operating conditions. In this context, the notion of 'proper scoring rule', see, for example, the work of Murphy and Winkler (1970), sheds some light on some performance metrics. Some proper scoring rules, such as the Brier Score (MSE loss), the logloss, boosting loss and error rate (0-1 loss) have been shown by Buja et al. (2005) to be special cases of an integral over a Beta density of costs, see, for example, the works of Gneiting and Raftery (2007), Reid and Williamson (2010, 2011) and Brümmer (2010). Each performance metric is derived as a special case of the Beta distribution. However, this analysis focusses on scoring rules which are 'proper', that is, metrics that are minimised for well-calibrated probability assessments or, in other words, get the best (lowest) score by forecasting the true beliefs. Much less is known (in terms of expected loss for varying distributions) about other performance metrics which are non-proper scoring rules, such as *AUC*. Moreover, even its role as a classification performance metric has been put into question (Hand, 2009, 2010; Hand and Anagnostopoulos, 2011).

All these approaches make some (generally implicit and poorly understood) assumptions on how the model will work for each operating condition. In particular, it is generally assumed that the threshold which is used to discriminate between the classes will be set according to the operating condition. In addition, it is assumed that the threshold will be set in such a way that the estimated probability where the threshold is set is made equal to the operating condition. This is natural if we focus on proper scoring rules. Once all this is settled and fixed, different performance metrics

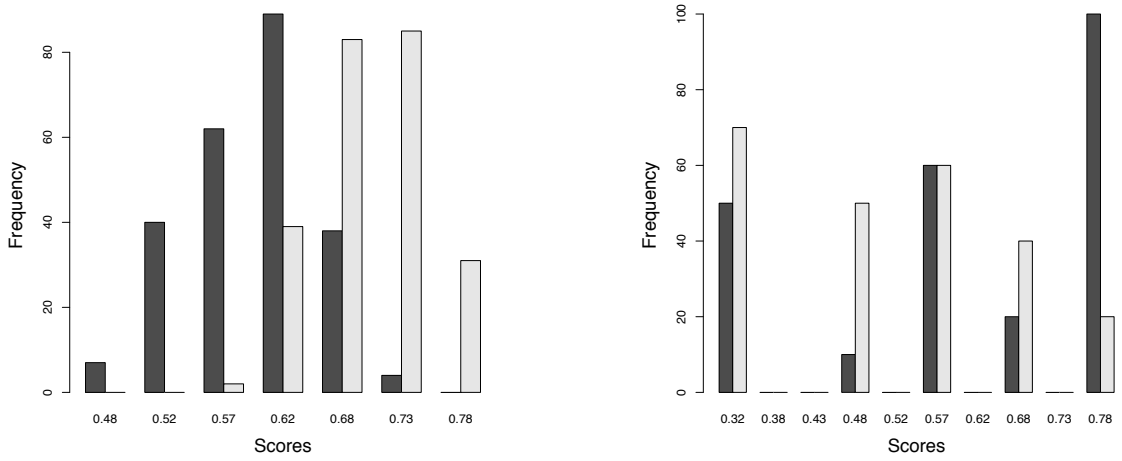


Figure 1: Histograms of the score distribution for model *A* (left) and model *B* (right).

represent different expected losses by using the distribution over the operating condition as a parameter. However, this *threshold choice* is only one of the many possibilities, as some other works have explored or mentioned in a more or less explicit way (Wieand et al., 1989; Drummond and Holte, 2006; Hand, 2009, 2010).

In our work we make these assumptions explicit through the concept of a *threshold choice method,* which systematically links performance metrics and expected loss. A threshold choice method sets a single threshold on the scores of a model in order to arrive at classifications, possibly taking circumstances in the deployment context into account, such as the operating condition (the class or cost distribution) or the intended proportion of positive predictions (the predicted positive rate). Building on this notion of threshold choice method, we are able to systematically explore how known performance metrics are linked to expected loss, resulting in a range of results that are not only theoretically well-founded but also practically relevant.

The basic insight is the realisation that there are many ways of converting a model (understood throughout this paper as a function assigning scores to instances) into a classifier that maps instances to classes (we assume binary classification throughout). Put differently, there are many ways of setting the threshold given a model and an operating condition. We illustrate this with an example concerning a very common scenario in machine learning research. Consider two models A and B, a naive Bayes model and a decision tree respectively (induced from a training data set), which are evaluated against a test data set, producing a score distribution for the positive and negative classes as shown in Figure 1. We see that scores are in the $[0,1]$ interval and in this example are interpreted as probability estimates for the negative class. ROC curves of both models are shown in Figure 2. We will assume that at this *evaluation time* we do not have information about the operating condition, but we expect that this information will be available at *deployment time*.

If we ask the question of which model is best we may rush to calculate its *AUC* and *BS* (and perhaps other metrics), as given by Table 1. However, we cannot give an answer because the question is *underspecified*. First, we need to know the range of operating conditions the model will work with. Second, we need to know how we will make the classifications, or in other words, we need a *deci-*

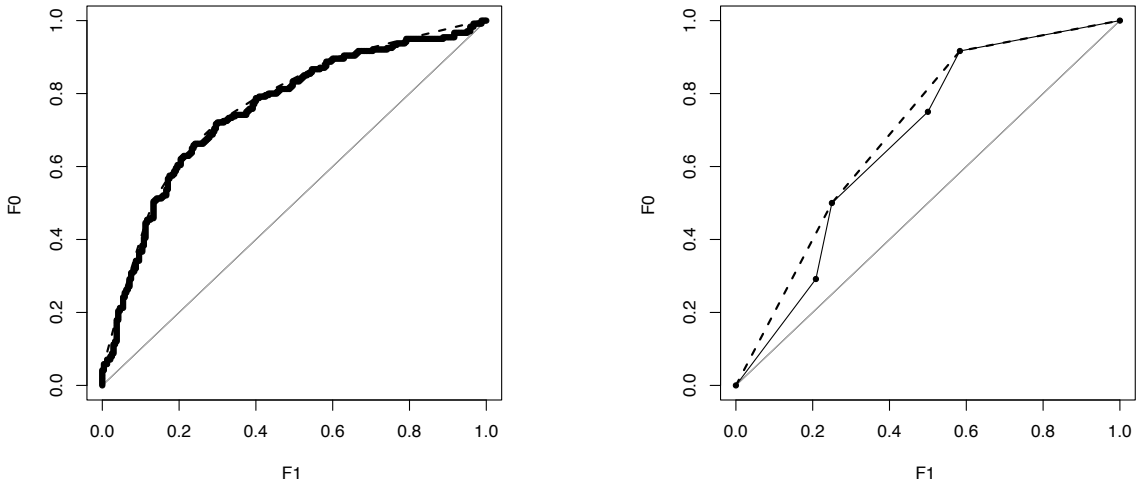


Figure 2: ROC Curves for model *A* (left) and model *B* (right).

sion rule, which can be implemented as a *threshold choice method* when the model outputs scores. For the first dimension (already considered by the work on proper scoring rules), if we have no precise knowledge about the operating condition, we can assume any of many distributions, depending on whether we have some information about how the cost may be distributed or no information at all. For instance, we can use a symmetric Beta distribution (Hand, 2009), an asymmetric Beta distribution (Hand and Anagnostopoulos, 2011) or, a partial or truncated distribution only considering a range of costs, or a simple uniform distribution (Drummond and Holte, 2006), as we also do here, which considers all operating conditions equally likely. For the second (new) dimension, we *also* have many options.

performance metric	model *A*	model *B*
AUC	0.791	0.671
Brier score	0.328	0.231

Table 1: Results from two models on a data set.

For instance, we can just set a fixed threshold at 0.5. This is what naive Bayes and decision trees do by default. This decision rule works as follows: if the score is greater than 0.5 then predict negative (1), otherwise predict positive (0). With this precise decision rule, we can now ask the question about the expected misclassification loss for a range of different misclassification costs (and/or class distributions), that is, for a distribution of operating conditions. Assuming a uniform distribution for operating conditions (cost proportions), we can effectively calculate the answer on the data set: 0.51.

But we can use other decision rules. We can use decision rules which adapt to the operating condition. One of these decision rules is the score-driven threshold choice method, which sets the threshold equal to the operating condition or, more precisely, to a cost proportion c. Another decision rule is the rate-driven threshold choice method, which sets the threshold in such a way that

the proportion of predicted positives (or predicted positive rate), simply known as 'rate' and denoted by r, equals the operating condition. Using these three different threshold choice methods for the models A and B, and assuming cost proportions are uniformly distributed, we get the expected losses shown in Table 2.

threshold choice method	expected loss model A	expected loss model B
Fixed ($T = 0.5$)	0.510	0.375
Score-driven ($T = c$)	0.328	0.231
Rate-driven (T s.t. $r = c$)	0.188	0.248

Table 2: Extension of Table 1 where two models are applied with three different threshold choice methods each, leading to six different classifiers and corresponding expected losses. In all cases, the expected loss is calculated over a range of cost proportions (operating conditions), which is assumed to be uniformly distributed. We denote the threshold by T, the cost proportion by c and the predicted positive rate by r).

In other words, only when we specify or assume a threshold choice method can we convert a model into a classifier for which it makes sense to consider its expected loss (for a range or distribution of costs). In fact, as we can see in Table 2, very different expected losses are obtained for the same model with different threshold choice methods. And this is the case even assuming the same uniform cost distribution for all of them.

Once we have made this (new) dimension explicit, we are ready to ask new questions. How many threshold choice methods are there? Table 3 shows six of the threshold choice methods we will analyse in this work, along with their notation. Only the score-fixed and the score-driven methods have been analysed in previous works in the area of proper scoring rules. The use of rates, instead of scores, is assumed in *screening* applications where an inspection, pre-diagnosis or coverage rate is intended (Murphy et al., 1987; Wieand et al., 1989), and the idea which underlies the distinction between rate-uniform and rate-driven is suggested by Hand (2010). In addition, a seventh threshold choice method, known as optimal threshold choice method, denoted by T^o, has been (implicitly) used in a few works (Drummond and Holte, 2000, 2006; Hand, 2009).

Threshold choice method	Fixed	Chosen uniformly	Driven by o.c.
Using scores	score-fixed (T^{sf})	score-uniform (T^{su})	score-driven (T^{sd})
Using rates	rate-fixed (T^{rf})	rate-uniform (T^{ru})	rate-driven (T^{rd})

Table 3: Possible threshold choice methods. The first family uses scores (as they were probabilities) and the second family uses rates (using scores as rank indicators). For both families we can fix a threshold or assume them ranging uniformly, which makes the threshold choice method independent from the operating condition. Only the last column takes the operating condition (o.c.) into account, and hence are the most interesting threshold choice methods.

We will see that each threshold choice method is linked to a specific performance metric. This means that if we decide (or are forced) to use a threshold choice method then there is a recommended performance metric for it. The results in this paper show that accuracy is the appropriate performance metric for the score-fixed method, *MAE* fits the score-uniform method, *BS* is the appropriate performance metric for the score-driven method, and *AUC* fits both the rate-uniform and the rate-driven methods. The latter two results assume a uniform cost distribution. It is important to make this explicit since the uniform cost distribution may be unrealistic in many particular situations and it is only one of many choices for a reference standard in the general case. As we will mention at the end of the paper, this suggests that new metrics can be derived by changing this distribution, as Hand (2009) has already done for the optimal threshold choice method with a Beta distribution.

The good news is that inter-comparisons are still possible: given a threshold choice method we can calculate expected loss from the relevant performance metric. The results in Table 2 allow us to conclude that model *A* achieves the lowest expected loss for uniformly sampled cost proportions, *if* we are wise enough to choose the appropriate threshold choice method (in this case the rate-driven method) to turn model *A* into a successful classifier. Notice that this cannot be said by just looking at Table 1 because the metrics in this table are not comparable to each other. In fact, there is no single performance metric that ranks the models in the correct order, because, as already said, expected loss cannot be calculated for models, only for classifiers.

1.1 Contributions and Structure of the Paper

The contributions of this paper to the subject of model evaluation for classification can be summarised as follows.

1. The expected loss of a model can only be determined if we select a distribution of operating conditions and a threshold choice method. We need to set a point in this two-dimensional space. Along the second (usually neglected) dimension, several new threshold choice methods are introduced in this paper.

2. We answer the question: "if one is choosing thresholds in a particular way, which performance metric is appropriate?" by giving an explicit expression for the expected loss for each threshold choice method. We derive linear relationships between expected loss and many common performance metrics.

3. Our results reinvigorate AUC as a well-founded measure of expected classification loss for both the rate-uniform and rate-driven methods. While Hand (2009, 2010) raised objections against AUC for the optimal threshold choice method only, noting that AUC can be consistent with other threshold choice methods, we encountered a widespread misunderstanding in the machine learning community that the AUC is fundamentally flawed as a performance metric—a clear misinterpretation of Hand's papers that we hope that this paper helps to further rectify.

4. One fundamental and novel result shows that the refinement loss of the convex hull of a ROC curve is equal to expected *optimal* loss as measured by the area under the optimal cost curve. This sets an optimistic (but also unrealistic) bound for the expected loss.

5. Conversely, from the usual calculation of several well-known performance metrics we can derive expected loss. Thus, classifiers and performance metrics become easily comparable.

With this we do not choose the best model but rather the best classifier (a model with a particular threshold choice method).

6. By cleverly manipulating scores we can connect several of these performance metrics, either by the notion of evenly-spaced scores or perfectly calibrated scores. This provides an additional way of analysing the relation between performance metrics and, of course, threshold choice methods.

7. We use all these connections to better understand which threshold choice method should be used, and in which cases some are better than others. The analysis of calibration plays a central role in this understanding, and also shows that non-proper scoring rules do have their role and can lead to lower expected loss than proper scoring rules, which are, as expected, more appropriate when the model is well-calibrated.

This set of contributions provides an integrated perspective on performance metrics for classification around the systematic exploration of the notion of threshold choice method that we develop in this paper.

The remainder of the paper is structured as follows. Section 2 introduces some notation, the basic definitions for operating condition, threshold, expected loss, and particularly the notion of threshold choice method, which we will use throughout the paper. Section 3 investigates expected loss for fixed threshold choice methods (score-fixed and rate-fixed), which are the base for the rest. We show that, not surprisingly, the expected loss for these threshold choice method are the 0-1 loss (weighted or unweighted accuracy depending on whether we use cost proportions or skews). Section 4 presents the results that the score-uniform threshold choice method has *MAE* as associate performance metric and the score-driven threshold choice method leads to the Brier score. We also show that one dominates over the other. Section 5 analyses the non-fixed methods based on rates. Somewhat surprisingly, both the rate-uniform threshold choice method and the rate-driven threshold choice method lead to linear functions of *AUC*, with the latter always been better than the former. All this vindicates the rate-driven threshold choice method but also *AUC* as a performance metric for classification. Section 6 uses the optimal threshold choice method, connects the expected loss in this case with the area under the optimal cost curve, and derives its corresponding metric, which is refinement loss, one of the components of the Brier score decomposition. Section 7 analyses the connections between the previous threshold choice methods and metrics by considering several properties of the scores: evenly-spaced scores and perfectly calibrated scores. This also helps to understand which threshold choice method should be used depending on how good scores are. Finally, Section 8 closes the paper with a thorough discussion of results, related work, and an overall conclusion with future work and open questions. Two appendices include a derivation of univariate operating conditions for costs and skews and some technical results for the optimal threshold choice method.

2. Background

In this section we introduce some basic notation and definitions we will need throughout the paper. Further definitions will be introduced when needed. The most important definitions we will need are introduced below: the notion of threshold choice method and the expression of expected loss.

2.1 Notation and Basic Definitions

A *classifier* is a function that maps instances x from an instance space X to classes y from an output space Y. For this paper we will assume binary classifiers, that is, $Y = \{0,1\}$. A *model* is a function $m : X \to \mathbb{R}$ that maps examples to real numbers (scores) on an unspecified scale. We use the convention that higher scores express a stronger belief that the instance is of class 1. A *probabilistic model* is a function $m : X \to [0,1]$ that maps examples to estimates $\hat{p}(1|x)$ of the probability of example x to be of class 1. Throughout the paper we will use the term *score* (usually denoted by s) both for unscaled values (in an unbounded interval) and probability estimates (in the interval [0,1]). Nonetheless, we will make the interpretation explicit whenever we use them in one way or the other. We will do similarly for thresholds. In order to make predictions in the Y domain, a model can be converted to a classifier by fixing a decision threshold t on the scores. Given a predicted score $s = m(x)$, the instance x is classified in class 1 if $s > t$, and in class 0 otherwise.

For a given, unspecified model and population from which data are drawn, we denote the score density for class k by f_k and the cumulative distribution function by F_k. Thus, $F_0(t) = \int_{-\infty}^{t} f_0(s)ds = P(s \leq t|0)$ is the proportion of class 0 points correctly classified if the decision threshold is t, which is the sensitivity or true positive rate at t. Similarly, $F_1(t) = \int_{-\infty}^{t} f_1(s)ds = P(s \leq t|1)$ is the proportion of class 1 points incorrectly classified as 0 or the false positive rate at threshold t; $1 - F_1(t)$ is the true negative rate or specificity. Note that we use 0 for the positive class and 1 for the negative class, but scores increase with $\hat{p}(1|x)$. That is, $F_0(t)$ and $F_1(t)$ are monotonically non-decreasing with t. This has some notational advantages and is the same convention as used by, for example, Hand (2009).

Given a data set $D \subset \langle X, Y \rangle$ of size $n = |D|$, we denote by D_k the subset of examples in class $k \in \{0,1\}$, and set $n_k = |D_k|$ and $\pi_k = n_k/n$. Clearly $\pi_0 + \pi_1 = 1$. We will use the term *class proportion* for π_0 (other terms such as 'class ratio' or 'class prior' have been used in the literature). Given a model and a threshold t, we denote by $R(t)$ the predicted positive rate, that is, the proportion of examples that will be predicted positive (class 0) if the threshold is set at t. This can also be defined as $R(t) = \pi_0 F_0(t) + \pi_1 F_1(t)$. The average score of actual class k is $\bar{s}_k = \int_0^1 s f_k(s)ds$. Given any strict order for a data set of n examples we will use the index i on that order to refer to the i-th example. Thus, s_i denotes the score of the i-th example and y_i its true class.

We define partial class accuracies as $Acc_0(t) = F_0(t)$ and $Acc_1(t) = 1 - F_1(t)$. From here, (weighted or micro-average) accuracy is defined as $Acc(t) = \pi_0 Acc_0(t) + \pi_1 Acc_1(t)$ and (unweighted or macro-average) accuracy as $uAcc(t) = (Acc_0(t) + Acc_1(t))/2$ (also known as 'average recall', Flach, 2012), which computes accuracy while assuming balanced classes.

We denote by $U_S(x)$ the continuous uniform distribution of variable x over an interval $S \subset \mathbb{R}$. If this interval S is $[0,1]$ then S can be omitted. The family of continuous distributions Beta is denoted by $\beta_{\alpha,\beta}$. The Beta distributions are always defined in the interval $[0,1]$. Note that the uniform distribution is a special case of the Beta family, that is, $\beta_{1,1} = U$.

2.2 Operating Conditions and Expected Loss

When a model is deployed for classification, the conditions might be different to those during training. In fact, a model can be used in several deployment contexts, with different results. A context can entail different class distributions, different classification-related costs (either for the attributes, for the class or any other kind of cost), or some other details about the effects that the application of a model might entail and the severity of its errors. In practice, a deployment context or *operating*

condition is usually defined by a misclassification cost function and a class distribution. Clearly, there is a difference between operating when the cost of misclassifying 0 into 1 is equal to the cost of misclassifying 1 into 0 and doing so when the former is ten times the latter. Similarly, operating when classes are balanced is different from when there is an overwhelming majority of instances of one class.

One general approach to cost-sensitive learning assumes that the cost does not depend on the example but only on its class. In this way, misclassification costs are usually simplified by means of cost matrices, where we can express that some misclassification costs are higher than others (Elkan, 2001). Typically, the costs of correct classifications are assumed to be 0. This means that for binary models we can describe the cost matrix by two values $c_k \geq 0$ with at least one of both being strictly greater than 0, representing the misclassification cost of an example of class k. Additionally, we can normalise the costs by setting $b = c_0 + c_1$, which will be referred to as the *cost magnitude* (which is clearly strictly greater than 0), and $c = c_0/b$; we will refer to c as the *cost proportion* Since this can also be expressed as $c = (1 + c_1/c_0)^{-1}$, it is often called 'cost ratio' even though, technically, it is a proportion ranging between 0 and 1.

Under these assumptions, an operating condition can be defined as a tuple $\theta = \langle b, c, \pi_0 \rangle$. The space of operating conditions is denoted by Θ. These three parameters are not necessarily independent, as we will discuss in more detail below. The loss for an operating condition is defined as follows:

$$\begin{aligned} Q(t;\theta) = Q(t;\langle b,c,\pi_0\rangle) &\triangleq b\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\} \\ &= c_0\pi_0(1-F_0(t)) + c_1\pi_1 F_1(t). \end{aligned} \tag{1}$$

It is important to distinguish the information we may have available at each stage of the process. At evaluation time we may not have access to some information that is available later, at deployment time. In many real-world problems, when we have to evaluate or compare models, we do not know the operating condition that will apply during deployment. One general approach is to evaluate the model on a range of possible operating points. In order to do this, we have to set a weight or distribution for operating conditions.

A key issue when applying a model under different operating conditions is how the threshold is chosen in each of them. If we work with a classifier, this question vanishes, since the threshold is already settled. However, in the general case when we work with a model, we have to decide how to establish the threshold. The key idea proposed in this paper is the notion of a threshold choice method, a function which converts an operating condition into an appropriate threshold for the classifier.

Definition 1 Threshold choice method. *A threshold choice method*[1] *is a (possibly non-deterministic) function $T : \Theta \to \mathbb{R}$ such that given an operating condition it returns a decision threshold.*

When we say that T may be non-deterministic, it means that the result may depend on a random variable and hence may itself be a random variable according to some distribution. We introduce

1. The notion of threshold choice method could be further generalised to cover situations where we have some information about the operating condition which cannot be expressed in terms of a specific value of Θ, such a distribution on Θ or information about $\mathbb{E}\{b\}$, $\mathbb{E}\{bc\}$, etc. This generalisation could be explored, but it is not necessary for the cases discussed in this paper.

the threshold choice method as an abstract concept since there are several reasonable options for the function T, essentially because there may be different degrees of information about the model and the operating conditions at evaluation time. We can set a fixed threshold ignoring the operating condition; we can set the threshold by looking at the ROC curve (or its convex hull) and using the cost proportion to intersect the ROC curve (as ROC analysis does); we can set a threshold looking at the estimated scores; or we can set a threshold independently from the rank or the scores. The way in which we set the threshold may dramatically affect performance. But, not less importantly, the performance metric used for evaluation must be in accordance with the threshold choice method.

Given a threshold choice function T, the loss for a particular operating condition θ is given by $Q(T(\theta);\theta)$. However, if we do not know the operating condition precisely, we can define a distribution for operating conditions as a multivariate distribution, $w(\theta)$. From here, we can now calculate expected loss as a weighted average over operating conditions (Adams and Hand, 1999):

$$L \triangleq \int_{\Theta} Q(T(\theta);\theta)w(\theta)d\theta. \tag{2}$$

Calculating this integral for a particular case depends on the threshold choice method and the kind of model, but particularly on the space of operating conditions Θ and its associated distribution $w(\theta)$. Typically, the representation of operating conditions is simplified from a three-parameter tuple $\langle b, c, \pi_0 \rangle$ to a single parameter. This reduces w to a univariate distribution. However, this reduction must carry some assumptions. For instance, the cost magnitude b is not always independent of c and π_0, since costs in very imbalanced cases tend to have higher magnitude. For instance, we may have two different operating conditions, one with $c_0 = 10$ and $c_1 = 1$ and another with $c_0 = 5$ and $c_1 = 50$. While the cost ratios are symmetric (10:1 with $c = 10/11$ for the first case, 1:10 with $c = 1/11$ for the second), the second operating condition will clearly have more impact on the expected loss, because its magnitude is five times higher. Moreover, c is usually closely linked to π_0, since the higher the imbalance (class proportion), the higher the cost proportion. For instance, if positives are rare, we usually want them to be detected (especially in diagnosis and fault detection applications), and false negatives (i.e., a positive which has not been detected but misclassified as negative) will have higher cost.

Despite these dependencies, one common option for this simplified operating condition is to consider that costs are normalised (the cost matrix always sums up to a constant, that is, the cost magnitude b is constant), or less strongly, that b and c are independent. Another option which does not require independence of b and c relies on noticing that b is a multiplicative factor in Equation (1). From here, we just need to assume that the threshold choice method is independent of b. This is not a strong assumption, since all the threshold choice methods that have been used systematically in the literature (e.g., the optimal threshold choice method and the score-driven method) are independent of b and so are the rest of methods we work with in this paper. With this, as deployed in appendix A, we can incorporate b in a bivariate distribution $v_H(\langle c, \pi_0 \rangle)$ for cost and class proportions. This does not mean that we ignore the magnitude b or assume it constant, but that we can embed its variability in v_H. From here, we just derive two univariate cost functions and the corresponding expected losses.

The first one assumes π_0 constant, leading to a loss expression which only depends on cost proportions c:

$$Q_c(t;c) \triangleq \mathbb{E}\{b\}\{c\pi_0(1 - F_0(t)) + (1 - c)\pi_1 F_1(t)\}. \tag{3}$$

Using this expression, expected loss will be derived as an integral using the univariate distribution w_c, which incorporates the variability of b jointly with c (see appendix A for details).

A different approach to reducing the operating condition to a single parameter is the notion of *skew*, which is a normalisation of the product between cost proportion and class proportion:

$$z \triangleq \frac{c\pi_0}{c\pi_0 + (1-c)(1-\pi_0)}.$$

This means that π_0 is no longer fixed, but neither is it independent of c. What z does is to combine both parameters. This is a different way of reducing the operating condition to one single parameter. We thus (re-)define loss as depending solely on z.

$$Q_z(t;z) \triangleq z(1-F_0(t)) + (1-z)F_1(t).$$

Similarly, we also define a weight $w_z(z)$ which also incorporates the variability of b and π_0 (see appendix A for details), which will be used in the integral for calculating the expected loss below.

As a result, in what follows, we will just work with operating conditions which are either defined by the cost proportion c (assuming a fixed class distribution π_0) or by the skew z (which combines c and π_0). For convenience, as justified in appendix A, we will assume $\mathbb{E}\{b\} = 2$. Interestingly, we can relate both approaches (using costs and skews) with the following lemma (proven in appendix A):

Lemma 2 *Assuming $\mathbb{E}\{b\} = 2$, if $\pi_0 = \pi_1$ then $z = c$ and $Q_z(t;z) = Q_c(t;c)$.*

This will allow us to translate the results for cost proportions to skews.

From now on, since the operating condition can be either a cost proportion c or a skew z we will use the subscript c or z to differentiate them. In fact, threshold choice methods will be represented by T_c and T_z and they will be defined as $T_c : [0,1] \to \mathbb{R}$ and $T_z : [0,1] \to \mathbb{R}$ respectively. Superscripts will be used to identify particular threshold choice methods. Some threshold choice methods we consider in this paper take additional information into account, such as a default threshold or a target predicted positive rate; such information is indicated by square brackets. So, for example, the score-fixed threshold choice method for cost proportions considered in the next section is indicated thus: $T_c^{sf}[t](c)$. In the rest of this paper, we explore a range of different methods to choose the threshold (some deterministic and some non-deterministic). We will give proper definitions of all these threshold choice methods in its due section.

The expected loss for costs and skews is then adapted from Equation (2) as follows:

Definition 3 *Given a threshold choice method for cost proportions T_c and a probability density function over cost proportions w_c,* expected loss *L_c is defined as*

$$L_c \triangleq \int_0^1 Q_c(T_c(c);c)w_c(c)dc. \tag{4}$$

Incorporating the class distribution into the operating condition as skews and defining a distribution over skews w_z, we obtain expected loss over a distribution of skews:

$$L_z \triangleq \int_0^1 Q_z(T_z(z);z)w_z(z)dz. \tag{5}$$

It is worth noting that if we plot Q_c or Q_z against c and z, respectively, we obtain *cost curves* as defined by Drummond and Holte (2000, 2006). Cost curves are also known as risk curves (e.g., Reid and Williamson, 2011, where the plot can also be shown in terms of *priors*, that is, class proportions).

Equations (4) and (5) illustrate the space we explore in this paper. Two parameters determine the expected loss: $w_c(c)$ and $T_c(c)$ (respectively $w_z(z)$ and $T_z(z)$). While much work has been done on a first dimension, by changing $w_c(c)$ or $w_z(z)$, particularly in the area of proper scoring rules, no work has *systematically* analysed what happens when changing the second dimension, $T_c(c)$ or $T_z(z)$.

This means that in this paper we focus on this second dimension, and just make some simple choices for the first dimension. Except for cases where the threshold choice is independent of the operating condition, we will assume a uniform distribution for $w_c(c)$ and $w_z(z)$. This is of course just one possible choice, but not an arbitrary choice for a number of reasons:

- The uniform distribution is arguably the simplest distribution for a value between 0 and 1 and requires no parameters.
- This distribution makes the representation of the loss straightforward, since we can plot Q on the y-axis versus c (or z) on the x-axis, where the x-axis can be shown linearly from 0 to 1, without any distribution warping. This makes metrics correspond exactly with the areas under many cost curves, such as the optimal cost curves (Drummond and Holte, 2006), the Brier curves (Hernández-Orallo et al., 2011) or the rate-driven/Kendall curves (Hernández-Orallo et al., 2012).
- The uniform distribution is a reasonable choice if we want a model to behave well in a wide range of situations, from high costs for false positives to the other extreme. In this sense, it gives more relevance to models which perform well when the cost matrices or the class proportions are highly imbalanced.
- Most of the connections with the existing metrics are obtained with this distribution and not with others, which is informative about what the metrics implicitly assume (if understood as measures of expected loss).

Many expressions in this paper can be fine-tuned with other distributions, such as the Beta distribution $\beta(2,2)$, as suggested by Hand (2009), or using imbalance (Hand, 2010). However, it is the uniform distribution which leads us to many well-known evaluation metrics.

3. Expected Loss for Fixed-Threshold Classifiers

The easiest way to choose the threshold is to set it to a pre-defined value t_{fixed}, independently from the model and also from the operating condition. This is, in fact, what many classifiers do (e.g., Naive Bayes chooses $t_{\text{fixed}} = 0.5$ independently from the model and independently from the operating condition). We will see the straightforward result that this threshold choice method corresponds to 0-1 loss. Part of these results will be useful to better understand some other threshold choice methods.

Definition 4 *The* score-fixed threshold choice method *is defined as follows:*

$$T_c^{sf}[t](c) \triangleq T_z^{sf}[t](z) \triangleq t. \tag{6}$$

This choice has been criticised in two ways, but is still frequently used. Firstly, choosing 0.5 as a threshold is not generally the best choice even for balanced data sets or for applications where the test distribution is equal to the training distribution (see, for example, the work of Lachiche and Flach, 2003 on how to get much more from a Bayes classifier by simply changing the threshold). Secondly, even if we are able to find a better value than 0.5, this does not mean that this value is best for every skew or cost proportion—this is precisely one of the reasons why ROC analysis is used (Provost and Fawcett, 2001). Only when we know the deployment operating condition at evaluation time is it reasonable to fix the threshold according to this information. So either by common choice or because we have this latter case, consider then that we are going to use the same threshold t independently of skews or cost proportions. Given this threshold choice method, then the question is: *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?* This is what we answer below.

If we plug T_c^{sf} (Equation 6) into the general formula of the expected loss for a range of cost proportions (Equation 4) we have:

$$L_c^{sf}(t) \triangleq \int_0^1 Q_c(T_c^{sf}[t](c);c)w_c(c)dc.$$

We obtain the following straightforward result.

Theorem 5 *If a classifier sets the decision threshold at a fixed value t irrespective of the operating condition or the model, then expected loss for any cost distribution w_c is given by:*

$$L_c^{sf}(t) = 2\mathbb{E}_{w_c}\{c\}\,(1-Acc(t)) + 4\pi_1 F_1(t)\left(\frac{1}{2} - \mathbb{E}_{w_c}\{c\}\right).$$

Proof

$$\begin{aligned}
L_c^{sf}(t) &= \int_0^1 Q_c(T_c^{sf}[t](c);c)w_c(c)dc = \int_0^1 Q_c(t;c)w_c(c)dc \\
&= \int_0^1 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\}w_c(c)dc \\
&= 2\pi_0(1-F_0(t))\int_0^1 cw_c(c)dc + 2\pi_1 F_1(t)\int_0^1 (1-c)w_c(c)dc \\
&= 2\pi_0(1-F_0(t))\mathbb{E}_{w_c}\{c\} + 2\pi_1 F_1(t)(1-\mathbb{E}_{w_c}\{c\}) \\
&= 2\pi_0(1-F_0(t))\mathbb{E}_{w_c}\{c\} + 2\pi_1 F_1(t)\left(\mathbb{E}_{w_c}\{c\} + 2\left(\frac{1}{2} - \mathbb{E}_{w_c}\{c\}\right)\right) \\
&= 2\mathbb{E}_{w_c}\{c\}\,(\pi_0(1-F_0(t)) + \pi_1 F_1(t)) + 4\pi_1 F_1(t)\left(\frac{1}{2} - \mathbb{E}_{w_c}\{c\}\right) \\
&= 2\mathbb{E}_{w_c}\{c\}\,(1-Acc(t)) + 4\pi_1 F_1(t)\left(\frac{1}{2} - \mathbb{E}_{w_c}\{c\}\right).
\end{aligned}$$

■

This gives an expression of expected loss which depends on error rate and false positive rate at t and the expected value for the distribution of costs.[2] Similarly, if we plug T_z^{sf} (Equation 6) into the general formula of the expected loss for a range of skews (Equation 5) we have:

$$L_z^{sf}(t) \triangleq \int_0^1 Q_z(T_z^{sf}[t](z);z)w_z(z)dz.$$

Using Lemma 2 we obtain the equivalent result for skews:

$$L_z^{sf}(t) = 2\mathbb{E}_{w_z}\{z\}\,(1-uAcc(t)) + 2F_1(t)\left(\frac{1}{2} - \mathbb{E}_{w_z}\{z\}\right).$$

Corollary 6 *If a classifier sets the decision threshold at a fixed value irrespective of the operating condition or the model, then expected loss under a distribution of cost proportions w_c with expected value $\mathbb{E}_{w_c}\{c\} = 1/2$ is equal to the error rate at that decision threshold.*

$$L^{sf}_{\mathbb{E}\{c\}=1/2}(t) = \pi_0(1-F_0(t)) + \pi_1 F_1(t) = 1 - Acc(t).$$

Using Lemma 2 we obtain the equivalent result for skews:

$$L^{sf}_{\mathbb{E}\{z\}=1/2}(t) = (1-F_0(t))/2 + F_1(t)/2 = 1 - uAcc(t).$$

So the expected loss under a distribution of cost proportions with mean 1/2 for the *score-fixed threshold choice method* is the error rate of the classifier at that threshold. Clearly, a uniform distribution is a special case, but the result also applies to, for instance, a symmetric Beta distribution centered at 1/2. That means that accuracy can be seen as a measure of classification performance in a range of cost proportions when we choose a fixed threshold. This interpretation is reasonable, since accuracy is a performance metric which is typically applied to classifiers (where the threshold is fixed) and not to models outputting scores. This is exactly what we did in Table 2. We calculated the expected loss for the fixed threshold at 0.5 for a uniform distribution of cost proportions, and we obtained $1 - Acc = 0.51$ and 0.375 for models A and B respectively.

The previous results show that 0-1 losses are appropriate to evaluate models in a range of operating conditions if the threshold is fixed for all of them and we do not have any information about a possible asymmetry in the cost matrix at deployment time. In other words, accuracy and unweighted accuracy can be the right performance metrics for classifiers even in a cost-sensitive learning scenario. The situation occurs when one assumes a particular operating condition at evaluation time while the classifier has to deal with a range of operating conditions in deployment time.

In order to prepare for later results we also define a particular way of setting a fixed classification threshold, namely to achieve a particular predicted positive rate. One could say that such a method *quantifies* the proportion of positive predictions made by the classifier. For example, we could say that our threshold is fixed to achieve a rate of 30% positive predictions and the rest negatives. This

2. As mentioned above, the value of t is usually calculated disregarding the information (if any) about the operating condition, and frequently set to 0.5. In fact, this threshold choice method is called 'fixed' because of this. However, we can estimate and fix the value of t by taking the expected value for the operating condition $\mathbb{E}_{w_c}\{c\}$ into account, if we have some information about the *distribution* w_c. For instance, we may choose $t = \mathbb{E}_{w_c}\{c\}$ or we may choose the value of t which minimises the expression of expected loss in Theorem 5.

of course involves ranking the examples by their scores and setting a cutting point at the appropriate position, something which is frequent in 'screening' applications (Murphy et al., 1987; Wieand et al., 1989).

Let us denote the predicted positive rate at threshold t as $R(t) = \pi_0 F_0(t) + \pi_1 F_1(t)$. Then,

Definition 7 *If R is invertible, then we define the* rate-fixed threshold choice method *for rate r as:*

$$T_c^{rf}[r](c) \triangleq R^{-1}(r).$$

Similarly to the cost case, the rate-fixed threshold choice method for skews, assuming R is invertible, is defined as:

$$T_z^{rf}[r](z) \triangleq R_z^{-1}(r).$$

where $R_z(t) = F_0(t)/2 + F_1(t)/2$.

If R is not invertible, it has plateaus and so does R. This can be handled by deriving t from the centroid of a plateau. Nonetheless, in what follows, we will explicitly state when the invertibility of R is necessary. The corresponding expected loss for cost proportions is

$$L_c^{rf} \triangleq \int_0^1 Q_c(T_c^{rf}[r](c); c) w_c(c) dc = \int_0^1 Q_c(R^{-1}(r); c) w_c(c) dc.$$

As already mentioned, the notion of setting a threshold based on a rate is typically seen in screening applications but it also closely related to the task of class prevalence estimation (Neyman, 1938; Tenenbein, 1970; Alonzo et al., 2003), which is also known as quantification in machine learning and data mining (Forman, 2008; Bella et al., 2010). The goal of this task is to correctly estimate the proportion for each of the classes. This threshold choice method allows the user to set the quantity of positives, which might be known (from a sample of the test) or can be estimated using a quantification method. In fact, some quantification methods can be seen as methods to determine an absolute fixed threshold t that ensures a correct proportion for the test set. Fortunately, it is immediate to get the threshold which produces a rate; it can just be derived by sorting the examples by their scores and placing the cutpoint where the rate equals the rank divided by the number of examples (e.g., if we have n examples, the cutpoint i makes $r = i/n$).

4. Threshold Choice Methods Using Scores

In the previous section we looked at accuracy and error rate as performance metrics for classifiers and gave their interpretation as expected losses. In this and the following sections we consider performance metrics for models that do not require fixing a threshold choice method in advance. Such metrics include *AUC* which evaluates ranking performance and the Brier score or mean squared error which evaluates the quality of probability estimates. We will deal with the latter in this section. For the rest of this section, we will therefore assume that scores range between 0 and 1 and represent posterior probabilities for class 1, unless otherwise stated. This means that we can sample thresholds uniformly or derive them from the operating condition. We first introduce two performance metrics that are applicable to probabilistic scores.

The Brier score is a well-known performance metric for probabilistic models. It is an alternative name for the Mean Squared Error or MSE loss (Brier, 1950), especially for binary classification.

Definition 8 *The Brier score, BS, is defined as follows:*

$$BS \triangleq \pi_0 BS_0 + \pi_1 BS_1.$$

where the partial expressions for the positive and negative class are given by:

$$BS_0 \triangleq \int_0^1 s^2 f_0(s) ds.$$
$$BS_1 \triangleq \int_0^1 (1-s)^2 f_1(s) ds.$$

From here, we can define a prior-independent version of the Brier score (or an unweighted Brier score) as follows:

$$uBS \triangleq \frac{BS_0 + BS_1}{2}.$$

The Mean Absolute Error (*MAE*) is another simple performance metric which has been rediscovered many times under different names.

Definition 9 *The Mean Absolute Error, MAE, is defined as follows:*

$$MAE \triangleq \pi_0 MAE_0 + \pi_1 MAE_1.$$

where the partial expressions for the positive and negative class are given by:

$$MAE_0 \triangleq \int_0^1 s f_0(s) ds = \bar{s}_0.$$
$$MAE_1 \triangleq \int_0^1 (1-s) f_1(s) ds = 1 - \bar{s}_1.$$

We can define an unweighted MAE as follows:

$$uMAE \triangleq \frac{MAE_0 + MAE_1}{2} = \frac{\bar{s}_0 + (1-\bar{s}_1)}{2}.$$

It can be shown that *MAE* is equivalent to the Mean Probability Rate (MPR) (Lebanon and Lafferty, 2002) for discrete classification (Ferri et al., 2009).

4.1 The Score-Uniform Threshold Choice Method Leads to MAE

We now demonstrate how varying a model's threshold leads to an expected loss that is different from accuracy. First, we explore a threshold choice method which considers that we have no information at all about the operating condition, neither at evaluation time nor at deployment time. We just employ the interval between the maximum and minimum value of the scores, and we randomly select the threshold using a uniform distribution over this interval. It can be argued that this threshold choice method is unrealistic, because we almost always have some information about the operating condition, especially at deployment time. A possible interpretation is that this threshold choice method is useful to make a *worst-case* evaluation. In other words, expected loss using this method gives a robust assessment for situations where the information about the operating condition is not only unavailable, but maybe unrealiable or even malicious. So what we show next is that there are evaluation metrics which can be expressed as an expected loss under these assumptions, adding support to the idea that the metrics related to this threshold choice method are blind to (or unaware of) any cost information.

Definition 10 *Assuming a model's scores are expressed on a bounded scale* $[l,u]$*, the* score-uniform threshold choice method *is defined as follows:*

$$T_c^{su}(c) \triangleq T_z^{su}(z) \triangleq T_c^{sf}[U_{l,u}](c).$$

Given this threshold choice method, then the question is: *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?*

Theorem 11 *Assuming probabilistic scores and the score-uniform threshold choice method, expected loss under a distribution of cost proportions* w_c *is equal to:*

$$L_c^{su} = 2\{\mathbb{E}_{w_c}\{c\}\pi_0(\bar{s}_0) + (1-\mathbb{E}_{w_c}\{c\})\pi_1(1-\bar{s}_1)\}.$$

Proof First we derive Q_c:

$$\begin{aligned} Q_c(T_c^{su}(c);c) &= Q_c(T_c^{sf}[U_{l,u}](c);c) = \int_l^u Q_c(T_c^{sf}[t](c);c)\frac{1}{u-l}dt \\ &= \frac{1}{u-l}\int_l^u Q_c(t;c)dt = \frac{1}{u-l}\int_l^u 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\}dt \\ &= 2\frac{c\pi_0(\bar{s}_0 - l) + (1-c)\pi_1(u-\bar{s}_1)}{(u-l)}. \end{aligned}$$

The last step makes use of the following useful property.

$$\int_l^u F_k(t)dt = [tF_k(t)]_l^u - \int_l^u t f_k(t)dt = uF_k(u) - lF_k(l) - \bar{s}_k = u - \bar{s}_k.$$

Setting $l=0$ and $u=1$ for probabilistic scores, we obtain the final result:

$$Q_c(T_c^{su}(c);c) \quad = 2\{c\pi_0(\bar{s}_0) + (1-c)\pi_1(1-\bar{s}_1)\}.$$

And now, we calculate the expected loss for the distribution $w_c(c)$.

$$\begin{aligned} L_c^{su} &= \int_0^1 Q_c(T_c^{su}(c);c)w_c(c)dc \\ &= \int_0^1 2\{c\pi_0(\bar{s}_0) + (1-c)\pi_1(1-\bar{s}_1)\}w_c(c)dc \\ &= 2\{\mathbb{E}_{w_c}\{c\}\pi_0(\bar{s}_0) + (1-\mathbb{E}_{w_c}\{c\})\pi_1(1-\bar{s}_1)\}. \end{aligned}$$

■

Corollary 12 *Assuming probabilistic scores and the score-uniform threshold choice method, expected loss under a distribution of cost proportions* w_c *with expected value* $\mathbb{E}_{w_c}\{c\} = 1/2$ *is equal to the model's mean absolute error.*

$$L^{su}_{\mathbb{E}\{c\}=1/2} = \pi_0\bar{s}_0 + \pi_1(1-\bar{s}_1) = MAE.$$

This gives a baseline loss if we choose thresholds randomly and independently of the model. Using Lemma 2 we obtain the equivalent result for skews:

$$L^{su}_{\mathbb{E}\{z\}=1/2} \quad = \quad \frac{\bar{s}_0 + (1-\bar{s}_1)}{2} = uMAE.$$

4.2 The Score-Driven Threshold Choice Method Leads to the Brier Score

We will now consider the first threshold choice method to take the operating condition into account. Since we are dealing with probabilistic scores, this method simply sets the threshold equal to the operating condition (cost proportion or skew). This is a natural criterion as it has been used especially when the model is a probability estimator and we expect to have perfect information about the operating condition at deployment time. In fact, this is a direct choice when working with proper scoring rules, since when rules are proper, scores are assumed to be a probabilistic assessment. The use of this threshold choice method can be traced back to Murphy (1966) and, perhaps, implicitly, much earlier. More recently, and in a different context from proper scoring rules, Drummond and Holte (2006) say "the performance independent criterion, in this case, is to set the threshold to correspond to the operating conditions. For example, if $PC(+) = 0.2$ the Naive Bayes threshold is set to 0.2". The term $PC(+)$ is equivalent to our 'skew'.

Definition 13 *Assuming the model's scores are expressed on a probability scale* $[0,1]$*, the* score-driven threshold choice method *is defined for cost proportions as follows:*

$$T_c^{sd}(c) \triangleq c \tag{7}$$

and for skews as

$$T_z^{sd}(z) \triangleq z.$$

Given this threshold choice method, then the question is: *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?* This is what we answer below.

Theorem 14 (Hernández-Orallo et al., 2011) *Assuming probabilistic scores and the score-driven threshold choice method, expected loss under a uniform distribution of cost proportions is equal to the model's Brier score.*

Proof If we plug T_c^{sd} (Equation 7) into the general formula of the expected loss (Equation 4) we have the expected score-driven loss:

$$L_c^{sd} \triangleq \int_0^1 Q_c(T_c^{sd}(c);c)w_c(c)dc = \int_0^1 Q_c(c;c)w_c(c)dc. \tag{8}$$

And if we use the uniform distribution and the definition of Q_c (Equation 3):

$$L_{U(c)}^{sd} = \int_0^1 Q_c(c;c)U(c)dc = \int_0^1 2\{c\pi_0(1-F_0(c)) + (1-c)\pi_1 F_1(c)\}dc. \tag{9}$$

In order to show this is equal to the Brier score, we expand the definition of BS_0 and BS_1 using integration by parts:

$$\begin{aligned} BS_0 &= \int_0^1 s^2 f_0(s)ds = \left[s^2 F_0(s)\right]_{s=0}^1 - \int_0^1 2sF_0(s)ds = 1 - \int_0^1 2sF_0(s)ds \\ &= \int_0^1 2sds - \int_0^1 2sF_0(s)ds = \int_0^1 2s(1-F_0(s))ds. \\ BS_1 &= \int_0^1 (1-s)^2 f_1(s)ds = \left[(1-s)^2 F_1(s)\right]_{s=0}^1 + \int_0^1 2(1-s)F_1(s)ds = \int_0^1 2(1-s)F_1(s)ds. \end{aligned}$$

Taking their weighted average, we obtain

$$BS = \pi_0 BS_0 + \pi_1 BS_1 = \int_0^1 \{\pi_0 2s(1-F_0(s)) + \pi_1 2(1-s)F_1(s)\}ds. \quad (10)$$

which, after reordering of terms and change of variable, is the same expression as Equation (9). ■

It is now clear why we just put the Brier score from Table 1 as the expected loss in Table 2. We calculated the expected loss for the score-driven threshold choice method for a uniform distribution of cost proportions as its Brier score.

Theorem 14 was obtained by Hernández-Orallo et al. (2011) (the threshold choice method there was called 'probabilistic') but it is not completely new in itself. Murphy (1966) found a similar relation to expected utility (in our notation, $-(1/4)PS + (1/2)(1+\pi_0)$, where the so-called probability score $PS = 2BS$). Apart from the sign (which is explained because Murphy works with utilities and we work with costs), the difference in the second constant term is explained because Murphy's utility (cost) model is based on a cost matrix where we have a cost for one of the classes (in meteorology the class 'protect') independently of whether we have a right or wrong prediction ('adverse' or 'good' weather). The only case in the matrix with a 0 cost is when we have 'good' weather and 'no protect'. It is interesting to see that the result only differs by a constant term, which supports the idea that whenever we can express the operating condition with a cost proportion or skew, the results will be portable to each situation with the inclusion of some constant terms (which are the same for all classifiers). In addition to this result, it is also worth mentioning another work by Murphy (1969) where he makes a general derivation for the Beta distribution.

After Murphy, in the last four decades, there has been extensive work on the so-called proper scoring rules, where several utility (cost) models have been used and several distributions for the cost have been used. This has led to relating Brier score (square loss), logarithmic loss, 0-1 loss and other losses which take the scores into account. For instance, Buja et al. (2005) give a comprehensive account of how all these losses can be obtained as special cases of the Beta distribution. The result given in Theorem 14 would be a particular case for the uniform distribution (which is a special case of the Beta distribution) and a variant of Murphy's results. In fact, the *BS* decomposition can also be connected to more general decompositions of Bregman divergences (Reid and Williamson, 2011). Nonetheless, it is important to remark that the results we have just obtained in Section 4.1 (and those we will get in Section 5) are new because they are not obtained by changing the cost distribution but rather by changing the threshold choice method. The threshold choice method used (the score-driven one) is not put into question in the area of proper scoring rules. But Theorem 14 can now be seen as a result which connects these two different dimensions: cost distribution and threshold choice method, so placing the Brier score at an even more predominant role.

Hernández-Orallo et al. (2011) derive an equivalent result using empirical distributions. In that paper we show how the loss can be plotted in cost space, leading to the *Brier curve* whose area underneath is the Brier score.

Finally, using skews we arrive at the prior-independent version of the Brier score.

Corollary 15 $L^{sd}_{U(z)} = uBS = (BS_0 + BS_1)/2$.

It is interesting to analyse the relation between $L^{su}_{U(c)}$ and $L^{sd}_{U(c)}$ (similarly between $L^{su}_{U(z)}$ and $L^{sd}_{U(z)}$). Since the former gives the *MAE* and the second gives the Brier score (which is the MSE),

from the definitions of *MAE* and Brier score, we get that, assuming scores are between 0 and 1:

$$MAE = L^{su}_{U(c)} \geq L^{sd}_{U(c)} = BS.$$

$$uMAE = L^{su}_{U(z)} \geq L^{sd}_{U(z)} = uBS.$$

Since *MAE* and *BS* have the same terms but the second squares them, and all the values which are squared are between 0 and 1, then the *BS* must be lower or equal. This is natural, since the expected loss is lower if we get reliable information about the operating condition at deployment time. So, the difference between the Brier score and *MAE* is precisely the gain we can get by having (and using) the information about the operating condition at deployment time. Notice that all this holds regardless of the quality of the probability estimates.

Finally, the difference between the results of Section 3 (Corollary 6) and these results fits well with a phenomenon which is observed when trying to optimise classification models: good probability estimation does not imply good classification and vice versa (see, for example, the work of Friedman, 1997). In the context of these results, we can re-interpret this phenomenon from a new perspective. The Brier score is seen as expected loss for the score-driven threshold choice method, while accuracy assumes a fixed threshold. The expected losses shown in Table 2 are a clear example of this.

5. Threshold Choice Methods Using Rates

We show in this section that *AUC* can be translated into expected loss for varying operating conditions in more than one way, depending on the threshold choice method used. We consider two threshold choice methods, where each of them sets the threshold to achieve a particular predicted positive rate: the rate-uniform method, which sets the rate in a uniform way; and the rate-driven method, which sets the rate equal to the operating condition. Some of these approaches have been used or mentioned in the literature, but choosing or ranging over sensitivity (or, complementary, specificity) instead of ranging over the *rate* (which is a weighted sum of sensitivity, that is, F_0, and $1-$ specificity, that is, F_1). For instance, Wieand et al. (1989) take a uniform distribution on a restricted range of sensitivities (or, similarly, specificities, Wieand et al., 1989). Also, Hand (2010) mentions that *AUC* can be seen as 'the mean specificity value, assuming a uniform distribution for the sensitivity'.

We recall the definition of a ROC curve and its area first.

Definition 16 *The ROC curve (Swets et al., 2000; Fawcett, 2006) is defined as a plot of $F_1(t)$ (i.e., false positive rate at decision threshold t) on the x-axis against $F_0(t)$ (true positive rate at t) on the y-axis, with both quantities monotonically non-decreasing with increasing t (remember that scores increase with $\hat{p}(1|x)$ and 1 stands for the negative class). The Area Under the ROC curve (AUC) is defined as:*

$$\begin{aligned} AUC &\triangleq \int_0^1 F_0(s)dF_1(s) = \int_{-\infty}^{+\infty} F_0(s)f_1(s)ds = \int_{-\infty}^{+\infty}\int_{-\infty}^{s} f_0(t)f_1(s)dtds \\ &= \int_0^1 (1-F_1(s))dF_0(s) = \int_{-\infty}^{+\infty} (1-F_1(s))f_0(s)ds = \int_{-\infty}^{+\infty}\int_{s}^{+\infty} f_1(t)f_0(s)dtds. \end{aligned}$$

Note that in this section scores are not necessarily assumed to be probability estimates and so s ranges from $-\infty$ to ∞.

5.1 The Rate Uniform Threshold Choice Method Leads to AUC

The rate-fixed threshold choice method places the threshold in such a way that a given predictive positive rate is achieved. However, this proportion may change, or we might not have reliable information about the operating condition *at deployment time*. An option in this case is to fix a predictive positive rate equal to 0.5 (predict exactly half of the examples as positive), which boils down to a special case of Theorem 5, but another option is to consider a non-deterministic choice or a distribution for this quantity. One natural choice can be a uniform distribution. This complete absence of information will hardly ever be the case, as we discussed for the score-uniform threshold choice method, but it is still instructive to explore what the outcome would be with this choice.

Definition 17 *The rate-uniform threshold choice method non-deterministically sets the threshold to achieve a uniformly randomly selected rate:*

$$T_c^{ru}(c) \triangleq T_c^{rf}[U_{0,1}](c).$$
$$T_z^{ru}(z) \triangleq T_z^{rf}[U_{0,1}](z).$$

In other words, it sets a relative quantity (from 0% positives to 100% positives) in a uniform way, and obtains the threshold from this uniform distribution over rates. Note that for a large number of examples, this is the same as defining a uniform distribution over examples or, alternatively, over cutpoints (between examples), as explored by Flach et al. (2011).

This threshold choice method is a generalisation of the rate-fixed threshold choice method which considers all the imbalances (class proportions) equally likely whenever we make a classification. It is important to clearly state that this makes the strong assumption that we will not have any information about the operating condition at deployment time.

As done before for other threshold choice methods, we analyse the question: given this threshold choice method, *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?*

The corresponding expected loss for cost proportions is (assuming R is invertible)

$$L_c^{ru} \triangleq \int_0^1 Q_c(T_c^{ru}(c);c)w_c(c)dc = \int_0^1 \int_0^1 Q_c(R^{-1}(r);c)U(r)w_c(c)dr\,dc.$$

We then have the following result.

Theorem 18 *Assuming the rate-uniform threshold choice method and invertible R, expected loss under a distribution of cost proportions w_c decreases linearly with AUC as follows:*

$$L_c^{ru} = \pi_0\pi_1(1-2AUC) + \pi_0\mathbb{E}_{w_c}\{c\} + \pi_1(1-\mathbb{E}_{w_c}\{c\}).$$

Proof First of all we note that $r = R(t)$ and hence $U(r)dr = R'(t)dt = \{\pi_0 f_0(t) + \pi_1 f_1(t)\}dt$. Under the same change of variable, $Q_c(R^{-1}(r);c) = Q_c(t;c) = 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\}$. Hence:

$$\begin{aligned}
L_c^{ru} &= \int_0^1 \int_{-\infty}^{\infty} \{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\} w_c(c) \{\pi_0 f_0(t) + \pi_1 f_1(t)\} dt\, dc \\
&= \int_{-\infty}^{\infty} \int_0^1 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\} \{\pi_0 f_0(t) + \pi_1 f_1(t)\} w_c(c) dc\, dt \\
&= \int_{-\infty}^{\infty} 2\{\mathbb{E}_{w_c}\{c\}\pi_0(1-F_0(t)) + (1-\mathbb{E}_{w_c}\{c\})\pi_1 F_1(t)\} \{\pi_0 f_0(t) + \pi_1 f_1(t)\} dt \\
&= 2\pi_0\pi_1 \mathbb{E}_{w_c}\{c\} \int_{-\infty}^{\infty} (1-F_0(t)) f_1(t) dt + 2\pi_0\pi_1 (1-\mathbb{E}_{w_c}\{c\}) \int_{-\infty}^{\infty} F_1(t) f_0(t) dt \\
&+ 2\pi_0^2 \mathbb{E}_{w_c}\{c\} \int_{-\infty}^{\infty} (1-F_0(t)) f_0(t)\, dt + 2\pi_1^2 (1-\mathbb{E}_{w_c}\{c\}) \int_{-\infty}^{\infty} F_1(t) f_1(t)\, dt.
\end{aligned}$$

The first two integrals in this last expression are both equal to $1 - AUC$. The remaining two integrals reduce to a constant:

$$\int_{-\infty}^{\infty} (1-F_0(t)) f_0(t)\, dt = -\int_1^0 (1-F_0(t))\, d(1-F_0(t)) = 1/2.$$

$$\int_{-\infty}^{\infty} F_1(t) f_1(t)\, dt = \int_0^1 F_1(t)\, dF_1(t) = 1/2.$$

Putting everything together we obtain

$$\begin{aligned}
L_c^{ru} &= 2\pi_0\pi_1 \mathbb{E}_{w_c}\{c\}(1-AUC) + 2\pi_0\pi_1 (1-\mathbb{E}_{w_c}\{c\})(1-AUC) + \pi_0^2 \mathbb{E}_{w_c}\{c\} + \pi_1^2 (1-\mathbb{E}_{w_c}\{c\}) \\
&= 2\pi_0\pi_1 (1-AUC) + \pi_0^2 \mathbb{E}_{w_c}\{c\} + \pi_1^2 (1-\mathbb{E}_{w_c}\{c\}) \\
&= \pi_0\pi_1 (1-2AUC) + \pi_0\pi_1 + \pi_0^2 \mathbb{E}_{w_c}\{c\} + \pi_1^2 (1-\mathbb{E}_{w_c}\{c\}) \\
&= \pi_0\pi_1 (1-2AUC) + \pi_0\pi_1 \mathbb{E}_{w_c}\{c\} + \pi_0^2 \mathbb{E}_{w_c}\{c\} + \pi_0\pi_1 (1-\mathbb{E}_{w_c}\{c\}) + \pi_1^2 (1-\mathbb{E}_{w_c}\{c\})
\end{aligned}$$

and the result follows. ■

The following two results were originally obtained by Flach, Hernández-Orallo, and Ferri (2011) for the special case of uniform cost and skew distributions. We are grateful to David Hand for suggesting that the earlier results might be generalised.

Corollary 19 *Assuming the rate-uniform threshold choice method, invertible R, and a distribution of cost proportions w_c with expected value $\mathbb{E}_{w_c}\{c\} = 1/2$, expected loss decreases linearly with AUC as follows:*

$$L^{ru}_{\mathbb{E}\{c\}=1/2} = \pi_0\pi_1 (1-2AUC) + 1/2.$$

Corollary 20 *For any distribution of skews w_z, assuming the rate-uniform threshold choice method and invertible R, expected loss decreases linearly with AUC as follows:*

$$L_z^{ru} = (1-2AUC)/4 + 1/2.$$

Proof By assuming a uniform class distribution in Theorem 18 we obtain:

$$L_c^{ru} = \frac{1}{2}\frac{1}{2}(1-2AUC) + \frac{1}{2}\mathbb{E}_{w_c}\{c\} + \frac{1}{2}(1-\mathbb{E}_{w_c}\{c\}) = (1-2AUC)/4 + 1/2.$$

By Lemma 2 this is equal to L_z^{ru}. ■

Notice that Corollary 20 does not make any assumption about the expected value of w_z, and in that sense is more general than Corollary 19 for cost proportions. We see that expected loss for uniform skew ranges from 1/4 for a perfect ranker that is harmed by sub-optimal threshold choices, to 3/4 for the worst possible ranker that puts positives and negatives the wrong way round, yet gains some performance by putting the threshold at or close to one of the extremes.

Intuitively, a result like Corollary 20 can be understood as follows. Setting a randomly sampled rate is equivalent to setting the decision threshold to the score of a randomly sampled example. With probability π_0 we select a positive and with probability π_1 we select a negative. If we select a positive, then the expected true positive rate is $1/2$ (as on average we select the middle one); and the expected false positive rate is $1-AUC$ (as one interpretation of *AUC* is the expected proportion of negatives ranked correctly wrt. a random positive). Similarly, if we select a negative then the expected true positive rate is *AUC* and the expected false positive rate is $1/2$. Put together, the expected true positive rate is $\pi_0/2+\pi_1 AUC$ and the expected false positive rate is $\pi_1/2+\pi_0(1-AUC)$. The proportion of true positives among all examples is thus

$$\pi_0\,(\pi_0/2+\pi_1 AUC) = \frac{\pi_0^2}{2} + \pi_0\pi_1 AUC$$

and the proportion of false positives is

$$\pi_1\,(\pi_1/2+\pi_0(1-AUC)) = \frac{\pi_1^2}{2} + \pi_0\pi_1(1-AUC).$$

We can summarise these expectations in the following contingency table (all numbers are proportions relative to the total number of examples):

	Predicted +	Predicted −	
Actual +	$\pi_0^2/2+\pi_0\pi_1 AUC$	$\pi_0^2/2+\pi_0\pi_1(1-AUC)$	π_0
Actual −	$\pi_1^2/2+\pi_0\pi_1(1-AUC)$	$\pi_1^2/2+\pi_0\pi_1 AUC$	π_1
	$1/2$	$1/2$	1

The column totals are, of course, as expected: if we randomly select an example to split on, then the expected split is in the middle.

While in this paper we concentrate on the case where we have access to population densities $f_k(s)$ and distribution functions $F_k(t)$, in practice we have to work with empirical estimates. Flach et al. (2011) provides an alternative formulation of the main results in this section, relating empirical loss to the *AUC* of the empirical ROC curve. For instance, the expected loss for uniform skew and uniform instance selection is calculated by Flach et al. (2011) to be $\left(\frac{n}{n+1}\right)\frac{1-2AUC}{4}+\frac{1}{2}$, showing that for smaller samples the reduction in loss due to *AUC* is somewhat smaller.

5.2 The Rate-Driven Threshold Choice Method Leads to AUC

Naturally, if we can have precise information of the operating condition at deployment time, we can use the information about the skew or cost to adjust the rate of positives and negatives to that proportion. This leads to a new threshold selection method: if we are given skew (or cost proportion) z (or c), we choose the threshold t in such a way that we get a proportion of z (or c) positives. This is an elaboration of the rate-fixed threshold choice method which *does* take the operating condition into account.

Definition 21 *The* rate-driven threshold choice method *for cost proportions is defined as*

$$T_c^{rd}(c) \triangleq T_c^{rf}[c](c) = R^{-1}(c). \tag{11}$$

The rate-driven threshold choice method for skews is defined as

$$T_z^{rd}(z) \triangleq T_z^{rf}[z](z) = R_z^{-1}(z).$$

Given this threshold choice method, the question is again: *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?* This is what we answer below.

If we plug T_c^{rd} (Equation 11) into the general formula of the expected loss for a range of cost proportions (Equation 4) we have:

$$L_c^{rd} \triangleq \int_0^1 Q_c(T_c^{rd}(c);c)w_c(c)dc.$$

And now, from this definition, if we use the uniform distribution for $w_c(c)$, we obtain this new result.

Theorem 22 *Expected loss for uniform cost proportions using the rate-driven threshold choice method is linearly related to AUC as follows:*

$$L_{U(c)}^{rd} = \pi_1\pi_0(1-2AUC) + 1/3.$$

Proof

$$\begin{aligned} L_{U(c)}^{rd} &= \int_0^1 Q_c(T_c^{rd}(c);c)U(c)dc = \int_0^1 Q_c(R^{-1}(c);c)dc \\ &= 2\int_0^1 \{c\pi_0(1-F_0(R^{-1}(c))) + (1-c)\pi_1F_1(R^{-1}(c))\}dc \\ &= 2\int_0^1 \{c\pi_0 - c\pi_0F_0(R^{-1}(c))) + \pi_1F_1(R^{-}1(c)) - c\pi_1F_1(R^{-1}(c))\}dc. \end{aligned}$$

Since $\pi_0F_0(R^{-}1(c))) + \pi_1F_1(R^{-1}(c)) = R(R^{-1}(c)) = c$,

$$\begin{aligned} L_{U(c)}^{rd} &= 2\int_0^1 \{c\pi_0 - c^2 + \pi_1F_1(R^{-1}(c))\}dc \\ &= \pi_0 - \frac{2}{3} + 2\pi_1\int_0^1 F_1(R^{-1}(c))dc. \end{aligned}$$

Taking the rightmost term and using the change of variable $R^{-1}(c) = t$ we have $c = R(t)$ and hence $dc = R'(t)dt = \{\pi_0 f_0(t) + \pi_1 f_1(t)\}dt = R'(t)dt$, and thus this term is rewritten as

$$\begin{aligned} 2\pi_1 \int_0^1 F_1(R^{-1}(c))dc &= 2\pi_1 \int_\infty^\infty F_1(t)\{\pi_0 f_0(t) + \pi_1 f_1(t)\}dt \\ &= 2\pi_1\pi_0 \int_0^1 F_1(t)dF_0(t) + 2\pi_1^2 \int_0^1 F_1(t)dF_1(t) \\ &= 2\pi_1\pi_0(1-AUC) + 2\pi_1^2\frac{1}{2} = 2\pi_1\pi_0(1-AUC) + \pi_1(1-\pi_0). \end{aligned}$$

Putting everything together we have:

$$\begin{aligned} L^{rd}_{U(c)} &= \pi_0 - \frac{2}{3} + 2\pi_1\pi_0(1-AUC) + \pi_1(1-\pi_0) \\ &= \frac{1}{3} + \pi_1\pi_0(1-2AUC). \end{aligned}$$

■

Now we can unveil and understand how we obtained the results for the expected loss in Table 2 for the rate-driven method. We just took the *AUC* of the models and applied the previous formula: $\pi_1\pi_0(1-2AUC) + \frac{1}{3}$.

Corollary 23 *Expected loss for uniform skews using the rate-driven threshold choice method is linearly related to AUC as follows:*

$$L^{rd}_{U(z)} = (1-2AUC)/4 + 1/3.$$

If we compare Corollary 20 with Corollary 23, we see that $L^{ru}_{U(z)} > L^{rd}_{U(z)}$, more precisely:

$$L^{ru}_{U(z)} = (1-2AUC)/4 + 1/2 = L^{rd}_{U(z)} + 1/6.$$

So we see that taking the operating condition into account when choosing thresholds based on rates reduces the expected loss with $1/6$, *regardless of the quality of the model* as measured by *AUC*. This term is clearly not negligible and demonstrates that the rate-driven threshold choice method is superior to the rate-uniform method. Figure 3 illustrates this. Logically, $L^{rd}_{U(c)}$ and $L^{rd}_{U(z)}$ work upon information about the operating condition at deployment time, while $L^{ru}_{U(c)}$ and $L^{ru}_{U(z)}$ may be suited when this information is unavailable or unreliable.

6. The Optimal Threshold Choice Method

The last threshold choice method we investigate is based on the optimistic assumption that (1) we are having complete information about the operating condition (class proportions and costs) at deployment time and (2) we are able to use that information (also at deployment time) to choose the threshold that will minimise the loss using the current model. ROC analysis is precisely based on these two points since we can calculate the threshold which gives the smallest loss by using the skew and the convex hull.

This threshold choice method, denoted by T^o_c, is defined as follows:

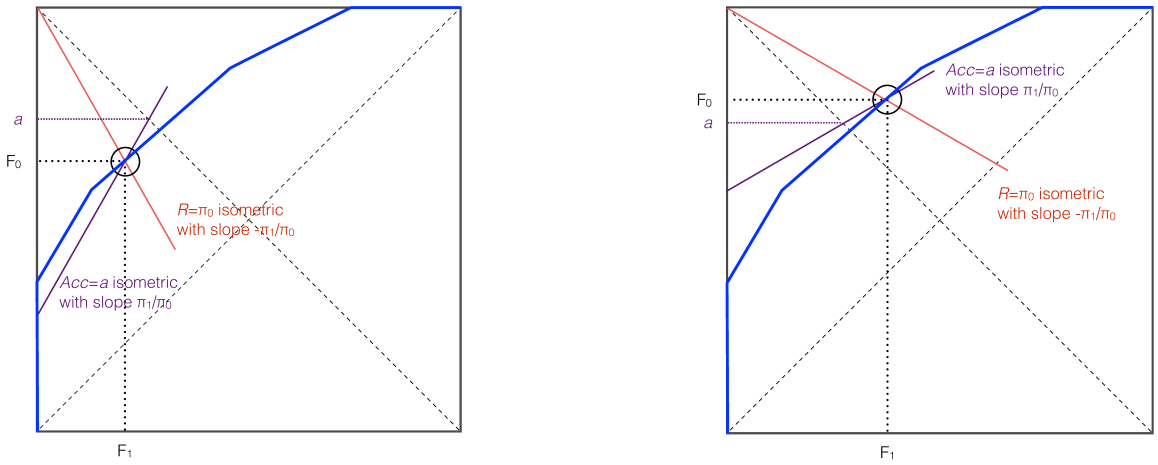


Figure 3: Illustration of the rate-driven threshold choice method. We assume uniform misclassification costs ($c_0 = c_1 = 1$), and hence skew is equal to the proportion of positives ($z = \pi_0$). The majority class is class 1 on the left and class 0 on the right. Unlike the rate-uniform method, the rate-driven method is able to take advantage of knowing the majority class, leading to a lower expected loss.

Definition 24 *The optimal threshold choice method is defined as:*

$$T_c^o(c) \triangleq \arg\min_t \{Q_c(t;c)\} = \arg\min_t 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\} \tag{12}$$

and similarly for skews:

$$T_z^o(z) \triangleq \arg\min_t \{Q_z(t;z)\}.$$

Note that in both cases, the arg min will typically give a range (interval) of values which give the same optimal value. So these methods can be considered non-deterministic. This threshold choice method is analysed by Fawcett and Provost (1997), and used by Drummond and Holte (2000, 2006) for defining their cost curves and by Hand (2009) to define a new performance metric.

If we plug Equations (12) and (3) into Equation (4) using a uniform distribution for cost proportions, we get:

$$\begin{aligned} L^o_{U(c)} &= \int_0^1 Q_c(\arg\min_t\{Q_c(t,c)\};c)dc = \int_0^1 \min_t\{Q_c(t;c)\}dc \\ &= \int_0^1 \min_t\{2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\}dc. \end{aligned} \tag{13}$$

The connection with the convex hull of a ROC curve (ROCCH) is straightforward. The convex hull is a construction over the ROC curve in such a way that all the points on the convex hull have minimum loss for some choice of c or z. This means that we restrict attention to the *optimal* threshold for a given cost proportion c, as derived from Equation (12).

6.1 Convexification

We can give a corresponding, and more formal, definition of the convex hull as derived from the score distributions. First, we need a more precise definition of a convex model. For that, we rely on the ROC curve, and we use the slope of the curve, defined as usual:

$$slope(T) = \frac{f_0(T)}{f_1(T)}.$$

A related expression we will also use is:

$$c(T) = \frac{\pi_1 f_1(T)}{\pi_0 f_0(T) + \pi_1 f_1(T)}.$$

Sometimes we will use subindices for $c(T)$ depending on the model we are using. In this way, we have, $\frac{\pi_0}{\pi_1} slope(T) = \frac{\pi_0 f_0(T)}{\pi_1 f_1(T)} = \frac{1}{c(T)} - 1$.

Definition 25 (Convex model) *A model m is convex, if for every threshold T, we have that $c(T)$ is non-decreasing (or, equivalently, slope(T) is non-increasing).*

In order to make any model convex, it is not sufficient to repair local concavities, we need to calculate the convex hull. This is clear if we categorise the types of segments. Some threshold values t will never minimise $Q_c(t;c) = 2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\}$ for any value of c. These values will be in one or more intervals of which only the end points will minimise $Q_c(t;c)$ for some value of c. We will call these intervals *non-hull intervals*, and all the rest will be referred to as *hull intervals*. It clearly holds that hull intervals are convex. Non-hull intervals may contain convex and concave subintervals.

From here, a definition of convex hull for continuous distributions is given as follows:

Definition 26 (Convexification) *Let m be any model with score distributions $f_0(T)$ and $f_1(T)$. Define convexified score distributions $e_0(T)$ and $e_1(T)$ as follows.*

1. *For every hull interval $t_{i-1} \leq s \leq t_i$: $e_0(T) = f_0(T)$ and $e_1(T) = f_1(T)$.*
2. *For every non-hull interval $t_{j-1} \leq s \leq t_j$:*

$$e_0(T) = e_{0,j} = \frac{1}{t_j - t_{j-1}} \int_{t_{j-1}}^{t_j} f_0(T) dT.$$
$$e_1(T) = e_{1,j} = \frac{1}{t_j - t_{j-1}} \int_{t_{j-1}}^{t_j} f_1(T) dT.$$

The function Conv *returns the model* Conv(m) *defined by the score distributions $e_0(T)$ and $e_1(T)$.*

We can also define the cumulative distributions $E_x(t) = \int_0^t e_x(T) dT$, where x represents either 0 or 1. By construction we have that for every interval $[t_{j-1}, t_j]$ identified above:

$$[E_x(t)]_{t_{j-1}}^{t_j} = \int_{t_{j-1}}^{t_j} e_x(T) dT = (t_j - t_{j-1}) e_{x,j} = \int_{t_{j-1}}^{t_j} f_x(T) dT = [F_x(t)]_{t_{j-1}}^{t_j} \quad (14)$$

and so the convexified score distributions are proper distributions. Furthermore, since the new score distributions are constant in the convexified intervals—and hence monotonically non-decreasing for the new $c(T)$, denoted by $c_{\text{Conv}(m)}(T)$—so is

$$c_{\text{Conv}(m)}(T) = c_j = \frac{\pi_1 e_{1,j}}{\pi_0 e_{0,j} + \pi_1 e_{1,j}}.$$

It follows that $\text{Conv}(m)$ is everywhere convex. In addition,

Theorem 27 *Optimal loss is invariant under* Conv*, that is:* $L^o_{U(c)}(\text{Conv}(m)) = L^o_{U(c)}(m)$ *for every* m.

Proof By Equation (13) we have that optimal loss is:

$$L^o_{U(c)}(m) \quad = \quad \int_0^1 \min_t \{2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\} dc.$$

By definition, the hull intervals have not been modified by $\text{Conv}(m)$. Only the non-hull intervals have been modified. A non-hull interval was defined as those where there is no t which minimises $Q_c(t;c) = 2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\}$ for any value of c, and only the endpoints attained the minimum. Consequently, we only need to show that the new $e_0(T)$ and $e_1(T)$ do not introduce any new minima.

We now focus on each non-hull segment (t_{j-1}, t_j) using the definition of Conv. We only need to check the expression for the minimum:

$$\min_{t_j \le t \le t_{j-1}} \{2c\pi_0(1-E_0(t)) + 2(1-c)\pi_1 E_1(t)\}.$$

From Equation (14) we derive that $E_x(t) = E_x(t_{j-1}) + (t_j - t_{j-1})e_{x,j}$ inside the interval (they are straight lines in the ROC curve), and we can see that the expression to be minimised is constant (it does not depend on t). Since the end points were the old minima and were equal, we see that this expression cannot find new minima. ■

It is not difficult to see that if we plot $\text{Conv}(m)$ in the cost space defined by Drummond and Holte (2006) with $Q_z(t;z)$ on the y-axis against skew z on the x-axis, we have a cost curve. Its area is then the expected loss for the optimal threshold choice method. In other words, this is the area under the (optimal) cost curve. Similarly, the new performance metric H introduced by Hand (2009) is simply a rescaled version of the area under the optimal cost curve using the $\beta_{2,2}$ distribution instead of the $\beta_{1,1}$ (i.e., uniform) distribution, and using cost proportions instead of skews (so being dependent to class priors). This is further discussed by Flach et al. (2011). While all of these distributions are symmetric, the Beta distribution can be non-symmetric if required by a specific application. In fact, Hand and Anagnostopoulos (2012) suggest that the parameters of the distribution should be linked to class proportion π_0.

6.2 The Optimal Threshold Choice Method Leads to Refinement Loss

Once again, the question now must be stated clearly. Assume that the optimal threshold choice method is set as the method we will use for every application of our model. Furthermore, assume

that each and every application of the model is going to find the perfect threshold. Then, *if we must evaluate a model before application for a wide range of skews and cost proportions, which performance metric should be used?* In what follows, we will find the answer by relating this expected loss with a genuine performance metric: refinement loss. We will now introduce this performance metric.

The Brier score, being a sum of squared probabilistic residuals, can be decomposed in various ways. The most common decomposition of the Brier score is due to Murphy (1973) and decomposes the Brier score into Reliability, Resolution and Uncertainty. Frequently, the two latter components are joined together and the decomposition gives two terms: calibration loss and refinement loss.

This decomposition is usually applied to empirical distributions, requiring a binning of the scores. Scores are assumed to be probability estimates in the interval $[0,1]$. The decomposition is based on a partition $\mathcal{P}_D = \{b_j\}_{j=1..B}$ where D is the data set, B the number of bins, and each bin is denoted by $b_j \subset D$. Since it is a partition $\bigcup_{j=1}^{B} b_j = D$. With this partition the decompoistion is:

$$BS \approx CL^{\mathcal{P}_D} + RL^{\mathcal{P}_D} = \frac{1}{n}\sum_{j=1}^{B} |b_j| \left(s_{b_j} - y_{b_j}\right)^2 + \frac{1}{n}\sum_{j=1}^{B} |b_j| y_{b_j} \left(1 - y_{b_j}\right).$$

Here we use the notation $s_{b_j} = \frac{1}{|b_j|}\sum_{i \in b_j} s_i$ and $y_{b_j} = \frac{1}{|b_j|}\sum_{i \in b_j} y_i$ for the average predicted scores and the average actual classes respectively for bin b_j.

For many partitions the empirical decomposition is not exact. It is only exact for partitions which are coarser than the partition induced by the ROC curve (i.e., ties cannot be spread over different partitions), as shown by Flach and Matsubara (2007). We denote by CL^{ROC} and RL^{ROC} the calibration loss and the refinement loss, respectively, using the segments of the empirical ROC curve as bins. In this case, $BS = CL^{ROC} + RL^{ROC}$.

In this paper we will use a variant of the above decomposition based on the ROC convex hull of a model. In this decomposition, we take each bin as each segment in the convex hull. Naturally, the number of bins in this decomposition is lower or equal than the number of bins in the ROC decomposition. In fact, we may find different values of s_i in the same bin. In some way, we can think about this decomposition as an optimistic/optimal version of the ROC decomposition, as Flach and Matsubara (2007, Th. 3) show. We denote by CL^{ROCCH} and RL^{ROCCH} the calibration loss and the refinement loss, respectively, using the segments of the convex hull of the empirical ROC curve as bins (Flach et al., 2011).

We can define the same decomposition in continuous terms considering Definition 8. We can see that in the continuous case, the partition is irrelevant. Any partition will give the same result, since the composition of consecutive integrals is the same as the whole integral.

Theorem 28 *The continuous decomposition of the Brier Score, $BS = CL + RL$, is exact and gives CL and RL as follows.*

$$CL = \int_0^1 \frac{\left(s(\pi_0 f_0(s) + \pi_1 f_1(s)) - \pi_1 f_1(s)\right)^2}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds.$$

$$RL = \int_0^1 \frac{\pi_1 f_1(s) \pi_0 f_0(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds.$$

Proof

$$\begin{aligned}
BS &= \int_0^1 \left[s^2\pi_0 f_0(s) + (1-s)^2\pi_1 f_1(s)\right] ds \\
&= \int_0^1 \left[s^2(\pi_0 f_0(s) + \pi_1 f_1(s)) - 2s\pi_1 f_1(s) + \pi_1 f_1(s)\right] ds \\
&= \int_0^1 \frac{s^2(\pi_0 f_0(s) + \pi_1 f_1(s))^2 - 2s(\pi_0 f_0(s) + \pi_1 f_1(s))\pi_1 f_1(s) + \pi_1 f_1(s)(\pi_1 f_1(s) + \pi_0 f_0(s))}{(\pi_0 f_0(s) + \pi_1 f_1(s))} ds \\
&= \int_0^1 \frac{(s(\pi_0 f_0(s) + \pi_1 f_1(s)) - \pi_1 f_1(s))^2 + \pi_1 f_1(s)\pi_0 f_0(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds \\
&= \int_0^1 \frac{(s(\pi_0 f_0(s) + \pi_1 f_1(s)) - \pi_1 f_1(s))^2}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds + \int_0^1 \frac{\pi_1 f_1(s)\pi_0 f_0(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds.
\end{aligned}$$

■

This proof keeps the integral from start to end. That means that the decomposition is not only true for the integral as a whole, but also pointwise for every single score s. Note that y_{b_j} in the empirical case (see Definition 15) corresponds to $c(s) = \frac{\pi_1 f_1(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)}$ (as given by Equation 14) in the continuous case above, and also note that s_{b_j} corresponds to the cardinality $\pi_0 f_0(s) + \pi_1 f_1(s)$. The decomposition for empirical distributions as introduced by Murphy (1973) is still predominant for any reference to the decomposition. To our knowledge this is the first explicit derivation of a continuous version of the decomposition.

And now we are ready for relating the optimal threshold choice method with a performance metric as follows:

Theorem 29 *For every convex model m, we have that:*

$$L^o_{U(c)}(m) = RL(m).$$

The proof of this theorem is found in the appendix as Theorem 48.

Corollary 30 *For every model m the expected loss for the optimal threshold choice method $L^o_{U(c)}$ is equal to the refinement loss using the convex hull.*

$$L^o_{U(c)}(m) = RL(\mathrm{Conv}(m)) \triangleq RL_{\mathrm{Conv}}(m).$$

Proof We have $L^o_{U(c)}(m) = L^o_{U(c)}(\mathrm{Conv}(m))$ by Theorem 27, and $L^o_{U(c)}(\mathrm{Conv}(m)) = RL(\mathrm{Conv}(m))$ by Theorem 29 and the convexity of $\mathrm{Conv}(m)$. ■

It is possible to obtain a version of this theorem for empirical distributions which states that $L^o_{U(c)} = RL^{ROCCH}$ where RL^{ROCCH} is the refinement loss of the empirical distribution using the segments of the convex hull for the decomposition.

Before analysing what the meaning of this threshold choice method is and how it relates to the rest, we have to consider whether this threshold choice method is realistic or not. In the beginning of this section we said that the optimal method assumes that (1) we are having complete information about the operating condition at deployment time and (2) we are able to use that information to choose the threshold that will minimise the loss at deployment time.

While (1) is not always true, there are many occasions where we know the costs and distributions at application time. This is the base of the score-driven and rate-driven methods. However, having this information does not mean that the optimal threshold for a data set (e.g., the training or validation data set) ensures an optimal choice for a test set (2). Drummond and Holte (2006) are conscious of this problem and they reluctantly rely on a threshold choice method which is based on "the ROC convex hull [...] only if this selection criterion happens to make cost-minimizing selections, which in general it will not do". But even if these cost-minimising selections are done, as mentioned above, it is not clear how reliable they are for a test data set. As Drummond and Holte (2006, p. 122) recognise: "there are few examples of the practical application of this technique. One example is given by Fawcett and Provost (1997), in which the decision threshold parameter was tuned to be optimal, empirically, for the test distribution".

In the example shown in Table 2 in Section 1, the evaluation technique was training and test. However, with cross-validation, the convex hull cannot be estimated reliably in general, and the thresholds derived from each fold might be inconsistent. Even with a large validation data set, the decision threshold may be suboptimal. This is one of the reasons why the area under the convex hull has not been used as a performance metric. In any case, we can calculate the values as an optimistic limit, leading to $L^o_{U(c)} = RL^{ROCCH} = 0.0953$ for model A and 0.2094 for model B.

7. Relating Performance Metrics

So far, we have repeatedly answered the following question: "If threshold choice method X is used, which is the corresponding performance metric?" The answers are summarised in Table 4. The seven threshold choice methods are shown in the first column (the two fixed methods are grouped in the same row). The integrated view of performance metrics for classification is given by the next two columns. The expected loss of a model for a uniform distribution of cost proportions or skews for each of these seven threshold choice methods produces most of the common performance metrics in classification: 0-1 loss (either weighted or unweighted accuracy), the Mean Absolute Error (equivalent to Mean Probability Rate), the Brier score, *AUC* (which equals the Wilcoxon-Mann-Whitney statistic and the Kendall tau distance of the model to the perfect model, and is linearly related to the Gini coefficient) and, finally, the refinement loss using the bins given by the convex hull.

All the threshold choice methods seen in this paper consider model scores in different ways. Some of them disregard the score, since the threshold is fixed, some others consider the 'magnitude' of the score as an (accurate) estimated probability, leading to the score-based methods, and others consider the 'rank', 'rate' or 'proportion' given by the scores, leading to the rate-based methods. Since the optimal threshold choice is also based on the convex hull, it is apparently more related to the rate-based methods. This is consistent with the taxonomy proposed by Ferri et al. (2009) based on correlations over more than a dozen performance metrics, where three families of metrics were recognised: performance metrics which account for the quality of classification (such as accuracy), performance metrics which account for a ranking quality (such as *AUC*), and performance metrics which evaluate the quality of scores or how well the model does in terms of probability estimation (such as the Brier score or logloss).

This suggests that the way scores are distributed is crucial in understanding the differences and connections between these metrics. In addition, this may shed light on which threshold choice method is best. We have already seen some relations, such as $L^{su}_{U(c)} \geq L^{sd}_{U(c)}$, and $L^{ru}_{U(c)} > L^{rd}_{U(c)}$, but

Threshold choice method	Cost proportions	Skews	Equivalent (or related) performance metrics
fixed	$L^{sf}_{U(c)} = 1 - Acc$	$L^{sf}_{U(z)} = 1 - uAcc$	0-1 loss: Weighted and unweighted accuracy.
score-uniform	$L^{su}_{U(c)} = MAE$	$L^{su}_{U(z)} = uMAE$	Absolute error, Average score, *pAUC* (Ferri et al., 2005) , Probability Rate (Ferri et al., 2009).
score-driven	$L^{sd}_{U(c)} = BS$	$L^{sd}_{U(z)} = uBS$	Brier score (Brier, 1950), Mean Squared Error (*MSE*).
rate-uniform	$L^{ru}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{2}$	$L^{ru}_{U(z)} = \frac{1-2AUC}{4} + \frac{1}{2}$	*AUC* (Swets et al., 2000) and variants (*wAUC*) (Fawcett, 2001; Ferri et al., 2009), Kendall tau, WMW statistic, Gini coefficient.
rate-driven	$L^{rd}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{3}$	$L^{rd}_{U(z)} = \frac{1-2AUC}{4} + \frac{1}{3}$	*AUC* (Swets et al., 2000) and variants (*wAUC*) (Fawcett, 2001; Ferri et al., 2009), Kendall tau, WMW statistic, Gini coefficient.
optimal	$L^{o}_{U(c)} = RL_{\text{Conv}}$	$L^{o}_{U(z)} = uRL_{\text{Conv}}$	ROCCH Refinement loss (Flach and Matsubara, 2007), Refinement Loss (Murphy, 1973), Area under the Cost Curve ('Total Expected Cost') (Drummond and Holte, 2006), Hand's H (Hand, 2009).

Table 4: Threshold choice methods and their expected loss for cost proportions and skews. The *u* in *uAcc*, *uMAE*, *uBS* and *uRL* mean that these metrics are unweighted, that is, calculated as if $\pi_0 = \pi_1$, while the *w* in *wAUC* refers to a weighted version of the AUC, and the *x*-axis and *y*-axis are proportional to π_0 and π_1.

what about $L^{sd}_{U(c)}$ and $L^{rd}_{U(c)}$? Are they comparable? And what about $L^{o}_{U(c)}$? It gives the minimum expected loss by definition over the training (or validation) data set, but when does it become a good estimation of the expected loss for the test data set?

In order to answer these questions we need to analyse transformations on the scores and see how these affect the expected loss given by each threshold choice method. For the rest of the section we assume that scores are in the interval $[0,1]$. Given a model, its scores establish a total order over the examples: $\sigma = (s_1, s_2, ..., s_n)$ where $s_i \leq s_{i+1}$. Since there might be ties in the scores, this total order is not necessarily strict. A monotonic transformation is any alteration of the scores, such that the order is kept. We will consider two transformations: the evenly-spaced transformation and PAV calibration.

7.1 Evenly-Spaced Scores. Relating Brier Score, MAE and AUC

If we are given a ranking or order, or we are given a set of scores but its reliability is low, a quite simple way to assign (or re-assign) the scores is to set them evenly-spaced (in the $[0,1]$ interval).

Definition 31 *A discrete evenly-spaced transformation is a procedure* $\text{EST}(\sigma) \to \sigma'$ *which converts any sequence of scores* $\sigma = (s_1, s_2, ..., s_n)$ *where* $s_i < s_{i+1}$ *into scores* $\sigma' = (s'_1, s'_2, ..., s'_n)$ *where* $s'_i = \frac{i-1}{n-1}$.

Notice that such a transformation does not affect the ranking and hence does not alter the *AUC*.

The previous definition can be applied to continuous score distribution as follows:

Definition 32 *A continuous evenly-spaced transformation is a any strictly monotonic transformation function on the score distribution, denoted by* Even*, such that for the new scores* s' *it holds that* $P(s' \leq t) = t$.

It is easy to see that EST is idempotent, that is, $\text{EST}(\text{EST}(\sigma)) = \text{EST}(\sigma)$. So we say a set of scores σ is evenly-spaced if $\text{EST}(\sigma) = \sigma$.

Lemma 33 *Given a model and data set with set of scores σ, such that they evenly-spaced, when $n \to \infty$ then we have $R(t) = t$.*

Proof Remember that by definition the true positive rate $F_0(t) = P(s \leq t|0)$ and the false positive rate $F_1(t) = P(s \leq t|1)$. Consequently, from the definition of rate we have $R(t) = \pi_0 F_0(t) + \pi_1 F_1(t) = \pi_0 P(s \leq t|0) + \pi_1 P(s \leq t|1) = P(s \leq t)$. But, since the scores are evenly-spaced, the number of scores such that $s \leq t$ is $\sum_{i=1}^{n} I(s_i \leq t) = \sum_{i=1}^{n} I(\frac{i-1}{n-1} \leq t)$ with I being the indicator function (1 when true, 0 otherwise). This number of scores is $\sum_{i=1}^{tn} 1$ when $n \to \infty$, which clearly gives tn. So the probability $P(s \leq t)$ is $tn/n = t$. Consequently $R(t) = t$. ■

The following results connect the score-driven threshold choice method with the rate-driven threshold choice method:

Theorem 34 *Given a model and data set with set of scores σ, such that they are evenly-spaced, when $n \to \infty$:*

$$BS = L^{sd}_{U(c)} = L^{rd}_{U(c)} = \pi_0 \pi_1 (1 - 2AUC) + \frac{1}{3}.$$

Proof By Lemma 33 we have $R(t) = t$, and so the rate-driven and score-driven threshold choice methods select the same thresholds. ■

Corollary 35 *Given a model and data set with set of scores σ such that they are evenly-spaced, when $n \to \infty$:*

$$uBS = L^{sd}_{U(z)} = L^{rd}_{U(z)} = \frac{1 - 2AUC}{4} + \frac{1}{3}.$$

These straightforward results connect *AUC* and Brier score for evenly-spaced scores. This connection is enlightening because it says that *AUC* and *BS* are equivalent performance metrics (linearly related) when we set the scores in an evenly-spaced way. In other words, it says that *AUC* is like a Brier score which considers all the scores evenly-spaced. Although the condition is strong, this is the first linear connection which, to our knowledge, has been established so far between *AUC* and the Brier score.

Similarly, we get the same results for the score-uniform threshold choice method and the rate-uniform threshold choice method.

Theorem 36 *Given a model and data set with set of scores σ such that they are evenly-spaced, when $n \to \infty$:*

$$MAE = L^{su}_{U(c)} = L^{ru}_{U(c)} = \pi_0 \pi_1 (1 - 2AUC) + \frac{1}{2}$$

with similar results for skews. This also connects *MAE* with *AUC* and clarifies when they are linearly related.

7.2 Perfectly-Calibrated Scores. Relating BS, CL and RL

In this section we will work with a different condition on the scores. We will study what interesting connections can be established if we assume the scores to be perfectly calibrated.

The informal definition of perfect calibration usually says that a model is calibrated when the estimated probabilities are close to the true probabilities. From this informal definition, we would derive that a model is perfectly calibrated if the estimated probability given by the scores (i.e., $\hat{p}(1|x)$) equals the true probability. However, if this definition is applied to single instances, it implies not only perfect calibration but a perfect model. In order to give a more meaningful definition, the notion of calibration is then usually defined in terms of groups or bins of examples, as we did, for instance, with the Brier score decomposition. So, we need to apply this correspondence between estimated and true (actual) probabilities over bins. We say a bin partition is invariant on the scores if for any two examples with the same score they are in the same bin. In other words, two equal scores cannot be in different bins (equivalence classes cannot be broken). From here, we can give a definition of perfect calibration:

Definition 37 (Perfectly-calibrated for empirical distribution models) *A model is* perfectly calibrated *if for any invariant bin partition $\mathcal{P}$ we have that $y_{b_j} = s_{b_j}$ for all its bins: that is, the average actual probability equals the average estimated probability, thus making $CL^{\mathcal{P}} = 0$.*

Note that it is not sufficient to have $CL = 0$ for one partition, but for all the invariant partitions. Also notice that the bins which are generated by a ROC curve are the minimal invariant partition on the scores (i.e., the quotient set). So, we can give an alternative definition of perfectly calibrated model: a model is perfectly calibrated if and only if $CL^{ROC} = 0$. For the continuous case, the partition is irrelevant and the definition is as follows:

Definition 38 (Perfectly-calibrated for continuous distribution models) *A continuous model is* perfectly calibrated *if and only if $s = \frac{\pi_1 f_1(s)}{\pi_0 f_0(s)+\pi_1 f_1(s)}$ which is exactly $c(s)$ given by Equation (14).*

Note that the previous definition is equivalent to saying that $CL = 0$, as in the empirical case, since CL can be rewritten as follows, following the decomposition of Theorem 28:

$$\begin{aligned} CL &= \int_0^1 \frac{(s(\pi_0 f_0(s) + \pi_1 f_1(s)) - \pi_1 f_1(s))^2}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds \\ &= \int_0^1 (\pi_0 f_0(s) + \pi_1 f_1(s)) \left(s - \frac{\pi_1 f_1(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)} \right)^2 ds. \end{aligned}$$

Lemma 39 *For a perfectly calibrated classifier m:*

$$\frac{1-s}{s} = \frac{f_0(s)}{f_1(s)} \frac{\pi_0}{\pi_1}$$

and m is convex.

Proof The expression is a direct transformation of Definition 38 and convexity just follows from Definition 25. ■

Now that we have two proper and operational definitions of perfect calibration, we define a calibration transformation as follows.

Definition 40 Cal *is a monotonic function over the scores which converts any model m into another calibrated model m^* such that $CL = 0$ and RL is not modified.*

Cal always produces a convex model, so $\text{Conv}(\text{Cal}(m)) = \text{Cal}(m)$, but a convex model is not always perfectly calibrated (e.g., a binormal model with same variances is always convex but it can be uncalibrated), so $\text{Cal}(\text{Conv}(m) \neq \text{Conv}(m)$. This is summarised in Table 5. If the model is strictly convex, then Cal is strictly monotonic. An instance of the function Cal is the transformation $T \mapsto s = c(T)$ where $c(T) = \frac{\pi_1 f_1(T)}{\pi_0 f_0(T) + \pi_1 f_1(T)}$ as given by Equation (14). This transformation is shown to keep *RL* unchanged in the appendix and makes $CL = 0$.

The previous function is defined for continuous score distributions. The corresponding function for empirical distributions is known as the Pool Adjacent Algorithm (PAV) (Ayer et al., 1955). Following Fawcett and Niculescu-Mizil (2007), the *PAV* function converts any model *m* into another calibrated model m^* such that the following property $s_{b_j} = y_{b_j}$ holds for every segment in its convex hull.

	Evenly-spaced	Convexification	Perfect Calibration
Continuous distributions	Even	Conv	Cal
Empirical distributions	EST	ROCCH	PAV

Table 5: Transformations on scores. Perfect calibration implies a convex model but not vice versa.

Fawcett and Niculescu-Mizil (2007) have shown that isotonic-based calibration (Robertson et al., 1988) is equivalent to the PAV algorithm, and closely related to ROCCH, since, for every *m* and data set, we have:

$$\begin{aligned} BS(PAV(m)) &= CL^{ROC}(PAV(m)) + RL^{ROC}(PAV(m)) = CL^{ROCCH}(PAV(m)) + RL^{ROCCH}(PAV(m)) \\ &= RL^{ROC}(PAV(m)) = RL^{ROCCH}(PAV(m)). \end{aligned}$$

It is also insightful to see that isotonic regression (calibration) is the monotonic function defined as $\arg\min_f \sum (y_i - f(s_i))^2$, that is, the monotonic function over the scores which minimises the Brier score. This leads to the same function if we use any other proper scoring function (such as logloss).

The similar expression for the continuous case is

$$BS(\text{Cal}(m)) = CL(\text{Cal}(m)) + RL(\text{Cal}(m)) = RL(\text{Cal}(m)).$$

Now we analyse what happens with perfectly calibrated models for the score-driven threshold choice and the score-uniform threshold choice methods. This will help us understand the similarities and differences between the threshold choices and their relation with the optimal method. Along the way, we will obtain some straightforward, but interesting, results.

Theorem 41 *If a model is perfectly calibrated then we have:*

$$\pi_0 s_0 = \pi_1 (1 - s_1) \tag{15}$$

or equivalently,

$$\pi_0 MAE_0 = \pi_1 MAE_1.$$

Proof For perfectly calibrated models, we have that for every bin in an invariant partition on the scores we have that $y_{b_j} = s_{b_j}$. Just taking a partition consisting of one single bin (which is an invariant partition), we have that this is the same as saying that $\pi_1 = \pi_1 s_1 + \pi_0 s_0$. This leads to $\pi_1(1-s_1) = \pi_0 s_0$. ■

Equation (15) is an interesting formula in its own right. It gives a necessary condition for calibration: the extent to which the average score over all examples (which is the weighted mean of per-class averages $\pi_0\bar{s}_0 + \pi_1\bar{s}_1$) deviates from π_1.

We now give a first result which connects two performance metrics:

Theorem 42 *If a model is perfectly calibrated then we have:*

$$BS = \pi_0 s_0 = \pi_1(1-s_1) = MAE/2.$$

Proof We use the continuous decomposition (Theorem 28):

$$BS = CL + RL.$$

Since it is perfectly calibrated, $CL = 0$. Then we have:

$$\begin{aligned} BS &= RL = \int_0^1 \frac{\pi_1 f_1(s)\pi_0 f_0(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds = \int_0^1 (\pi_1 f_1(s)) \left(1 - \frac{\pi_1 f_1(s)}{\pi_0 f_0(s) + \pi_1 f_1(s)}\right) ds \\ &= \int_0^1 \left(\pi_1 f_1(s) - \frac{[\pi_1 f_1(s)]^2}{\pi_0 f_0(s) + \pi_1 f_1(s)}\right) ds = \int_0^1 \pi_1 f_1(s) ds - \int_0^1 \frac{[\pi_1 f_1(s)]^2}{\pi_0 f_0(s) + \pi_1 f_1(s)} ds \\ &= \pi_1 - \int_0^1 \frac{\pi_1 f_1(s)}{\frac{\pi_0 f_0(s)}{\pi_1 f_1(s)} + 1} ds. \end{aligned}$$

Since it is perfectly calibrated, we have, by Lemma 39:

$$\frac{f_0(s)}{f_1(s)} = \frac{1-s}{s}\frac{\pi_1}{\pi_0}.$$

So:

$$\begin{aligned} BS &= \pi_1 - \int_0^1 \frac{\pi_1 f_1(s)}{\frac{\pi_0}{\pi_1}\frac{1-s}{s}\frac{\pi_1}{\pi_0} + 1} ds = \pi_1 - \int_0^1 \frac{s\pi_1 f_1(s)}{(1-s)+s} ds \\ &= \pi_1 - \pi_1 \int_0^1 s f_1(s) ds = \pi_1(1 - \int_0^1 s f_1(s)) ds = \pi_1(1-s_1). \end{aligned}$$

■

We will now use the expressions for expected loss to analyse where this result comes from exactly. In the following result, we see that for a calibrated model the optimal threshold T for a given cost proportion c is $T = c$, which is exactly the score-driven threshold choice method. In other words:

Theorem 43 *For a perfectly calibrated model:*
$T_c^o(c) = T_c^{sd}(c) = c$.

Proof We first take Equation (12):

$$T_c^o(c) \quad = \quad \arg\min_t \; 2\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\}.$$

We calculate the derivative and equal it to 0 to get t, but we isolate c:

$$2\{c\pi_0(1-f_0(t)) + (1-c)\pi_1 f_1(t)\} = 0$$
$$c = \frac{\pi_1 f_1(t)}{\pi_0 f_0(t) + \pi_1 f_1(t)}.$$

From Definition 38 (perfect calibration) we have that the right expression above equals t, so we have $t = c$. The proof is identical for T_c^{sd}. ■

And now we can express and relate many of the expressions for the expected loss seen so far. Starting with the expected loss for the optimal threshold choice method, that is, L_c^o (which uses T_c^o), we have, from Theorem 43, that $T_c^o(c) = T_c^{sd}(c) = c$ when the model is perfectly calibrated. Consequently, we have the same as Equation (8), and since we know that $BS = \pi_0 s_0$ for perfectly calibrated models, we have:

$$L^o_{U(c)} = BS = \pi_0 s_0 = MAE/2.$$

The following theorem summarises all the previous results.

Theorem 44 *For perfectly calibrated models:*

$$L^{sd}_{U(c)} = L^o_{U(c)} = RL = \frac{L^{su}_{U(c)}}{2} = \frac{MAE}{2} = BS = \pi_0 s_0 = \pi_1(1-\bar{s}_1).$$

Proof Since $L^{sd}_{U(c)} = BS$ it is clear that $L^{sd}_{U(c)} = \pi_0 s_0$, as seen above for $L^o_{U(c)}$ as well. Additionally, from Theorem 12, we have that $L^{su}_{U(c)} = \pi_0\bar{s}_0 + \pi_1(1-\bar{s}_1)$, which reduces to $2L^{su}_{U(c)} = 2BS = 2\pi_0\bar{s}_0$. We also use the result of Theorem 29 which states that, in general (not just for perfectly calibrated models), $L^o_{U(c)}(m) = RL(\text{Conv}(m))$. ■

All this gives an interpretation of the optimal threshold choice method as a method which calculates expected loss by assuming perfect calibration. Note that this is clearly seen by the relation $L^{sd}_{U(c)} = L^o_{U(c)} = \frac{L^{su}_{U(c)}}{2}$, since the loss drops to the half if we use scores to adjust to the operating condition. In this situation, we get the best possible result.

7.3 Choosing a Threshold Choice Method

It is enlightening to see that many of the most popular classification performance metrics are just expected losses by changing the threshold choice method and the use of cost proportions or skews. However, it is even more revealing to see how (and under which conditions) these performance

General relations:

$$L^{ru}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{2} > L^{rd}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{3} \geq L^{o}_{U(c)} = RL_{\text{Conv}}$$

$$L^{su}_{U(c)} = MAE \geq L^{sd}_{U(c)} = BS \geq L^{o}_{U(c)} = RL_{\text{Conv}}$$

If scores are evenly-spaced:

$$L^{ru}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{2} = L^{su}_{U(c)} = MAE = \pi_0 s_0 + \pi_1(1-\bar{s}_1)$$

$$L^{rd}_{U(c)} = \pi_0\pi_1(1-2AUC) + \frac{1}{3} = L^{sd}_{U(c)} = BS$$

If scores are perfectly calibrated:

$$L^{sd}_{U(c)} = L^{o}_{U(c)} = RL = \frac{L^{su}_{U(c)}}{2} = \frac{MAE}{2} = BS = \pi_0 s_0 = \pi_1(1-\bar{s}_1)$$

If the model has perfect ranking:

$$L^{ru}_{U(c)} = \frac{1}{4} > L^{rd}_{U(c)} = \frac{1}{12} > L^{o}_{U(c)} = 0$$

If the model is random (and $\pi_0 = \pi_1$):

$$L^{su}_{U(c)} = L^{sd}_{U(c)} = L^{ru}_{U(c)} = \frac{1}{2} > L^{rd}_{U(c)} = \frac{1}{3} > L^{o}_{U(c)} = \frac{1}{4}$$

Figure 4: Comparison of losses and performance metrics, in general and under several score conditions.

metrics can be related (in some cases with inequalities and in some other cases with equalities). The notion of score transformation is the key idea for these connections, and is more important that it might seem at first sight. Some threshold choice methods can be seen as a score transformation followed by the score-driven threshold choice method. Even the fixed threshold choice method can be seen as a crisp transformation where scores are set to 1 if $s_i > t$ and 0 otherwise. Another interesting point of view is to see the values of extreme models, such as a model with perfect ranking ($AUC = 1, RL^{ROCCH} = 0$) and a random model ($AUC = 0.5, RL^{ROCCH} = 0.25$ when $\pi_0 = \pi_1$). Figure 4 summarises all the relations found so far and these extreme cases.

The first apparent observation is that $L^{o}_{U(c)}$ seems the best loss, since it derives from the optimal threshold choice method. We already argued in Section 6 that this is unrealistic. The result given by Theorem 29 is a clear indication of this, since this makes expected loss equal to RL_{Conv}. Hence, this threshold choice method assumes that the calibration which is performed with the convex hull over the training (or a validation data set) is going to be perfect and hold for the test set. Figure 4 also gives the impression that $L^{su}_{U(c)}$ and $L^{ru}_{U(c)}$ are so bad that their corresponding threshold choice methods and metrics are useless. In order to refute this simplistic view, we must realise (again) that not every threshold choice method can be applied in every situation. Some require more information

or more assumptions than others. Table 6 completes Table 3 to illustrate the point. If we know the deployment operating condition at evaluation time, then we can fix the threshold and get the expected loss. If we do not know this information at evaluation time, but we expect to be able to have it and use it at deployment time, then the score-driven, rate-driven and optimal threshold choice methods seem the appropriate ones. Finally, if no information about the operating condition is going to be available at any time then the score-uniform and the rate-uniform may be alternative options, which could account for a worst-case estimation.

Threshold choice method	Fixed	Driven by o.c.	Chosen uniformly
Using scores	score-fixed (T^{sf})	score-driven (T^{sd})	score-uniform (T^{su})
Using rates	rate-fixed (T^{rf})	rate-driven (T^{rd})	rate-uniform (T^{ru})
Using optimal thresholds		optimal (T^{o})	
Required information	o.c. at evaluation time	o.c. at deployment time	no information

Table 6: Information which is required (and when) for the seven threshold choice methods so that they become reasonable (or just not totally unreasonable). Operating condition is denoted by o.c.

From the cases shown in Table 6, the methods driven by the operating condition require further discussion. The relations shown in Figure 4 illustrate that, in addition to the optimal threshold choice method, the other two methods that seem more competitive are the score-driven and the rate-driven. One can argue that the rate-driven threshold choice has an expected loss which is always greater than $1/12$ (if $AUC = 1$, we get $-1/4 + 1/3$), while the others can be 0. But things are not so clear-cut.

- The score-driven threshold choice method considers that the scores are estimated probabilities and that they are reliable, in the tradition of proper scoring rules. So it just uses these probabilities to set the thresholds.

- The rate-driven threshold choice method completely ignores the scores and only considers their order. It assumes that the ranking is reliable while the scores are not accurate probabilities. It derives the thresholds using the predictive positive rate. It can be seen as the score-driven threshold choice method where the scores have been set evenly-spaced by a transformation.

- The optimal threshold choice method also ignores the scores completely and only considers their order. It assumes that the ranking is reliable while the scores are not accurate probabilities. However, this method derives the thresholds by keeping the order and using the slopes of the segments of the convex hull (typically constructed over the training data set or a validation data set). It can be seen as the score-driven threshold choice method where the scores have been calibrated by the PAV method.

Now that we better understand the meaning of the threshold choice methods we may state the difficult question more clearly: given a model, which threshold choice method should we use to make classifications? The answer is closely related to the calibration problem. Some theoretical and experimental results (Robertson et al., 1988; Ayer et al., 1955; Platt, 1999; Zadrozny and Elkan, 2001,

2002; Niculescu-Mizil and Caruana, 2005; Niculescu-Mizil and Caruana, 2005; Bella et al., 2009; Gebel, 2009) have shown that the PAV method (also known as isotonic regression) is not frequently the best calibration method. Some other calibration methods could do better, such as Platt's calibration or binning averaging. In particular, it has been shown that "isotonic regression is more prone to overfitting, and thus performs worse than Platt scaling, when data is scarce" (Niculescu-Mizil and Caruana, 2005). Even with a large validation data set which allows the construction of an accurate ROC curve and an accurate convex hull, the resulting choices are not necessarily optimal for the test set, since there might be problems with outliers (Rüping, 2006). In fact, if the validation data set is much smaller (or biased) than the training set, the resulting probabilities can be even worse than the original probabilities, as it may happen with cross-validation. So, we have to feel free to use other (possibly better) calibration methods instead and do not stick to the PAV method just because it is linked to the optimal threshold choice method.

So the question of whether we keep the scores or not (and how we replace them in case) depends on our expectations on how well-calibrated the model is, and whether we have tools (calibration methods and validation data sets) to calibrate the scores.

But we can turn the previous question into a much more intelligent procedure. Calculating the three expected losses discussed above (and perhaps the other threshold choice methods as well) provides a rich source of information about how our models behave. This is what performance metrics are all about. It is only after the comparison of all the results and the availability of (validation) data sets when we can make a decision about which threshold choice method to use.

This is what we did with the example shown in Table 2 in Section 1. We evaluated the model for several threshold choice methods and from there we clearly saw which models were better calibrated and we finally made a decision about which model to use and with which threshold choice methods.

In any case, note that the results and comparisons shown in Figure 4 are for expected loss; the actual loss does not necessarily follow these inequalities. In fact, the expected loss calculated over a validation data set may not hold over the test data set, and even some threshold choice methods we have discarded from the discussion above (the fixed ones or the score-uniform and rate-uniform, if probabilities or rankings are very bad respectively) could be better in some particular situations.

8. Discussion

This paper builds upon the notion of threshold choice method and the expected loss we can obtain for a range of cost proportions (or skews) for each of the threshold choice methods we have investigated. The links between threshold choice methods, between performance metrics, in general and for specific score arrangements, have provided us with a much broader (and more elaborate) view of classification performance metrics and the way thresholds can be chosen. In this last section we link our results to the extensive bulk of work on classification evaluation and analyse the most important contributions and open questions which are derived from this paper.

8.1 Related Work

One decade ago there was a scattered view of classification evaluation. Many performance metrics existed and it was not clear what their relationships were. One first step in understanding some of these performance metrics in terms of costs was the notion of cost isometrics (Flach, 2003). With cost isometrics, many classification metrics (and decision tree splitting criteria) are characterised by its skew landscape, that is, the slope of its isometric at any point in the ROC space. Another com-

prehensive view was the empirical evaluation made by Ferri et al. (2009). The analysis of Pearson and Spearman correlations between 18 different performance metrics shows the pairs of metrics for which the differences are significant. However, this work does not elaborate, at least theoretically, on what exactly each metric measures, but rather on whether they give different choices in general.

In addition to these, there have been three lines of research in this area which provide further pieces to understand the whole picture.

- First, the notion of 'proper scoring rules' (which was introduced in the sixties, see for example, the work by Murphy and Winkler, 1970), has been developed to a degree (Buja et al., 2005) in which it has been shown that the Brier score (MSE loss), logloss, boosting loss and error rate (0-1 loss) are all special cases of an integral over a Beta density, and that all these performance metrics can be understood as averages (or integrals), at least theoretically, over a range of cost proportions (see, e.g., the works of Gneiting and Raftery, 2007; Reid and Williamson, 2010 and Brümmer, 2010), so generalising the early works by Murphy on probabilistic predictions when cost-loss ratio is unknown (Murphy, 1966, 1969). Additionally, further connections have been found between proper scoring rules and distribution divergences (f-divergences and Bregman divergences) (Reid and Williamson, 2011).

- Second, the translation of the Brier decomposition using ROC curves (Flach and Matsubara, 2007) suggests a connection between the Brier score and ROC curves, and particularly between refinement loss and *AUC*, since both are performance metrics which do not require the magnitude of the scores of the model.

- Third, an important coup d'effet has been given by Hand (2009), stating that the *AUC* cannot be used as a performance metric for evaluating models (for a range of cost proportions), *assuming the optimal threshold choice method*, because the distribution for these cost proportions depends on the model. This seemed to suggest a definitive rupture between ranking quality and classification performance over a range of cost proportions.

Each of the three lines mentioned above provides a partial view of the problem of classifier evaluation, and suggests that some important connections between performance metrics were waiting to be unveiled. The starting point of this unifying view is that all the previous works above worked with only two threshold choice methods, which we have called the score-driven threshold choice method and the optimal threshold choice method. Only a few works mention these two threshold choice methods together. For instance, Drummond and Holte (2006) talk about 'selection criteria' (instead of 'threshold choice methods') and they distinguish between 'performance-independent' selection criteria and 'cost-minimizing' selection criteria. Hand (personal communication) says that 'Hand (2009) (top of page 122) points out that there are situations where one might choose thresholds independently of cost, and go into more detail in Hand (2010)'. This is related to the fixed threshold choice method, or the rate-uniform and score-uniform threshold choice methods used here. Finally, Flach et al. (2011) explore the rate-uniform threshold choice method while Hernández-Orallo et al. (2011) explore the score-driven threshold choice method.

The notion of proper scoring rule works with the score-driven threshold choice method. This implies that this notion cannot be applied to *AUC*—Reid and Williamson (2011) connects the area under the convex hull (*AUCH*) with other proper scoring rules but not *AUC*—and to RL. As a consequence, the Brier score, log-loss, boosting loss and error rate would only be minor choices depending on the information about the distribution of costs.

Hand (2009) takes a similar view of the cost distribution, as a choice that depends on the information we may have about the problem, but makes an important change over the tradition in proper scoring rules tradition. He considers 'optimal thresholds' (see Equation 12) instead of the score-driven choice. With this threshold choice method, Hand is able to derive *AUC* (or yet again *AUCH*) as a measure of aggregated classification performance, but the distribution he uses (and criticises) depends on the model itself. Then he defines a new performance metric which is proportional to the area under the optimal cost curve. Hand (2010) and Hand and Anagnostopoulos (2011) elaborate on this by the consideration of asymmetries in the cost distribution.

8.2 A Plethora of Evaluation Metrics

The unifying view under the systematic exploration of threshold choice methods in this paper has established a set of connections which portray a much more comprehensive view of the landscape of evaluation metrics for classification. However, it has to be emphasised that each connection between a metric and a kind of expected loss is associated to a particular set of assumptions. The most important assumption is the cost model. For the whole paper, we have assumed that the operating condition θ is simplified to a single parameter, c or z, from a three-dimensional vector $\theta = \langle b, c, \pi_0 \rangle$. In order to make this reduction, we have assumed that the threshold choice method ignores the magnitude b. In addition, we have either assumed π_0 fixed or have linked it to c through the notion of skew (see appendix A). However, in general, we could consider trivariate distributions for the parameters in θ. We could also consider threshold choice methods which are sensitive to the magnitude b or other combinations of the three parameters. For instance, we could consider a threshold choice method which is more 'conservative' when b is high and more 'risky' when b is low. Moreover, in some applications, the operating condition can have even more parameters, since it may be instance-dependent (Turney, 2000) or can change depending on previous errors. Certainly, for a specific application one must consider the distribution which better fits the expectation or knowledge about the possible operating conditions. This first *dimension*, the distribution of operating conditions, has been varied in many different ways by Wieand et al. (1989), Gneiting and Raftery (2007), Reid and Williamson (2010), Reid and Williamson (2011), Brümmer (2010), Hand (2009), Hand (2010) and Hand and Anagnostopoulos (2011), as mentioned above. Here we have considered the simplest option, a uniform distribution (except for the fixed threshold choice methods where the results are more general), but many other options can be explored, including partial, asymmetric or multimodal distributions.

As said in the introduction, this paper works and varies on a different dimension, by changing the threshold choice method systematically. The choice of a particular threshold choice method reflects many things: the information one may have at evaluation time or deployment time, the reliability or calibration degree one expect from a model or the very character of the model, which may be a crisp classifier, a ranker or a probability estimator. Also, the choice can also be just a matter of practice, since some threshold choice methods are simpler than others and develop into simpler decision rules. In fact, we have explored seven possibilities here. Some of them may look more reasonable than others, and some may correspond to frequent practical situations, while others have just been set in order to derive the relation between expected loss and a relevant evaluation metric. And there might be other possibilites. For instance, a particular case of the fixed threshold choice method can be defined by choosing the threshold at $\mathbb{E}\{c\}$, which can be generalised, for example,

in an online setting, if this expected value evolves (or is refined) after the information we get from the actual costs example after example.

All this suggests that many other combinations could be explored by using different options for the two dimensions, and possibly relaxing some other assumptions, such as Hand did with his measure H (Hand, 2009), when using the $\beta_{2,2}$ distribution for the optimal threshold choice method instead of the uniform ($\beta_{1,1}$) distribution. We think that the same thing could be done with the rate-driven threshold choice method, possibly leading to new variants of the *AUC*. This is related to the extensive work where several distributions are presented for calculating an average of sensitivities over a restricted range of specificities (Wieand et al., 1989), leading to other variants of *AUC* (such as partial *AUC*). And, of course, this has also been done with proper scoring rules for the score-driven threshold choice method with many loss functions.

It is, however, also worthwhile to underline the limitations of aggregated metrics for comparing classification models. Graphical plots, where performance is shown in a range of operating conditions, are more powerful than aggregated metrics, since we can keep several methods provided they are not completely dominated by others. Many threshold choice methods give rise to particular kinds of curves that provide at each operating point, rather than just an aggregate.

8.3 Conclusions and Future Work

As a conclusion, if we want to evaluate a model for a wide range of operating conditions (i.e., cost proportion or skews), we have to determine first which threshold choice method is to be used. If it is fixed because we have a non-probabilistic classifier or we are given the actual operating condition at evaluation time, then we get accuracy (and unweighted accuracy) as a good performance metric. If we have no access to the operating condition at evaluation time but neither do we at deployment time, then the score-uniform and the rate-uniform may be considered, with *MAE* and *AUC* as corresponding performance metrics. Finally, in the common situation when we do not know the operating condition at evaluation time but we expect that it will be known and used at deployment time, then we have more options. If a model has no reliable scores or probability estimations, we recommend the refinement loss (RL_{Conv}, which is equivalent to area under the optimal cost curve) if thresholds are being chosen using the convex hull of a reliable ROC curve, or, alternatively, we recommend the area under the ROC curve (*AUC*) if the estimation of this convex hull is not reliable enough to choose thresholds confidently. More readily, if a model has reliable scores because it is a good probability estimator or it has been processed by a calibration method, then we recommend to choose the thresholds according to scores. In this case, the corresponding performance metric is the Brier score.

From this paper, now we have a much better understanding on the relation between the Brier score, the *AUC* and refinement loss. We also know much better what is happening when models are not convex and/or not calibrated. In addition, we find that using evenly-spaced scores, we get that the Brier score and the *AUC* are linearly related. Furthermore, we see that if the model is perfectly calibrated, the expected loss using the score-driven threshold choice method equals the optimal threshold choice method.

The collection of new findings introduced in this paper leads to many other avenues to follow and some questions ahead. For instance, the duality between cost proportions and skews suggests that we could work with loglikelihood ratios as well. Also, there is always the problem of multi-class evaluation. This is as challenging as interesting, since there are many more threshold choice

methods in the multiclass case and the corresponding expected losses could be connected to some multiclass extensions of the binary performance metrics. Finally, more work is needed on the relation between the ROC space and the cost space, and the representation of all these expected losses in the latter space. The notion of Brier curve (Hernández-Orallo et al., 2011) is a first step in this direction, but all the other threshold choice methods also lead to other curves.

Acknowledgments

We thank the anonymous reviewers for their comments, which have helped to improve this paper significantly. We are also grateful to David Hand for insightful discussion. This work was supported by the MEC/MINECO projects CONSOLIDER-INGENIO CSD2007-00022 and TIN 2010-21062-C02-02, GVA project PROMETEO/2008/051, and the *REFRAME* project granted by the European Coordinated Research on Long-term Challenges in Information and Communication Sciences & Technologies ERA-Net (CHIST-ERA), and funded by the Engineering and Physical Sciences Research Council in the UK and the Ministerio de Economía y Competitividad in Spain.

Appendix A. Univariate Operating Conditions Using Cost Proportions and Skews

This appendix develops the expressions for univariate operating conditions from the general notion of operating condition introduced in Section 2.2. One possibility of reducing an operating condition with three parameters $\theta = \langle b, c, \pi_0 \rangle$ into one single parameter, without assuming independence of b and c, relies on noticing that b is a multiplicative factor in Equation (1). So, we can express loss as follows:

$$Q(t;\theta) = b\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\} = bQ_\eta(t;\eta). \tag{16}$$

where $\eta = \langle c, \pi_0 \rangle$. The set of these normalised operating conditions is denoted by H. In other words, loss is just a product of the cost magnitude b and normalised loss. When calculating expected loss for a particular threshold choice method, we can write

$$Q(T(\theta);\theta) = bQ_\eta(T_\eta(\eta),\eta). \tag{17}$$

Note that this *assumes that the threshold choice method is defined in terms of* η, *and hence it is independent of* b.

From here, we can just work with Equation (17) and derive expected loss from Equation (2) as follows:

$$\begin{aligned} L &= \int_\Theta Q(T(\theta);\theta)w(\theta)d\theta = \int_\Theta bQ_\eta(T_\eta(\eta),\eta)w(\theta)d\theta \\ &= \int_H \left\{ \int_0^\infty bQ_\eta(T_\eta(\eta),\eta)w_{B|H}(b|\eta)db \right\} w_H(\eta)d\eta \\ &= \int_H Q_\eta(T_\eta(\eta),\eta) \left\{ \int_0^\infty bw_{B|H}(b|\eta)db \right\} w_H(\eta)d\eta \\ &= \int_H Q_\eta(T_\eta(\eta),\eta)\mathbb{E}\{b|\eta\}w_H(\eta)d\eta. \end{aligned}$$

with $w_H(\eta)$ being the marginal distribution density for η, that is, $w_H(\eta) = \int_0^\infty w(\theta)db$, and $w_{B|H}(b|\eta)$ the conditional density for b given η, that is, $w_{B|H}(b|\eta) = w(\theta)/w_H(\eta)$. And now, let us define

$v_H(\eta) = w_H(\eta)\mathbb{E}\{b|\eta\}/\mathbb{E}\{b\}$ with $\mathbb{E}\{b\} = \int_H \mathbb{E}\{b|\eta\}w_H(\eta)d\eta$, leading to the following expression:

$$L = \mathbb{E}\{b\}\int_H Q_\eta(T_\eta(\eta),\eta)v_H(\eta)d\eta. \quad (18)$$

So now we have an expression which seems to isolate b (more precisely, $\mathbb{E}\{b\}$) as a constant factor. Obviously, this is possible since we have constructed $v_H(\eta)$ in terms of the conditional expected value b, incorporating the variability of b, while isolating its average magnitude. This is interesting and very useful, because if we have knowledge about the dependency between b and η, we can incorporate that information into v, without affecting Q_η at all. For instance, if π_0 is fixed and we assume (or know) $w_H(\eta)$ to be a Beta distribution $\beta_{2,2}$ and we also assume (or know) that the values for b are higher for extreme values of c (closer to 1 or 0), then the resulting distribution v could be assumed to account for these two things. This would make increase the probability for extreme values of c, making the resulting distribution flatter and closer, for example, to a uniform distribution. Consequently, in this paper we will frequently assume $v_H(\eta)$ to be uniform (either fixing π_0 or combining π_0 and c into a single parameter z). This really makes it explicit that it is $\frac{w_H(\eta)\mathbb{E}\{b|\eta\}}{\mathbb{E}\{b\}}$ what we are assuming to be uniform.

Now, we will derive the two approaches for univariate operating conditions (costs and skews) that we use in this paper. In one of them, we assume that the class proportion (i.e., π_0) is fixed, leading to the marginal distribution $w_c(c) = v_H(\langle c,\pi_0\rangle)$.

Since we now only have a relevant free parameter c in Q_η, we can now express the normalised loss as a function of c. However, for a mere convenience that will become clear below, we include the factor $\mathbb{E}\{b\}$ in the loss produced at a decision threshold t and a cost proportion c, adapting Equation (16):

$$Q_c(t;c) \triangleq \mathbb{E}\{b\}Q_\eta(t;\langle c,\pi_0\rangle) = \mathbb{E}\{b\}\{c\pi_0(1-F_0(t)) + (1-c)\pi_1 F_1(t)\}. \quad (19)$$

With this inclusion of $\mathbb{E}\{b\}$ we just have the following simple expression for the calculation of expected loss, derived from Equation (18):

$$L_c = \int_0^1 Q_c(T(c);c)w_c(c)dc. \quad (20)$$

Recall that w_c incorporates the variability of b jointly with c.

A different approach to reducing the operating condition to a single parameter is the notion of *skew*, which is a normalisation of the product between cost proportion and class proportion:

$$z \triangleq \frac{c_0\pi_0}{c_0\pi_0 + c_1\pi_1} = \frac{c\pi_0}{c\pi_0 + (1-c)(1-\pi_0)}.$$

This means that π_0 is no longer fixed, but neither is it independent of c. What z does is to combine both parameters. This is a different way of reducing the operating condition to one single parameter. We thus define loss as depending solely on z. From Equation (19) we obtain

$$\frac{Q_c(t;c)}{\mathbb{E}\{b\}[c\pi_0 + (1-c)(1-\pi_0)]} = z(1-F_0(t)) + (1-z)F_1(t) \triangleq Q_z(t;z). \quad (21)$$

This gives an expression for *standardised*] loss at a threshold t and a skew z.

We then have the following simple but useful result.

Lemma 45 *If $\pi_0 = \pi_1$ then $z = c$ and $Q_z(t;z) = \frac{2}{\mathbb{E}\{b\}} Q_c(t;c)$.*

Proof If classes are balanced we have $c\pi_0 + (1-c)(1-\pi_0) = 1/2$, and the result follows from Equation (21). ■

This justifies taking the expected value of the cost magnitude $\mathbb{E}\{b\} = 2$, which means that Q_z and Q_c are expressed on the same 0-1 scale, and are also commensurate with error rate which assumes $c = 1/2$. The upshot of Lemma 45 is that we can transfer any expression for loss in terms of cost proportion to an equivalent expression in terms of skew by just setting $\pi_0 = \pi_1 = 1/2$ and $z = c$. Notice that if $c = 1/2$ then $z = \pi_0$, so in that case skew denotes the class distribution as operating condition.

In fact, we can also define $w_z(z)$ by incorporating the variability of b (and also π_0 and c). We could choose $w_z(z)dz = \frac{1}{\mathbb{E}\{b\}[c\pi_0+(1-c)(1-\pi_0)]} w_c(c)dc$, but we can use any other distribution. In the paper we will use the uniform distribution for $w_z(z)$. In any case, this leads to the corresponding expression of standardised expected loss (as Equation 20):

$$L_z \;=\; \int_0^1 Q_z(T(z);z)w_z(z)dz.$$

So, with this isolation of the average magnitude of b, and the incorporation of its variability into the univariate distribution, in the paper we will just work with operating conditions which are either defined by the cost proportion c (assuming a fixed class distribution π_0) or by the skew z (which combines c and π_0).

Appendix B. Proof of Theorem 29

In this appendix, we give the proof for Theorem 29 in the paper. The theorem works with convex models as given by Definition 25.

In this appendix, we will use:

$$c(T) = \frac{\pi_1 f_1(T)}{\pi_0 f_0(T) + \pi_1 f_1(T)}.$$

throughout, as introduced by Equation (14). Sometimes we will use subindices for $c(T)$ depending on the model we are using. We will also use $slope(T) = \pi_1 f_1(T) = \frac{\pi_1}{\pi_0}\left(\frac{1}{c(T)} - 1\right)$. A convex model is the same as saying that $c(T)$ is non-decreasing or that $slope(T)$ is non-increasing.

We use $c^{-1}(s)$ for the inverse of $c(T)$ (wherever it is well defined). We will use the following transformation $T \mapsto s = c(T)$ and the resulting model will be denoted by $m^{(c)}$. We will use s, c or σ for elements in the codomain of this transformation (cost proportions or scores between 0 and 1) and we will use T or τ for elements in the domain.

For continuous and strictly convex models for which $c(0) = 0$ and $c(1) = 1$, the proof is significantly simpler. In general, for any convex model, including discontinuities and straight segments, things become a little bit more elaborate, as we see below.

B.1 Intervals

Since the model is convex, we know that $c(T)$ is monotone, more precisely, non-decreasing. We can split the codomain and domain of this function into intervals. Intervals in the codomain of

thresholds will be represented with the letter τ and intervals in the domain of cost proportions or scores between 0 and 1 will be denoted by letter σ. The series of intervals are denoted as follows:

$$\begin{aligned} I_\sigma &= (\sigma_0,\sigma_1),(\sigma_1,\sigma_2)\ldots(\sigma_i,\sigma_{i+1})\ldots(\sigma_{n-1},\sigma_n) \\ &\qquad\qquad \Uparrow c(\tau) \qquad\quad \Downarrow c^{-1}(\sigma) \\ I_\tau &= (\tau_0,\tau_1),(\tau_1,\tau_2)\ldots(\tau_i,\tau_{i+1})\ldots(\tau_{n-1},\tau_n) \end{aligned}$$

where $\sigma_0 = 0$, $\sigma_n = 1$, $\tau_0 = -\infty$ and $\tau_n = \infty$. Even though we cannot make a bijective mapping for every point, we can construct a bijective mapping between I_σ and I_τ. Because of this bijection, we may occasionally drop the subindex for I_σ and I_τ.

We need to distinguish three kinds of intervals:

- Intervals where $c(T)$ is strictly increasing, denoted by $\acute{I}$. We call these intervals *bijective*, since $c(T)$ is invertible. These correspond to non-straight parts of the ROC curve. Each point inside these segments is optimal for one specific cost proportion.
- Intervals where $c(T)$ is constant, denoted by I. We call these non-injective intervals *constant*. These correspond to straight parts of the ROC curve. All the points inside these segments are optimal for just one cost proportion, and we only need to consider any of them (e.g., the extremes).
- Intervals in the codomain where no value T for $c(T)$ has an image, denoted by $\dot{I}$. We call these 'intervals' *singular*, and address non-surjectiveness. In the codomain they may usually correspond to one single point, but also can correspond to an actual interval when the density functions are 0 for some intervals in the codomain. In the end, these correspond to discontinuous points of the ROC curve. The points at $(0,0)$ and $(1,1)$ are generally (but not always) discontinuous. These points are optimal for many cost proportions.

Table 7 shows how these three kinds of intervals work.

bijective $\acute{I}$	constant I	singular $\dot{I}$
$]\sigma_i,\sigma_{i+1}[$	$[\sigma_i,\sigma_{i+1}]$	$]\sigma_i,\sigma_{i+1}[$
↑↑↑↑↑	↗↖	↖↗
$]\tau_i,\tau_{i+1}[$	$]\tau_i,\tau_{i+1}[$	$[\tau_i,\tau_{i+1}]$

Table 7: Illustration for the three types of intervals.

Now we are ready to get some results:

Lemma 46 *If the model m is convex, we have that minimal expected loss can be expressed as:*

$$L^o_{U(c)}(m) = \acute{\Lambda}(m) + \dot{\Lambda}(m)$$

where:

$$\acute{\Lambda}(m) = \sum_{]\tau_i,\tau_{i+1}[\in\acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} 2c(T)\pi_0(1-F_0(T)) + 2(1-c(T))\pi_1 F_1(T)\}c'(T)dT \tag{22}$$

where $c'(T)$ is the derivative of $c(T)$ and:

$$\dot{\Lambda}(m) = \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2c\pi_0(1-F_0(\tau_i)) + 2(1-c)\pi_1 F_1(\tau_i)\}dc \quad (23)$$

$$= \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \left\{\pi_0(1-F_0(\tau_i))({\sigma_{i+1}}^2 - {\sigma_i}^2) + \pi_1 F_1(\tau_i)(2\sigma_{i+1} - {\sigma_{i+1}}^2 - 2\sigma_i + {\sigma_i}^2)\right\}. \quad (24)$$

Note that the constant intervals in I_σ are not considered (their loss is 0).

Proof We take the expression for optimal loss from Equation (13):

$$L^o_{U(c)} = \int_0^1 \min_t \{2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\}dc. \quad (25)$$

In order to calculate the minimum, we make the derivative of the min expression equal to 0:

$$\begin{aligned} 2c\pi_0(0-f_0(t)) + 2(1-c)\pi_1 f_1(t) &= 0 \\ -2c\cdot\frac{\pi_0}{\pi_1} slope(T) + 2(1-c) &= 0 \\ \frac{\pi_0}{\pi_1} slope(T) &= \frac{1-c}{c} \\ \frac{1}{c(T)} - 1 &= \frac{1-c}{c} \\ c(T) &= c. \end{aligned}$$

We now check the sign of the second derivative, which is:

$$-2c\cdot\frac{\pi_0}{\pi_1} slope'(t) = -2c \times (\frac{1}{c(T)} - 1)' = -2c\frac{-c'(T)}{{c(T)}^2} = 2c\frac{c'(T)}{{c(T)}^2}.$$

For the bijective intervals $\acute{I}_\sigma$, where the model is strictly convex and $c(T)$ is strictly decreasing, its derivative is > 0. Also, c is always between 0 and 1, so the above expression is positive, and it is a minimum. And this cannot be a 'local' minimum, since the model is convex.

For the constant intervals I_σ where the model is convex (but not strictly), this means that $c(T)$ is constant, and its derivative is 0. That means that the minimum can be found at any point T in these intervals $]\tau_i,\tau_{i+1}[$ for the same $[\sigma_i = \sigma_{i+1}]$. But their contribution to the loss will be 0, as can be seen since $c'(T)$ equals 0.

For the singular intervals $\dot{I}_\sigma$, on the contrary, all the values in each interval $]\sigma_i,\sigma_{i+1}[$ will give a minimum for the same $[\tau_i = \tau_{i+1}]$.

So we decompose the loss with the bijective and singular intervals only:

$$L^o_{U(c)}(m) = \acute{\Lambda}(m) + \dot{\Lambda}(m).$$

For the strictly convex (bijective) intervals, we now know that the minimum is at $c(T) = c$, and $c(T)$ is invertible. We can use exactly this change of variable over Equation (25) and express this for the series of intervals $]\tau_i,\tau_{i+1}[$.

$$\acute{\Lambda}(m) \quad = \quad \sum_{]\tau_i,\tau_{i+1}[\in \acute{I}_\sigma} \int_{\tau_i}^{\tau_{i+1}} 2c(T)\pi_0(1-F_0(T)) + 2(1-c(T))\pi_1 F_1(T)\}c'(T)dT.$$

which corresponds to Equation (22). Note that when there is only one bijective interval (the model is continuous and strictly convex), we have that there is only one integral in the sum and its limits go from $c^{-1}(0)$ to $c^{-1}(1)$, which in some cases can go from $-\infty$ to ∞, if the scores are not understood as probabilities.

For the singular intervals, we can work from Equation (25):

$$\dot{\Lambda}(m) \quad = \quad \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} \min\{2c\pi_0(1-F_0(t)) + 2(1-c)\pi_1 F_1(t)\}dc.$$

As said, all the values in each interval $]\sigma_i,\sigma_{i+1}[$ will give a minimum for the same $[\tau_i = \tau_{i+1}]$, so this reduces to:

$$\begin{aligned}
\dot{\Lambda}(m) \quad &= \quad \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} \{2c\pi_0(1-F_0(\tau_i)) + 2(1-c)\pi_1 F_1(\tau_i)\}dc \\
&= \quad 2 \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \left\{ \pi_0(1-F_0(\tau_i)) \int_{\sigma_i}^{\sigma_{i+1}} c\,dc + \pi_1 F_1(\tau_i) \int_{\sigma_i}^{\sigma_{i+1}} (1-c)dc \right\} \\
&= \quad 2 \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \left\{ \pi_0(1-F_0(\tau_i)) \left[\frac{c^2}{2}\right]_{\sigma_i}^{\sigma_{i+1}} + \pi_1 F_1(\tau_i) \left[c - \frac{c^2}{2}\right]_{\sigma_i}^{\sigma_{i+1}} \right\} \\
&= \quad \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \left\{ \pi_0(1-F_0(\tau_i))({\sigma_{i+1}}^2 - {\sigma_i}^2) + \pi_1 F_1(\tau_i)(2\sigma_{i+1} - {\sigma_{i+1}}^2 - 2\sigma_i + {\sigma_i}^2) \right\}.
\end{aligned}$$

which corresponds to Equation (24). ■

B.2 $c(T)$ is Idempotent

Now we work with the transformation $T \mapsto s = c(T)$. The resulting model using this transformation will be denoted by $m^{(c)}$. We will use $H_0(s)$ and $H_1(s)$ for the cumulative distributions, which are defined as follows. Since $s = c(T)$ by definition we have that $F_0(T) = H_0(c(T)) = H_0(s)$ and similarly $F_1(T) = H_1(c(T)) = H_1(s)$.

For the intervals $]\tau_i,\tau_{i+1}[$ in $\acute{I}_\tau$, we have $c(T)$ is strictly convex we just use $c^{-1}(s)$ to derive H_0 and H_1. This may imply discontinuities at τ_i or τ_{i+1} for those values of s for which constant intervals have been mapped, namely σ_i and σ_{i+1}. So, we need to define the density functions as follows. For the bijective intervals we just use $h_0(s)ds = f_0(T)dT$ and $h_1(s)ds = f_1(T)dT$ as a shorthand for a change of variable, and we can clear h_0 and h_1 using $c^{-1}(s)$. We do that using open intervals $]\tau_i,\tau_{i+1}[$ in T. These correspond to $]c(\tau_i), c(\tau_{i+1})[=]\sigma_i,\sigma_{i+1}[$.

The constant intervals are $[\tau_i, \tau_{i+1}]$ in I_τ. There is probability mass for every constant interval $[\tau_i, \tau_{i+1}]$ mapping to a point $s_i = c(\tau_i) = c(\tau_{i+1}) = \sigma_i = \sigma_{i+1}$, as follows:

$$[H_0(T)]_{\sigma_i}^{\sigma_{i+1}} = \int_{\tau_i}^{\tau_{i+1}} f_0(T)dt = [F_0(T)]_{\tau_i}^{\tau_{i+1}} = F_0(\tau_{i+1}) - F_0(\tau_i). \quad (26)$$

$$[H_1(T)]_{\sigma_i}^{\sigma_{i+1}} = \int_{\tau_i}^{\tau_{i+1}} f_1(T)dt = [F_1(T)]_{\tau_i}^{\tau_{i+1}} = F_1(\tau_{i+1}) - F_1(\tau_i). \quad (27)$$

Finally, we just define $h_0(s) = h_1(s) = 0$ for those $s \in [\sigma_i, \sigma_{i+1}] \in \dot{I}_\sigma$, since for the singular intervals there is only one point τ_i and the mass to share is 0.

This makes $m^{(c)}$ well-defined for convex models (not necessarily continuous and strictly convex).

Lemma 47 *For model $m^{(c)}$ we have that, for the non-singular intervals, $c_{m^{(c)}}(s) = \frac{\pi_1 h_1(s)}{\pi_0 h_0(s) + \pi_1 h_1(s)}$ is idempotent, that is:*

$$c_{m^{(c)}}(s) = s.$$

Proof For the bijective (strictly convex) intervals $]\tau_i, \tau_{i+1}[$ mapped into $]c(\tau_i), c(\tau_{i+1})[$, that is, $]\sigma_i, \sigma_{i+1}[$:

$$\begin{aligned} c_{m^{(c)}}(s) &= \frac{\pi_1 h_1(s)}{\pi_0 h_0(s) + \pi_1 h_1(s)} = \frac{\pi_1 h_1(s)ds}{\pi_0 h_0(s)ds + \pi_1 h_1(s)ds} \\ &= \frac{\pi_1 f_1(T)dT}{\pi_0 f_0(T)dT + \pi_1 f_1(T)dT} = \frac{\pi_1 f_1(T)}{\pi_0 f_0(T) + \pi_1 f_1(T)} = c(T) = s. \end{aligned}$$

For the points $s_i = c(\tau_i) = c(\tau_{i+1})$ corresponding to constant intervals, we have that using Equation (26) and (27):

$$c_{m^{(c)}}(s_i) = \frac{\pi_1 h_1(s_i)}{\pi_0 h_0(s_i) + \pi_1 h_1(s_i)} = \frac{\pi_1 [F_1(T)]_{\tau_i}^{\tau_{i+1}}}{\pi_0 [F_0(T)]_{\tau_{i+1}}^{a_{i+1}} + \pi_1 [F_1(T)]_{\tau_i}^{\tau_{i+1}}}.$$

Since $c(T)$ is constant in the interval $]\tau_i, \tau_{i+1}[$, we have:

$$c_{m^{(c)}}(s_i) = \frac{\pi_1 f_1(T)}{\pi_0 f_0(T) + \pi_1 f_1(T)} = c(T) = s_i.$$

■

B.3 Main Result

Finally, we are ready to prove the theorem treating the three kinds of intervals.

Theorem 48 *(Theorem 29 in the paper) For every convex model m, we have that:*

$$L^o_{U(c)}(m) = RL(m).$$

Proof

Let us start from Lemma 46:

$$L^o_{U(c)}(m) = \acute{\Lambda}(m) + \dot{\Lambda}(m).$$

working with Equation (22) first for the bijective intervals:

$$\acute{\Lambda}(m) = \sum_{]\tau_i,\tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} 2c(T)\pi_0(1-F_0(T)) + 2(1-c(T))\pi_1 F_1(T)\}c'(T)dT.$$

Since this only includes the bijective intervals, we can use the correspondence between the H and the F, and making the change $s = c(T)$.

$$\begin{aligned}
\acute{\Lambda}(m) &= \sum_{]\tau_i,\tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} 2c(T)\pi_0(1-H_0(c(T))) + 2(1-c(T))\pi_1 H_1(c(T))\}c'(T)dT \\
&= \sum_{]c(\tau_i),c(\tau_{i+1})[\in \acute{I}_\sigma} \int_{c(\tau_i)}^{c(\tau_{i+1})} 2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds \\
&= \sum_{]\sigma_i,\sigma_{i+1}[\in \acute{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds.
\end{aligned}$$

and now working with Equation (23) for the singular intervals and also using the correspondence between the H and the F:

$$\begin{aligned}
\dot{\Lambda}(m) &= \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2c\pi_0(1-F_0(\tau_i)) + 2(1-c)\pi_1 F_1(\tau_i)\}dc \\
&= \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2c\pi_0(1-H_0(c(\tau_i))) + 2(1-c)\pi_1 H_1(c(\tau_i))\}dc \\
&= \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2s\pi_0(1-H_0(\sigma_i)) + 2(1-s)\pi_1 H_1(\sigma_i)\}ds.
\end{aligned}$$

The last step also uses the renaming of the variable. But since $h_0(s) = h_1(s) = 0$ for the singular intervals, we have that $H_0(s)$ and $H_1(s)$ are constant in these intervals, so this can be rewritten as:

$$\dot{\Lambda}(m) = \sum_{]\sigma_i,\sigma_{i+1}[\in \dot{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds.$$

Putting $\acute{\Lambda}(m)$ and $\dot{\Lambda}(m)$ together, because the constant intervals (I_σ) have length 0 (and loss 0), we have:

$$L^o_{U(c)}(m) = \sum_{]\sigma_i,\sigma_{i+1}[\in I_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} 2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds.$$

We can join the integrals into a single one, even though the whole integral has to be calculated by intervals if it is discontinuous:

$$\begin{aligned}
L^o_{U(c)}(m) &= \int_{\sigma_0}^{\sigma_n} \{2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds \\
&= \int_0^1 \{2s\pi_0(1-H_0(s)) + 2(1-s)\pi_1 H_1(s)\}ds.
\end{aligned}$$

By Theorem 14 (Equation 10) in the paper (and also because this theorem holds pointwise) we have that the last expression equals the Brier score, so this leads to:

$$L^o_{U(c)}(m) = BS(m^{(c)}).$$

And now we have that using Definition 8 for the *BS*:

$$BS(m^{(c)}) = \int_0^1 \{\pi_0 s^2 h_0(s) + \pi_1 (1-s)^2 h_1(s)\} ds.$$

This is 0 when $h_0(s) = h_1(s) = 0$, so we can ignore the singular intervals for the rest of the proof. The calibration loss for model $m^{(c)}$ can be expanded as follows, and using Lemma 47 (which is applicable except for non-singular intervals) we have:

$$\begin{aligned} CL(m^{(c)}) &= \int_0^1 \left(s - \frac{\pi_1 h_1(s)}{\pi_0 h_0(s) + \pi_1 h_1(s)} \right)^2 (\pi_0 h_0(s) + \pi_1 h_1(s))\, ds \\ &= \int_0^1 (s-s)^2 (\pi_0 h_0(s) + \pi_1 h_1(s))\, ds = 0. \end{aligned}$$

So, we have that:

$$L^o_{U(c)}(m) = RL(m^{(c)}). \tag{28}$$

And now we need to work with *RL*:

$$\begin{aligned} RL(m^{(c)}) &= \int_0^1 \frac{\pi_1 h_1(s) \pi_0 h_0(s)}{\pi_0 h_0(s) + \pi_1 h_1(s)} ds = \int_0^1 \pi_0 h_0(s) \frac{\pi_1 h_1(s)}{\pi_0 h_0(s) + \pi_1 h_1(s)} ds \\ &= \int_0^1 \pi_0 h_0(s) c_{m^{(c)}}(s) ds = \int_0^1 \pi_0 h_0(s) s ds. \end{aligned}$$

The last step applies Lemma 47 again.

We now need to treat the bijective and the constant intervals separately, otherwise the integral cannot be calculated when h_0 and h_1 are discontinuous.

$$RL(m^{(c)}) = \sum_{]\sigma_i, \sigma_{i+1}[\in \acute{I}_\sigma} \int_{\sigma_i}^{\sigma_{i+1}} \pi_0 h_0(s) s ds + \sum_{]\sigma_i, \sigma_{i+1}[\in I_\sigma} \pi_0 h_0(\sigma_i) \sigma_i.$$

We apply the variable change $s = c(T)$ for the expression on the left:

$$\begin{aligned} \sum_{]c(\tau_i), c(\tau_{i+1})[\in \acute{I}_\sigma} \int_{c(\tau_i)}^{c(\tau_{i+1})} \pi_0 h_0(s) s ds &= \sum_{]\tau_i, \tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 h_0(c(T)) c(T) \frac{dc(T)}{dT} dT \\ &= \sum_{]\tau_i, \tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 h_0(c(T)) \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} \frac{dc(T)}{dT} dT \\ &= \sum_{]\tau_i, \tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 h_0(c(T)) \frac{dc(T)}{dT} \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT \\ &= \sum_{]\tau_i, \tau_{i+1}[\in \acute{I}_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 f_0(T) \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT. \end{aligned}$$

We now work with the expression on the right using Equation (26):

$$\begin{aligned}\sum_{]c(\tau_i),c(\tau_{i+1})[\in I_\sigma} \pi_0 h_0(c(\tau_i))c(\tau_i) &= \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \pi_0 [F_0(T)]_{\tau_i}^{\tau_{i+1}} c(\tau_i) = \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \pi_0 \int_{\tau_i}^{\tau_{i+1}} f_0(T)dT\, c(\tau_i) \\ &= \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 f_0(T) c(T) dT \\ &= \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 f_0(T) \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT.\end{aligned}$$

The change from $c(\tau_i)$ to $c(T)$ inside the integral can be performed since $c(T)$ is constant, because here we are working with the constant intervals.

Putting everything together again:

$$\begin{aligned}RL(m^{(c)}) &= \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 f_0(T) \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT + \sum_{]\tau_i,\tau_{i+1}[\in I_\tau} \int_{\tau_i}^{\tau_{i+1}} \pi_0 f_0(T) \frac{\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT \\ &= \int_{\tau_0}^{\tau_n} \frac{\pi_0 f_0(T)\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT = \int_{-\infty}^{\infty} \frac{\pi_0 f_0(T)\pi_1 f_1(T)}{\pi_1 f_1(T) + \pi_0 f_0(T)} dT = RL(m).\end{aligned}$$

This and Equation (28) complete the proof. ■

References

N. M. Adams and D. J. Hand. Comparing classifiers when the misallocation costs are uncertain. *Pattern Recognition*, 32(7):1139–1147, 1999.

T. A. Alonzo, M. S. Pepe, and T. Lumley. Estimating disease prevalence in two-phase studies. *Biostatistics*, 4(2):313–326, 2003.

M. Ayer, H.D. Brunk, G.M. Ewing, W.T. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 5:641–647, 1955.

A. Bella, C. Ferri, J. Hernandez-Orallo, and M.J. Ramirez-Quintana. Calibration of machine learning models. In *Handbook of Research on Machine Learning Applications*, pages 128–146. IGI Global, 2009.

A. Bella, C. Ferri, J. Hernández-Orallo, and M.J. Ramírez-Quintana. Quantification via probability estimators. In *2010 IEEE International Conference on Data Mining*, pages 737–742. IEEE, 2010.

G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

N. Brümmer. Measuring, refining and calibrating speaker and language information extracted from speech. *Ph.D. Dissertation, Department of Electrical and Electronic Engineering, University of Stellenbosch*, 2010.

A. Buja, W. Stuetzle, and Y. Shen. Loss functions for binary class probability estimation: structure and applications. http://www-stat.wharton.upenn.edu/∼buja/PAPERS/paper-proper-scoring.pdf, 2005.

I. Cohen and M. Goldszmidt. Properties and benefits of calibrated classifiers. *Knowledge Discovery in Databases: PKDD 2004*, pages 125–136, 2004.

C. Drummond and R. C. Holte. Explicitly representing expected cost: an alternative to ROC representation. In *Knowledge Discovery and Data Mining*, pages 198–207, 2000.

C. Drummond and R. C. Holte. Cost curves: an improved method for visualizing classifier performance. *Machine Learning*, 65(1):95–130, 2006.

C. Elkan. The foundations of cost-sensitive searning. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence (IJCAI-01)*, pages 973–978, San Francisco, CA, 2001.

T. Fawcett. Using rule sets to maximize ROC performance. In *2001 IEEE International Conference on Data Mining (ICDM-01)*, pages 131–138, 2001.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

T. Fawcett and A. Niculescu-Mizil. PAV and the ROC convex hull. *Machine Learning*, 68(1): 97–106, July 2007.

T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3): 291–316, 1997.

C. Ferri, P.A. Flach, J. Hernández-Orallo, and A. Senad. Modifying ROC curves to incorporate predicted probabilities. In *Second Workshop on ROC Analysis in ML, ROCML*, pages 33–40, 2005.

C. Ferri, J. Hernández-Orallo, and R. Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38, 2009. ISSN 0167-8655.

P. A. Flach. The geometry of ROC space: understanding machine learning metrics through ROC isometrics. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003)*, pages 194–201, 2003.

P. A. Flach. *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. Cambridge University Press, to appear, 2012.

P. A. Flach and E. T. Matsubara. A simple lexicographic ranker and probability estimator. In *18th European Conference on Machine Learning*, pages 575–582. Springer, 2007.

P. A. Flach, J. Hernández-Orallo, and C. Ferri. A coherent interpretation of AUC as a measure of aggregated classification performance. In *Proceedings of the 28th International Conference on Machine Learning, ICML2011*, 2011.

G. Forman. Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206, 2008.

J. H. Friedman. On bias, variance, 0/1loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1):55–77, 1997.

M. Gebel. *Multivariate Calibration of Classifier Scores into the Probability Space*. PhD thesis, University of Dortmund, 2009.

T. Gneiting and A.E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. ISSN 0162-1459.

I. J. Good. Rational decisions. *Journal of the Royal Statistical Society, Series B*, 14:107–114, 1952.

D. J. Hand. *Construction and Assessment of Classification Rules*. John Wiley & Sons Inc, 1997.

D. J. Hand. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Machine Learning*, 77(1):103–123, 2009.

D. J. Hand. Evaluating diagnostic tests: the area under the ROC curve and the balance of errors. *Statistics in Medicine*, 29(14):1502–1510, 2010.

D. J. Hand and C. Anagnostopoulos. When is the area under the ROC curve an appropriate measure of classifier performance? *Technical report, Department of Mathematics, Imperial College, London*, 2011.

D. J. Hand and C. Anagnostopoulos. A better Beta for the H measure of classification performance. *arXiv:1202.2564v1 [stat] 12 Feb 2012*, page 9, 2012.

J. Hernández-Orallo, P. A. Flach, and C. Ferri. Brier curves: a new cost-based visualisation of classifier performance. In *Proceedings of the 28th International Conference on Machine Learning, ICML2011*, 2011.

J. Hernández-Orallo, P. Flach, and C. Ferri. ROC curves in cost space. In *Submitted*, 2012.

N. Lachiche and P. Flach. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. In *International Conference on Machine Learning*, pages 416–423, 2003.

G. Lebanon and J. D. Lafferty. Cranking: combining rankings using conditional probability models on permutations. In *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002)*, pages 363–370, 2002.

A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Fifth European Conference on Speech Communication and Technology*, 1997.

A. H. Murphy. A note on the utility of probabilistic predictions and the probability score in the cost-loss ratio decision situation. *Journal of Applied Meteorology*, 5:534–536, 1966. ISSN 0894-8763.

A. H. Murphy. Measures of the utility of probabilistic predictions in cost-loss ratio decision situations in which knowledge of the cost-loss ratios is incomplete. *Journal of Applied Meteorology*, 8:863–873, 1969. ISSN 0894-8763.

A. H. Murphy. A new vector partition of the probability score. *Journal of Applied Meteorology*, 12: 595–600, 1973.

A. H. Murphy and R. L. Winkler. Scoring rules in probability assessment and evaluation. *Acta Psychologica*, 34:273–286, 1970.

J. M. Murphy, D. M. Berwick, M. C. Weinstein, J. F. Borus, S. H. Budman, and G. L. Klerman. Performance of screening and diagnostic tests: application of receiver operating characteristic analysis. *Archives of General Psychiatry*, 44(6):550, 1987.

J. Neyman. Contribution to the theory of sampling human populations. *Journal of the American Statistical Association*, 33(201):101–116, 1938.

A. Niculescu-Mizil and R. Caruana. Obtaining calibrated probabilities from boosting. In *The 21st Conference on Uncertainty in Artificial Intelligence (UAI 05), AUAI Press*, pages 413–420. AUAI Press, 2005.

A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine learning*, pages 625–632. ACM, 2005.

G. Piatetsky-Shapiro and B. Masand. Estimating campaign benefits and modeling lift. In *Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 193. ACM, 1999.

J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, Boston, 1999.

R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. A survey on graphical methods for classification predictive performance evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 23:1601–1618, 2011.

F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42 (3):203–231, 2001.

M. D. Reid and R. C. Williamson. Composite binary losses. *The Journal of Machine Learning Research*, 11:2387–2422, 2010.

M. D. Reid and R. C. Williamson. Information, divergence and risk for binary experiments. *The Journal of Machine Learning Research*, 12:731–817, 2011.

T. Robertson, F. Wright, and R. Dykstra. *Order Restricted Statistical Inference*. John Wiley and Sons, New York, 1988.

S. Rüping. Robust probabilistic calibration. *Machine Learning: ECML 2006*, pages 743–750, 2006.

J. A. Swets, R. M. Dawes, and J. Monahan. Better decisions through science. *Scientific American*, 283(4):82–87, October 2000.

A. Tenenbein. A double sampling scheme for estimating from binomial data with misclassifications. *Journal of the American Statistical Association*, pages 1350–1361, 1970.

P. Turney. Types of cost in inductive concept learning. *Canada National Research Council Publications Archive*, 2000.

S. Wieand, M.H. Gail, B.R. James, and K.L. James. A family of nonparametric statistics for comparing diagnostic markers with paired or unpaired data. *Biometrika*, 76(3):585–592, 1989.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 609–616, 2001.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699, 2002.

Journal of Machine Learning Research 13 (2012) 2871-2902 Submitted 8/11; Revised 6/12; Published 10/12

Local and Global Scaling Reduce Hubs in Space

Dominik Schnitzer DOMINIK.SCHNITZER@OFAI.AT
Arthur Flexer ARTHUR.FLEXER@OFAI.AT
Austrian Research Institute for Artificial Intelligence (OFAI)
Freyung 6/6, 1010 Vienna, Austria

Markus Schedl MARKUS.SCHEDL@JKU.AT
Gerhard Widmer GERHARD.WIDMER@JKU.AT
Department of Computational Perception
Johannes Kepler University
Altenbergerstraße 69, 4040 Linz, Austria

Editor: Gert Lanckriet

Abstract

'Hubness' has recently been identified as a general problem of high dimensional data spaces, manifesting itself in the emergence of objects, so-called hubs, which tend to be among the k nearest neighbors of a large number of data items. As a consequence many nearest neighbor relations in the distance space are asymmetric, that is, object y is amongst the nearest neighbors of x but not vice versa. The work presented here discusses two classes of methods that try to symmetrize nearest neighbor relations and investigates to what extent they can mitigate the negative effects of hubs. We evaluate local distance scaling and propose a global variant which has the advantage of being easy to approximate for large data sets and of having a probabilistic interpretation. Both local and global approaches are shown to be effective especially for high-dimensional data sets, which are affected by high hubness. Both methods lead to a strong decrease of hubness in these data sets, while at the same time improving properties like classification accuracy. We evaluate the methods on a large number of public machine learning data sets and synthetic data. Finally we present a real-world application where we are able to achieve significantly higher retrieval quality.

Keywords: local and global scaling, shared near neighbors, hubness, classification, curse of dimensionality, nearest neighbor relation

1. Introduction

In a recent publication in this journal, Radovanović et al. (2010) describe the so-called 'hubness' phenomenon and explore it as a general problem of machine learning in high-dimensional data spaces. Hubs are data points which keep appearing unwontedly often as nearest neighbors of a large number of other data points. This effect is particularly problematic in algorithms for similarity search (for example, similarity-based recommenders), as the same similar objects are found over and over again and other objects are never recommended. The effect has been shown to be a natural consequence of high dimensionality and as such is yet another aspect of the curse of dimensionality (Bellman, 1961).

A direct consequence of the presence of hubs is that a large number of nearest neighbor relations in the distance space are asymmetric, that is, object y is amongst the nearest neighbors of x but not vice versa. A hub is by definition the nearest neighbor of a large number of objects, but

these objects cannot possibly all be the nearest neighbor of the hub. This observation connects the hub problem to methods that attempt to symmetrize nearest neighbor relations, such as 'shared near neighbors' (Jarvis and Patrick, 1973) and 'local scaling' (Zelnik-Manor and Perona, 2005). While these methods require knowledge of the local neighborhood of every data point, we propose a global variant that combines the idea of 'shared near neighbor' approaches with a transformation of distances to nearest neighbor 'probabilities' to define a concept we call *Mutual Proximity*. The approach is unsupervised and transforms an arbitrary distance function to a probabilistic similarity (distance) measure. Contrary to the local variants, this new approach lends itself to fast approximation for very large data bases and enables easy combination of multiple distance spaces due to its probabilistic nature.

In experiments with a large number of public machine learning databases we show that both local and global scaling methods lead to: (i) a significant decrease of hubness, (ii) an increase of k-nearest neighbor classification accuracy, and (iii) a strengthening of the pairwise class stability of the nearest neighbors. To demonstrate the practical relevance, we apply our global scaling algorithm to a real-world music recommendation system and show that doing so significantly improves the retrieval quality.

To permit other researchers to reproduce the results of this paper, all databases and the main evaluation scripts used in this work have been made publicly available.[1]

2. Related Work

The starting point for our investigations is a field where the existence of hubs has been well documented and established, namely, Music Information Retrieval (MIR). One of the central notions in MIR is that of music similarity. Proper modeling of music similarity is at the heart of many applications involving the automatic organization and processing of music data bases. In Aucouturier and Pachet (2004), hub songs were defined as songs which are, according to an audio similarity function, similar to very many other songs and therefore keep appearing unwontedly often in recommendation lists, preventing other songs from being recommended at all. Such songs that do not appear in any recommendation list have been termed 'orphans'. Similar observations about false positives in music recommendation that are not perceptually meaningful have been made elsewhere (Pampalk et al., 2003; Flexer et al., 2010; Karydis et al., 2010). The existence of the hub problem has also been reported for music recommendation based on collaborative filtering instead of audio content analysis (Celma, 2008). Similar effects have been observed in image (Doddington et al., 1998; Hicklin et al., 2005) and text retrieval (Radovanović et al., 2010), making this phenomenon a general problem in multimedia retrieval and recommendation.

In the MIR literature, Berenzweig (2007) first suspected a connection between the hub problem and the high dimensionality of the feature space. The hub problem was seen as a direct result of the curse of dimensionality (Bellman, 1961), a term that refers to a number of challenges related to the high dimensionality of data spaces. Radovanović et al. (2010) were able to provide more insight by linking the hub problem to the property of *concentration* (François et al., 2007) which occurs as a natural consequence of high dimensionality. Concentration is the surprising characteristic of all points in a high dimensional space to be at almost the same distance to all other points in that space. It is usually measured as a ratio between some measure of spread and magnitude. For example, the ratio between the standard deviation of all distances to an arbitrary reference point and the mean of

1. Databases and scripts can be found at `http://www.ofai.at/~dominik.schnitzer/mp`.

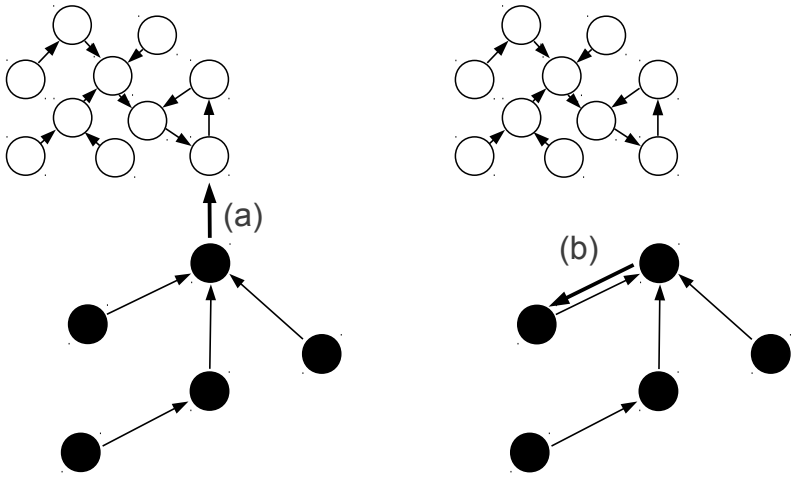
these distances. If this ratio converges to zero as the dimensionality goes to infinity, the distances are said to concentrate. For example, in the case of the Euclidean distance and growing dimensionality, the standard deviation of distances converges to a constant while the mean keeps growing. Thus the ratio converges to zero and the distances are said to concentrate.

The effect of distance concentration has been studied for Euclidean spaces and other ℓ^p norms (Aggarwal et al., 2001; François et al., 2007). Radovanović et al. (2010) presented the argument that in the finite case, due to this phenomenon some points are expected to be closer to the data-set mean than other points and are at the same time closer, on average, to all other points. Such points closer to the data-set mean have a high probability of being hubs, that is, of appearing in nearest neighbor lists of many other points. Points which are further away from that mean have a high probability of being 'orphans', that is, never appearing in any nearest neighbor list.

Nearest neighbor search is an essential method in many areas of computer science, such as pattern recognition, multimedia search, vector compression, computational statistics and data mining (Shakhnarovich et al., 2006) and, of course, information retrieval and recommendation. It is a well defined task: given an object x, find the most similar object in a collection of related objects. In the case of recommendation, the k most similar objects are retrieved with $k << n$ (n being the number of all objects in the data base). Since hubs appear in very many nearest neighbor lists, they tend to render many nearest neighbor relations asymmetric, that is, a hub y is the nearest neighbor of x, but the nearest neighbor of the hub y is another point a ($a \neq x$). This is because hubs are nearest neighbors to very many data points but only k data points can be nearest neighbors to a hub since the size of a nearest neighbor list is fixed. This behavior is especially problematic in classification or clustering if x and y belong to the same class but a does not, violating what Bennett et al. (1999) called the *pairwise stability* of clusters. Radovanović et al. (2010) coined the term *bad hubs* for points that show a disagreement of class information for the majority of data points they are nearest neighbors to. Figure 1 illustrates the effect: although a is, in terms of the distance measure, the correct answer to the nearest neighbor query for y, it may be beneficial to use a distance measure that enforces symmetric nearest neighbors. Thus a small distance between two objects should be returned only if their nearest neighbors concur.

This links the hub problem to 'shared near neighbor' (SNN) approaches, which try to symmetrize nearest neighbor relations. The first work to use common near neighbor information dates back to the 1970s. Jarvis and Patrick (1973) proposed a 'shared near neighbor' similarity measure to improve the clustering of 'non-globular' clusters. As the name suggests, the shared near neighbor (SNN) similarity is based on computing the overlap between the k nearest neighbors of two objects. Shared near neighbor similarity was also used by Ertöz et al. (2003) to find the most representative items in a set of objects. Pohle et al. (2006) define a related similarity measure based on the rank of nearest neighbors. They call their method 'proximity verification' and use it to enhance audio similarity search. Jin et al. (2006) use the reverse nearest neighbor (RNN) relation to define a general measure for outlier detection.

Related to SNN approaches are local scaling methods, which use local neighborhood information to rescale distances between data points. The intention is to find specific scaling parameters for each point, to be used to tune the pairwise distances in order to account for different local densities (scales) of the neighborhoods. Local scaling in this sense was first introduced as part of a spectral clustering method by Zelnik-Manor and Perona (2005). It transforms arbitrary distances using the distance between object x and its k'th nearest neighbor (see Section 3.1 below). In the context of image retrieval, Jegou et al. (2010) describe a related method called 'contextual dissimilarity measure'



(a) Original nearest neighbor relations

(b) Desired nearest neighbor relations

Figure 1: Schematic plot of two classes (black/white filled circles). Each circle has its nearest neighbor marked with an arrow: (a) violates the *pairwise stability* clustering assumption, (b) fulfills the assumption. In many classification and retrieval scenarios, (b) would be the desired nearest neighbor relation for the data set.

(CDM) and show that it reduces the error rates of the retrieval algorithm significantly, observing that "*the neighborhood symmetry rate increases*", while at the same time "*the percentage of never seen images decreases*", and in addition that "*the most frequent image is returned 54 times in the first 10 positions with the CDM, against 1062 times using the standard L1 distance*". While they do not explicitly make reference to the notion of hubs, their observations indicate the potential of local distance scaling to mitigate hub-related problems.

3. Scaling Methods

In the previous section we have seen that (i) the emergence of hubs leads to asymmetric nearest neighbor relations and that (ii) literature already contains hints that local scaling methods seem to improve the situation. However a detailed analysis of these facts and a systematic connection to the investigations of Radovanović et al. (2010) has not yet been done.

In what follows we review the local scaling methods and introduce a new global variant, which is also very simple to use. Due to its probabilistic modeling it possesses certain advantages over the local variant. Both methods are evaluated in regard to their effects on hubness in Section 4.

All the methods described here assume an underlying distance (divergence) measure with the following properties:

Definition 1 *Given a non-empty set M with n objects, each element $m_x \in M$ is assigned an index $x = 1 \ldots n$. We define a divergence measure $d : M \times M \to \mathbb{R}$ satisfying the condition of non-negativity in its distances:*

- *non-negativity:* $d(m_x, m_y) \geq 0, \quad m_x, m_y \in M,$

Individual objects $m_x \in M$ are referenced in the text by their index x. The distance between two objects x and y is denoted as $d_{x,y}$.

3.1 Local Scaling

Local scaling (Zelnik-Manor and Perona, 2005) transforms arbitrary distances to so-called *affinities* (that is, similarities) according to:

$$LS(d_{x,y}) = \exp\left(-\frac{{d_{x,y}}^2}{\sigma_x \sigma_y}\right), \tag{1}$$

where σ_x denotes the distance between object x and its k'th nearest neighbor. $LS(d_{x,y})$ tends to make neighborhood relations more symmetric by including local distance statistics of both data points x and y in the scaling. The exponent in Equation 1 can be rewritten as ${d_{x,y}}^2/\sigma_x\sigma_y = (d_{x,y}/\sigma_x)(d_{x,y}/\sigma_y)$: only when both parts in this product are small will the locally scaled similarity $LS(d_{x,y})$ be high. That is, x and y will be considered close neighbors only if the distance $d_{x,y}$ is small relative to both local scales σ_x and σ_y. Jegou et al. (2007) introduce a closely related variant called non-iterative contextual dissimilarity measure (NICDM). Instead of using the distance to the k'th nearest neighbor to rescale the distances, the average distance of the k nearest neighbors is used. This should return more stable scaling numbers and will therefore be used in all our evaluations. The non-iterative contextual dissimilarity measure (NICDM) transforms distances according to:

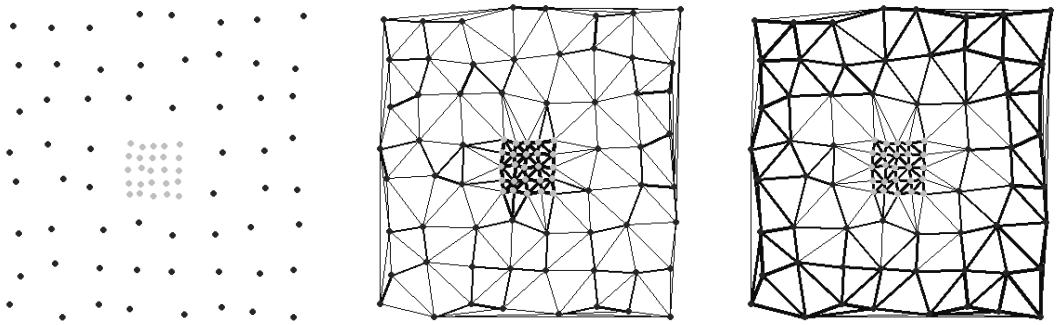
$$NICDM(d_{x,y}) = \frac{d_{x,y}}{\sqrt{\mu_x \mu_y}},$$

where μ_x denotes the average distance to the k nearest neighbors of object x. The iterative version of this algorithm performs the same transformation multiple times until a stopping criterion is met. Since these iterations yield only very minor improvements at the cost of increased computation time, we used the non-iterative version in our evaluations.

3.2 Global Scaling - Mutual Proximity

In this section we introduce a global scaling method that is based on: (i) transforming a distance between points x and y into something that can be interpreted as the probability that y is the closest neighbor to x given the distribution of the distances of all points to x in the data base; and (ii) combining these probabilistic distances from x to y and y to x via their joint probability. The result is a general unsupervised method to transform arbitrary distance matrices to matrices of probabilistic *mutual proximity* (MP). In contrast to local scaling methods, which use the local neighborhood information, MP uses information about all objects—thus the term global scaling.

The general idea of MP is to reinterpret the original distance space so that two objects sharing similar nearest neighbors are more closely tied to each other, while two objects with dissimilar neighborhoods are repelled from each other. This is done by reinterpreting the distance of two objects as a mutual proximity in terms of their distribution of distances.



(a) Original space: A tight cluster (gray dots in the center) placed in a loose background cluster (black dots).

(b) The affinity of two points in the original space is indicated by the thickness of the line connecting two points.

(c) Affinities after applying Mutual Proximity to the distances.

Figure 2: The effect of scaling techniques. Objects with similar nearest neighbors are tied together closely, while objects with dissimilar neighbors are repelled.

Figure 2 illustrates the effect of this reinterpretation in an example. The effect which can be seen here is similar to the intuitive repair of nearest neighbor relation as it was discussed in the beginning in Section 2 (Figure 1).

Figure 2a plots points from two classes on a two dimensional plane. A tight cluster (the gray dots in the center) is placed in a loose background cluster (black dots). Figure 2b connects close neighbors with lines according to a Delaunay triangulation.[2] The thickness of the lines shows the affinity of two neighboring points according to their Euclidean distance. The third plot (Figure 2c) plots the affinities after applying MP. It can be clearly seen that points from the loose cluster as well as points from the tight cluster now both have a high intra-class affinity. However, at the cluster borders there is weak affinity and strong separation as points from the tight cluster have different nearest neighbors than points from the background cluster.

This visible increase in class separation can also be measured in terms of classification rates. Simple two class k-nearest neighbor classification (tight vs. loose cluster) with this artificially generated data yields the following results: In the original space 96.4% of the nearest neighbors (at $k = 1$) are classified correctly; after applying MP, all (100%) of the nearest neighbors can be classified correctly. For $k = 5$ the classification rate increases from 95.2% to 98.8%.

3.2.1 COMPUTING MUTUAL PROXIMITY (MP)

To compute MP, we assume that the distances $d_{x,i=1..n}$ from an object x to all other objects in our collection follow a certain probability distribution. For example, Casey et al. (2008) and Cai et al. (2007) show that the ℓ^p distances they compute follow a Gamma distribution. Ferencz et al. (2005) used the Gamma distribution to model ℓ^2 distances from image regions. Pękalska and Duin (2000) show in general that based on the central limit theorem and if the feature vectors are independent and

2. A Delaunay triangulation ensures that the circumcircle associated with each triangle contains no other point in its interior, that is, no lines cross. This restriction is helpful for visualization purposes.

identically distributed (i.i.d.), their ℓ^2 distances approximately follow a normal distribution. As real data is not i.i.d., this can not be assumed. We, however, note that the accuracy of this approximation increases with increasing intrinsic dimensionality (François et al., 2007).

Under the assumption that all distances in a data set follow a certain distribution, any distance $d_{x,y}$ can now be reinterpreted as the probability of y being the nearest neighbor of x, given their distance $d_{x,y}$ and the probability distribution $P(X)$. $P(X)$ is defined by the distances of x to all other objects in the collection. In fact the probability that a randomly drawn element z will have a distance $d_{x,z} > d_{x,y}$ can then be computed:

$$P(X > d_{x,y}) = 1 - P(X \leq d_{x,y}) = 1 - \mathcal{F}_x(d_{x,y}).$$

$\mathcal{F}_x$ denotes the cumulative distribution function (cdf) which is assumed for the distribution of distances $d_{x,i=1..n}$. This way the probability of an element being a nearest neighbor of x increases with decreasing distance.

For illustration purposes we assume that in our collection the distances are normally distributed. Figure 3a shows a schematic plot of the probability density function (pdf) that was estimated for the distances of some object x. The mean distance or expected distance from x ($\hat{\mu}_x$) is in the center of the density function. Objects with a small distance to x (that is, objects with high similarity in the original space) find their distance towards the left of the density function. Note that the leftmost possible distance in this sketch is $d_{x,x} = 0$.[3] Figure 3b plots the probability of y being the nearest neighbor of x given $d_{x,y}$ (the gray filled area). The probability increases the smaller the distance to x is, or the farther left its distance is on the x-axis of the pdf.

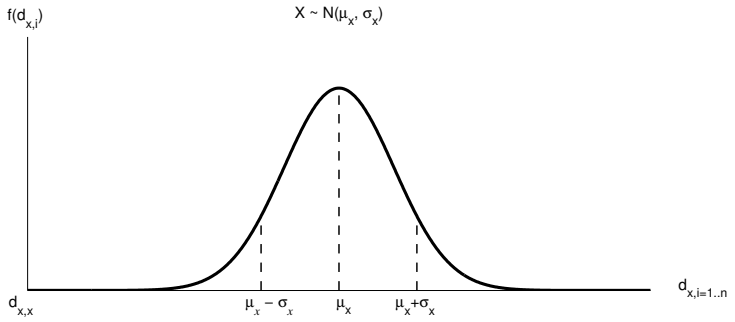
Note that this reinterpretation naturally leads to asymmetric probabilities for a given distance, as the distance distribution estimated for x may be different from the one estimated for y; x might be the nearest neighbor of y but not vice versa. Contrary to the original distances the probabilities now encode this asymmetric information. This allows for a convenient way to combine the asymmetric probabilities into a single expression, expressing the probability of x being a nearest neighbor of y and vice versa.

Definition 2 *We compute the probability that y is the nearest neighbor of x given $P(X)$ (the pdf defined by the distances $d_{x,i=1...n}$) and x is the nearest neighbor of y given $P(Y)$ (the pdf defined by the distances $d_{y,i=1...n}$), with their joint distribution $P(X,Y)$. The resulting probability is called* ***Mutual Proximity (MP)****:*

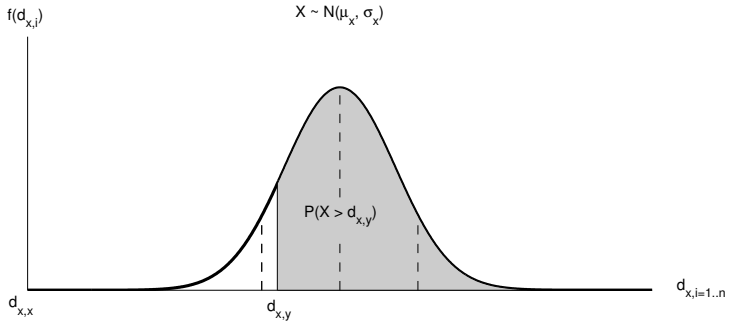
$$\begin{aligned} MP(d_{x,y}) &= P(X > d_{x,y} \cap Y > d_{y,x}) \\ &= 1 - P(X \leq d_{x,y} \cup Y \leq d_{x,y}) \\ &= 1 - \left[P(X \leq d_{x,y}) + P(Y \leq d_{y,x}) - P(X \leq d_{x,y} \cap Y \leq d_{y,x})\right]. \end{aligned}$$

Figure 4 illustrates MP for the distance $d_{x,y}$ and the joint distance distribution of X and Y, $P(X,Y)$. Each point of the plot refers to an object in the collection and its distance to points x and y. The shaded area (II) then defines the probability which is computed by MP. Sectors I+III correspond

3. Strictly speaking, then, the interpretation of this as a normal distribution is incorrect, since distances < 0 are not possible. However, we find the interpretation useful as a metaphor that helps understand why it makes sense to combine different views. We will do so in this section.



(a) The closer other elements are to x, the more to the left is their distance located on the x-axis of the density function plot. The leftmost possible observation in the data is the distance $d_{x,x} = 0$.



(b) The shaded area shows the probability that y is the nearest neighbor of x based on the distance $d_{x,y}$ and X. The closer y is to x (the smaller $d_{x,y}$) the higher the probability.

Figure 3: Schematic plot of the probability density function of a normal distribution which was estimated from the distances $d_{x,i=1...n}$: $X \backsim N(\hat{\mu}_x, \hat{\sigma}_x)$.

to the probability of x being the nearest neighbor of y, IV+III to the probability of y being a nearest neighbor of x and III to their joint probability:

$$II = MP(d_{x,y}) = 1 - [(I + III) + (IV + III) - III].$$

It is straightforward to compute MP using the empirical distribution, as illustrated in Figure 4. If the number of observations is large enough, we will tend to model the true underlying distribution closely. Computing MP for a given distance $d_{x,y}$ in a collection of n objects and using the empirical

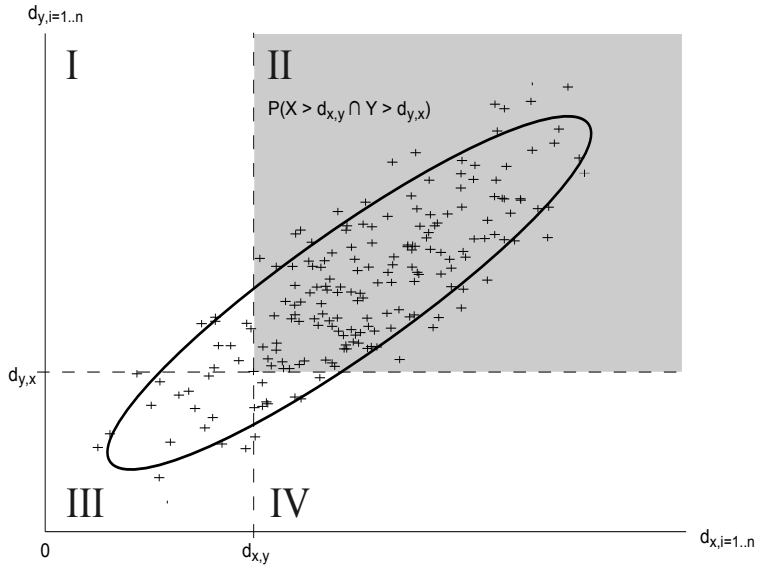


Figure 4: Visualizing the Mutual Proximity for the two points x, y, and their distances $d_{x,y}, d_{y,x}$.

distribution boils down to simply counting the number of objects j having a distance to x and y which is greater than $d_{x,y}$:

$$MP(d_{x,y}) = \frac{\left|\left\{j : d_{x,j} > d_{x,y}\right\} \cap \left\{j : d_{y,j} > d_{y,x}\right\}\right|}{n}.$$

For distances where the underlying distribution is known, estimating its parameters can be straightforward. For example, the parameters of normal distributions $N(\hat{\mu}, \hat{\sigma}^2)$, or Gamma distributions $\Gamma(\hat{k}, \hat{\theta})$ can be estimated quickly with the sample mean $\hat{\mu}_x$ and variance $\hat{\sigma}_x^2$ of the distances $d_{x,i=1..n}$:

$$\begin{aligned} N_x \backsim \quad & \hat{\mu}_x = \frac{1}{n}\sum_{i=1}^{n} d_{x,i}, \qquad \hat{\sigma}_x^2 = \frac{1}{n}\sum_{i=1}^{n}(d_{x,i} - \hat{\mu}_x)^2, \\ \Gamma_x \backsim \quad & \hat{k}_x = \frac{\hat{\mu}_x^2}{\hat{\sigma}_x^2}, \qquad \hat{\theta}_x = \frac{\hat{\sigma}_x^2}{\hat{\mu}_x}. \end{aligned} \tag{2}$$

In our experiments we will generally estimate MP directly from the empirical distribution. In addition we will also evaluate MP with different underlying parametric distance distributions, such as the Gauss or Gamma distribution (Section 4).

3.2.2 APPROXIMATIONS

The definition of MP (Definition 2) requires estimating a joint distribution $P(X,Y)$ for all distance pairs $d_{x,y}$, which is usually expensive to compute. On the other hand, if independence could be assumed between distributions $P(X)$ and $P(Y)$, the computation of MP would simplify in accordance with the product rule:

$$MP_I(d_{x,y}) = P(X > d_{x,y}) \cdot P(Y > d_{y,x}). \tag{3}$$

We will show in our experiments that assuming independence in the computation of MP does not affect the results in an adverse way (Section 4).

In the base case where MP is computed from the empirical distribution as well as all other variants presented so far, the computational cost of computing MP grows quadratically with the size of the data set as all methods require the full distance matrix (that is, all possible distances) to be computed. To circumvent this, we propose to estimate the distribution parameters by randomly selecting a small fraction of objects to compute the mean and standard deviation of distances for each object using only the subset of objects. We denote MP where the parameters have been estimated by sampling from the collection with MP_S. The parameter S specifies how many objects have been randomly sampled. The appropriate sample size is naturally dependent on the underlying distribution. However if a normal distribution may be assumed, a sample size as small as $S = 30$ will already yield stable results for MP.

The difference to the original estimation of the parameters in Equation 2 is that only a small fraction of distances ($S \times n$) needs to be computed, which, for constant S, reduces the complexity from quadratic to linear in n. This is also more efficient than local scaling, where the actual nearest neighbors of points x and y need to be identified. While local scaling methods can of course be used with fast nearest neighbor search algorithms indexing the high dimensional spaces, the complexity is far higher than randomly sampling only a constant number of distances.

Experimental verification that these approximations of the original idea are still valid will be presented in Section 4.

3.2.3 LINEAR COMBINATION OF DISTANCE MEASURES

Another nice property of MP which can be useful in some contexts is that MP yields $[0,1]$-normalized similarities. Thus, the MP transformation can easily be used to linearly combine multiple different distance measures d_1 and d_2 for some combination weights $\omega_{1,2}$:

$$d = \omega_1 MP(d_1) + \omega_2 MP(d_2).$$

Similar to a global zero-mean unit-variance normalization, each object's distances are also standardized by their respective mean and standard deviation. Thus, no distance measure can dominate the other in this combination. This property is useful in scenarios where multiple different distance measures (describing different aspects of a phenomenon) need to be linearly combined. A real-world example where this is necessary is presented in Section 5.

4. Evaluation

To investigate the effects of using local neighborhood scaling methods and MP, we first evaluate the methods on 30 public machine learning data sets. Each data set is characterized by the following parameters: name/origin, number of classes, size/number of items n and data dimensionality d. For each data set we evaluate the original distance space and compare it to the distances that are generated by the local scaling method and by MP.

After showing the impact of the scaling methods in regard to the hub problem on real data sets in the first set of experiments, a second series of experiments investigates the effects of the methods more deeply. Synthetic as well as real data is used.

4.1 Benchmarks

To quantify the impact of the two methods, a number of properties and quality measures are computed for the original and the new distances. The characteristics which we compute for each data set are:

4.1.1 LEAVE-ONE-OUT k-NEAREST NEIGHBOR CLASSIFICATION ACCURACY (C^k)

We report the k-nearest neighbor (kNN) classification accuracy using leave-one-out cross-validation, where classification is performed via a majority vote among the k nearest neighbors, with the class of the nearest neighbor used for breaking ties. We denote the k-NN accuracy as C^k. In the context of a retrieval problem, higher values would indicate better retrieval quality.[4]

To test for statistical significance differences in classification accuracy between two algorithms, we use McNemar's test (see Salzberg, 1997 and Dietterich, 1998 for a discussion of using this test in conjunction with leave-one-out classification). When comparing two algorithms A and B, only classification instances where A and B disagree are being analyzed. More specifically, it is tested whether the number of times that A classifies correctly and B does not is significantly different from the number of times B classifies correctly and A does not.

4.1.2 K-OCCURRENCE ($N^k(x)$)

Defines the k-occurrences of object x, that is, the number of times x occurs in the k nearest neighbor lists of all other objects in the collection.

4.1.3 HUBNESS (S^k)

We also compute the *hubness* of the distance space of each collection according to Radovanović et al. (2010). Hubness is defined as the skewness of the distribution of k-occurrences N_k:

$$S^k = \frac{\mathrm{E}\left[(N_k - \mu_{N_k})^3\right]}{\sigma_{N_k}^3}.$$

Positive skewness indicates high hubness, negative values low hubness.

4. To clarify the cross-validation (CV) process: We first compute the distance matrix for the entire data set of n instances, transform this into an MP matrix, and then perform leave-one-out cross-validation on the data set of n instances, in each iteration i using one of the n instances (x_i) as a test case to be classified by its nearest neighbors among the remaining $n-1$ instances. It might seem that the 'test' example x_i plays an undue role in this process, having contributed to the the normalization of the distance matrix before being used as a 'new' test case. However, in a 'real' classification scenario, where we would have a fixed training set X^{n-1} (consisting of $n-1$ instances) and are presented with a new object x_i to classify (not contained in X^{n-1}), we would also first have to compute the full distance matrix over $X^{n-1} \cup \{x_i\}$ in order to then be able to compute MP over this matrix. (That is because MP needs information about all distances to and from x_i.)

   The result of this would be exactly the MP matrix we compute beforehand in our cross-validation process—and it is the exact same matrix for all other 'test' instances from X. Thus, it is legitimate to compute this once and for all before the CV. On the other hand, the above means that using MP in a 'real' classification scenario is expensive, because before being able to classify a new instance, first a complete distance and MP matrix have to be computed. What makes this process feasible in practice is the MP approximation MP_S described in Section 3.2.2.

4.1.4 GOODMAN-KRUSKAL INDEX (I_{GK})

The Goodman-Kruskal Index (Günter and Bunke, 2003) is a clustering quality measure that relates the number of *concordant* (Q_c) and *discordant* (Q_d) tuples ($d_{i,j}, d_{k,l}$) of a distance matrix.

- A tuple is concordant if its items i, j are from the same class, items k, l are from different classes and $d_{i,j} < d_{k,l}$.
- A tuple is discordant if its items i, j are from the same class, items k, l are from different classes and $d_{i,j} > d_{k,l}$.
- A tuple is not counted if it is neither concordant nor discordant, that is, if $d_{i,j} = d_{k,l}$.

The Goodman-Kruskal Index (I_{GK}) is defined as:

$$I_{GK} = \frac{Q_c - Q_d}{Q_c + Q_d}.$$

I_{GK} is bounded to the interval $[-1,1]$, and the higher I_{GK}, the more concordant and fewer discordant quadruples are present in the data set. Thus a large index value indicates a good clustering (in terms of *pairwise stability*—see Section 2).

Other indices to compare clustering algorithms like the classic Dunn's Index or Davies-Bouldin Index (Bezdek and Pal, 1998) cannot be used here as their values do not allow a comparison across different distance measures.

4.1.5 INTRINSIC DIMENSIONALITY (d_{mle})

To further characterize each data set we compute an estimate of the intrinsic dimensionality of the feature spaces. Whereas the embedding dimension is the actual number of dimensions of a feature space, the intrinsic dimension is the—often much smaller—number of dimensions necessary to represent a data set without loss of information. It has also been demonstrated that hubness depends on the intrinsic rather than embedding dimensionality (Radovanović et al., 2010). We use the maximum likelihood estimator proposed by Levina and Bickel (2005), which was also used by Radovanović et al. (2010) to characterize the data sets.

4.1.6 PERCENTAGE OF SYMMETRIC NEIGHBORHOOD RELATIONS

We call a nearest neighbor relation between two points x and y 'symmetric' if the following holds: object x is a nearest neighbor of y if and only if y is also the nearest neighbor of x. As both examined methods aim at symmetrizing neighborhood relations, we report the percentage of symmetric relations at different neighborhood sizes k.

4.2 Public Machine Learning Data Sets

We evaluate the proposed method by applying it to 30 different public machine learning data sets. The data sets include problems from the general machine learning field, and the bio-medical, image, text and music retrieval domains. We use the following data sets:

- The UCI Machine Learning Repository (*UCI*, see Frank and Asuncion, 2010) data sets: *arcene*, *gisette*, *mfeat-pixels/karhunen/factors*, *dexter*, *mini-newsgroups*, *dorothea*, *reuters-transcribed*.[5]
- The Kent Ridge bio-medical data sets (*KR*): *amlall*, *lungcancer* and *ovarian-61902*.[6]
- The LibSVM data sets (*LibSVM*, see Chang and Lin, 2001): *australian*, *diabetes*, *german numbers*, *liver-disorders*, *breast-cancer*, *duke (train)*, *heart*, *sonar*, *colon-cancer*, *fourclass*, *ionosphere*, *splice*.[7]
- The Music Information Retrieval Evaluation eXchange (MIREX) data sets (*Mirex*, see Downie, 2008): *ballroom* and *ismir2004*.[8]
- Two music artist web pages and tweets data sets (*CP*, see Schedl, 2010): *c1ka-twitter* and *c224a-web*.[9]

For the general machine learning data sets from the statistical or biological domains no feature extraction is necessary. The feature vectors can be downloaded directly from the respective repositories. These general machine learning data sets use the standard Euclidean distance (denoted as ℓ^2) as similarity measure.

The text retrieval data sets (*reuters-transcribed*, *c224a-web*, *movie-reviews*, *dexter*, *mini-newsgroups*, *c1ka-twitter*) need to be preprocessed before evaluating them.[10] To this end we employ stop-word removal and stemming. They are transformed into the bag-of-words representation, and standard $tf \cdot idf$ (term frequency $\cdot$ inverse document frequency) weights are computed (see for example Baeza-Yates and Ribeiro-Neto, 1999). The word vectors are normalized to the average document length. Individual document vectors are compared with the cosine distance (denoted as *cos*).

For the image retrieval data set (*corel*) normalized color histograms are computed as features. They show reasonable classification performance despite their simplicity, as Chapelle et al. (1999) show. The three 64-dimensional color histograms are concatenated into a single vector and compared using the Euclidean distance (ℓ^2).

To extract the features for the two music information retrieval data sets (*ismir2004*, *ballroom*) we use a standard algorithm from Mandel and Ellis (2005) which computes Mel Frequency Cepstrum Coefficients (Logan, 2000) and models each object as a multivariate normal distribution over these. Objects (models) are compared using the symmetrized Kullback-Leibler divergence (denoted as *skl*).

4.3 Results

In the following experiments we compute all previously introduced benchmark numbers for the original data and the distance spaces after applying the scaling methods (NICDM, MP). We use MP as defined in Section 3.2 and model the distance distributions with the empirical distribution.

5. The UCI Repository can be found at `http://archive.ics.uci.edu/ml/`.
6. The Kent Ridge data sets can be found at `http://datam.i2r.a-star.edu.sg/datasets/krbd/`.
7. LibSVM can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.
8. The MIREX data sets can be found at `http://www.music-ir.org/mirex`.
9. These data sets can be found at `http://www.cp.jku.at/people/schedl/datasets.html`.
10. Set *c1ka-twitter* equals *c3ka-twitter* from *CP*, omitting artists classified as 'rock' to balance the data.

Tables 1 and 2 show the results of the evaluations conducted on the 30 previously introduced public data sets. The collections have very diverse properties. There are collections like *fourclass* or *liver-disorders* with very low dimensionality ($d = 2$ and $d = 6$), as well as data sets with very high embedding dimensionality, such as *dorothea* ($d = 100\,000$) or *c1ka-twitter* ($d = 49\,820$). Related to that, column d_{mle} lists the intrinsic dimensionality according to the maximum likelihood estimator. Using the intrinsic dimensionality estimate we can see that there are data sets where the data is originally represented using high-dimensional feature vectors, although the data's intrinsic dimensionality is quite low. For example the *ovarian_61902* data set has an embedding dimension of $d = 15\,154$ but its estimated intrinsic dimension is only $d_{mle} = 9$.

The evaluation results in Tables 1 and 2 are sorted by the hubness $S^{k=5}$ of the original distance space (printed in bold). In subsequent plots individual collections are referenced by their numbers as given in Tables 1 and 2. The columns $C^{k=1}/C^{k=5}$ show the k-nearest neighbor classification rates of the collections. The classification rates with the original distances, the local scaling (NICDM) and the global scaling (MP) are documented. For convenience the column +/- shows the difference in classification accuracy, in terms of absolute percentage points, between the original distances and NICDM/MP. All improvements compared to the original distances are printed in bold. Statistically significant differences are marked with an asterisk (McNemar's test, $df = 1$, $\alpha = .05$ error probability).

Looking at the tables, a first observation is that very high-dimensional data sets (in terms of their intrinsic as well as their embedding dimensionality) also tend to have high hubness. This is in agreement with the results of Radovanović et al. (2010) and the theory that hubness is a consequence of high dimensionality.

By looking at the classification rates (columns $C^{k=1}$ and $C^{k=5}$) it can also clearly be observed that the higher the hubness and intrinsic dimensionality, the more beneficial, in terms of classification accuracy, NICDM and MP. For data sets with high hubness (in the collections we used, a value above 1.4 seems to be a limit) the increase in classification rates is notable. For $C^{k=1}$, the accuracy gain ranges from rather moderate 1 to up to $7-8$ percentage points, and in the case of *c1ka-twitter* it is 15.9 percentage points for NICDM and 17.1 percentage points for MP. For $C^{k=5}$ the trend is even clearer. Whereas only three changes in accuracy (relative to the original distances) are significant for data sets with low hubness ($S^{k=5} \leq 1.4$, data sets 1–17), 34 changes in accuracy are significant for data sets with high hubness ($S^{k=5} > 1.4$, data sets 18–30). There is no statistically significant negative change in terms of classification accuracies.

Figures 5 and 6 (left hand sides) present these results in bar plots where the y-axis shows the index of the data sets (ordered according to hubness as in Tables 1 and 2) and the bars show the increase or decrease of classification rates. The bar plots also directly show how MP compares to NICDM in terms of classification accuracy for $k = 1, 5$. Generally speaking, results for MP and NICDM are very comparable. As for $k = 1$, MP and NICDM perform equally well and there is no statistically significant difference between MP and NICDM (McNemar's test, $df = 1$, $\alpha = .05$ error probability). Based on the same statistical testing procedure, results for NICDM and $k = 5$ are significantly better than for MP for data sets 18, 20, 22 (marked with asterisks in Figure 6). The general tendency of both MP and NICDM is comparable in the sense that if there is an improvement compared to the original distances, it can be seen for both MP and NICDM.

Another observation from the results listed in Tables 1 and 2 is that both NICDM and MP reduce the hubness of the distance space for all data sets to relatively low values. The hubness $S^{k=5}$ decreases from an average value of 2.5 (original) to 0.29 (MP) and 0.94 (NICDM), indicating a

Name/Src.	Cls.	n	d	d_{mle}	Dist.	$C^{k=1}$	+/- %-pts	$C^{k=5}$	+/- %-pts	$S^{k=5}$	I_{GK}
fourclass (sc)	2	862	2	2	ℓ^2	100%		100%		**0.15**	0.22
1. *LibSVM*					NICDM	100%	0	100%	0	0.06	0.21
					MP	100%	0	100%	0	0.04	0.23
arcene	2	100	10 000	399	ℓ^2	82.0%		75.0%		**0.25**	0.07
2. *UCI*					NICDM	81.0%	-1.0	77.0%	**2.0**	-0.27	0.06
					MP	80.0%	-2.0	81.0%	**6.0**	0.15	0.10
liver-disorders (sc)	2	345	6	6	ℓ^2	62.6%		60.6%		**0.39**	0.00
3. *UCI*					NICDM	63.2%	**0.6**	65.8%	***5.2**	-0.04	0.03
					MP	62.9%	**0.3**	65.5%	***4.9**	-0.03	0.01
australian	2	690	14	3	ℓ^2	65.5%		68.8%		**0.44**	0.13
4. *LibSVM*					NICDM	65.7%	**0.2**	69.4%	**0.6**	-0.09	0.14
					MP	65.4%	-0.1	68.4%	-0.4	0.08	0.14
diabetes (sc)	2	768	8	6	ℓ^2	70.6%		74.1%		**0.49**	0.20
5. *UCI*					NICDM	69.8%	-0.8	74.1%	0	0.04	0.15
					MP	70.3%	-0.3	73.2%	-0.9	-0.02	0.19
heart (sc)	2	270	13	7	ℓ^2	75.6%		80.0%		**0.50**	0.35
6. *LibSVM*					NICDM	75.9%	**0.3**	79.3%	-0.7	-0.00	0.27
					MP	75.6%	0	80.4%	**0.4**	0.08	0.39
ovarian-61902	2	253	15 154	10	ℓ^2	95.3%		93.7%		**0.66**	0.20
7. *KR*					NICDM	95.7%	**0.4**	93.3%	-0.4	-0.10	0.19
					MP	94.1%	-1.2	94.1%	**0.4**	-0.28	0.19
breast-cancer (sc)	2	683	10	5	ℓ^2	95.6%		97.4%		**0.71**	0.89
8. *LibSVM*					NICDM	95.8%	**0.2**	97.1%	-0.3	0.19	0.42
					MP	96.0%	**0.4**	97.1%	-0.3	0.22	0.91
mfeat-factors	10	2 000	216	7	ℓ^2	95.0%		94.7%		**0.79**	0.71
9. *UCI*					NICDM	94.8%	-0.2	94.7%	0	0.15	0.76
					MP	94.5%	-0.5	94.9%	**0.2**	0.01	0.77
colon-cancer	2	62	2 000	11	ℓ^2	72.6%		77.4%		**0.81**	0.19
10. *LibSVM*					NICDM	69.4%	-3.2	82.3%	**4.9**	0.08	0.18
					MP	67.7%	-4.9	82.3%	**4.9**	-0.11	0.19
ger.num (sc)	2	1 000	24	8	ℓ^2	67.5%		71.7%		**0.81**	0.07
11. *LibSVM*					NICDM	66.8%	-0.7	72.0%	**0.3**	0.32	0.03
					MP	67.6%	**0.1**	71.4%	-0.3	0.11	0.07
amlall	2	72	7 129	32	ℓ^2	91.7%		93.1%		**0.83**	0.31
12. *KR*					NICDM	93.1%	**1.4**	97.2%	**4.1**	0.56	0.33
					MP	88.9%	-2.8	91.7%	-1.4	-0.01	0.34
mfeat-karhunen	10	2 000	64	15	ℓ^2	97.4%		97.4%		**0.84**	0.76
13. *UCI*					NICDM	97.2%	-0.2	97.6%	**0.2**	0.27	0.74
					MP	97.0%	-0.4	97.5%	**0.1**	0.08	0.79
lungcancer	2	181	12 533	60	ℓ^2	98.9%		100%		**1.07**	0.56
14. *KR*					NICDM	99.4%	**0.5**	98.9%	-1.1	0.31	0.50
					MP	98.3%	-0.6	97.8%	-2.2	0.01	0.56
c224a-web	14	224	1 244	41	cos	86.2%		89.3%		**1.09**	0.79
15. *CP*					NICDM	87.9%	**1.7**	92.4%	***3.1**	0.42	0.89
					MP	88.4%	**2.2**	92.4%	**3.1**	0.22	0.89

Table 1: Evaluation results ordered by ascending hubness ($S^{k=5}$) of the original distance space. This table reports data sets with small hubness. Each evaluated data set (*Name*/*Src*) is described by its number of classes (*Cls.*), its size (n), its extrinsic (d) and intrinsic (d_{mle}) data dimension and the distance measure used (*Dist*). Columns C^k report the classification accuracies at a given k, the respective adjacent column +/- the difference in classification accuracy between the original distances and NICDM/MP (in percentage points), column I_{GK} the Goodman-Kruskal Index. See Section 4.1 for an explanation of the individual benchmarks.

Name/Src.	Cls.	n	d	d_{mle}	Dist.	$C^{k=1}$	+/- %-pts	$C^{k=5}$	+/- %-pts	$S^{k=5}$	I_{GK}
mfeat-pixels	10	2 000	240	12	ℓ^2	97.6%		97.7%		**1.25**	0.75
16. *UCI*					NICDM	97.2%	-0.4	97.8%	**0.1**	0.28	0.75
					MP	97.2%	-0.4	97.5%	-0.2	0.13	0.79
duke (train)	2	38	7 129	16	ℓ^2	73.7%		68.4%		**1.37**	0.02
17. *UCI*					NICDM	81.6%	**7.9**	68.4%	0	0.43	0.06
					MP	76.3%	**2.6**	68.4%	0	0.21	0.03
corel1000	10	1 000	192	9	ℓ^2	70.7%		65.2%		**1.45**	0.33
18. *Corel*					NICDM	72.9%	***2.2**	72.0%	***6.8**	0.39	0.47
					MP	71.6%	**0.9**	70.3%	***5.1**	0.31	0.50
sonar (sc)	2	208	60	11	ℓ^2	87.5%		82.2%		**1.54**	0.07
19. *UCI*					NICDM	87.0%	-0.5	87.0%	**4.8**	0.47	0.08
					MP	87.5%	0	84.1%	**1.9**	0.32	0.08
ionosphere (sc)	2	351	34	13	ℓ^2	86.9%		85.5%		**1.55**	0.31
20. *UCI*					NICDM	92.3%	***5.4**	94.3%	***8.8**	0.28	0.07
					MP	91.7%	***4.8**	89.7%	***4.2**	0.50	0.27
reuters-transcribed	10	201	2 730	70	cos	44.3%		49.3%		**1.61**	0.38
21. *UCI*					NICDM	45.3%	**1.0**	52.7%	**3.4**	0.63	0.32
					MP	42.3%	-2.0	55.2%	***5.9**	0.18	0.43
ballroom	8	698	820	12	skl	54.3%		48.1%		**2.98**	0.15
22. *Mirex*					NICDM	57.2%	**2.9**	51.6%	***3.5**	1.09	0.20
					MP	56.6%	**2.3**	54.3%	***6.2**	0.30	0.18
ismir2004	6	729	820	25	skl	80.4%		74.1%		**3.20**	0.37
23. *Mirex*					NICDM	83.8%	***3.4**	79.0%	***4.9**	0.77	0.21
					MP	83.4%	***3.0**	77.0%	***2.9**	0.46	0.45
movie-reviews	2	2 000	10 382	28	cos	71.1%		75.7%		**4.07**	0.05
24. *PaBo*					NICDM	72.0%	**0.9**	76.0%	**0.3**	1.22	0.07
					MP	71.8%	**0.7**	76.7%	**1.0**	0.36	0.07
dexter	2	300	20 000	161	cos	80.3%		80.3%		**4.22**	0.10
25. *UCI*					NICDM	84.3%	**4.0**	86.0%	***5.7**	2.02	0.13
					MP	83.0%	**2.7**	90.0%	***9.7**	0.58	0.13
gisette	2	6 000	5 000	149	ℓ^2	96.0%		96.3%		**4.48**	0.16
26. *UCI*					NICDM	97.2%	***1.2**	98.1%	***1.8**	0.78	0.20
					MP	97.4%	***1.4**	97.9%	***1.6**	0.34	0.20
splice (sc)	2	1 000	60	27	ℓ^2	69.6%		69.4%		**4.55**	0.07
27. *LibSVM*					NICDM	73.3%	***3.7**	79.3%	***9.9**	1.51	0.11
					MP	72.4%	**2.8**	77.2%	***7.8**	0.48	0.10
mini-newsgroups	20	2 000	8 811	188	cos	64.4%		65.6%		**5.14**	0.47
28. *UCI*					NICDM	67.2%	***2.8**	68.5%	***2.9**	1.32	0.52
					MP	67.7%	***3.3**	68.4%	***2.8**	0.60	0.57
dorothea	2	800	100 000	201	ℓ^2	90.6%		90.2%		**12.91**	0.21
29. *UCI*					NICDM	92.2%	**1.6**	93.0%	***2.8**	11.72	0.21
					MP	91.5%	**0.9**	93.1%	***2.9**	1.66	0.20
c1ka-twitter	17	969	49 820	46	cos	31.9%		26.6%		**14.63**	0.08
30. *CP*					NICDM	47.8%	***15.9**	53.0%	***26.4**	2.94	0.33
					MP	49.0%	***17.1**	50.8%	***24.2**	1.79	0.16

Table 2: Evaluation results ordered by ascending hubness ($S^{k=5}$) of the original distance space. This table reports data sets with large hubness. Each evaluated data set (*Name*/*Src*) is described by its number of classes (*Cls.*), its size (n), its extrinsic (d) and intrinsic (d_{mle}) data dimension and the distance measure used (*Dist*). Columns C^k report the classification accuracies at a given k, the respective adjacent column +/- the difference in classification accuracy between the original distances and NICDM/MP (in percentage points), column I_{GK} the Goodman-Kruskal Index. See Section 4.1 for an explanation of the individual benchmarks.

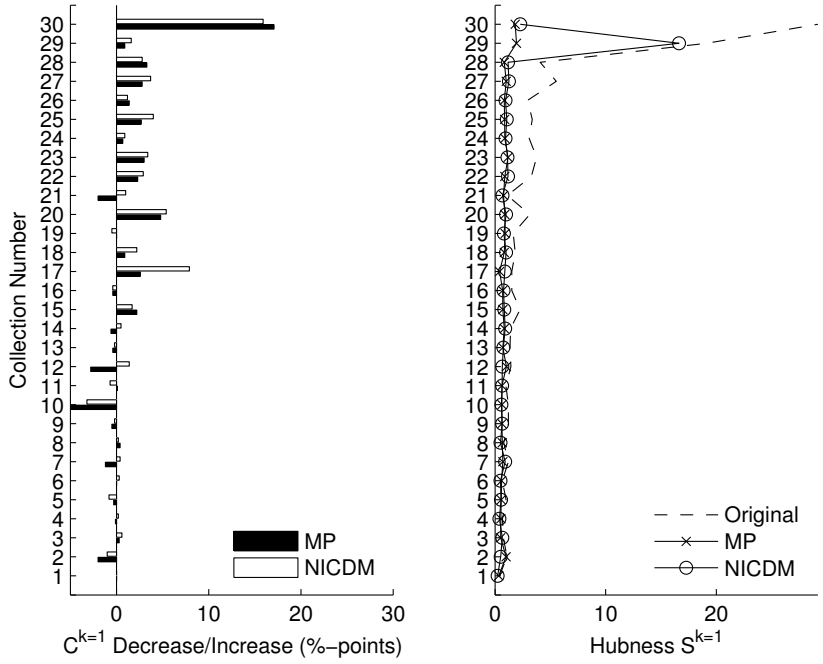


Figure 5: Improvements in accuracy (absolute percentage points) and hubness evaluated with $k = 1$.

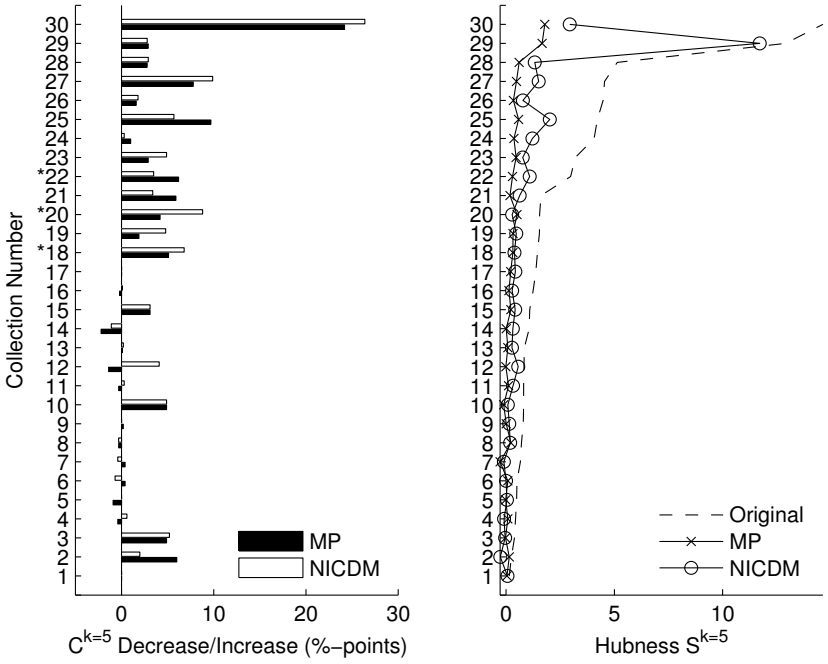


Figure 6: Improvements in accuracy (absolute percentage points, significant differences marked with an asterisk) and hubness evaluated with $k = 5$.

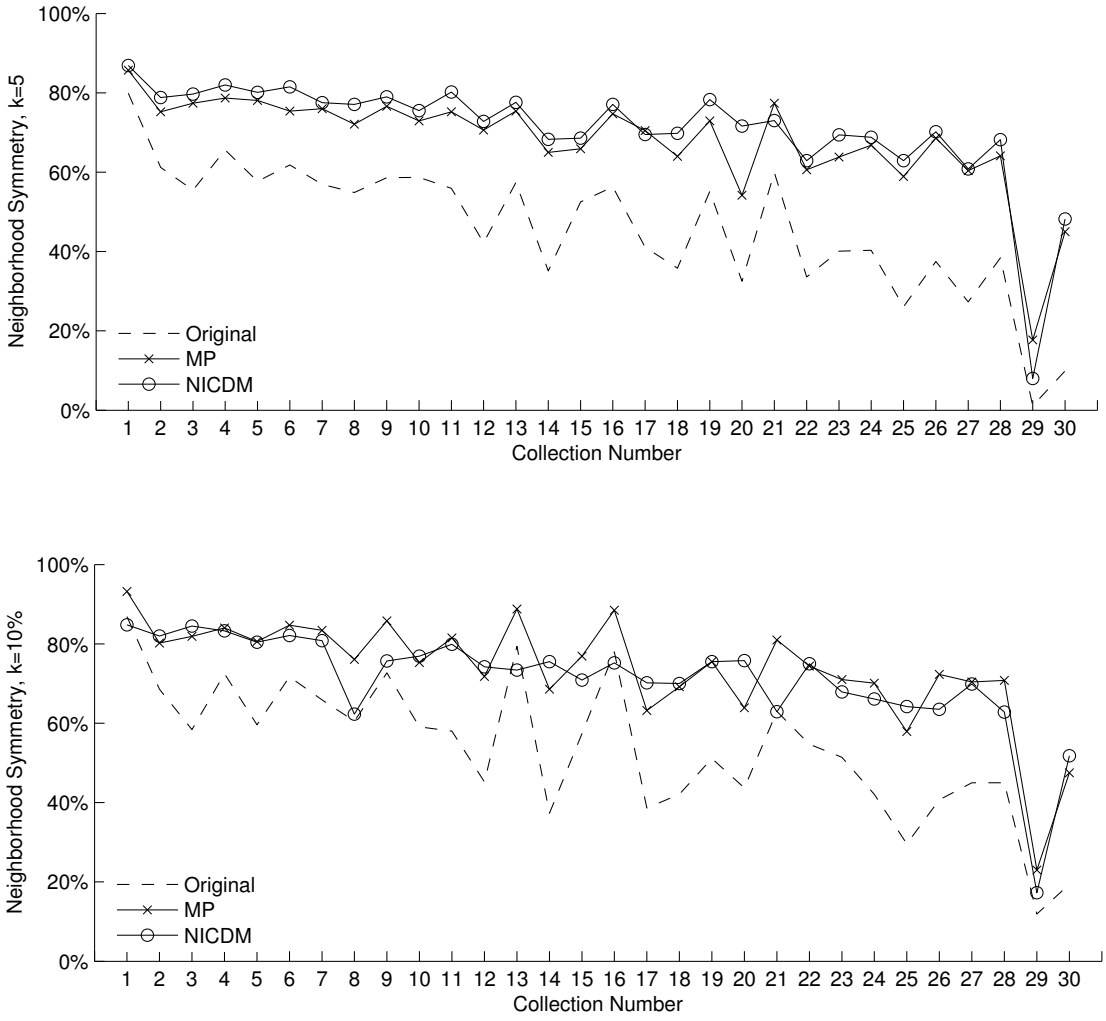


Figure 7: Percentage of symmetric neighborhood relations at $k = 5$ (above) and $k = 10\%$ (below) of the respective collection size.

well balanced distribution of nearest neighbors. The impact of MP and NICDM on the hubness per data set is plotted in Figures 5 and 6 (right hand sides). It can be seen that both MP and NICDM lead to lower hubness (measured for $S^{k=1,5}$) compared to the original distances. The effect is more pronounced for data sets having large hubness values according to the original distances.[11]

11. A notable exception is data set 29 ('dorothea') where the reduction in hubness is not so pronounced. This may be due to the extremely unbalanced distribution of its two classes (9:1).

More positive effects in the distances can also be seen in the increase of concordant (see Section 4 for the definition) distance quadruples indicated by higher Goodman-Kruskal index values (I_{GK}). This index improves or remains unchanged for 27 out of 30 data sets in the case of using MP. The effect is not so clear for NICDM, which improves the index or leaves it unchanged for only 17 out of 30 data sets. The effect of NICDM on I_{GK} is especially unclear for data with low hubness (data sets 1–17).

Finally we also checked whether both MP and NICDM are able to raise the percentage of symmetric neighborhood relations. Results for $k = 5$ and k set to 10% of the collection size (denoted by $k = 10\%$) are shown in Figure 7. As can be seen, the symmetry in the nearest neighbors for all data sets increases with both MP and NICDM. For NICDM there are two cases (data set 13 and 16) where the neighborhood symmetry does not increase. The average percentage of symmetric neighborhoods across all data sets for $k = 5$ is 46% for the original distances, 69% for MP, and 70.8% for NICDM. The numbers for $k = 10\%$ are 53% (original), 73.7% (MP), and 71.1% (NICDM).

4.4 Approximations

The general definition of MP (Definition 2, Section 3.2.1) allows for more specific uses if the underlying distribution of distances is known. All experiments conducted up until now use MP with the all-purpose empirical distribution. This section evaluates the use of different distributions in MP. Specifically, we will compare a Gaussian and a Gamma modeling to using the empirical distribution. For the two selected distributions, parameter estimation is straightforward (see Section 3.2.1). In case of the Gaussian, we will compute MP as it was defined. In our experiments this configuration will be denoted and referenced with 'MP (Gauss)'. As this variant involves computing a joint distribution in every step and this is expensive to calculate, there is no advantage to the original MP. Where things get interesting from a computational point of view, is using MP and assuming independence (MP_I, see Equation 3). In this case computing the joint distribution can be omitted. In our experiments we use the Gamma (denoted with, 'MP (i.Gamma)') and Gauss (denoted with 'MP (i.Gauss)') distribution with MP assuming independence.

Figure 8 plots the result of this experiment in the same way as we have done in the previous section. We compare the decrease/increase of classification accuracies and hubness at $k = 5$. Looking at the results, we can see that all methods seem to perform equally in terms of reducing hubness and increasing classification accuracies. More importantly, we notice that the simple variant ('MP (i.Gauss)'), which assumes a Gaussian distribution of distances and independence, performs similarly to all other variants.

This leads to the next experiment where we compare MP to a very simple approximation MP_S (see Section 3.2.2). As discussed in Section 3.2.2, assuming a Gaussian or Gamma distance distribution requires only a small sample size ($S = 30$) for a good estimate of the distribution parameters. Paired with the already evaluated simplification of MP assuming independence when computing the joint probability, MP is ready to be used instantly with any data collection. Figure 9 shows the results of a comparison of MP_S to MP. The classification accuracies are averages over ten approximations, that is, based on using ten times thirty randomly drawn data points for every data set. As can be seen, accuracy results are very comparable. We recorded three statistically significantly different results for MP_S using the approximative Gamma and Gauss variant (data sets 2, 10, 21, McNemar's test, $df = 1, \alpha = .05$ error probability). We also notice that with a sample size of $S = 30$ the decrease in hubness is not as pronounced for MP_S as for MP.

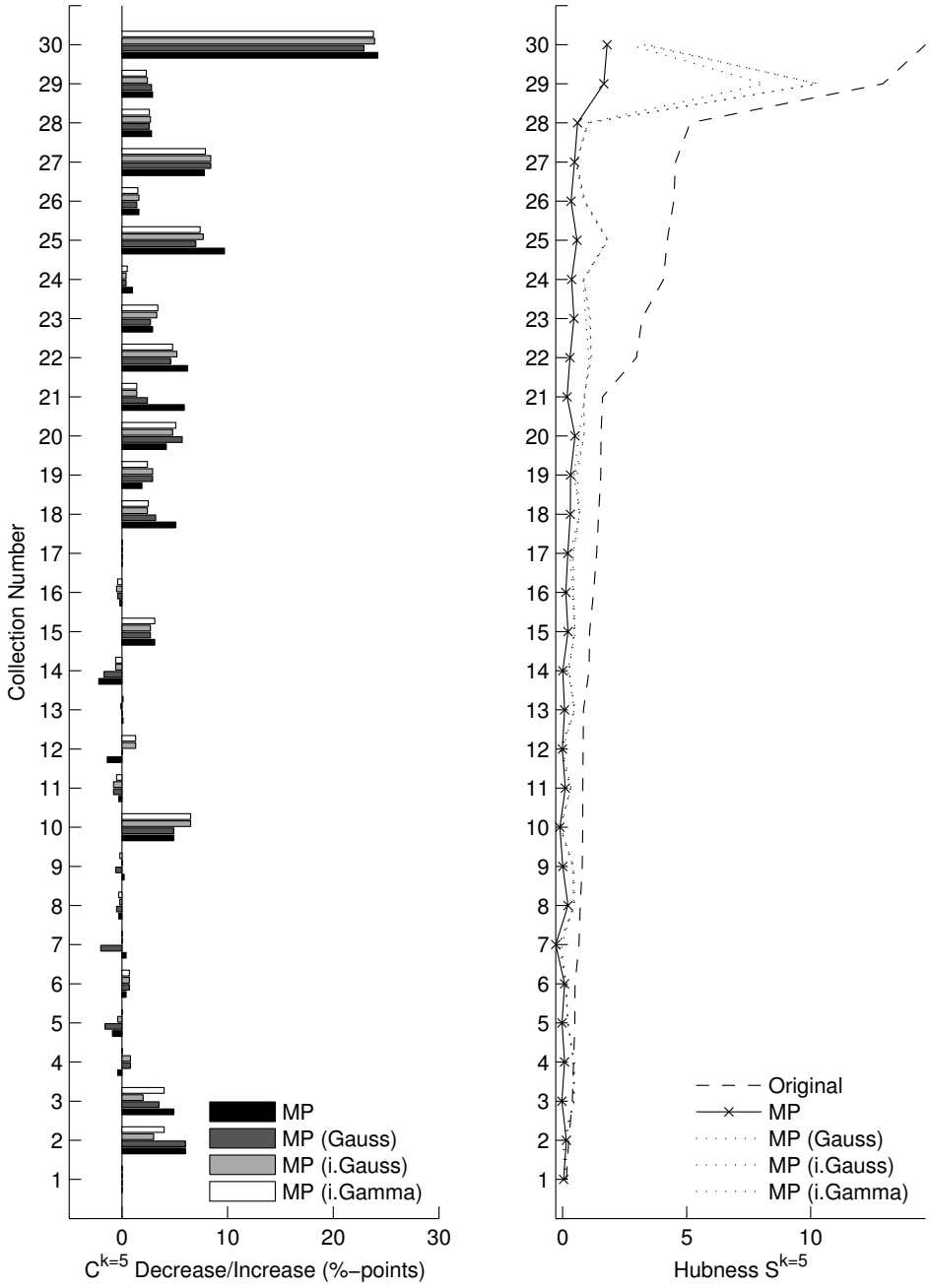


Figure 8: Comparison of different distance distributions in MP in terms of classification rates and hubness.

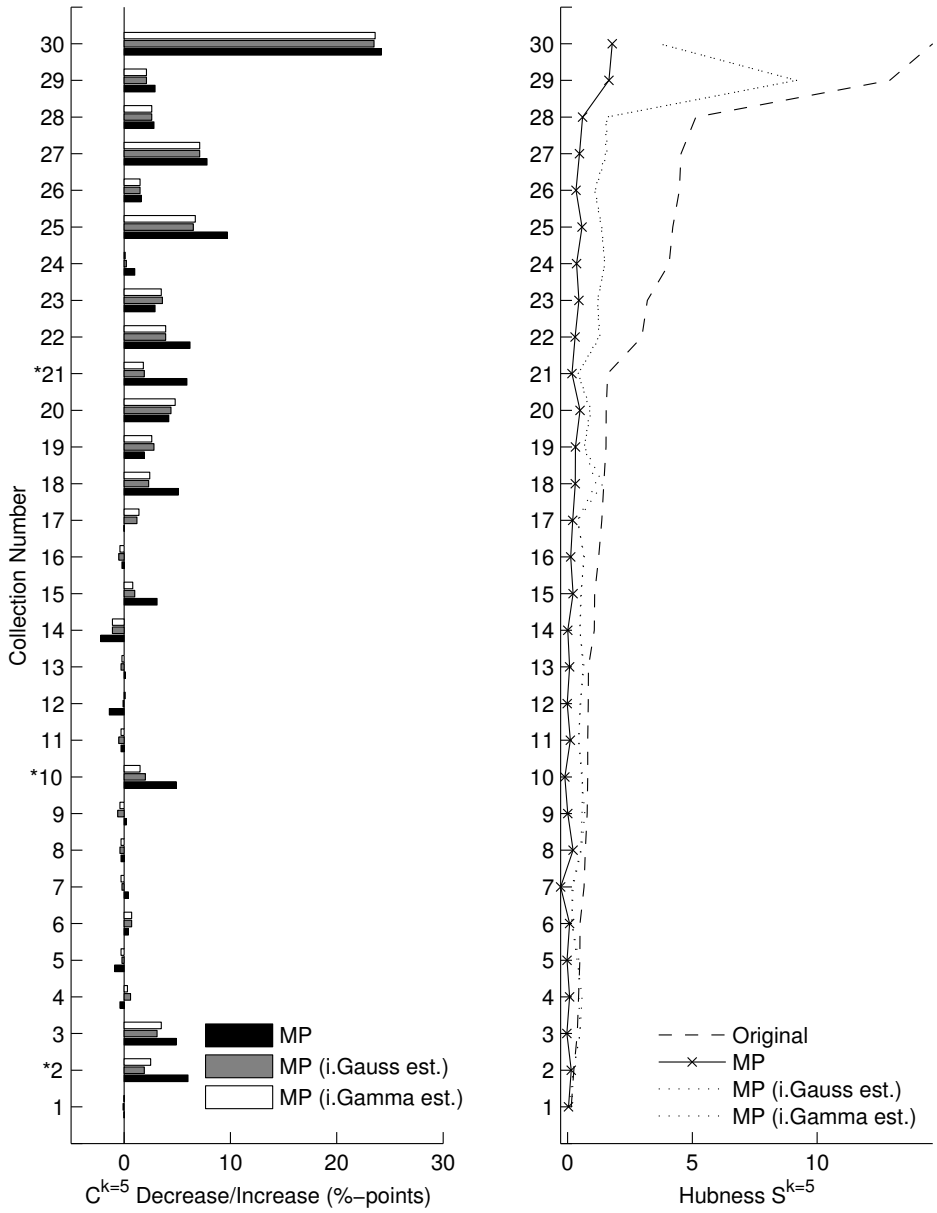


Figure 9: Improvements in accuracy (absolute percentage points, significant differences marked with an asterisk) and hubness evaluated with $k = 5$ for MP (black) and its approximative variant MP_S (gray).

4.5 Further Evaluations and Discussion

The previous experimental results suggest that the considered distance scaling methods work well as they tend to reduce hubness and improve the classification/retrieval accuracy. In the following three experiments we examine the scaling methods on artificial data as well as real data in order to investigate the following three questions:

1. Does NICDM/MP work by effectively reducing the intrinsic dimensionality of the data?
2. What is the impact of NICDM/MP on hubs and orphans?
3. Is the changing role of hubs responsible for improved classification accuracy?

The artificial data used in the experiments is generated by randomly sampling i.i.d. high-dimensional data vectors ($n = 1000$) in the hypercube $[0,1]^d$ from the standard uniform distribution. We use the Euclidean distance function and MP with the empirical distribution in all experiments.

4.5.1 DIMENSIONALITY

As we have already shown that hubs tend to occur in high dimensional spaces, the first experiment examines the consequential question if the scaling methods actually reduce the intrinsic dimensionality of the data. In order to test this hypothesis, the following simple experiment was performed: We increase the dimensions of artificial data (generated as described above) to create high hubness, and measure the intrinsic dimensionality of the data spaces before and after scaling the distances with NICDM/MP.

We start with a low data dimensionality ($d = 5$) and increase the dimensionality to a maximum of $d = 50$. In each iteration we measure the hubness of the data and its intrinsic dimensionality. The maximum likelihood estimator proposed by Levina and Bickel (2005) is used to estimate the intrinsic dimensionality of the generated vector spaces.

In Figure 10a we can see that a vector space dimension as low as 30 already leads to a distance space with very high hubness ($S^{k=5} > 2$). We can further see that NICDM/MP are able to reduce the hubness of the data spaces as expected. Figure 10b shows the measured intrinsic dimensionality of the original data. As anticipated it increases with its embedding dimensionality. However, to measure the intrinsic dimensionality of the data spaces created by MP and NICDM, we first have to map their distance space to a vector space. We perform this vector mapping using multidimensional scaling (MDS), doubling the target dimensionality to ensure a good mapping.

Figure 11 shows the results. For verification purposes, we (i) also map the original distance space with MDS and (ii) re-compute the hubness for the new data spaces (Figure 11a). Figure 11b finally compares the measured intrinsic dimensionality. We can clearly see that neither MP or NICDM decreases the intrinsic dimensionality notably. In none of the experiments does the estimated intrinsic dimensionality of the new distance space fall below the one measured in the original space.

4.5.2 IMPACT ON HUBS/ORPHANS

In the second experiment, we evaluate the question of what exactly happens to the hub and anti-hub (orphan) objects. Do hubs, after scaling the distances, still remain hubs (but 'less severely' so), or do they stop being hubs altogether? To look into this, we repeatedly generate a random, artificial, and

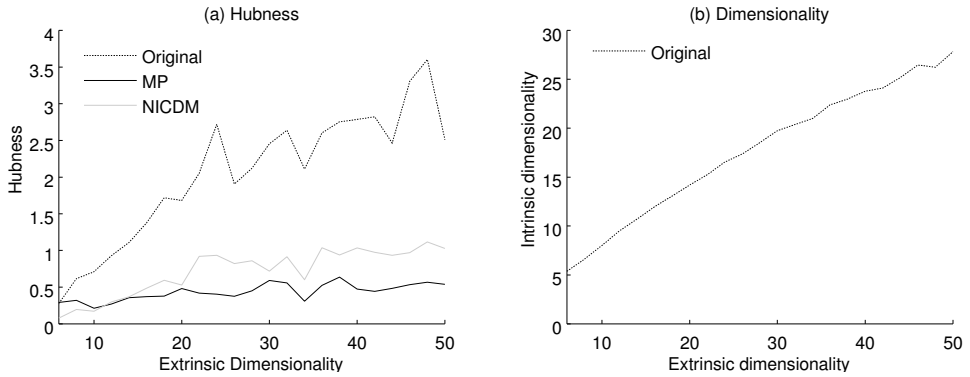


Figure 10: Increasing the dimensionality of artificially generated random data. Measuring (a) hubness of the original and scaled data, (b) the intrinsic dimensionality of the original data.

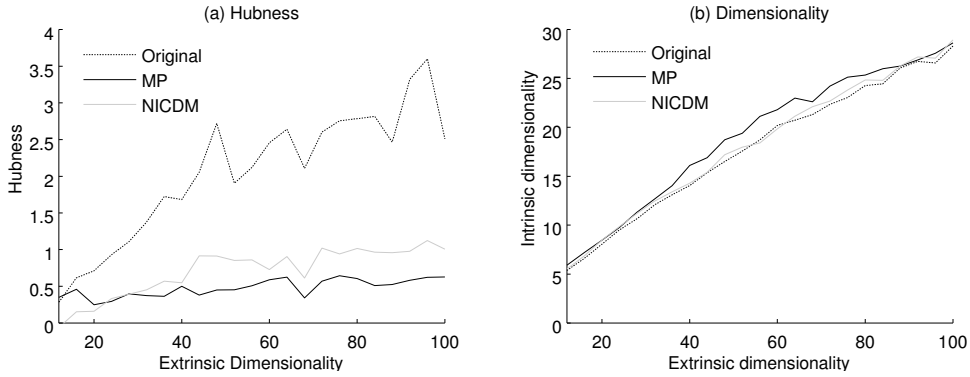


Figure 11: A vector space mapping of the distance spaces generated in Figure 10 allows to compare the intrinsic dimensionality of the original, MP, and NICDM data-spaces. No decrease of the intrinsic data dimensionality by using NICDM/MP can be observed.

high-dimensional ($d = 50$) data sample to (i) track hub and anti-hub objects and (ii) compute their k-occurrence (N^k) in the original space and in the distance spaces created after applying MP and NICDM. We define 'hub' objects as objects with a k-occurrence in the nearest neighbors greater than $5k$ and 'orphan' objects as having a k-occurrence of zero ($k = 5$). The experiment is repeated 100 times and for each iteration the observed mean k-occurrence of hubs/orphans is plotted in Figure 12.

Looking at the figure we can confirm that for the two studied cases (hubs/orphans) a weakening of the effects can be observed: after scaling the distances, hubs do not occur as often as nearest neighbors any more, while orphans re-appear in some nearest-neighbor lists. The k-occurrence of all other objects stays constant. Another observation is that in no instance of the experiment do hubs become orphans or orphans become hubs, as the measured $N^{k=5}$ never cross for the two classes.

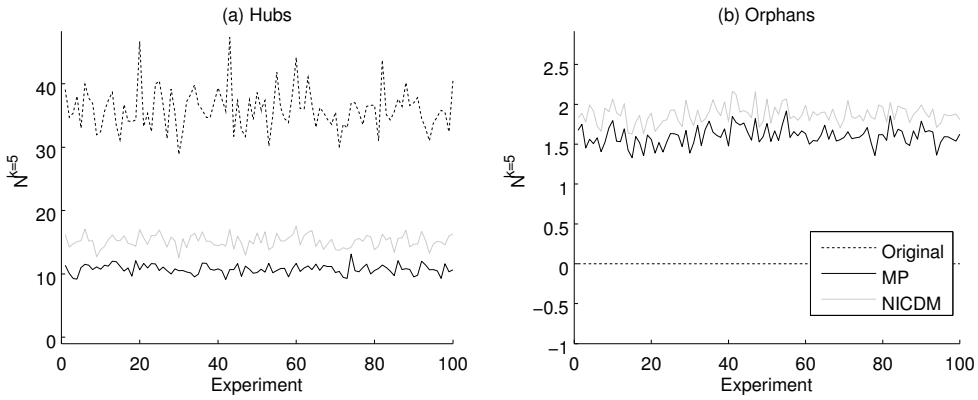


Figure 12: The k-occurrence ($N^{k=5}$) of hub and orphan data points before and after applying any of the scaling methods (NICDM, MP). Orphans re-appear in the nearest neighbor lists and the strength of hubs is reduced.

4.5.3 IMPACT OF HUBS/ORPHANS

In the final experiment we examine the increase in classification accuracies we observed previously when using NICDM or MP on the high dimensional machine learning data sets. To learn where the increase in classification accuracy came from, we distinguish between hubs, orphans, and all other objects. For each of the three classes we compute the so-called 'badness' ($BN^{k=5}$) as defined by Radovanović et al. (2010). Badness of an object x is the number of its occurrences as nearest neighbor at a given k where it appears with a different (that is, 'bad') class label. As this experiment makes only sense in collections with more than one class showing high hubness, we select machine learning data sets with high hubness of $S^{k=5} > 2$ from the 30 previously used databases. Table 3 documents the results of this experiment on the nine selected data sets.

For each collection the table shows the absolute number of hubs, orphans, and all other objects in the original data space. We then compute their badness before (columns *Orig.*) and after applying MP and NICDM. It can be clearly seen that indeed in each of the tested collections the badness of hubs decreases noticeably. In fact, on average $BN^{k=5}$ decreases more than 10 percentage points from 46.3% in the original space to 35.6% (NICDM) and 35.3% (MP). Another visible effect is that orphans re-appear in the nearest neighbor lists (see previous experiment, Figure 12) with an average badness of 36.5% (NICDM) and 35.1% (MP). The measured badness of orphan objects is comparable to the values reported for hubs, but is still notably higher than the numbers computed for the rest of the objects ('Other'). The badness of all other objects tends to stay the same: In three cases the badness increases slightly, in all other cases a slight decrease in badness can be observed. On average, badness decreases from 29.3% to 28.4% for both methods (MP and NICDM).

4.6 Summary of Results

Our main result is that both global (MP) and local (NICDM) scaling show very beneficial effects concerning hubness on data sets that exhibit high hubness in the original distance space. Both methods are able to decrease the hubness, raise classification accuracy, and improve other indicators

	Hubs, $BN^{k=5}$ *(%)*				*Orphans*, $BN^{k=5}$ *(%)*				*Other*, $BN^{k=5}$ *(%)*			
Data Set	#	Orig.	NICDM	MP	#	Orig.	NICDM	MP	#	Orig.	NICDM	MP
c1ka-twitter	13	83.5	54.0	55.7	540	/	59.3	59.2	416	46.2	47.9	50.1
dorothea	19	10.2	9.7	6.8	730	/	10.4	10.6	51	8.6	7.1	4.9
mini-newsgroups	38	67.2	62.2	60.7	304	/	45.6	43.5	1 658	42.2	41.5	41.6
splice (sc)	28	36.5	29.3	28.6	289	/	31.8	30.9	683	35.0	31.5	31.7
gisette	49	18.9	10.9	9.8	635	/	7.9	8.1	5316	4.7	4.0	3.9
dexter	11	44.3	27.9	28.4	80	/	33.5	30.5	209	18.2	18.1	17.7
movie-reviews	50	37.5	35.4	36.2	293	/	36.0	36.3	1 657	31.5	32.0	32.3
ismir2004	10	50.3	27.8	27.3	120	/	44.2	38.0	599	25.7	24.4	25.0
ballroom	12	67.9	62.8	63.8	148	/	59.5	58.6	538	51.6	49.0	48.3
Average (%-points):		46.3	35.6	35.3		/	36.5	35.1		29.3	28.4	28.4

Table 3: Relative badness ($BN^{k=5}$) of hub objects ($N^{k=5} > 5k$), orphan objects ($N^{k=5} = 0$), and all other objects. Data sets with $S^{k=5} > 2$.

like percentage of concordant distance quadruples or symmetric neighborhood relations. In case of MP, its approximation MP_S is able to perform at equal level with substantially less computational cost ($O(n)$, as opposed to $O(n^2)$ for both MP and local scaling). For data sets exhibiting low hubness in the original distance space, improvements are much smaller or non-existent, but there is no degradation of performance.

We have also shown that while MP and NICDM reduce hubness, which tends to occur as a consequence of high dimensional data, both methods do not decrease the intrinsic dimensionality of the distance spaces (at least for the type of data and measure of intrinsic dimensionality used in our experiments). By enforcing symmetry in the neighborhood of objects, both methods are able to naturally reduce the occurrence of hubs in nearest neighbor lists. Interestingly, at the same time as the occurrence of hubs in nearest neighbor lists decreases, hubs also lose their badness in terms of classification accuracy.

5. Mutual Proximity and Content-Based Music Similarity

This section presents an application where we can use Mutual Proximity, its approximation MP_S (Section 3.2.2) and a linear combination of multiple similarity measures (Section 3.2.3) to improve the retrieval quality of the similarity algorithm significantly. We chose to include this example as it demonstrates how MP_S with all its aspects introduced above can improve the quality of a real world application: the FM4 Soundpark.

The FM4 Soundpark is a web platform run by the Austrian public radio station FM4, a subsidiary of the Austrian Broadcasting Corporation (ORF).[12] The FM4 Soundpark was launched in 2001 and has gained significant public attention since then. Registered artists can upload and present their music free of charge. After a short editorial review period, new tracks are published on the front-page of the website. Older tracks remain accessible in the order of their publication date and in a large alphabetical list. Visitors of the website can listen to and download all the music at no cost. The FM4 Soundpark attracts a large and lively community interested in up and coming music,

12. FM4 Soundpark can be found at `http://fm4.orf.at/soundpark`.

Figure 13: The FM4 Soundpark music player web interface.

and the radio station FM4 also picks out selected artists and plays them on terrestrial radio. At the time of writing, there are more than 11 000 tracks by about 5 000 artists listed in the on-line catalog.

Whereas chronological publishing is suitable to promote new releases, older releases tend to disappear from the users' attention. In the case of the FM4 Soundpark this had the effect of users mostly listening to music that is advertised on the front-page, and therefore missing the full musical bandwidth. To allow access to the full database regardless of publication date of a song, we implemented a recommendation system using a content-based music similarity measure (see Gasser and Flexer, 2009 for a more detailed discussion of the system).

The user interface to the music recommender has been implemented as an Adobe Flash-based MP3 player with integrated visualization of the five songs most similar to the one currently playing. This web player can be launched from within an artist's web page on the Soundpark website by clicking on one of the artist's songs. Additionally to offering the usual player interface (start, stop, skipping forward/backward) it shows songs similar to the currently playing one in a text list and in a graph-based visualization (see Figure 13). The similar songs are retrieved by using an audio similarity function.

The graph visualization displays an incrementally constructed nearest neighbor graph (number of nearest neighbors = 5).

5.1 Similarity

The distance function used in the Soundpark to quantify music similarity was described by Pampalk (2006). To compute a similarity value between two music tracks (x, y), the method linearly combines rhythmic (d_r) and musical timbre (d_t) similarities into a single general music similarity (d) value. To combine the different similarities, they are normalized to zero-mean and unit-variance using static

normalization values ($\mu_r/\sigma_r, \mu_t/\sigma_t$) precomputed from a fixed training collection:

$$d(x,y) = 0.3\frac{d_r(x,y)-\mu_r}{\sigma_r} + 0.7\frac{d_t(x,y)-\mu_t}{\sigma_t}. \quad (4)$$

5.2 Limitations

The above algorithm for computing music similarity creates nearest neighbor lists which exhibit very high hubness. In the case of the FM4 Soundpark application, which always displays the top-5 nearest neighbors of a song, a similarity space with high hubness has an immediate negative impact on the quality of the results of the application. High hubness leads to circular recommendations and to the effect that some songs never occur in the nearest neighbor lists at all—hubs push out other objects from the $k = 5$ nearest neighbors. As a result of high hubness only 72.63% of the songs are reachable in the recommendation interface using the standard algorithm, that is, over a quarter of songs can never be reached in the application (more details are discussed in the next section).

In the following, we show that MP can improve this considerably. We use MP with two of the above mentioned aspects: (i) the linear combination of multiple similarity measures to combine timbre and rhythm similarities, and (ii) the approximation of the MP parameters, as computing all pairwise similarities would be highly impractical in a collection of this size.

5.3 Evaluation and Results

To evaluate the impact of MP_S on the application, we use MP_S in the linear combination of the rhythmic d_r and timbre d_t similarities:

$$d_{MP_S}(x,y) = 0.3\,MP_{S=30}(d_r(x,y) + 0.7\,MP_{S=30}(d_t(x,y)),$$

and compare the result to the standard variant (Equation 4) of the algorithm. Table 5 shows the results of the comparison (including a random baseline algorithm). As with the machine learning data sets evaluated previously, we observe that the hubness $S^{k=5}$ (which is particularly relevant for the application) decreases from 5.65 to 2.32. This is also visible in the k-occurrence (N^k) of the biggest hub object, N^k_{max}, which for $k = 5$ decreases from 242, with the standard algorithm, to 70.

We also compute the *retrieval accuracy* R^k (the average ratio of song genre labels matching the query object's genre) for $k = 1, 5, 10$. For a query song x and a list of recommendations $i = 1 \ldots k$, the retrieval accuracy over their multiple genres is computed as:

$$R^k(x) = \frac{1}{k}\sum_{i=1}^{k}\frac{|Genres(x) \cap Genres(i)|}{|Genres(x) \cup Genres(i)|}.$$

Similarly to the increase in classification accuracies for the machine learning data sets, R^k increases in all configurations. The music genre labels used in this evaluations originate from the artists who uploaded their songs to the Soundpark (see Table 4 for the music genres and their distribution in the collection).

The decrease of hubness produced by MP_S leads to a concurrent increase in the reachability of songs in the nearest neighbor graphs. Instead of only 72.6%, 86.2% of all songs are reachable via $k = 5$ nearest neighbor recommendation lists. If the application were to randomly sample 5 recommendations from the $k = 10$ nearest neighbors, the reachability with MP_S would even increase

Pop	*Rock*	*Electronica*	*Hip-Hop*	*Funk*	*Reggae*
37.6%	46.3%	44.0%	14.3%	19.7%	5.3%

Table 4: Music genre/class distribution of the songs in the FM4 Soundpark collection used for our experiments. Each artist can assign a newly uploaded song to one or more of these predefined genres. There are a total of 11 229 songs in our collection snapshot. As every song is allowed to belong to more than one genre, the percentages in the table add up to more than 100%.

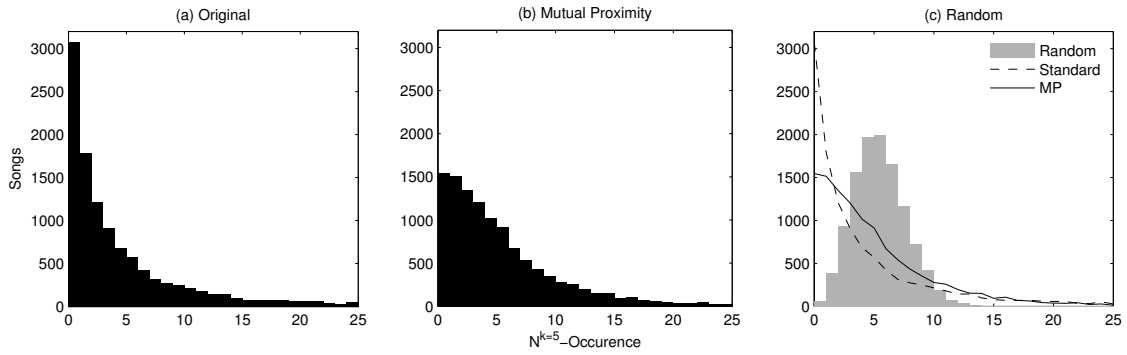


Figure 14: $N^{k=5}$-occurrences of songs in the nearest neighbors for (a) the standard algorithm, (b) the linear combination using $\mathrm{MP_S}$, and (c) random distances.

to 93.7% (from 81.9%) while the retrieval accuracy R^k for $k = 10$ would only slightly drop compared to $k = 5$.

Figure 14 shows a histogram plot of the $N^{k=5}$ occurrence of songs for the standard algorithm and $\mathrm{MP_S}$. The decrease of skewness is clearly visible as the number of songs that are never recommended drops from about 3000 to 1500—thus a more even distribution of objects in the nearest neighbors is achieved. The positive effects of using $\mathrm{MP_S}$ in this application are thus clearly visible: we obtain an improvement of retrieval accuracy and a decrease of hubness, paired with an increase of reachability in the nearest neighbors.

6. Conclusion

We have presented a possible remedy for the 'hubness' problems, which tend to occur when learning in high-dimensional data spaces. Considerations on the asymmetry of neighbor relations involving hub objects led us to evaluate a recent local scaling method, and to propose a new global variant named 'mutual proximity' (MP). In a comprehensive empirical study we showed that both scaling methods are able to reduce hubness and improve classification accuracy as well as other performance indices. Local and global methods perform at about the same level. Both methods are fully unsupervised and very easy to implement. Our own global scaling variant MP presented in this paper offers the additional advantage of being easy to approximate for large data sets which we show in an application to a real-world music recommendation service.

Characteristic	*Standard*	MP_S	(*Random*)
Retrieval Accuracy $R^{k=1}$	51.9%	54.5%	29.0%
Retrieval Accuracy $R^{k=5}$	48.2%	50.1%	28.5%
Retrieval Accuracy $R^{k=10}$	47.1%	48.6%	28.4%
Hubness $S^{k=5}$	5.65	2.31	0.46
Maximum Hub size $N_{max}^{k=5}$	242	70	17
Reachability $k = 5$	72.6%	86.2%	99.4%
Hubness $S^{k=10}$	5.01	2.14	0.36
Maximum Hub size $N_{max}^{k=10}$	416	130	25
Reachability $k = 10$	81.9%	93.7%	99.9%

Table 5: Evaluation results of the FM4 Soundpark data set comparing the standard method to MP_S. A random algorithm is added as baseline.

Our results indicate that both global and local scaling show very beneficial effects concerning hubness on a wide range of diverse data sets. They are especially effective for data sets of high dimensionality which are most affected by hubness. There is only little impact but no degradation of performance with data sets of low dimensionality. It is our hope that this empirical study will be the starting point of more theoretical work and consideration concerning the connection between hubness, asymmetric neighbor relations, and the benefits of similarity space transformations.

The main evaluation scripts used in this work are publicly available to permit reproduction of our results.[13]

Acknowledgments

This research is supported by the Austrian Research Fund (FWF) (grants P22856-N23, P24095, L511-N15, Z159) and the Vienna Science and Technology Fund WWTF (Audiominer, project number MA09-024). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Transport, Innovation and Technology. We would also like to thank our anonymous reviewers for their comments, which helped to improve this publication substantially.

References

Charu Aggarwal, Alexander Hinneburg, and Daniel Keim. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory - ICDT 2001*, Lecture Notes in Computer Science, pages 420–434. Springer Berlin/Heidelberg, 2001. doi: 10.1007/3-540-44503-X_27.

Jean-Julien Aucouturier and François Pachet. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

13. The scripts can be found at `http://www.ofai.at/~dominik.schnitzer/mp`.

Richard Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

Kristin P. Bennett, Usama Fayyad, and Dan Geiger. Density-based indexing for approximate nearest-neighbor queries. In *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)*, KDD '99, pages 233–243, New York, NY, USA, 1999. ACM. ISBN 1-58113-143-7. doi: 10.1145/312129.312236.

Adam Berenzweig. *Anchors and Hubs in Audio-based Music Similarity*. PhD thesis, Columbia University, 2007.

James C. Bezdek and Nihil R. Pal. Some new indexes of cluster validity. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 28(3):301 –315, 1998. ISSN 1083-4419. doi: 10.1109/3477.678624.

Rui Cai, Chao Zhang, Lei Zhang, and Wei-Ying Ma. Scalable music recommendation by search. In *Proceedings of the 15th International Conference on Multimedia*, pages 1065–1074, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-702-5. doi: 10.1145/1291233.1291466.

Michael Casey, Christophe Rhodes, and Malcolm Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech, and Language Processing*, 16 (5):1015–1028, 2008. doi: 10.1109/TASL.2008.925883.

Òscar Celma. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2008.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: A Library for Support Vector Machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Olivier Chapelle, Patrick Haffner, and Vladimir N. Vapnik. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5):1055 –1064, 1999. ISSN 1045-9227. doi: 10.1109/72.788646.

Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998. doi: 10.1162/089976698300017197.

George Doddington, Walter Liggett, Alvin Martin, Mark Przybocki, and Douglas Reynolds. Sheep, goats, lambs and wolves a statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation. In *Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP-98)*, 1998.

J. Stephen Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

Levent Ertöz, Michael Steinbach, and Vipin Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proceedings of the Third SIAM International Conference on Data Mining*, San Francisco, CA, 2003.

Andras Ferencz, Erik G. Learned-Miller, and Jitendra Malik. Building a classification cascade for visual identification from one example. In *Proceedings of the 10th IEEE International Conference on Computer Vision*, volume 1, pages 286–293. IEEE, 2005. doi: 10.1109/ICCV.2005.52.

Arthur Flexer, Dominik Schnitzer, Martin Gasser, and Tim Pohle. Combining features reduces hubness in audio similarity. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, 2010.

Damien François, Vincent Wertz, and Michel Verleysen. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering*, 19:873–886, 2007. ISSN 1041-4347. doi: 10.1109/TKDE.2007.1037.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. Repository located at: `http://archive.ics.uci.edu/ml`.

Martin Gasser and Arthur Flexer. FM4 SoundPark: Audio-based music recommendation in everyday use. In *Proceedings of the 6th Sound and Music Computing Conference*, pages 23–25, 2009.

Simon Günter and Horst Bunke. Validation indices for graph clustering. *Pattern Recognition Letters*, 24(8):1107 – 1113, 2003. ISSN 0167-8655. doi: DOI:10.1016/S0167-8655(02)00257-X. Graph-based Representations in Pattern Recognition.

Austin Hicklin, Craig Watson, and Brad Ulery. *The Myth of Goats: How Many People have Fingerprints that are Hard to Match?* US Dept. of Commerce, National Institute of Standards and Technology (NIST), 2005.

R.A. Jarvis and Edward A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, 22:1025–1034, 1973. ISSN 0018-9340. doi: 10.1109/T-C.1973.223640.

Herve Jegou, Hedi Harzallah, and Cordelia Schmid. A contextual dissimilarity measure for accurate and efficient image search. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007. doi: 10.1109/CVPR.2007.382970.

Herve Jegou, Cordelia Schmid, Hedi Harzallah, and Jakob Verbeek. Accurate image search using the contextual dissimilarity measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1), 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2008.285.

Wen Jin, Anthony Tung, Jiawei Han, and Wei Wang. Ranking outliers using symmetric neighborhood relationship. In *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 577–593. Springer Berlin/Heidelberg, 2006. doi: 10.1007/11731139_68.

Ioannis Karydis, Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. Looking through the "glass ceiling": A conceptual framework for the problems of spectral similarity. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, 2010.

Elizaveta Levina and Peter J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 17*, pages 777–784. MIT Press, Cambridge, MA, 2005.

Beth Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the first International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, MA, 2000.

Michael Mandel and Dan Ellis. Song-level features and support vector machines for music classification. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, UK, 2005.

Elias Pampalk. *Computational Models of Music Similarity and Their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, 2006.

Elias Pampalk, Simon Dixon, and Gerhard Widmer. On the Evaluation of Perceptual Similarity Measures for Music. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03)*, pages 7–12, 2003.

Elżbieta Pękalska and Robert P.W. Duin. Classifiers for dissimilarity-based pattern recognition. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 2, pages 12–16. IEEE, 2000. doi: 10.1109/ICPR.2000.906008.

Tim Pohle, Peter Knees, Markus Schedl, and Gerhard Widmer. Automatically adapting the structure of audio similarity spaces. In *Proceedings of the First Learning the Semantics of Audio Signals (LSAS) International Workshop*, Athens, Greece, 2006.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11:2487–2531, December 2010. ISSN 1532-4435.

Steven Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1:317–328, 1997. ISSN 1384-5810. doi: 10.1023/A:1009752403260.

Markus Schedl. On the Use of Microblogging Posts for Similarity Estimation and Artist Labeling. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, the Netherlands, August 2010.

Gregory Shakhnarovich, Trevor Darrell, and Piotr Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing)*. MIT press, 2006.

Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems*, volume 17, pages 1601–1608. MIT Press, Cambridge, MA, 2005.

Journal of Machine Learning Research 13 (2012) 2903-2922 Submitted 12/11; Revised 7/12; Published 10/12

Online Submodular Minimization

Elad Hazan* EHAZAN@IE.TECHNION.AC.IL
Technion - Israel Inst. of Tech.
Technion City
Haifa, 32000, Israel

Satyen Kale† SCKALE@US.IBM.COM
IBM T.J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

Editor: Francis Bach

Abstract

We consider an online decision problem over a discrete space in which the loss function is submodular. We give algorithms which are computationally efficient and are Hannan-consistent in both the full information and partial feedback settings.

Keywords: submodular optimization, online learning, regret minimization

1. Introduction

Online decision-making is a learning problem in which one needs to choose a decision repeatedly from a given set of decisions, in an effort to minimize costs over the long run, even in the face of complete uncertainty about future outcomes. The performance of an online learning algorithm is measured in terms of its *regret*, which is the difference between the total cost of the decisions it chooses, and the cost of the optimal decision chosen in hindsight. A *Hannan-consistent* algorithm is one that achieves sublinear regret (as a function of the number of decision-making rounds). Hannan-consistency implies that the average per round cost of the algorithm converges to that of the optimal decision in hindsight.

In the past few decades, a variety of Hannan-consistent algorithms have been devised for a wide range of decision spaces and cost functions, including well-known settings such as prediction from expert advice (Littlestone and Warmuth, 1989), online convex optimization (Zinkevich, 2003), and more (see the book by Cesa-Bianchi and Lugosi, 2006 for an extensive survey of prediction algorithms). Most of these algorithms are based on an online version of convex optimization algorithms. Despite this success, many online decision-making problems still remain open, especially when the decision space is discrete and large (say, exponential size in the problem parameters) and the cost functions are non-linear.

In this paper, we consider just such a scenario. Our decision space is now the set of all subsets of a ground set of n elements, and the cost functions are assumed to be *submodular*. This property is widely seen as the discrete analogue of convexity, and has proven to be a ubiquitous property in various machine learning tasks (see Guestrin and Krause, 2008 for references). A crucial compo-

∗. Supported by ISF Grant 810/11 and a Google Research Award.
†. Part of this work was done while the author was at Microsoft Research.

nent in these latter results are the celebrated polynomial time algorithms for submodular function minimization (Iwata et al., 2001).

To motivate the online decision-making problem with submodular cost functions, here is an example from the survey by McCormick (2006). Consider a factory capable of producing any subset from a given set of n products E. Let $f : 2^E \mapsto \mathbb{R}$ be the cost function for producing any such subset (here, 2^E stands for the set of all subsets of E). Economics tells us that this cost function should satisfy the law of diminishing returns: that is, the additional cost of producing an additional item is lower the more we produce. Mathematically stated, for all sets $S, T \subseteq E$ such that $T \subseteq S$, and for all elements $i \in E$, we have

$$f(T \cup \{i\}) - f(T) \;\geq\; f(S \cup \{i\}) - f(S).$$

Such cost functions are called *submodular*, and frequently arise in real-world economic and other scenarios. Now, for every item i, let p_i be the market price of the item, which is only determined in the future based on supply and demand. Thus, the profit from producing a subset S of the items is $P(S) = \sum_{i \in S} p_i - f(S)$. Maximizing profit is equivalent to minimizing the function $-P$, which is submodular as well.

The online decision problem which arises is now to decide which set of products to produce, to maximize profits in the long run, without knowing in advance the cost function or the market prices. A more difficult version of this problem, perhaps more realistic, is when the only information obtained is the actual profit of the chosen subset of items, and no information on the profit possible for other subsets.

In general, the Online Submodular Minimization problem is the following. In each iteration, we choose a subset of a ground set of n elements, and then observe a submodular cost function which gives the cost of the subset we chose. The goal is to minimize the regret, which is the difference between the total cost of the subsets we chose, and the cost of the best subset in hindsight. Depending on the feedback obtained, we distinguish between two settings, full-information and bandit. In the full-information setting, we can query each cost function at as many points as we like. In the bandit setting, we only get to observe the cost of the subset we chose, and no other information is revealed.

Obviously, if we ignore the special structure of these problems, standard algorithms for learning with expert advice and/or with bandit feedback can be applied to this setting. However, the computational complexity of these algorithms would be proportional to the number of subsets, which is 2^n. In addition, for the submodular bandits problem, even the regret bounds have an exponential dependence on n. It is hence of interest to design *efficient* algorithms for these problems. For the bandit version an even more basic question arises: does there exist an algorithm with regret which depends only polynomially on n?

In this paper, we answer these questions in the affirmative. We give efficient algorithms for both problems, with regret which is bounded by a polynomial in n, the underlying dimension, and sublinearly in the number of iterations. For the full information setting, we give two different randomized algorithms.

One of these algorithms is based on the follow-the-perturbed-leader approach (Hannan, 1957; Kalai and Vempala, 2005). We give a new way of analyzing such an algorithm. We hope this analysis technique will be applicable to other problems with large decision spaces as well. This algorithm is combinatorial, strongly polynomial, and can be generalized to arbitrary distributive lattices, rather than just all subsets of a given set.

The second algorithm is based on convex analysis. We make crucial use of a continuous extension of a submodular function known as the *Lovász extension*. We obtain our regret bounds by running a (sub)gradient descent algorithm in the style of Zinkevich (2003). The expected regret of this latter algorithm is shown to be bounded by $O(\sqrt{nT})$, and we show this to be optimal.

For the bandit setting, we give a randomized algorithm with expected regret at most $O(nT^{2/3})$. This algorithm also makes use of the Lovász extension and gradient descent. The algorithm folds exploration and exploitation steps into a single sample and obtains the stated regret bound. We also give high-probability bounds on regret of the same order for both settings of online submodular minimization.

An extended abstract of the results of this paper was originally presented in NIPS 2009 (Hazan and Kale, 2009). The present paper contains additional results with detailed proofs and tighter bounds, as well as several corrections.

1.1 Related Work

Submodular optimization has found numerous applications in machine learning and optimization in recent years, see, for example, the survey of Krause and Guestrin (2011). The prediction framework of online convex optimization was put forth by Zinkevich (2003), and found numerous applications since. Flaxman et al. (2005) show how to obtain sub linear regret bounds in the bandit setting. The latter technique is applicable to our setting when applied to the Lovász extension of a submodular function, although this gives weaker regret bounds than the ones presented hereby. Following our work, Jegelka and Bilmes (2011) study constrained submodular minimization over specific combinatorial structures.

2. Preliminaries and Problem Statement

In this section we review the basic concepts of submodular functions, prediction and online convex optimization, and state our main results.

2.1 Submodular Functions

The decision space is the set of all subsets of a universe of n elements, $[n] = \{1, 2, \ldots, n\}$. The set of all subsets of $[n]$ is denoted $2^{[n]}$. For a set $S \subseteq [n]$, denote by χ_S its characteristic vector in $\{0,1\}^n$, that is, $\chi_S(i) = 1$ if $i \in S$, and 0 otherwise.

A function $f : 2^{[n]} \to \mathbb{R}$ is called *submodular* if for all sets $S, T \subseteq [n]$ such that $T \subseteq S$, and for all elements $i \in [n]$, we have

$$f(T+i) - f(T) \;\geq\; f(S+i) - f(S).$$

Here, we use the shorthand notation $S+i$ to indicate $S \cup \{i\}$. An explicit description of f would take exponential space. We assume therefore that the only way to access f is via a *value oracle*, that is, an oracle that returns the value of f at any given set $S \subseteq [n]$.

Given access to a value oracle for a submodular function, it is possible to minimize it in polynomial time (Grötschel et al., 1988), and indeed, even in strongly polynomial time (Grötschel et al., 1988; Iwata et al., 2001; Schrijver, 2000; Iwata, 2003; Orlin, 2009; Iwata and Orlin, 2009). The current fastest strongly polynomial algorithm is due to Orlin (2009) and takes time $O(n^5 \mathrm{EO} + n^6)$, where EO is the time taken to run the value oracle. The fastest weakly polynomial algorithm are given by Iwata (2003) and Iwata and Orlin (2009) and run in time $\tilde{O}(n^4 \mathrm{EO} + n^5)$.

2.2 Online Submodular Minimization

In the Online Submodular Minimization problem, over a sequence of iterations $t = 1, 2, \ldots$, an online decision maker has to repeatedly chose a subset $S_t \subseteq [n]$. In each iteration, after choosing the set S_t, the cost of the decision is specified by a submodular function $f_t : 2^{[n]} \to [-M, M]$. The decision maker incurs cost $f_t(S_t)$. The *regret* of the decision maker is defined to be

$$\text{Regret}_T \ := \ \sum_{t=1}^{T} f_t(S_t) - \min_{S \subseteq [n]} \sum_{t=1}^{T} f_t(S).$$

If the sets S_t are chosen by a randomized algorithm, then we consider the expected regret over the randomness in the algorithm.

An online algorithm to choose the sets S_t will be said to be Hannan-consistent if it ensures that $\text{Regret}_T = o(T)$. The algorithm will be called *efficient* if it computes each decision S_t in $\text{poly}(n,t)$ time. Depending on the kind of feedback the decision maker receives, we distinguish between two settings of the problem:

- *Full information setting*. In this case, in each round t, the decision maker has unlimited access to the value oracles of the previously seen cost function $f_1, f_2, \ldots f_{t-1}$.
- *Bandit setting*. In this case, in each round t, the decision maker only observes the cost of her decision S_t, viz. $f_t(S_t)$, and receives no other information.

2.3 Statement of Main Results

In the setup of the Online Submodular Minimization, we have the following results:

Theorem 1 *In the full information setting of Online Submodular Minimization, there is an efficient randomized algorithm that attains the following regret bound:*

$$\mathbf{E}[Regret_T] \ = \ O(M\sqrt{nT}).$$

Furthermore, $Regret_T = O(M(\sqrt{n} + \sqrt{\log(1/\varepsilon)})\sqrt{T})$ *with probability at least* $1 - \varepsilon$.

We also prove a lower bound that shows that the algorithm of Theorem 1 has optimal regret up to constants:

Theorem 2 *In the full information setting of Online Submodular Minimization, for any algorithm, there is a sequence of submodular cost functions such that the algorithm has regret at least* $\Omega(M\sqrt{nT})$.

Theorem 3 *In the bandit setting of Online Submodular Minimization, there is an efficient randomized algorithm that attains the following regret bound:*

$$\mathbf{E}[Regret_T] \ = \ O(MnT^{2/3}).$$

Furthermore, $Regret_T = O(M(n + \sqrt{n\log(1/\varepsilon)})T^{2/3})$ *with probability at least* $1 - \varepsilon$.

Both Theorem 1 and Theorem 3 hold against both oblivious as well as adaptive adversaries, that is, the cost functions can be chosen adversarially with knowledge of the distribution over subsets chosen by the decision maker.

2.4 The Lovász Extension

A major technical construction we need for the algorithms is the *Lovász extension* $\hat{f}$ of the submodular function f. This is defined on the unit hypercube $\mathcal{K} = [0,1]^n$ and takes real values. Before defining the Lovász extension, we need the concept of a chain of subsets of $[n]$:

Definition 4 *A* **chain** *of subsets of $[n]$ is a collection of sets $A_0, A_1, \ldots, A_p$ such that*

$$A_0 \subset A_1 \subset A_2 \subset \cdots \subset A_p.$$

A **maximal chain** *is one where $p = n$. For a maximal chain, we have $A_0 = \emptyset$, $A_n = [n]$, and there is a unique associated permutation $\pi : [n] \to [n]$ such that for all $i \in [n]$, we have $A_{\pi(i)} = A_{\pi(i)-1} + i$. For this permutation π, we have $A_{\pi(i)} = \{j \in [n] : \pi(j) \leq \pi(i)\}$ for all $i \in [n]$.*

Now let $x \in \mathcal{K}$. There is a unique chain $A_0 \subset A_1 \subset \cdots A_p$ such that x can be expressed as a convex combination $x = \sum_{i=0}^{p} \mu_i \chi_{A_i}$ where $\mu_i > 0$ and $\sum_{i=0}^{p} \mu_i = 1$. A nice way to construct this combination is the following random process: choose a threshold $\tau \in [0,1]$ uniformly at random, and consider the level set $S_\tau = \{i : x_i > \tau\}$. The sets in the required chain are exactly the level sets which are obtained with positive probability, and for any such set A_i, $\mu_i = \mathbf{Pr}[S_\tau = A_i]$. In other words, we have $x = \mathbf{E}_\tau[\chi_{S_\tau}]$. This follows immediately by noting that for any i, we have $\mathbf{Pr}_\tau[i \in S_\tau] = x_i$. Of course, the chain and the weights μ_i can also be constructed deterministically simply by sorting the coordinates of x.

Now, we are ready to define[1] the Lovász extension $\hat{f}$:

Definition 5 *Let $x \in \mathcal{K}$. Let $A_0 \subset A_1 \subset \cdots A_p$ such that x can be expressed as a convex combination $x = \sum_{i=0}^{p} \mu_i \chi_{A_i}$ where $\mu_i > 0$ and $\sum_{i=0}^{p} \mu_i = 1$. Then the value of the Lovász extension $\hat{f}$ at x is defined to be*

$$\hat{f}(x) := \sum_{i=0}^{p} \mu_i f(A_i).$$

The preceding discussion gives an equivalent way of defining the Lovász extension: choose a threshold $\tau \in [0,1]$ uniformly at random, and consider the level set $S_\tau = \{i : x_i > \tau\}$. Then we have

$$\hat{f}(x) = \mathbf{E}_\tau[f(S_\tau)].$$

Note that the definition immediately implies that for all sets $S \subseteq [n]$, we have $\hat{f}(\chi_S) = f(S)$.

We will also need the notion of a maximal chain associated to a point $x \in \mathcal{K}$ in order to define subgradients of the Lovász extension:

Definition 6 *Let $x \in \mathcal{K}$, and let $A_0 \subset A_1 \subset \cdots A_p$ be the unique chain such that $x = \sum_{i=0}^{p} \mu_i \chi_{A_i}$ where $\mu_i > 0$ and $\sum_{i=0}^{p} \mu_i = 1$. A* **maximal chain associated with** *x is any maximal completion of the A_i chain, that is, a maximal chain $\emptyset = B_0 \subset B_1 \subset B_2 \subset \cdots B_n = [n]$ such that all sets A_i appear in the B_j chain.*

We have the following key properties of the Lovász extension. For proofs, refer to the book by Fujishige (2005, chapter IV).

Proposition 7 *For a submodular function f, the following properties of its Lovász extension $\hat{f} : \mathcal{K} \to \mathbb{R}$ hold:*

1. Note that this is not the standard definition of the Lovász extension, but an equivalent characterization.

1. *$\hat{f}$ is convex.*

2. *Let $x \in \mathcal{K}$. Let $\emptyset = B_0 \subset B_1 \subset B_2 \subset \cdots B_n = [n]$ be an arbitrary maximal chain associated with x, and let $\pi : [n] \to [n]$ be the corresponding permutation. Then, a subgradient g of $\hat{f}$ at x is given as follows:*

$$g_i \;=\; f(B_{\pi(i)}) - f(B_{\pi(i)-1}).$$

With the notation above, the following Lemma is from the paper by Jegelka and Bilmes (2011):

Lemma 8 (Lemma 1 from Jegelka and Bilmes, 2011) *The subgradients g of the Lovász extension $\hat{f} : \mathcal{K} \to [-M, M]$ of a submodular function are bounded by $\|g\|_2 \leq \|g\|_1 \leq 4M$.*

We provide a proof of this lemma in the appendix for completeness.

3. The Full Information Setting

In this section we give two algorithms for regret minimization in the full information setting. The first is a randomized combinatorial algorithm, based on the "follow the leader" approach of Hannan (1957) and Kalai and Vempala (2005) which attain the regret bound of $O(Mn\sqrt{T})$.

The second is an analytical algorithm based on (sub)gradient descent on the Lovász extension. It attains the regret bound of $O(M\sqrt{nT})$. We also prove a lower bound of $\Omega(M\sqrt{nT})$ on the regret of any algorithm for online submodular minimization, implying that the analytical algorithm is optimal up to constants.

Both algorithms have pros and cons: while the second algorithm is much simpler and more efficient, we do not know how to extend it to distributive lattices, for which the first algorithm readily applies.

3.1 A Combinatorial Algorithm

In this section we analyze a combinatorial, strongly polynomial, algorithm for minimizing regret in the full information Online Submodular Minimization setting:

Algorithm 1 Submodular Follow-The-Perturbed-Leader

1: Input: parameter $\eta > 0$.
2: Initialization: For every $i \in [n]$, choose a random number $r_i \in [-M/\eta, M/\eta]$ uniformly at random. Define $R : 2^{[n]} \to \mathbb{R}$ as $R(S) = \sum_{i \in S} r_i$.
3: **for** $t = 1$ to T **do**
4: Use the set $S_t = \arg\min_{S \subseteq [n]} \sum_{\tau=1}^{t-1} f_\tau(S) + R(S)$, and obtain cost $f_t(S_t)$.
5: **end for**

Define $\Phi_t : 2^{[n]} \to \mathbb{R}$ as $\Phi_t(S) = \sum_{\tau=1}^{t-1} f_\tau(S) + R(S)$. Note that R is a submodular function, and Φ_t, being the sum of submodular functions, is itself submodular. Furthermore, it is easy to construct a value oracle for Φ_t simply by using the value oracles for the f_τ. Thus, the optimization in step 3 is poly-time solvable given oracle access to Φ_t.

While the algorithm itself is a simple extension of Hannan (1957) follow-the-perturbed-leader algorithm, previous analysis (such as the one given by Kalai and Vempala, 2005), which rely on linearity of the cost functions, cannot be made to work here. Instead, we introduce a new analysis

technique: we divide the decision space using n different cuts so that any two decisions are separated by at least one cut, and then we give an upper bound on the probability that the chosen decision switches sides over each such cut. This new technique may have applications to other problems as well. We now prove the regret bound of Theorem 1:

Theorem 9 *Algorithm 1 run with parameter* $\eta = \frac{1}{\sqrt{T}}$ *achieves the following regret bound:*

$$\mathbf{E}[Regret_T] \leq 6Mn\sqrt{T}.$$

Proof We note that the algorithm is essentially running a "follow-the-leader" algorithm on the cost functions $f_0, f_1, \ldots, f_{t-1}$, where $f_0 = R$ is a fictitious "period 0" cost function used for regularization. The first step to analyzing this algorithm is to use a stability lemma, essentially proved by Kalai and Vempala (2005) and reproved in the appendix as Lemma 21 for completeness, which bounds the regret as follows:

$$\text{Regret}_T \leq \sum_{t=1}^{T} [f_t(S_t) - f_t(S_{t+1})] + R(S^*) - R(S_1).$$

Here, $S^* = \arg\min_{S \subseteq [n]} \sum_{t=1}^{T} f_t(S)$.

To bound the expected regret, by linearity of expectation, it suffices to bound $\mathbf{E}[f(S_t) - f(S_{t+1})]$, where for the purpose of analysis, we assume that we re-randomize in every round (that is, choose a fresh random function $R : 2^{[n]} \to \mathbb{R}$). Naturally, the expectation $\mathbf{E}[f(S_t) - f(S_{t+1})]$ is the same regardless of when R is chosen.

To bound this, we need the following lemma:

Lemma 10

$$\mathbf{Pr}[S_t \neq S_{t+1}] \leq 2n\eta.$$

Proof First, we note the following simple union bound:

$$\mathbf{Pr}[S_t \neq S_{t+1}] \leq \sum_{i \in [n]} \mathbf{Pr}[i \in S_t \text{ and } i \notin S_{t+1}] + \mathbf{Pr}[i \notin S_t \text{ and } i \in S_{t+1}]. \tag{1}$$

Now, fix any i, and we aim to bound $\mathbf{Pr}[i \in S_t \text{ and } i \notin S_{t+1}]$. For this, we condition on the randomness in choosing r_j for all $j \neq i$. Define $R' : 2^{[n]} \to \mathbb{R}$ as $R'(S) = \sum_{j \in S, j \neq i} r_j$, and $\Phi'_t : 2^{[n]} \to \mathbb{R}$ as $\Phi'_t(S) = \sum_{\tau=1}^{t-1} f_\tau(S) + R'(S)$. Note that if $i \notin S$, then $R'(S) = R(S)$ and $\Phi'_t(S) = \Phi_t(S)$. Let

$$A = \arg\min_{S \subseteq [n] : i \in S} \Phi'(S) \quad \text{and} \quad B = \arg\min_{S \subseteq [n] : i \notin S} \Phi'(S).$$

Now, we note that the event $i \in S_t$ happens only if $\Phi'_t(A) + r_i < \Phi'_t(B)$, and $S_t = A$. But if $\Phi'_t(A) + r_i < \Phi'_t(B) - 2M$, then we must have $i \in S_{t+1}$, since for any C such that $i \notin C$,

$$\Phi_{t+1}(A) = \Phi'_t(A) + r_i + f_t(A) < \Phi'_t(B) - M < \Phi'_t(C) + f_t(C) = \Phi_t(C).$$

The inequalities above use the fact that $f_t(S) \in [-M, M]$ for all $S \subseteq [n]$. Thus, if $v := \Phi'_t(B) - \Phi'_t(A)$, we have

$$\mathbf{Pr}[i \in S_t \text{ and } i \notin S_{t+1} \mid r_j, j \neq i] \leq \mathbf{Pr}[r_i \in [v - 2M, v] \mid r_j, j \neq i] \leq \eta,$$

since r_i is chosen uniformly from $[-M/\eta, M/\eta]$. We can now remove the conditioning on r_j for $j \neq i$, and conclude that

$$\mathbf{Pr}[i \in S_t \text{ and } i \notin S_{t+1}] \leq \eta.$$

Similarly, we can bound $\mathbf{Pr}[i \notin S_t \text{ and } i \in S_{t+1}] \leq \eta$. Finally, the union bound (1) over all choices of i yields the required bound on $\mathbf{Pr}[S_t \neq S_{t+1}]$. ■

Continuing the proof, we have (since $|f(S)| \leq M$)

$$\begin{aligned}\mathbf{E}[f(S_t) - f(S_{t+1})] &= \mathbf{E}[f(S_t) - f(S_{t+1}) \mid S_t \neq S_{t+1}] \cdot \mathbf{Pr}[S_t \neq S_{t+1}] \\ &\leq 2M \cdot \mathbf{Pr}[S_t \neq S_{t+1}] \\ &\leq 4Mn\eta.\end{aligned}$$

The last inequality follows from Lemma 10. Now, we have $R(S^*) - R(S_1) \leq 2Mn/\eta$, and so

$$\begin{aligned}\mathbf{E}[\text{Regret}_T] &\leq \sum_{t=1}^{T} \mathbf{E}[f(S_t) - f(S_{t+1})] + \mathbf{E}[R(S^*) - R(S_1)] \\ &\leq 4Mn\eta T + \frac{2Mn}{\eta} \\ &\leq 6Mn\sqrt{T},\end{aligned}$$

since $\eta = \frac{1}{\sqrt{T}}$. ■

3.2 An Analytical Algorithm

In this section, we give a different algorithm based on the Online Gradient Descent method of Zinkevich (2003). We apply this technique to the Lovász extension of the cost function coupled with a simple randomized construction of the subgradient, as given in definition 5. This algorithm requires the concept of a *Euclidean projection* of a point in $\mathbb{R}^n$ on to the set $\mathcal{K}$, which is a function $\Pi_{\mathcal{K}} : \mathbb{R}^n \to \mathcal{K}$ defined by

$$\Pi_{\mathcal{K}}(y) := \arg\min_{x \in \mathcal{K}} \|x - y\|.$$

Since $\mathcal{K} = [0,1]^n$, it is easy to implement this projection: indeed, for a point $y \in \mathbb{R}^n$, the projection $x = \Pi_{\mathcal{K}}(y)$ is defined by

$$x_i = \begin{cases} y_i & \text{if } y_i \in [0,1] \\ 0 & \text{if } y_i < 0 \\ 1 & \text{if } y_i > 1. \end{cases}$$

Algorithm 2 Submodular Subgradient Descent

1: Input: parameter $\eta > 0$. Let $x_1 \in \mathcal{K}$ be an arbitrary initial point.
2: **for** $t = 1$ to T **do**
3: Choose a threshold $\tau \in [0,1]$ uniformly at random, and use the set $S_t = \{i : x_t(i) > \tau\}$ and obtain cost $f_t(S_t)$.
4: Find a maximal chain associated with x_t, $\emptyset = B_0 \subset B_1 \subset B_2 \subset \cdots B_n = [n]$, and use $f_t(B_0), f_t(B_1), \ldots, f_t(B_n)$ to compute a subgradient g_t of $\hat{f}_t$ at x_t as in part 2 of Proposition 7.
5: Update: set $x_{t+1} = \Pi_{\mathcal{K}}(x_t - \eta g_t)$.
6: **end for**

In the analysis of the algorithm, we need the following regret bound. It is a simple extension of Zinkevich's analysis of Online Gradient Descent to vector-valued random variables whose expectation is the subgradient of the cost function (the generality to random variables is not required for this section, but it will be useful in the next section):

Lemma 11 *Let $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_T : \mathcal{K} \to \mathbb{R}$ be a sequence of convex cost functions over the cube $\mathcal{K}$. Let $x_1, x_2, \ldots, x_T \in \mathcal{K}$ be defined by $x_1 = 0$ and $x_{t+1} = \Pi_{\mathcal{K}}(x_t - \eta \hat{g}_t)$, where $\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_T$ are vector-valued random variables such that $\mathbf{E}[\hat{g}_t | x_t] = g_t$, where g_t is a subgradient of $\hat{f}_t$ at x_t. Then the expected regret of playing $x_1, x_2, \ldots, x_T$ is bounded by*

$$\sum_{t=1}^{T} \mathbf{E}[\hat{f}_t(x_t)] - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} \hat{f}_T(x) \;\leq\; \frac{n}{2\eta} + 2\eta \sum_t \mathbf{E}[\|\hat{g}_t\|^2].$$

Proof Let $y_{t+1} = x_t - \eta \hat{g}_t$, so that $x_{t+1} = \Pi_{\mathcal{K}}(y_{t+1})$. Note that

$$\|y_{t+1} - x^*\|^2 \;=\; \|x_t - x^*\|^2 - 2\eta \hat{g}_t^\top (x_t - x^*) + \eta^2 \|\hat{g}_t\|^2.$$

Rearranging,

$$\begin{aligned} \hat{g}_t^\top (x_t - x^*) \;&=\; \frac{1}{2\eta}[\|x_t - x^*\|^2 - \|y_{t+1} - x^*\|^2] + \frac{\eta}{2}\|\hat{g}_t\|^2 \\ &\leq\; \frac{1}{2\eta}[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2] + \frac{\eta}{2}\|\hat{g}_t\|^2, \end{aligned}$$

since $\|x_{t+1} - x^*\| \leq \|y_{t+1} - x^*\|$ by the properties of Euclidean projections onto convex sets. Hence, we have

$$\begin{aligned} \sum_{t=1}^{T} \hat{g}_t^\top (x_t - x^*) \;&\leq\; \sum_{t=1}^{T} \frac{\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2}{2\eta} + \frac{\eta}{2}\|\hat{g}_t\|^2 \\ &\leq\; \frac{n}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\hat{g}_t\|^2, \end{aligned}$$

since $\|x_1 - x^*\|^2 \leq n$, both x_1 and x^* being in the cube $\mathcal{K}$. Next, since $\mathbf{E}[\hat{g}_t | x_t] = g_t$, a subgradient of $\hat{f}_t$ at x_t, we have

$$\mathbf{E}[\hat{g}_t^\top (x_t - x^*) | x_t] \;=\; g_t^\top (x_t - x^*) \;\geq\; \hat{f}_t(x_t) - \hat{f}_t(x^*),$$

since $\hat{f}_t$ is a convex function. Taking expectation over the choice of x_t, we have

$$\mathbf{E}[\hat{g}_t^\top (x_t - x^*)] \;\geq\; \mathbf{E}[\hat{f}_t(x_t)] - \hat{f}_t(x^*).$$

Thus, we can bound the expected regret as follows:

$$\sum_{t=1}^{T} \mathbf{E}[\hat{f}_t(x_t)] - \hat{f}_t(x^*) \;\leq\; \mathbf{E}\left[\sum_{t=1}^{T} \hat{g}_t^\top (x_t - x^*)\right] \;\leq\; \frac{n}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \mathbf{E}[\|\hat{g}_t\|^2].$$

■

We can now prove the following regret bound:

Theorem 12 *Algorithm 2, run with parameter* $\eta = \sqrt{\frac{n}{16MT}}$, *achieves the following regret bound:*

$$\mathbf{E}[Regret_T] \leq 4M\sqrt{nT}.$$

Furthermore, with probability at least $1-\varepsilon$, *we have*

$$Regret_T \leq 4M\sqrt{nT} + M\sqrt{2T\log(1/\varepsilon)}.$$

Proof Note that be Definition 5, we have that $\mathbf{E}[f_t(S_t)] = \hat{f}_t(x_t)$. Since the algorithm runs Online Gradient Descent (from Lemma 11) with $\hat{g}_t = g_t$ (that is, no randomness), we get the following bound on the regret. Here, we use the bound of Lemma 8 $\|\hat{g}_t\|^2 = \|g_t\|^2 \leq 16M^2$.

$$\begin{aligned}
\mathbf{E}[\text{Regret}_T] &= \sum_{t=1}^{T} \mathbf{E}[f_t(S_t)] - \min_{S\subseteq[n]} \sum_{t=1}^{T} f(S) \\
&\leq \sum_{t=1}^{T} \hat{f}_t(x_t) - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} \hat{f}_T(x) \\
&\leq \frac{n}{2\eta} + \frac{16}{2}\eta M^2 T \\
&\leq 4M\sqrt{nT},
\end{aligned}$$

where the last inequality is due to the choice of η as in the theorem statement.

We proceed to give a high probability bound. The following Theorem is by Hoeffding (see the book by Cesa-Bianchi and Lugosi 2006, Appendix A):

Theorem 13 (Hoeffding) *Let* $X_1,\ldots,X_T$ *be independent random variables such that* $|X_t| \leq M$. *Then, for* $\varepsilon > 0$, *we have*

$$\Pr\left[\sum_{t=1}^{T} X_t - \mathbf{E}\left[\sum_{t=1}^{T} X_t\right] > M\sqrt{2T\log(1/\varepsilon)}\right] \leq \varepsilon.$$

Note that the sequence of points $x_1, x_2, \ldots, x_T$ is deterministic since it is obtained by deterministic gradient descent. The sets S_t are obtained by independent randomized rounding on the x_t's, and so the random variables $X_t = f_t(S_t)$ are independent. Note that $|X_t| \leq M$. Applying the Hoeffding bound above we get that with probability at least $1-\varepsilon$,

$$\sum_{t=1}^{T} f_t(S_t) \leq \sum_{t=1}^{T} \mathbf{E}[f_t(S_t)] + M\sqrt{2T\log(1/\varepsilon)},$$

which implies the high probability regret bound. ■

3.3 Lower Bound on Regret

We give a simple lower bound (which is reminiscent of the lower bounds for the setting of prediction from expert advice as in the book by Cesa-Bianchi and Lugosi, 2006), that in the full-information setting any algorithm for online submodular minimization can be made to have regret $\Omega(M\sqrt{nT})$. This show that the upper bound of Theorem 12 is optimal up to constants.

Theorem 14 *In the full-information setting, for any algorithm for online submodular minimization, there is a sequence of submodular cost functions $f_1, f_2, \ldots, f_T : 2^{[n]} \to [-M, M]$ such that the regret of the algorithm is at least $\Omega(M\sqrt{nT})$.*

Proof Consider the following randomized sequence of cost functions. In round t, choose the element $i(t) = (t \bmod n) + 1 \in [n]$, and a Rademacher random variable $\sigma_t \in \{-1, 1\}$ chosen independently of all other random variables. Then, define $f_t : 2^{[n]} \to [-M, M]$ as:

$$\forall S \subseteq [n] : \; f_t(S) \; = \; \begin{cases} -\sigma_t M & \text{if } i(t) \notin S \\ \sigma_t M & \text{if } i(t) \in S. \end{cases}$$

It is easy to check that f_t is submodular (in fact, it is modular). Note that for any set S_t played by the algorithm in round t, we have $\mathbf{E}[f_t(S_t)] = 0$, where the expectation is taken over the choice of σ_t. Thus, in expectation, the cost of the algorithm is 0. But now consider the set $S \subseteq [n]$ defined as follows. For all $i \in [n]$, let $X_i = \sum_{t:i(t)=i} \sigma_t$. Then let $S = \{i : \; X_i \leq 0\}$. Observe that by construction,

$$\sum_t f_t(S) \; = \; \sum_i -M|X_i|,$$

and hence

$$\mathbf{E}\left[\sum_t f_t(S)\right] = \mathbf{E}\left[\sum_i -M|X_i|\right] = n \cdot -M \cdot \Omega\left(\sqrt{\frac{T}{n}}\right) = -\Omega(M\sqrt{nT}).$$

Here, we used the fact that each X_i is a sum of at least $\lfloor \frac{T}{n} \rfloor$ independent Rademacher random variables, and Khintchine's inequality (see Cesa-Bianchi and Lugosi, 2006, Appendix A) implies that if Y is a sum of m independent Rademacher random variables, then $\mathbf{E}[|Y|] \geq \sqrt{m/2}$. Hence, the expected regret of the algorithm is $\Omega(M\sqrt{nT})$. In particular, there is a specific choice of the Rademacher random variables σ_t such that the algorithm incurs regret at least $\Omega(M\sqrt{nT})$. ■

4. The Bandit Setting

We now present an algorithm for the Bandit Online Submodular Minimization problem. The algorithm is based on the Online Gradient Descent algorithm of Zinkevich (2003). The main idea is to use just one sample for both exploration (to construct an unbiased estimator for the subgradient) and exploitation (to construct an unbiased estimator for the point chosen by the Online Gradient Descent algorithm).

Algorithm 3 Bandit Submodular Subgradient Descent

1: Input: parameters $\eta, \delta > 0$. Let $x_1 \in \mathcal{K}$ be arbitrary.
2: **for** $t = 1$ to T **do**
3: Find a maximal chain associated with x_t, $\emptyset = B_0 \subset B_1 \subset B_2 \subset \cdots B_n = [n]$, and let π be the associated permutation as in part 2 of Proposition 7. Then x_t can be written as $x_t = \sum_{i=0}^{n} \mu_i \chi_{B_i}$, where $\mu_i = 0$ for the extra sets B_i that were added to complete the maximal chain for x_t.
4: Choose the set S_t as follows:

$$S_t = B_i \quad \text{with probability} \quad \rho_i = (1-\delta)\mu_i + \frac{\delta}{n+1}.$$

Use the set S_t and obtain cost $f_t(S_t)$.
5: If $S_t = B_0$, then set $\hat{g}_t = -\frac{1}{\rho_0} f_t(S_t) e_{\pi(1)}$, and if $S_t = B_n$ then set $\hat{g}_t = \frac{1}{\rho_n} f_t(S_t) e_{\pi(n)}$. Otherwise, $S_t = B_i$ for some $1 \le i \le n-1$. Choose $\varepsilon_t \in \{+1, -1\}$ uniformly at random, and set:

$$\hat{g}_t = \begin{cases} \frac{2}{\rho_i} f_t(S_t) e_{\pi(i)} & \text{if } \varepsilon_t = 1 \\ \\ -\frac{2}{\rho_i} f_t(S_t) e_{\pi(i+1)} & \text{if } \varepsilon_t = -1. \end{cases}$$

6: Update: set $x_{t+1} = \Pi_{\mathcal{K}}(x_t - \eta \hat{g}_t)$.
7: **end for**

Before launching into the analysis, we define some convenient notation first. Define the filtration $\mathcal{F} = (\mathcal{F}_{t \le T})$, where $\mathcal{F}_t$ is the smallest σ-field with respect to which the random coin tosses of the algorithm in rounds $1, 2, \ldots, t$ are measurable, and let $\mathbf{E}_t[\cdot] = \mathbf{E}[\cdot | \mathcal{F}_{t-1}]$, and $\mathrm{VAR}_t[\cdot] = \mathrm{VAR}[\cdot | \mathcal{F}_{t-1}]$.

A first observation is that in expectation, the regret of the algorithm above is almost the same as if it had played x_t all along and the loss functions were replaced by the Lovász extensions of the actual loss functions.

Lemma 15 *For all t, we have $\mathbf{E}[f_t(S_t)] \le \mathbf{E}[\hat{f}_t(x_t)] + 2\delta M$.*

Proof From Definition 5 we have that $\hat{f}_t(x_t) = \sum_i \mu_i f_t(B_i)$. On the other hand, $\mathbf{E}_t[f_t(S_t)] = \sum_i \rho_i f_t(B_i)$, and hence:

$$\mathbf{E}_t[f_t(S_t)] - \hat{f}_t(x_t) \;=\; \sum_{i=0}^{n} (\rho_i - \mu_i) f_t(B_i) \;\le\; \delta \sum_{i=0}^{n} \left[\frac{1}{n+1} + \mu_i \right] |f_t(B_i)| \;\le\; 2\delta M. \tag{2}$$

The lemma now follows by taking expectations on both sides with respect to the randomness up to round $t-1$. ■

Next, by Proposition 7, the subgradient of the Lovász extension of f_t at point x_t corresponding to the maximal chain $B_0 \subset B_1 \subset \cdots \subset B_n$ is given by $g_t(i) = f(B_{\pi(i)}) - f(B_{\pi(i)-1})$. Using this fact, it is easy to check that the random vector $\hat{g}_t$ is constructed in such a way that $\mathbf{E}[\hat{g}_t | x_t] = \mathbf{E}_t[\hat{g}_t] = g_t$. Furthermore, we can bound the norm of this estimator as follows:

$$\mathbf{E}_t[\|\hat{g}_t\|^2] \;\le\; \sum_{i=0}^{n} \frac{4}{\rho_i^2} f_t(B_i)^2 \cdot \rho_i \;\le\; \frac{4M^2(n+1)^2}{\delta} \;\le\; \frac{16M^2 n^2}{\delta}. \tag{3}$$

We can now remove the conditioning, and conclude that $\mathbf{E}[\|\hat{g}_t\|^2] \leq \frac{16M^2n^2}{\delta}$.

Theorem 16 *Algorithm 3, run with parameters* $\delta = \frac{n}{T^{1/3}}$, $\eta = \frac{1}{4MT^{2/3}}$, *achieves the following regret bound:*

$$\mathbf{E}[Regret_T] \;\leq\; 6MnT^{2/3}.$$

Proof We bound the expected regret as follows: using Lemma 15), we have

$$\begin{aligned}
\sum_{t=1}^{T} \mathbf{E}[f_t(S_t)] - \min_{S\subseteq[n]} \sum_{t=1}^{T} f_t(S) \;&\leq\; 2\delta MT + \sum_{t=1}^{T} \mathbf{E}[\hat{f}_t(x_t)] - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} \hat{f}_t(x) \\
&\leq\; 2\delta MT + \frac{n}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \mathbf{E}[\|\hat{g}_t\|^2] \qquad \text{(By Lemma 11)} \\
&\leq\; 2\delta MT + \frac{n}{2\eta} + \frac{8n^2M^2\eta T}{\delta}. \qquad \text{(By (3))}
\end{aligned}$$

The bound is now obtained using the stated values for η, δ. ■

4.1 High Probability Bounds on the Regret

The theorem of the previous section gave a bound on the expected regret. However, a much stronger claim can be made that essentially the same regret bound holds with very high probability (exponential tail). The following gives high probability bounds against an adaptive adversary.

Theorem 17 *With probability at least* $1-4\varepsilon$, *Algorithm 3, run with parameters* $\delta = \frac{n}{T^{1/3}}$, $\eta = \frac{1}{4MT^{2/3}}$, *achieves the following regret bound:*

$$Regret_T \;\leq\; 38MnT^{2/3} + 44M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)}.$$

To prove the high probability regret bound, we require the following concentration lemma which can be found in the book by Cesa-Bianchi and Lugosi (2006, Appendix A):

Lemma 18 (Bernstein inequality for martingales) *Let* $X_1, \ldots, X_T$ *be a sequence of bounded random variables adapted to a filtration* $\mathcal{F} = (\mathcal{F}_t)_{t\leq T}$. *Let* $\mathbf{E}_t[\cdot] := \mathbf{E}[\cdot|\mathcal{F}_{t-1}]$. *Suppose that* $|X_t| \leq b$ *and let* $\mathbf{E}_t[X_t^2] \leq V$ *for all* $t \leq T$. *Then, for* $\varepsilon > 0$, *we have*

$$\Pr\left[\left|\sum_{t=1}^{T} X_t - \mathbf{E}_t[X_t]\right| > \sqrt{2TV\log(1/\varepsilon)} + b\log(1/\varepsilon)\right] \;\leq\; \varepsilon.$$

The following simple corollary will be useful in the analysis:

Corollary 19 *In the setup of Lemma 18, assume that the parameters* T, V, b *and* ε *satisfy* $\sqrt{TV} > b\sqrt{\log(1/\varepsilon)}$. *Then*

$$\Pr\left[\left|\sum_{t=1}^{T} X_t - \mathbf{E}_t[X_t]\right| > 4\sqrt{TV\log(1/\varepsilon)}\right] \;\leq\; \varepsilon.$$

Proof [Theorem 17] If $T \leq 2\log^{3/2}(1/\varepsilon)$, then the regret can be trivially bounded by

$$2MT \;\leq\; 4MT^{2/3}\sqrt{\log(1/\varepsilon)} \leq\; 38MnT^{2/3} + 44M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)}.$$

So from now on we assume that $T > 2\log^{3/2}(1/\varepsilon)$. We need the following lemma:

Lemma 20 *If $T > 2\log^{3/2}(1/\varepsilon)$, then with probability at least $1-4\varepsilon$, all of the following inequalities hold:*

$$\sum_{t=1}^{T} \|\hat{g}_t\|^2 \;\leq\; \sum_{t=1}^{T} \mathbf{E}_t[\|\hat{g}_t\|^2] + 64M^2T^{4/3}\sqrt{\log(1/\varepsilon)}, \tag{4}$$

$$\sum_{t=1}^{T} g_t^\top x_t \;\leq\; \sum_{t=1}^{T} \hat{g}_t^\top x_t + 16M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)}, \tag{5}$$

$$\forall S \subseteq [n], \quad \sum_{t=1}^{T} \hat{g}_t^\top \chi_S \;\leq\; \sum_{t=1}^{T} g_t^\top \chi_S + 16M\sqrt{n}T^{2/3}\sqrt{\log(2^n/\varepsilon)}, \tag{6}$$

$$\textit{and} \quad \sum_{t=1}^{T} f_t(S_t) \;\leq\; \sum_{t=1}^{T} \mathbf{E}_t[f_t(S_t)] + 4M\sqrt{T\log(1/\varepsilon)}. \tag{7}$$

Proof We use Lemma 18 to bound the probability of each of the four events *not* happening by ε, and then we apply a union bound. In the following, we use the lower bound

$$\rho_i \;\geq\; \frac{\delta}{n+1} \;\geq\; \frac{1}{2T^{1/3}}. \tag{8}$$

Recall the filtration $\mathcal{F} = (\mathcal{F}_{t\leq T})$, where $\mathcal{F}_t$ is the smallest σ-field with respect to which the random coin tosses of the algorithm in rounds $1,2,\ldots,t$ are measurable. In the following we will consider sequences of random variables $X_1, X_2, \ldots, X_T$ adapted to $\mathcal{F}$.

Proof of (4). Consider the random variables $X_t := \|\hat{g}_t\|^2$ for $t \leq T$, that are adapted to $\mathcal{F}$. To apply Corollary 19, we estimate the parameters b, V. If B_i was sampled in step t, we have, using (3),

$$|X_t| \;=\; \|\hat{g}_t\|^2 \;\leq\; \frac{4}{\rho_i^2} f_t^2(B_i) \;\leq\; 16M^2T^{2/3},$$

using (8). Thus, we can choose $b = 16M^2T^{2/3}$. Next, we have

$$\mathbf{E}[\|\hat{g}_t\|^4 | \mathcal{F}_{t-1}] \;\leq\; \sum_{i=0}^{n} \frac{16}{\rho_i^4} f_t(B_i)^4 \cdot \rho_i \;\leq\; (n+1)\cdot 16 \cdot 8T \cdot M^4 \;\leq\; 256M^4nT.$$

Thus, we can choose $V = 256M^4nT$. Now, $\sqrt{TV} = 16M^2\sqrt{n}T > b\sqrt{\log(1/\varepsilon)}$ for $T > \log^{3/2}(1/\varepsilon)$. The required bound follows from Corollary 19 using the overestimation $\sqrt{n} \leq T^{1/3}$.

Proof of (5). Consider the random variables $X_t := \hat{g}_t^\top x_t$ for $t \leq T$, that are adapted to $\mathcal{F}$. First note that $\mathbf{E}_t[\hat{g}_t^\top x_t] = g_t^\top x_t$. To apply Corollary 19, we estimate the parameters b, V. First, if B_i was sampled in step t, then we have, using Hölder's inequality,

$$|X_t| \;\leq\; \|\hat{g}_t\|_1 \|x_t\|_\infty \;\leq\; \frac{2}{\rho_i}|f_t(B_i)| \;\leq\; 4MT^{1/3},$$

using (8). Thus, we choose $b = 4MT^{1/3}$. Next, again using Hölder's inequality we have

$$\mathbf{E}_t[(\hat{g}_t^\top x_t)^2] \;\le\; \mathbf{E}_t[\|\hat{g}_t\|_1^2 \|x_t\|_\infty^2] \;\le\; \sum_{i=0}^{n} \frac{4}{\rho_i^2} f_t^2(B_i) \cdot \rho_i \;\le\; 16M^2 nT^{1/3},$$

using (8). Thus, we can choose $V = 16M^2 nT^{1/3}$. Note that $\sqrt{TV} = 4M\sqrt{n}T^{2/3} > b\sqrt{\log(1/\varepsilon)}$ for $T > \log^{3/2}(1/\varepsilon)$. The required bound follows from Corollary 19.

Proof of (6). This bound follows exactly as the previous one, except we use the random variables $X_t := \hat{g}_t^\top \chi_S$ for every fixed set $S \subseteq [n]$, with error parameter $\varepsilon/2^n$. With this value of the error parameter, the conditions of Corollary 19 are met for $T > 2\log^{3/2}(1/\varepsilon)$. We then take a union bound over the 2^n choices of S to obtain the required bound.

Proof of (7). Consider the random variables $X_t := f_t(S_t)$ for $t \le T$, that are adapted to $\mathcal{F}$. To apply Corollary 19, we estimate the parameters b, V. We have

$$|X_t| \;=\; |f_t(S_t)| \;\le\; M.$$

So we can use $b = M$. As for V, we use the trivial bound $V = b^2 = M^2$. Again, $\sqrt{TV} = \sqrt{T}b > b\sqrt{\log(1/\varepsilon)}$ for $T > \log(1/\varepsilon)$. The required bound follows from Corollary 19. ■

Finally, we can imagine the points $x_1, x_2, \ldots, x_T$ as being produced by running Online Gradient Descent with linear cost functions $\hat{g}_t^\top x$, thinking of $\hat{g}_t$ as deterministic vectors. Thus, by Lemma 11, we get that for any $S \subseteq [n]$, we have

$$\sum_{t=1}^{T} \hat{g}_t^\top (x_t - \chi_S) \;\le\; \frac{n}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\hat{g}_t\|^2. \tag{9}$$

Thus, with probability $1-4\varepsilon$, for any $S \subseteq [n]$, we have

$$\begin{aligned}
&\sum_{t=1}^{T} f_t(S_t) - f_t(S) \\
&\leq \sum_{t=1}^{T} \mathbf{E}_t[f_t(S_t)] - f_t(S) + 4M\sqrt{T\log(1/\varepsilon)} && \text{(By (7))} \\
&\leq \sum_{t=1}^{T} \hat{f}_t(x_t) - \hat{f}_t(\chi_S) + 2nMT^{2/3} + 4M\sqrt{T\log(1/\varepsilon)} && \text{(By (2))} \\
&\leq \sum_{t=1}^{T} g_t^\top (x_t - \chi_S) + 2MnT^{2/3} + 4M\sqrt{T\log(1/\varepsilon)} && \text{(by convexity of } \hat{f}_t) \\
&\leq \sum_{t=1}^{T} \hat{g}_t^\top (x_t - \chi_S) + 2MnT^{2/3} + 4M\sqrt{T\log(1/\varepsilon)} \\
&\qquad + 32M\sqrt{n}T^{2/3}\sqrt{\log(2^n/\varepsilon)} && \text{(By (5), (6))} \\
&\leq \sum_{t=1}^{T} \hat{g}_t^\top (x_t - \chi_S) + 34MnT^{2/3} + 36M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)} \\
&\leq \frac{n}{\eta} + \frac{\eta}{2}\sum_{t=1}^{T} \|\hat{g}_t\|^2 + 34MnT^{2/3} + 36M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)} && \text{(By (9))} \\
&\leq \frac{n}{2\eta} + \frac{\eta}{2}\left[\sum_{t=1}^{T} \mathbf{E}_t[\|\hat{g}_t\|^2] + 64M^2T^{4/3}\sqrt{\log(1/\varepsilon)}\right] \\
&\qquad + 34MnT^{2/3} + 36M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)} && \text{(By (4))} \\
&\leq 4MnT^{2/3} + 8MT^{2/3}\sqrt{\log(1/\varepsilon)} \\
&\qquad + 34MnT^{2/3} + 36M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)} && \text{(C.f. proof of Thm 16)} \\
&\leq 38MnT^{2/3} + 44M\sqrt{n}T^{2/3}\sqrt{\log(1/\varepsilon)}.
\end{aligned}$$

This gives the required bound. ■

5. Conclusions and Open Questions

We have described efficient regret minimization algorithms for submodular cost functions, in both the bandit and full information settings. This parallels the work of Streeter and Golovin (2008) who study two specific instances of online submodular *maximization* (for which the offline problem is NP-hard), and give (approximate) regret minimizing algorithms. We leave it as an open question whether there exists an efficient algorithm that attains $O(\sqrt{T})$ regret bounds for online submodular minimization in the bandit setting.

Acknowledgments

The first author would like to acknowledge support for this project from the Israeli Science Foundation (ISF grant 810/11).

Appendix A. Additional Lemmas

In this section we prove auxiliary lemmas that were used in the paper for completeness.

A.1 The FTL-BTL Lemma

The following stability lemma was essentially proved in Theorem 1.1 of Kalai and Vempala (2005). We reprove it here for completeness:

Lemma 21 *Let* $S_t = \arg\min_{S\subseteq[n]}\{\sum_{\tau=1}^{t-1} f_\tau(S) + R(S)\}$ *as in Algorithm 1. Then*

$$Regret_T \;\leq\; \sum_{t=1}^{T} [f_t(S_t) - f_t(S_{t+1})] + R(S^*) - R(S_1).$$

Where $S^* = \arg\min_{S\subseteq[n]} \sum_{t=1}^{T} f_t(S)$.

Proof For convenience, denote by $f_0 = R$, and assume we start the algorithm from $t = 0$ with an arbitrary S_0. The lemma is now proved by induction on T.
Induction base: Note that by definition, we have that $S_1 = \arg\min_S\{R(S)\}$, and thus $f_0(S_1) \leq f_0(S^*)$ for all S^*, thus $f_0(S_0) - f_0(S^*) \leq f_0(S_0) - f_0(S_1)$.
Induction step: Assume that that for T, we have

$$\sum_{t=0}^{T} f_t(S_t) - f_t(S^*) \;\leq\; \sum_{t=0}^{T} f_t(S_t) - f_t(S_{t+1})$$

and let us prove for $T+1$. Since $S_{T+2} = \arg\min_S\{\sum_{t=0}^{T+1} f_t(S)\}$ we have:

$$\begin{aligned}
\sum_{t=0}^{T+1} f_t(S_t) - \sum_{t=0}^{T+1} f_t(S^*) &\leq \sum_{t=0}^{T+1} f_t(S_t) - \sum_{t=0}^{T+1} f_t(S_{T+2}) \\
&= \sum_{t=0}^{T} (f_t(S_t) - f_t(S_{T+2})) + f_{T+1}(S_{t+1}) - f_{T+1}(S_{T+2}) \\
&\leq \sum_{t=0}^{T} (f_t(S_t) - f_t(S_{t+1})) + f_{T+1}(S_{t+1}) - f_{T+1}(S_{T+2}) \\
&= \sum_{t=0}^{T+1} f_t(S_t) - f_t(S_{t+1}).
\end{aligned}$$

Where in the third line we used the induction hypothesis for $S^* = S_{T+2}$. We conclude that

$$\begin{aligned}
\sum_{t=1}^{T} f_t(S_t) - f_t(S^*) &\leq \sum_{t=1}^{T} f_t(S_t) - f_t(S_{t+1}) + [-f_0(S_0) + f_0(S^*) + f_0(S_0) - f_0(S_1)] \\
&= \sum_{t=1}^{T} f_t(S_t) - f_t(S_{t+1}) + [R(S^*) - R(S_1)].
\end{aligned}$$

■

A.2 Proof of Lemma 8

Next, we give a proof of Lemma 8 from the paper of Jegelka and Bilmes (2011), for completeness:

Lemma 8 restatement: *The subgradients g of the Lovász extension $\hat{f} : \mathcal{K} \to [-M, M]$ of a submodular function are bounded by $\|g\|_2 \leq \|g\|_1 \leq 4M$.*

Proof Recall the subgradient definition of proposition 7: Let $x \in \mathcal{K}$. Let $\emptyset = B_0 \subset B_1 \subset B_2 \subset \cdots B_n = [n]$ be an arbitrary maximal chain associated with x, and let $\pi : [n] \to [n]$ be the corresponding permutation. Note that $B_{\pi(i)} = \{j \in [n] : \pi(j) \leq \pi(i)\}$. Then, a subgradient g of $\hat{f}$ at x is given by:

$$g_i \;=\; f(B_{\pi(i)}) - f(B_{\pi(i)-1}).$$

Let $S^+ = \{i : \; g_i \geq 0\}$. First, we claim:

Proposition 22

$$\sum_{i \in S^+} g_i \;\leq\; M - f(\emptyset).$$

Proof Let $\sigma : S^+ \to \{1, 2, \ldots, |S^+|\}$ be the one-to-one mapping that orders the elements of S^+ according to π, that is, for $i, j \in S^+$, we have $\sigma(i) < \sigma(j)$ if and only if $\pi(i) < \pi(j)$. For $i \in [S^+]$, define $C_i = \{j \in S^+ : \; \sigma(j) \leq i\}$, and define $C_0 = \emptyset$. Since σ respects the ordering given by π, for all $i \in S^+$ we have

$$C_{\sigma(i)-1} \;=\; \{j \in S^+ : \; \sigma(j) \leq \sigma(i) - 1\} \;\subseteq\; \{j \in [n] : \; \pi(i) \leq \pi(j) - 1\} \;=\; B_{\pi(i)-1}.$$

Note that $C_{\sigma(i)} = C_{\sigma(i)-1} + i$ and $B_{\pi(i)} = B_{\pi(i)-1} + 1$. Thus by the submodularity of f, we have

$$g_i \;=\; f(B_{\pi(i)}) - f(B_{\pi(i)-1}) \;\leq\; f(C_{\sigma(i)}) - f(C_{\sigma(i)-1}).$$

Thus, we have

$$\begin{aligned}
\sum_{i \in S^+} g_i \;&\leq\; \sum_{i \in S^+} f(C_{\sigma(i)}) - f(C_{\sigma(i)-1}) \\
&=\; \sum_{i=1}^{|S^+|} f(C_i) - f(C_{i-1}) \\
&=\; f(S^+) - f(\emptyset) \\
&\leq\; M - f(\emptyset).
\end{aligned}$$

■

Now let $S^- := [n] \setminus S^+$ be the subset of indices of all negative entries of g. We have

$$\sum_{i \in S^-} g_i \;=\; \sum_{i \in [n]} g_i - \sum_{i \in S^+} g_i \;=\; f([n]) - f(\emptyset) - \sum_{i \in S^+} g_i \;\geq\; -2M.$$

The second equality above follows by the definition of g. Hence, we have

$$\|g\|_1 = \sum_{i \in S^+} g_i - \sum_{i \in S^-} g_i \leq 3M - f(\emptyset) \leq 4M.$$

■

References

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan. McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA*, pages 385–394, 2005. ISBN 0-89871-585-7.

Satoru Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.

Martin Grötschel, Laszlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, 1988.

Carlos Guestrin and Andreas Krause. Beyond convexity - submodularity in machine learning. In *Tutorial given in the 25rd International Conference on Machine Learning (ICML)*, 2008.

James Hannan. Approximation to Bayes risk in repeated play. *In M. Dresher, A. W. Tucker, and P. Wolfe, editors, Contributions to the Theory of Games, volume III*, pages 97–139, 1957.

Elad Hazan and Satyen Kale. Beyond convexity: Online submodular minimization. In *Twenty-Fourth Annual Conference on. Neural Information Processing Systems (NIPS)*, pages 700–708, 2009.

Satoru Iwata. A faster scaling algorithm for minimizing submodular functions. *SIAM J. Comput.*, 32(4):833–840, 2003.

Satoru Iwata and James B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *SODA '09: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms*, pages 1230–1237, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48:761–777, 2001.

Stefanie Jegelka and Jeff Bilmes. Online submodular minimization for combinatorial structures. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 345–352, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

Andreas Krause and Carlos Guestrin. Submodularity and its applications in optimized information gathering. *ACM Transactions on Intelligent Systems and Technology*, 2(4), July 2011.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. In *FOCS*, pages 256–261. IEEE Computer Society, 1989.

S. Thomas McCormick. Submodular function minimization. In *Chapter 7 in the Handbook on Discrete Optimization*, pages 321–391. Elsevier, 2006.

James B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Math. Program.*, 118(2):237–251, 2009. ISSN 0025-5610. doi: http://dx.doi.org/10.1007/s10107-007-0189-2.

Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

Matthew J. Streeter and Daniel Golovin. An online algorithm for maximizing submodular functions. In *NIPS*, pages 1577–1584, 2008.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference (ICML)*, pages 928–936, 2003.

Journal of Machine Learning Research 13 (2012) 2923-2954 Submitted 5/11; Revised 2/12; Published 10/12

Efficient Methods for Robust Classification Under Uncertainty in Kernel Matrices

Aharon Ben-Tal ABENTAL@IE.TECHNION.AC.IL
William Davidson Faculty of Industrial Engineering and Management,
Technion- Israel Institute of Technology
Technion City, Haifa 32000, Israel

Sahely Bhadra SAHELY@CSA.IISC.ERNET.IN
Chiranjib Bhattacharyya CHIRU@CSA.IISC.ERNET.IN
Department of Computer Science and Automation,
Indian Institute of Science, Bangalore -560012
Karnataka, India

Arkadi Nemirovski NEMIROVS@ISYE.GATECH.EDU
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology, Georgia 30332-0205
Atlanta, Georgia, USA

Editor: John Shawe-Taylor

Abstract

In this paper we study the problem of designing SVM classifiers when the kernel matrix, $\mathbf{K}$, is affected by uncertainty. Specifically $\mathbf{K}$ is modeled as a positive affine combination of given positive semi definite kernels, with the coefficients ranging in a norm-bounded uncertainty set. We treat the problem using the Robust Optimization methodology. This reduces the uncertain SVM problem into a deterministic conic quadratic problem which can be solved in principle by a polynomial time Interior Point (IP) algorithm. However, for large-scale classification problems, IP methods become intractable and one has to resort to first-order gradient type methods. The strategy we use here is to reformulate the robust counterpart of the uncertain SVM problem as a saddle point problem and employ a special gradient scheme which works directly on the convex-concave saddle function. The algorithm is a simplified version of a general scheme due to Juditski and Nemirovski (2011). It achieves an $O(1/T^2)$ reduction of the initial error after T iterations. A comprehensive empirical study on both synthetic data and real-world protein structure data sets show that the proposed formulations achieve the desired robustness, and the saddle point based algorithm outperforms the IP method significantly.

Keywords: robust optimization, uncertain classification, kernel functions

1. Introduction

The Support Vector Machine(SVM) formulation (Vapnik, 1998) learns a classifier of the form

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{1}$$

from a training data set $D = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{X}, y_i \in \{1,-1\}\ i = 1,\ldots,n\}$. The coefficients, α, are determined by solving

$$\max_{\alpha \in S_n, t} \quad \alpha^\top e - \tfrac{1}{2}t \qquad \text{s.t.} \quad \alpha^\top Y \mathbf{K} Y \alpha \leq t \tag{2}$$

where $S_n = \{\alpha | 0 \leq \alpha_i \leq C,\ \sum_{i=1}^n \alpha_i y_i = 0\}$ and $Y = diag(y_1,\ldots,y_n)$. Each entry of the matrix $\mathbf{K}$, is defined by $\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, is a kernel function and it defines a dot product in an associated *Reproducing Kernel Hilbert Space* (Mercer, 1909; Shawe-Taylor and Cristianini, 2000). As a consequence of $K(\cdot,\cdot)$ being a dot product, the matrix $\mathbf{K}$ needs to be positive semi-definite (see, e.g., Shawe-Taylor and Cristianini, 2000) for any positive integer n.

Observations emanating from real world data are often plagued by uncertainty. The problem of designing classifiers for uncertain observations remain an interesting open problem and has gained considerable interest in the recent past. Previous attempts (Ghaoui et al., 2003; Bhattacharyya et al., 2004; Shivaswamy et al., 2006; Bhadra et al., 2009; Ben-Tal et al., 2011) at designing robust classifiers have been limited to the case of linear classification where the uncertainty is specified over an explicitly stated feature map.

Consider the problem of automated protein structure classification, an important problem of Computational Biology, where no such feature map is available. Protein Structures are specified by a set of 3D coordinates and it is possible to design kernel functions for protein structures based on the coordinates (Qiu et al., 2007; Bhattacharya et al., 2007). Unfortunately the coordinates are not known precisely and this makes the kernel values uncertain. Motivated by this problem (Bhadra et al., 2010) initiated a study of designing robust classifiers when the entries of the kernel matrix are independently distributed random variables (a somewhat problematic assumption). The approach, based on Chance-Constraints (probabilistic) formalism, leads to a non-convex problem which may result in an invalid (i.e., indefinite) kernel matrix.

In this paper we propose a Robust Optimization(RO) approach which overcomes the above drawbacks. The approach employs a geometric description of uncertainty instead of the probabilistic description used earlier (Bhadra et al., 2010). The uncertainty in the kernel matrix $\mathbf{K}$ is modeled by a bounded convex set, which encompasses several possible realizations of $\mathbf{K}$. This new approach results first in a robust counterpart of the uncertain SVM which can be cast as a Conic Quadratic (CQ) problem. Such problems can be solved in polynomial time by Interior Point (IP) algorithm. However for large-scale problems IP methods become intractable. Our main contribution here is to reformulate the robust counterpart as a saddle point problem. Due to favorable conditions satisfied by the saddle function one can in principle refer to a gradient-based general scheme introduced in (Juditski and Nemirovskii, 2011) for solving such saddle point problems. Using this scheme we propose an algorithm, which has a much more simplified analysis, and achieves the same efficiency estimate, namely it achieves the $O(1/T^2)$ reduction in the initial error after T iterations. Experimental results performed on synthetic data, as well as real-world protein structure data sets, show that the saddle-point based algorithm outperforms the IP method considerably. We further conduct detailed experimental evaluation to test the robustness and scalability of the obtained classifiers.

The paper is structured as follows. To motivate the paper we start with a brief discussion on issues underlying protein structure classification and kernel based classifiers in Section 2. In Section 3 we review the formulation in Bhadra et al. (2010) and identify the key shortcomings of the approach. The RO approach for designing robust SVMs is discussed in Section 4. The RO approach leads to a minimax problem. In Section 5 we present the saddle point algorithm and discuss

its application to the minimax problem. In Section 6 we prepare the ground for a comprehensive computational study by introducing various prediction rules and related error metrics. The results of the computational study are described in Section 7.

1.1 Notation

The space of symmetric positive semi-definite $n \times n$ matrices will be denoted as $\mathcal{S}_+^n$. Let $A * B$ denote the the Hadamard prodcuct of two matrices $(A*B)_{ij} = A_{ij}B_{ij}$ where A and B are two square matrices. Frobenius norm of matrix A will be denoted as $\|A\|_F = \sqrt{\sum_{ij} A_{ij}^2}$. Let $\mathbf{1}_A$ be the indicator function for the event A. The uniform random variate will be represented as $U(a,b)$ $(a < b)$. We denote $diag(x_1,\ldots,x_n)$ to be a $n \times n$ diagonal matrix whose ith diagonal entry is x_i.

2. Motivation: Uncertain Kernels and Automated Protein Structure Classification

Classification of protein structures into various classes like families, superfamilies etc remains an important research challenge in computational biology (see Holm and Sander, 1996 for an introduction). Kernel based classifiers are becoming increasingly popular (Qiu et al., 2007; Bhattacharya et al., 2007), for addressing this problem.

Usually a protein structure is specified by the positions of alpha carbon (C^α) atoms. A formal description of C^α atoms and protein structures is beyond the scope of the paper and we refer the reader to Branden and Tooze (1999) for an introduction. In the sequel we will denote protein structure by a set

$$P = \{\mathbf{c}_i \in \mathbb{R}^3 | i = 1, \cdots, s\}, \tag{3}$$

where each C^α atom is determined by spatial coordinates $\mathbf{c}_i = \{\mathbf{c}_{i1}, \mathbf{c}_{i2}, \mathbf{c}_{i3}\}$ obtained by X-ray crystallography. Automated classification of such structures is an extremely useful and challenging problem in computational biology. In the recent past kernel based methods (Qiu et al., 2007; Bhattacharya et al., 2007) have emerged as an interesting alternative to this problem.

Biologists often determine the similarity between a pair of structures by first computing an *alignment* and then measuring the quality of the *alignment* by root mean square deviation(RMSD). We do not formally define the notion of alignment and RMSD in this paper but the refer the interested reader to Shindyalov and Bourne (1998) and Holm and Sander (1996) for an introduction. Though computing structural alignment is an intractable problem there are several heuristic algorithms like DALI (Holm and Sander, 1996), CE (Shindyalov and Bourne, 1998) etc, which works well in practice. Existing literature (Qiu et al., 2007; Bhattacharya et al., 2007) on kernel design rely on structural alignments computed by such programs.

All such procedures implicitly assume that the protein structures are specified exactly, that is, the location of the atoms constituting the structure is known precisely. Unfortunately in reality, the coordinates, $\mathbf{c}_i$, are difficult to determine with exact precision and is highly dependent on the *resolution* of X-ray diffraction experiment.[1] For a protein structure P, the resolution information r, specifies the error in each coordinate. More formally the position of the ith atom in a protein structure P (see (3)) could be anywhere in the uncertainty box $\{\mathbf{c} | \|\mathbf{c} - \mathbf{c}_i\|_\infty \leq r\}$, around the value $\mathbf{c}_i$. For any $r > 0$ one can now define the uncertainty set $U(P)$ for any P as follows

$$U(P) = \{R | R = \{\mathbf{z}_1, \ldots, \mathbf{z}_s\} \ \|\mathbf{z}_i - \mathbf{c}_i\|_\infty \leq r, \ \mathbf{z}_i \in \mathbb{R}^3, \ i = 1, \ldots, s\}. \tag{4}$$

1. See `http://www.rcsb.org/pdb/` for examples.

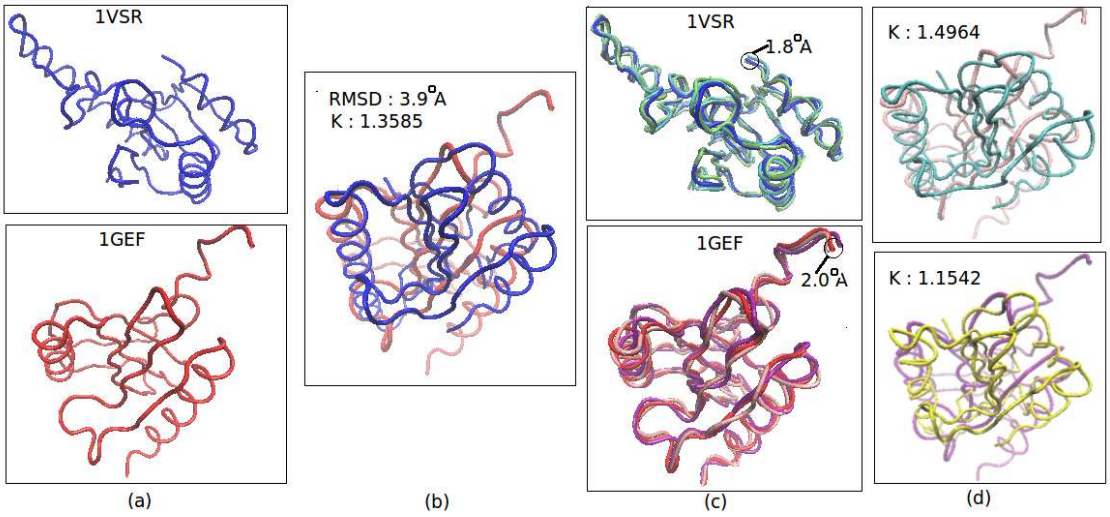


Figure 1: (a) Pictorial presentation of C^{α} atoms of protein **d1vsra1**(top) and **d1gefa1**(bottom). (b) Structural alignment between them. (c) Possible perturbation within resolution limit. (d) Alignment among perturbed structures

Furthermore we would refer to

$$P = \{\mathbf{c}_i \in \mathbb{R}^3 | i = 1, \ldots, n\}$$

as the *nominal* structure and $U(P)$ as the uncertainty set associated with it. The set $U(P)$ characterizes all alternative structures, including P for a given value of r.

The structural alignment between P and P' in presence of uncertainty sets $U(P)$ and $U(P')$ is not defined anymore. Even when r is small, the alignment scores between two *nominal* structures, P and P' can differ significantly from the alignment scores between an arbitrary R and R' where $R \in U(P), R' \in U(P')$. This difference in alignment scores leads to uncertain kernel values.

For example, consider two proteins[2] **d1vsra1**(denote it by P) and **d1gefa1**(denote it by P') belonging to protein superfamily **Restriction endonuclease-like**. The value of r for P is $1.8\mathring{A}$ and for P' it is $2.0\mathring{A}$ respectively. The program DALI computes a structural alignment with RMSD of 3.7 Å between these two structures. Figure 1(a) shows pictorial presentation of C^{α} atoms of these two proteins while Figure 1(b) shows structural alignment between them. If one ignores the uncertainty one obtains a kernel value of 1.3585, using the kernel function described in Bhattacharya et al. (2007). On randomly sampled structures, from the corresponding uncertainty box (4) we observe that the kernel value ranged from $1.1542 (= K_{min}) \leq K(P,P') \leq 1.4964 (= K_{max})$, see Figure 1(c,d). This variability is indeed substantial. In superfamily **Restriction endonuclease-like**, more than 60% of the kernel values, computed between any pair of nominal protein structures, lie between K_{min} and K_{max}.

2. One should refer to them as SCOP domains. But to lighten the discussion on the biology side we refer to them as proteins.

This demonstrates that accounting for resolution information leads to considerable uncertainty in kernel values. There is clearly a need for designing classifiers which can withstand this variation in kernel values. This is an extremely challenging problem which has not been well studied in the literature. Recently (Bhadra et al., 2010) initiated a study of this problem in a probabilistic setting. In the following section we review this work, to identify key shortcomings and subsequently propose a robust optimization procedure to address them.

3. Related Work

In Bhadra et al. (2010), the uncertainty is modeled by independent noise in each of the entries of the kernel matrix $\mathbf{K}$. The uncertain event $\left(\alpha^\top Y(\mathbf{K})Y\alpha \leq t\right)$ is then required to occur with high probability. This results in the following *chance constraint* problem:

$$p^* = \max_{t,\alpha\in S_n} \alpha^\top e - \frac{1}{2}t \tag{5}$$

$$\text{s.t. } Prob\left(\alpha^\top Y(\overline{\mathbf{K}}+Z)Y\alpha \leq t\right) \geq 1-\varepsilon \tag{6}$$

where $\varepsilon < 0.5$, and where Z is a random matrix variate.

Problem (5) is hard to solve since typically the feasible set is non-convex. The key result is the derivation of a lower bound on p^* under probabilistic assumptions on the entries of Z

Theorem 1 ***[Bhadra et al., 2010]*** *Let Z be an $n \times n$ random symmetric matrix with independent entries, Z_{ij}, each having finite support, $P(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1$ and $E(Z_{ij}) = 0$. Then the chance constraint in* (5) *is satisfied at any pair (α, t) which is feasible for the constraint*

$$\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \sqrt{2\log(1/\varepsilon)}\|\beta' * (Y\alpha\alpha^\top Y)\|_F \leq t$$

where $\beta'_{ij} = l_{ij}\gamma_{ij}$ where $l_{ij} = \frac{b_{ij}-a_{ij}}{2}$, $c_{ij} = \frac{b_{ij}+a_{ij}}{2}$, $\hat{\mu}_{ij} = -\frac{c_{ij}}{l_{ij}}$ and

$$\text{and } \gamma_{ij} = \quad \min\{\sigma \geq 0 \mid \tfrac{\sigma^2}{2}z^2 + \hat{\mu}_{ij}z - \log(\cosh(z) + \hat{\mu}_{ij}\sinh(z)) \geq 0,\ \forall\, z \in \mathbb{R}\}.$$

Proof See Bhadra et al. (2010). ■

Theorem 1 is used to replace (5) by the following problem, whose optimal value lower-bounds p^*. Specifically $p^* > \hat{p}$, where

$$\boxed{\begin{aligned} &\hat{p} = \max_{t,\alpha\in S_n} \frac{1}{2}t - \sum_i \alpha_i \\ &\text{s.t. } \sum_{ij} y_i y_j \alpha_i \alpha_j \overline{K}_{ij} + \kappa\sqrt{\sum_{ij}\beta_{ij}\alpha_i^2\alpha_j^2} \leq t \end{aligned}} \tag{7}$$

where, $\kappa = \sqrt{2\log(1/\varepsilon)}$ and $\beta_{ij} = \beta'^2_{ij}$. Any solution of the above problem is guaranteed to satisfy the chance constraint of problem (5).

This approach suffers from several drawbacks. First unless the matrix β is psd, problem (7) is not necessarily convex. Indeed in Bhadra et al. (2010) a locally optimal Quasi newton procedure

was suggested for solving this problem. In the sequel **RSVM** will denote the solution of (7) by the Quasi newton procedure. The second most important drawback is that the constraint (6) does not define a valid model of uncertainty.

To constitute a valid characterization of uncertainty set the constraint (6) needs to be modified as follows

$$Prob\left(\alpha^\top Y(\overline{\mathbf{K}}+Z)Y\alpha \le t\right) \ge 1-\varepsilon,\ \ \overline{\mathbf{K}}+Z \in \mathcal{S}_n^+$$

The formulation (5) solves a relaxed version of the above problem by ignoring the psd requirement. As a consequence the resultant optimization problem becomes non-convex. Thirdly, the assumption that entries of Z are independently distributed is extremely unrealistic; often the uncertainty in the entries are due to uncertainty in the observations hence $K(\mathbf{x}_i,\mathbf{x}_j)$ is seldom independent of $K(\mathbf{x}_i,\mathbf{x}_l)$ for distinct i,j,l.

In this work we pursue a RO methodology where the uncertainty is described by a geometric set. This allows us to alleviate the drawbacks associated with the probabilistic model. In the next section we describe a RO procedure for designing robust classifiers.

4. Affine Uncertainty Set Model for Uncertain Kernel Matrices

In this section we introduce an uncertainty set over psd matrices and study the resultant robust SVM problem using an RO approach.

4.1 Robust Optimization

Consider an uncertain optimization problem (8) where $f, g_i : S \subset (\mathbb{R}^n \times \mathbb{R}^k) \Rightarrow \mathbb{R}$

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x},\Psi) \tag{8}$$

$$g_i(\mathbf{x},\Psi) \le 0\ i=1,\ldots,m$$

where $\Psi \in \mathbb{R}^k$ is a vector of uncertain parameters. The UOP is in fact a *family* of problems -one for each realization of Ψ. In the RO framework the information related to Ψ is modelled as a *geometric* uncertainty set $\mathcal{E} \subset \mathbb{R}^k$ and the family of problems, (8) is replaced by its *robust counterpart*:

$$r^* = \min_{\mathbf{x}} \max_{\Psi\in\mathcal{E}}\ f(\mathbf{x},\Psi) \tag{9}$$

$$g_i(\mathbf{x},\Psi) \le 0\ \forall \Psi \in \mathcal{E}\ i=1,\ldots,m.$$

A solution of (9) is feasible to (8) for any realization of $\Psi \in \mathcal{E}$ and the objective function is guaranteed to be no worse than r^*. The uncertainty set $\mathcal{E}$ is typically a polytope or ellipsoid or intersection of such sets. These sets yield useful models of uncertainty, which lead to tractable optimization problems (Ben-Tal et al., 2009). A general representation of $\mathcal{E}$ is as follows

$$\mathcal{E} = \{\Psi = \Psi + \sum_{i=1}^{L} \eta_i \Psi^i | \|\eta\| \le \rho\}$$

where Ψ is the nominal value of the uncertain vector Ψ, the vectors Ψ^i are possible scenarios of it, and η is a perturbation vector. The norm is suitably defined to capture the geometry of the set. As an example, Consider the ellipsoidal set

$$\mathcal{E}_{ellipsoid} = \{\Psi | (\Psi-\Psi)^\top Q(\Psi-\Psi) \le \rho\}$$

where $Q \in \mathcal{S}_+^n$ and is positive definite. It is easily seen that the set can be represented by $\mathcal{E}$ with Ψ^i being the columns of $Q^{-\frac{1}{2}}$ and where $\|\cdot\|$ is the L_2 norm.

In general the RC of an UOP may have infinite number of constraints and is often NP hard. However in several important cases it reduces to a polynomially solvable convex optimization problem. We refer the reader to Ben-Tal et al. (2009) for a comprehensive treatment of RO problems.

4.2 Affine Uncertainty Set Model

Recall the setup for the problem, there is a black box which when presented with a pair of observations $z, z' \in \mathcal{X}$ computes the kernel value $K(z,z')$. We assume that if z and z' are noisy observations with uncertainty sets $U(z)$ and $U(z')$ with nominal values z_{nom} and z'_{nom} respectively then $\overline{K}(z,z') = K(z_{nom}, z'_{nom})$, defines a kernel function and will be called the *nominal kernel*. The difference between actual and the nominal kernel is expressed by a linear combination of known L kernel functions, $K_l, l = 1,\ldots,L$ evaluated at points z, z', as follows:

$$K(z,z') - \overline{K}(z,z') = \sum_{l=1}^{L} \eta_l K_l(z,z').$$

When there is no uncertainty $K(z,z') = \overline{K}(z,z')$ and $\eta = 0$. The value of $K(z,z')$ lies in the uncertainty set

$$\{\overline{K}(z,z') + \sum_{l=1}^{L} \eta_l K_l(z,z') | \|\eta\|_p \leq \kappa,\ \eta_l \geq 0\ \forall l = 1,\ldots,L\} \tag{10}$$

where $\|\eta\|_p, p \geq 1$ denotes the l_p norm on η. The constraint $\eta_l \geq 0$ is needed to ensure that each element in the set represents a valid kernel evaluation. The quantity κ measures the quality of approximation and hence the uncertainty. If $\kappa = 0$ then we have no uncertainty. As κ increases the uncertainty set increases. In the sequel we will refer to K, K_l as base kernels.

We impose the uncertainty set (10) to all examples of interest which immediately leads to the following model of uncertainty on the kernel matrix corresponding to the training set,

$$\mathcal{E}(\kappa) = \{\mathbf{K} = \overline{\mathbf{K}} + \sum_{l=1}^{L} \eta_l \mathbf{K}_l,\ \|\eta\|_p \leq \kappa\ \eta_l \geq 0, l = 1,\ldots,L\}. \tag{11}$$

The matrices $\overline{\mathbf{K}}, \mathbf{K}_l \in \mathcal{S}_n^+$ are obtained by evaluating the known kernel functions $\overline{K}, K_l$ on the training set. As any $\mathbf{K} \in \mathcal{E}(\kappa)$ is always positive semi-definite, the set $\mathcal{E}(\kappa)$ defines a valid model for describing uncertainty in psd matrices. In a later subsection we will discuss the relevance of this setup to protein structure classification problem.

The Robust SVM problem (2) with uncertain $\mathbf{K}$, as characterized in (11), can now be cast as follows

$$\max_{\alpha \in S_n} \min_{\mathbf{K} \in \mathcal{E}(\kappa)} \quad -\frac{1}{2}\alpha^\top Y \mathbf{K} Y \alpha + \alpha^\top e$$

or more explicitly

$$\max_{\alpha \in S_n} \quad \min_{\|\eta\|_p \leq \kappa} \quad -\frac{1}{2}\alpha^\top Y \overline{\mathbf{K}} Y \alpha - \frac{1}{2}\alpha^\top Y \sum_{l=1}^{L} (\eta_l \mathbf{K}_l) Y \alpha + \alpha^\top e \quad . \tag{12}$$

Note that in the latter problem the constraint $\eta \geq 0$ is dropped. Indeed if we define $a_l = \alpha^\top Y \mathbf{K}_l Y \alpha$ then $a_l \geq 0$ as $\mathbf{K}_l \in \mathcal{S}_n^+$. The optimal η is the solution of $\max_{\eta \in B_p} a^\top \eta$ where $B_p(\kappa) = \{\eta | \|\eta\|_p \leq \kappa\}$.

which occurs at $\eta_l = \kappa \frac{a_l^{q-1}}{\|a\|_q^{q-1}} \geq 0$ where $\|\cdot\|_q$ is the dual norm of $\|\cdot\|_p$, with $q = \frac{p}{p-1}$ for any $p > 1$. For $p = 1$ one needs to observe that optimality is achieved at $\eta_l \geq 0$. Indeed there exists some l such that $a_l = \|a\|_\infty$. The optimal η is given by the condition that $\sum_{l:a_l=\|a\|_\infty} \eta_l = 1$ and $\eta_l \geq 0$. If $a_l < \|a\|_\infty$ then η_l is strictly 0. At optimality $\eta_l \geq 0$, and hence the resulant optimal kernel lies in $\mathcal{E}(\kappa)$.

Before proceeding further it might be useful to discuss the Computation of b. Recall that one also needs to compute b in (1). The choice of b is governed by the following procedure. For a given $\overline{\mathbf{K}}, \mathbf{K}_1, \ldots, \mathbf{K}_L$ let the optimal solution of (12) be α^* and η^*. This η^* can be viewed as defining an effective kernel $\mathbf{K} = \overline{\mathbf{K}} + \sum_{l=1}^{L} \eta_l^* \mathbf{K}_l$ and hence b can be computed as

$$b = \frac{1}{\#\ SV} \sum_{j \in SV} [y_j - \sum_i y_i \alpha_i^* K_{ij}] \ \ SV = \{i | \alpha_i > 0\}.$$

For $p = 2$ the problem (12) can be solved as a Second Order Cone Program(SOCP)[3]

$$\begin{aligned} \min_{\alpha \in S_n, a, t} \quad & \tfrac{1}{2} t + \tfrac{1}{2} \alpha^\top Y \overline{\mathbf{K}} Y \alpha - \alpha^\top e \\ s.t \quad & \|a\|_2 \leq t \\ & \alpha^\top Y \mathbf{K}_l Y \alpha \leq a_l, \ \ \forall l = 1, \ldots, L. \end{aligned} \tag{13}$$

SOCP problems such as (13) can be solved by Interior point(IP) algorithms (e.g., CPLEX, MOSEK, Sedumi) and will be denoted by **Uncertainty-Set SVM** ($\mathbf{USSVM_{SOCP}}$).

Generic IP solvers are inadequate for large scale classification problems. Instead, we demonstrate here that the minimax reformulation, (12), admits algorithms which are better suited to large scale problems. For simplicity of exposition we will consider the case $p = 2$. The results can be easily extended for the general case, $p > 1$. Before we discuss the algorithmic aspects it maybe useful to discuss the computation of the base kernels.

4.3 Evaluation of Base Kernels

Recall that in the protein structure classification problem each observation is specified by a $(P, U(P), y)$, where P(see (3)) is the nominal structure, $U(P)$(see (4)) is the uncertainty set specified by the resolution and y is the label. We discuss this problem in a formal setup and motivate the uncertainty set described in (10).

To closely parallel the protein structure classification setting we consider the following setup. Given a data set $D = \{(\overline{\mathbf{x}}_i, U_i) | \overline{\mathbf{x}}_i \in U_i, U_i \in \mathcal{X}, i = 1, \ldots, m\}$ where an observation, $\mathbf{x}_i$, is not directly specified, instead a nominal value, $\overline{\mathbf{x}}_i$, and an uncertainty set U_i are given. When there is no uncertainty the set U_i reduces to only $\overline{\mathbf{x}}_i$. We are also given a kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. We make no assumptions about the functional form of $K(\cdot, \cdot)$, and we assume that there is a black box which when presented with any pair $z, z' \in \mathcal{X}$ returns a value $K(z, z')$.

Let us now consider the computation of $K(\mathbf{x}_i, \mathbf{x}_j)$. Since $\mathbf{x}_i, \mathbf{x}_j$ are uncertain the precise value of $K(\mathbf{x}_i, \mathbf{x}_j)$ is not known but it for sure lies in the set

$$\mathcal{U}(\mathbf{x}, \mathbf{x}') = \{K(z, z') | z \in U(\mathbf{x}), z' \in U(\mathbf{x}')\}.$$

3. This is also true for any $(p > 1)$ because any p-norm can be represented by conic quadratic inequalities. For a more detailed discussion on this issue see Ben-Tal and Nemirovski (2001).

If there is no uncertainty then the set $\mathcal{U}(\mathbf{x},\mathbf{x}')$ is a singelton, $K(\overline{\mathbf{x}},\overline{\mathbf{x}}')$. However as the functional form of K is not known, it is not clear how to characterize the elements of $\mathcal{U}$ which is amenable to the RO procedure. We propose to circumvent this problem by building an alternate description of the set $\mathcal{U}$ using a sampling procedure. We make L independent draws from the uncertainty sets. In the lth draw we obtain $O^l = \{z_1^l, \ldots, z_m^l\}$ where z_i^l is an independent and uniform draw from U_i. For a given O^l we invoke the kernel function K to obtain an $m \times m$ kernel matrix K_l where $K_l(\mathbf{x}_i,\mathbf{x}_j) = K(z_i^l, z_j^l)$. Once K_l are determined we propose to approximate $\mathcal{U}$ by uncertainty set described in (10).

5. An Algorithm for a Special Class of Convex-Concave Saddle Point Problems

In this section we describe a novel algorithm, which is essentially a special case of an algorithm presented in Juditski and Nemirovski (2011), for a class of convex-concave saddle point problems. The algorithm is iterative in nature, requires only first order information, and has $O(1/T^2)$ convergence where T is the total number of iterations. The proposed algorithm applies generally, more specifically we show that it can be used to solve (12) and the convex version of (7).

5.1 Assumptions

Let U be a closed convex set in Euclidean space E, $\|\cdot\|$ be a norm on E, and $\omega(u) : U \to \mathbf{R}$ be a function. We say that $\omega(\cdot)$ is a distance-generating function (d.-g.f.) for U compatible with $\|\cdot\|$, if ω is convex and continuous on U, admits a continuous on the set $U^o = \{u : \partial\omega(\cdot) \neq \emptyset\}$ selection of $\omega'(u) \in \partial\omega(u)$ and is strongly convex, modulus 1 w.r.t. $\|\cdot\|$:

$$\langle \omega'(u) - \omega'(u'), u - u' \rangle \geq \|u - u'\|^2 \ \forall u, u' \in U^o.$$

Assume that

- X is a closed and bounded convex subset in a Euclidean space $\mathcal{X}$, and Y is a closed convex subset in a Euclidean space $\mathcal{Y}$
- $\mathcal{X}, \mathcal{Y}$ are equipped with norms $\|\cdot\|_{\mathcal{X}}, \|\cdot\|_{\mathcal{Y}}$, the conjugate norms being $\|\cdot\|_{\mathcal{X},*}, \|\cdot\|_{\mathcal{Y},*}$;
- X is equipped with a d.-g.f. $\omega_X(\cdot)$ compatible with $\|\cdot\|_{\mathcal{X}}$, and $\mathcal{Y}$ is equipped with a d.-g.f. $\omega_{\mathcal{Y}}(\cdot)$. We denote by x_ω the minimizer of $\omega_X(\cdot)$ on X (note that $x_\omega \in X^o$) and set

$$\Omega_X = \max_{x \in X} \omega_X(x) - \min_{x \in X} \omega_X(x).$$

  We assume that the minimizer of $\omega_{\mathcal{Y}}(\cdot)$ is the origin in $\mathcal{Y}$, and set

$$\Omega_{\mathcal{Y}} = \max_{\|y\|_{\mathcal{Y}} \leq 1} \omega_{\mathcal{Y}}(y) - \min_{y \in \mathcal{Y}} \omega_{\mathcal{Y}}(y).$$

We are interested in solving a saddle point problem

$$\mathrm{SadVal} = \min_{x \in X} \max_{y \in Y} \phi(x,y), \tag{14}$$

where $\phi(\cdot,\cdot)$ satisfies the following assumptions:

A.1. $\phi(x,y) : Z := X \times Y \to \mathbf{R}$ is continuously differentiable function which is convex in $x \in X$ and is strongly concave, modulus $\theta > 0$ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, in $y \in Y$, that is,

$$\langle \phi'_y(x,y') - \phi'_y(x,y), y - y' \rangle \geq \theta \|y - y'\|^2_{\mathcal{Y}} \ \forall (x \in X, y, y' \in Y).$$

A.2. $\nabla\phi(\cdot,\cdot)$ is Lipschitz continuous on $Z = X \times Y$

A.3. $\phi(x,y)$ is affine in x: $\phi(x,y) = \langle x, a(y) \rangle + g(y)$.

In the sequel, we set

$$\overline{\phi}(x) = \max_{y \in Y} \phi(x,y), \quad \underline{\phi}(y) = \min_{x \in X} \phi(x,y),$$

so that $\overline{\phi}$ is a continuous convex function on X, $\underline{\phi}$ is a continuous strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, function on Y, and

$$\min_{x \in X} \overline{\phi}(x) = \text{SadVal} = \max_{y \in Y} \underline{\phi}(y)$$

and the set of saddle points of ϕ on $X \times Y$ is $X_* \times \{y_*\}$, where $X_* = \text{Argmin}_X \overline{\phi}$, and $y_* = \text{Argmax}_Y \underline{\phi}$. Finally, we denote by $\varepsilon_{\text{sad}}(z), z \in Z$, the natural saddle point proximity measure:

$$\varepsilon_{\text{sad}}(x,y) = \overline{\phi}(x) - \underline{\phi}(y) = \left[\overline{\phi}(x) - \min_X \overline{\phi}\right] + \left[\max_Y \underline{\phi} - \underline{\phi}(y)\right].$$

5.2 Fixed Step-size per Stage(FSS) Algorithm for Convex-Concave Saddle Point Problem

The MPb algorithm presented in Juditski and Nemirovski (2011) is an extremely fast algorithm for convex-concave saddle point problems. Computation proceeds in several stages, each stage consisting of multiple updates involving varying stepsizes. Here we introduce a variation of the MPb algorithm called FSS, which employs fixed stepsize at every stage of the algorithm. We prove that this apparent limitation does not harm the theoretical convergence. The proof (see Appendix A) needs milder assumptions and is much simpler than the original MPb algorithm.

We present a variation of the algorithm where the stepsizes are fixed at every stage without any loss of convergence efficency. In the following we present the Fixed stepsize per stage(FSS) version of the MPb algorithm along with convergence analysis.

5.2.1 FSS ALGORITHM

We begin by introducing some notation

$$G(x,y) = \left[G_x(x,y) := \frac{\partial \phi(x,y)}{\partial x} = a(y); G_y(x,y) := -\frac{\partial \phi(x,y)}{\partial y}\right] : Z := X \times Y \to \mathcal{Z} := \mathcal{X} \times \mathcal{Y}$$

be the monotone operator associated with the saddle point problem (14). By A.1, this operator is Lipschitz continuous, and, as we see, its x-component depends solely on y. As a result, we can specify the partial Lipschitz constants L_{xy}, L_{yy} such that

$$\begin{array}{l} \forall (x, x' \in X, y, y' \in Y): \\ \|G_x(x,y) - G_x(x,y')\|_{\mathcal{X},*} \leq L_{xy} \|y - y'\|_{\mathcal{Y}}; \ \|G_y(x,y) - G_y(x',y)\|_{\mathcal{Y},*} \leq L_{xy} \|x - x'\|_X; \\ \|G_y(x,y) - G_y(x,y')\|_{\mathcal{Y},*} \leq L_{yy} \|y - y'\|_{\mathcal{Y}}. \end{array} \tag{15}$$

We are now ready to present FSS algorithm. Execution of the algorithm is split into *stages*. At the beginning of stage $s = 0, 1, ...$ we have at our disposal positive R_s and a point $y_s \in Y$ such that

$$\|y_s - y_*\|_{\mathcal{Y}} \leq R_s/2. \tag{I_s}$$

These data define the quantities

$$\begin{array}{ll}
(a) & Z_s = \{(x;y) \in Z : \|y - y_s\|_{\mathcal{Y}} \leq R_s\}, \\
(b) & \mathcal{L}_s = 2L_{xy}\sqrt{\Omega_X \Omega_{\mathcal{Y}}} R_s + L_{yy}\Omega_{\mathcal{Y}} R_s^2,\ \tau_s = 1/\mathcal{L}_s \\
(c) & \alpha_s = [L_{xy}\sqrt{\Omega_X \Omega_{\mathcal{Y}}} R_s]/\mathcal{L}_s, \beta_s = [L_{xy}\sqrt{\Omega_X \Omega_{\mathcal{Y}}} R_s + L_{yy}\Omega_{\mathcal{Y}} R_s^2]/\mathcal{L}_s = 1 - \alpha_s, \\
(d) & \omega_s(x,y) = \left[\frac{\alpha_s}{\Omega_X}\omega_X(x) + \frac{\beta_s}{\Omega_{\mathcal{Y}}}\omega_{\mathcal{Y}}([y - y_s]/R_s)\right], \\
(e) & N_s = \mathrm{Ceil}\left(\frac{64 L_{xy}\sqrt{\Omega_X \Omega_{\mathcal{Y}}} R_s^{-1} + 32 L_{yy}\Omega_{\mathcal{Y}}}{\theta}\right).
\end{array} \tag{16}$$

At stage s we carry out N_s steps of the following recurrence:

1. *Initialization:* We set $z_{1,s} = (x_\omega, y_s) = \mathrm{argmin}_Z\, \omega_s(\cdot)$.

2. *Step* $t = 1, 2, ..., N_s$*:* Given $z_{t,s} \in Z^o$, we compute

$$\begin{array}{rcl}
w_{t,s} & = & \mathrm{argmin}_{u \in Z}\left\{\langle \tau_s G(z_{t,s}) - \omega_s'(z_{t,s}), u\rangle + \omega_s(u)\right\} \\
z_{t+1,s} & = & \mathrm{argmin}_{u \in Z}\left\{\langle \tau_s G(w_{t,s}) - \omega_s'(z_{t,s}), u\rangle + \omega_s(u)\right\}
\end{array} \tag{17}$$

   and pass to step $t+1$, provided $t < N_s$. When $t = N_s$, we define the approximate solution to (14) built in course of s stages as

$$(x^s, y^s) = N_s^{-1} \sum_{t=1}^{N_s} w_{t,s}, \tag{18}$$

   set

$$y^{s+1} = y^s,\ R_{s+1} = R_s/2$$

   and pass to stage $s+1$.

5.2.2 PROOF OF CONVERGENCE

In this section we discuss the convergence properties of the FSS algorithm. To this end we present the following proposition.

Proposition 2 *Let assumptions A.1-3 hold and let $y_0 \in Y$ satisfying (I_0) be given along with $R > 0$. Then, for every s, (I_s) takes place, and*

$$\epsilon_{\mathrm{sad}}(x^s, y^s) \leq \theta R_0^2 2^{-2s-5}, \tag{19}$$

while the total number $M_s = \sum_{i=0}^{s} N_i$ of steps of the algorithm needed to build (x^s, y^s) admits the bound

$$M_s \leq O(1)\left[\frac{L_{xy}\sqrt{\Omega_X \Omega_{\mathcal{Y}}}}{\theta R_0} 2^s + \frac{L_{yy}\Omega_{\mathcal{Y}} + \theta}{\theta}(s+1)\right].$$

In particular, setting

$$s^* = \max\left[s : \frac{L_{xy}\sqrt{\Omega_X\Omega_\mathcal{Y}}}{L_{yy}\Omega_\mathcal{Y}+\theta}2^s \le (s+1)R_0\right],$$

we have

$$\begin{aligned} s \le s^* \Rightarrow \quad & \varepsilon_{\text{sad}}(x^s,y^s) \le \theta R_0^2 2^{-2(s+2)} \ \&\ M_s \le O(1)\tfrac{L_{yy}\Omega_\mathcal{Y}+\theta}{\theta}(s+1) \\ s > s^* \Rightarrow \quad & \varepsilon_{\text{sad}}(x^s,y^s) \le O(1)\tfrac{L_{xy}^2\Omega_X\Omega_\mathcal{Y}}{\theta M_s^2} \ \&\ M_s \le O(1)\tfrac{L_{xy}\sqrt{\Omega_X\Omega_\mathcal{Y}}}{\theta R_0}2^s. \end{aligned} \tag{20}$$

Proof See Appendix A. ■

The above proposition points us to the convergence rate of $O(1/M_s^2)$ when number of stages is high. In the following, we explain how to apply the algorithm to the problem at hand.

5.3 Application of FSS Algorithm to (12)

In this section we discuss the application of FSS algorithm for solving formulation (12). It is *strictly concave* in α, provided $\overline{\mathbf{K}}$ is positive definite, and *affine* in η. The domain of α and η are non-empty convex sets. Clearly the formulation obeys all the assumptions, A.1-3, of the FSS procedure. In order to be consistent with the notation of FSS procedure we define the following map where LHS denotes quantities involving formulation (12) and the right hand side corresponds to the saddle point procedure detailed in the previous section. In particular we use $\eta \to x$, $\alpha \to y$, $y \to \mathbf{s}$, $YK_lY \to Q_l$, $\mathbf{E}_l = \mathbb{R}^L$, leading to the following definitions.

$$\begin{aligned} Y =& \{y \in \mathbb{R}^n : C \ge y_i \ge 0,\ \sum_i \mathbf{s}_i y_i = 0\} \subset \mathbb{R}^n \\ X =& \{x \in \mathbb{R}^L : x \ge 0, \|x\|_2 \le 1\} \subset \mathbb{R}^L \\ \phi(x,y) =& -\sum_{l=1}^{L} x_l(\frac{1}{2}y^\top Q_l y) + y^\top e \\ G_x(x,y) =& -\mathbf{d}\ ,\ \mathbf{d} = [d_1,\ldots,d_L]^T\ \ d_l = \frac{1}{2}y^\top Q_l y \\ G_y(x,y) =& (\sum_{l=1}^{L} x_l Q_l)y - \mathbf{e_1}\ ,\ \mathbf{e_1} = [1,\ldots,1]^\top \\ \|x\|_\mathcal{X} =& \|x\|_2,\ \omega_X(x) = \frac{1}{2}\|x\|_2^2\ \ x_\omega = 0\ ,\ \Omega_X = \frac{1}{2} \\ \|y\|_\mathcal{Y} =& \|y\|_2,\ \omega_\mathcal{Y}(y) = \frac{1}{2}\|y\|_2^2,\ \ \Omega_\mathcal{Y} = \frac{1}{2} \\ L_{xy} =& 2CL_{yy},\ \ L_{yy} = \sqrt{L}\max_l(\lambda_{\max}(Q_l))\ ,\ \ R_0 = 2C\sqrt{n} \end{aligned}$$

Proposition 3 *Computing the right-hand sides in every step in (17) is equivalent to solving problems of the form*

$$\begin{aligned} &(a) \quad x_+ = \text{argmin}_{x\in X}\left[\tfrac{1}{2}(x-x)^T(x-x) - p^T x\right], \\ &(b) \quad y_+ = \text{argmin}_{y\in Y}\left[\tfrac{1}{2}(y-y)^T(y-y) - q^T y\right], \end{aligned} \tag{21}$$

where $x \in X$, $y \in Y$ are solution in previous iteration and $p = -\frac{\Omega_X \tau_s}{\alpha_s} G_x(\widetilde{x},\widetilde{y})$ and $q = -\frac{\Omega_Y \tau_s}{\beta_s} R_s^2 G_y(\widetilde{x},\widetilde{y})$. The point $(\tilde{x},\tilde{y})$ are intermediate points in (17).

Proof See that we have $\omega_s(x,y) = \frac{\mu_s}{2}x^T x + \frac{\nu_s}{2}y^T y$ with certain positive μ_s, ν_s (see (16)), hence a computation of the type of (17), that is

$$\begin{aligned}&\{z = (x,y) \in Z^o, \tau > 0, G(\widetilde{x},\widetilde{y})\}\\ &\Rightarrow z_+ = (x_+,y_+) := \text{argmin}_{u \in Z}\left\{\langle \tau G(\widetilde{x},\widetilde{y}) - \omega_s'(z), u\rangle + \omega_s(u)\right\}\end{aligned}$$

reduces to

$$\begin{aligned}x_+ &= \text{argmin}_{x \in X}\left\{x^T[\tau G_x(\widetilde{x},\widetilde{y}) - \mu_s x] + \tfrac{\mu_s}{2}x^T x\right\}\\ &= \text{argmin}_{x \in X}\left\{-p^T x + \tfrac{1}{2}(x-x)^\top (x-x)\right\},\end{aligned}$$

and

$$\begin{aligned}y_+ &= \text{argmin}_{y \in Y}\left\{y^T[\tau G_y(\widetilde{x},\widetilde{y}) - \nu_s y] + \tfrac{\nu_s}{2}y^T y\right\}\\ &= \text{argmin}_{y \in Y}\left\{-y^T q + \tfrac{1}{2}(y-y)^\top (y-y)\right\}.\end{aligned}$$

■

Assuming that we start with a feasible point, that is, $x^0 \geq 0$ and $x^{0^\top} x^0 = 1$, then $x^{k+1} = \frac{p+x^k}{\sqrt{p^T p + 2p^T x^k + 1}}$ The computation steps involving y is solved by projecting a vector onto the constraint set of the Dual SVM problem. The problem can be solved by a Quadratic program. However we propose a line search procedure described in Appendix B which results in considerable saving of computation time. The solution of formulation (12) by FSS procedure will be referred to as **USSVM$_{\mathbf{MN}}$**.

5.4 Application of FSS Algorithm to the Chance Constraint Setting

The FSS procedure is fairly general and also applies to formulation (7). If β is psd then the formulation (7) can be posed as a SOCP. In particular the following is true,

Theorem 4 ***[Bhadra et al., 2010]*** *If both $\overline{\mathbf{K}}, \beta$ are symmetric psd matrices then formulation (7) is equivalent to*

$$\begin{array}{cc}\min_{t,\nu,t',\alpha \in S_n} & \frac{1}{2}t - \sum_i \alpha_i\\ s.t. & \kappa\|\beta^{\frac{1}{2}}\nu\| \leq t - t',\\ & \|Y(\overline{\mathbf{K}})^{\frac{1}{2}}\alpha\|_2^2 \leq t',\\ & \alpha_i^2 \leq \nu_i.\end{array} \tag{22}$$

One could recast this problem as a mini-max problem and solve it using the FSS procedure.

Theorem 5 *Formulation* (22) *is equivalent to*

$$\min_{\zeta \geq 0, \sqrt{\zeta\beta^{-1}\zeta} \leq \frac{\kappa}{2}} \max_{\alpha \in S_n} -\frac{1}{2}\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \sum_{i=1}^n \alpha_i - \sum_i \zeta_i \alpha_i^2. \tag{23}$$

Proof At the optimum of (22) the constraint involving t and t' is active. Using this we eliminate both t and t' and on further dualizing one obtains,

$$\max_{\zeta \geq 0} \min_{\alpha \in S_n} \quad L(\zeta,\alpha)\left(= \frac{1}{2}\left(\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \kappa\sqrt{\nu^\top \beta \nu}\right) - \sum_{i=1}^n \alpha_i + \sum_{i=1}^n \zeta_i(\alpha_i^2 - \nu_i)\right)$$

where ζ_i is the lagrange multiplier of the constraint $\alpha_i^2 \leq \nu_i$. Using first order conditions for optimality, $\frac{\partial L}{\partial \nu_i} = 0$ one obtains

$$\frac{\kappa}{2}(\nu^\top \beta \nu)^{-\frac{1}{2}} \beta \nu + \zeta = 0,$$
$$\sqrt{\zeta^\top \beta^{-1} \zeta} = \frac{\kappa}{2}.$$

Eliminating ν and noting that at optimality the constraint of (23) involving ζ is active, the proof is complete. ■

Note that the objective, in the previous theorem, is Lipschitz continuous, linear in ζ and *strongly concave* in α as long as $\overline{\mathbf{K}}$ is positive definite. Both ζ and α lie in a convex and compact set. In principle the FSS procedure applies and could be an interesting alternative to the SOCP procedure discussed in Bhadra et al. (2010).

5.5 Remarks

In this section we make some remarks on FSS algorithm and its suitability to the problem at hand.

A minimax problem $\min_{x\in X} \max_{y\in Y} \phi(x,y)$ can be approached as follows

$$\min_{x\in X} \left(g(x) = \max_{y\in Y} \phi(x,y) \right). \tag{24}$$

If $\nabla_x g$ were Lipschitz continuous, (24) could be solved at the rate $O(1/T^2)$ by the fast gradient algorithm for smooth convex minimization due to Nesterov (1983). However for our problem, g not necessarily possesses the desired smoothness. However, the situation still allows to achieve $O(1/T^2)$ convergence rate by applying the FSS algorithm to the saddle point reformulation of (24). The FSS algorithm is essentially a modification of the Mirror Prox (MP) algorithm presented in Nemirovski (2004). The prototype algorithm MP solves saddle point problem (24) with convex-concave and smooth (with Lipschitz continuous gradient) ϕ at the rate $O(1/T)$, with the hidden factor in $O(\cdot)$ depending on the distance from the starting point to the solution set of (5.1) (this distance is taken w.r.t. a norm $\|\cdot\|$ assembling $\|\cdot\|_X, \|\cdot\|_\mathcal{Y}$) and the Lipschitz constant of the gradient of ϕ (this constant is taken w.r.t. the conjugate norm $\|\cdot\|_*$). Now, when ϕ is strongly concave in y, the above convergence implies qualified convergence of the y-components y^t of approximate solutions to the y-component y_* of the saddle point of ϕ, so that eventually we know that $\|y^t - y_*\|_\mathcal{Y}$ is, say, twice smaller than (a priori upper bound R on) $\|y^1 - y_*\|_\mathcal{Y}$. When it happens, affinity of ϕ w.r.t. x (Assumption **A**.3) allows to rescale the problem and to restart MP *as if* we were working with a twice smaller domain than the original one, which results in $O(1/T)$ convergence *with reduced hidden factor*. In FSS, we iterate the outlined rescalings and restarts (this is where the stages come from), thus arriving at $O(1/T^2)$ convergence.

6. Prediction Rules and Error Metrics

Recall that in the classical SVM formulation the label of each observation is predicted by (1) where (α, b) is obtained by solving (2). As $K(\mathbf{x}, \mathbf{x}_i)$ is known the classifier predicts a unique label to each test example, $\mathbf{x}$. The quality of the classifier is measured by comparing the prediction with the actual label. In the problem setup considered here the kernel values are only approximately known

and hence the prediction process is not as straightforward as in the case of SVM. In this section we introduce several prediction rules and evaluation measures.

6.1 Prediction Rules

For each observation, $\mathbf{x}_t$, the values, $K(\mathbf{x}_t,\mathbf{x}_i)$, are only approximately known and lies in an uncertainty set (10). Given any (α,b), application of decision rule (1) in this setting is not clear. To this end we propose two heuristics for modelling the prediction process.

The essence of robust classification is that for any choice of $K(\mathbf{x}_t,\mathbf{x}_i)$, governed by (10), the classifer will give the same label. In other words the classifier is robust to uncertainty in the value of K.

The simplest case would be to use $K(\mathbf{x}_t,\mathbf{x}_i)=\overline{K}(\mathbf{x}_t,\mathbf{x}_i)$, which when used in conjunction with (1) gives the following labelling rule:

$$y_t^{pr}=sign\left(\sum_i y_i\alpha_i\overline{K}(\mathbf{x}_t,\mathbf{x}_i)+b\right), \tag{25}$$

which, in the sequel, will be referred to as the *nominal rule*. A more comprehensive process of labelling would involve evaluating all possible choices of $\mathbf{K}$ and see how robust the resultant prediction is. Let $\{\eta^{t1},\ldots,\eta^{tR}\}$, be R uniformly drawn instances of $\{\eta^t\in\mathbb{R}^L|\|\eta\|_p\leq\kappa,\eta_l\geq 0\}$. Each choice of η^t generates a realization of kernel of the form

$$K_t(\mathbf{x}_t,\mathbf{x}_i)=\overline{K}(\mathbf{x}_t,\mathbf{x}_i)+\sum_{l=1}^{L}\eta_l^tK_l(\mathbf{x}_t,\mathbf{x}_i).$$

One option for arriving at a label would be to take the *majority vote* with the above kernel function,

$$y_t^{pr}=\text{sign}\left(\sum_{s=1}^{R}y_t^s\right),\ \ y_t^s=sign\left(\sum_i\alpha_iy_iK_{ts}(\mathbf{x}_t,\mathbf{x}_i)+b\right). \tag{26}$$

Once we have defined these two prediction rule , namely *majority vote* and the *nominal rule*, it is important to devise measures for evaluating the resultant classifiers.

6.2 Error Metrics

Consider a test data set $\mathcal{D}=\{(\mathbf{x}_t,y_t)\ t=1,\ldots,n_{tst}\}$, where, y_t is the true label for observation $\mathbf{x}_t$. We wish to measure the performance of the classifier (26) or (25) on this test data set for a given choice of (α,b).

For the nominal classifier (25) the usual $0/1$ loss works well and we define

$$\textbf{NominalErr}(NE)=\frac{\sum_{t=1}^{n_{tst}}\mathbf{1}_{(y_t^{pr}\neq y_t)}}{n_{tst}}.$$

Similarly for the majority vote based classifier (26), we define

$$\textbf{MajorityErr}(ME)=\frac{\sum_{t=1}^{n_{tst}}\mathbf{1}_{(y_t^{pr}\neq y_t)}}{n_{tst}} \tag{27}$$

where, y_t is the true label for $\mathbf{x}_t$. However as noted before a robust classifier is expected to ensure that $y_t^s=y\ ,s=1,\ldots,R$ is equal for all s. To capture this notion of robustness, we propose another error

Formulation	Required information
Nominal $-$ **SVM** (2)	$\overline{\mathbf{K}}$
RSVM (5)	$\overline{\mathbf{K}}$ and Support
$\mathbf{USSVM_{SOCP}}$(13) $\mathbf{USSVM_{MN}}$(21)	$\overline{\mathbf{K}}$ and $\mathcal{K} = \{\mathbf{K}_1, \ldots, \mathbf{K}_L\}$ (set of valid kernels)

Table 1: Summary of Various formulations and associated Information required.

measure (**RobustErr**) which counts the fraction of data points in $\mathcal{D}$ for which there is atleast one error among R observations. More precisely

$$\mathbf{RobustErr}(RE) = \frac{\sum_{t=1}^{n_{tst}} \mathbf{1}_{(\exists s | y_t^s \neq y_t)}}{n_{tst}} \tag{28}$$

is a more appropriate measure than (ME) to evaluate *robustness*.

In the following section, we report experimental results for the algorithms developed in this paper and benchmark them against the state of the art with respect to the above mentioned metrics.

7. Experimental Evaluation

This section presents experimental evaluation of the formulations, namely **Nominal** $-$ **SVM**, $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$ and **RSVM**. The **Nominal** $-$ **SVM** formulation is the usual SVM formulation (2) with the *nominal* kernel. The minimax problem (12) when solved by the FSS procedure will be referred as $\mathbf{USSVM_{MN}}$. The solution of the SOCP (13) will be referred as $\mathbf{USSVM_{SOCP}}$. Though as discussed before the setup of Bhadra et al. (2010) does not apply here but for sake of completeness we have also included a comparison with **RSVM**. A brief summary of the formulations is presented in Table 7.[4]

In particular it would be interesting to explore the following questions.

1. Comparison of $\mathbf{USSVM_{SOCP}}$ and $\mathbf{USSVM_{MN}}$ against the non-robust **Nominal** $-$ **SVM**.

2. Convergence and scalability of $\mathbf{USSVM_{MN}}$ algorithm

The section is organized as follows. We begin by a brief description of data sets in Section 7.1. A comparative study of robustness is presented in Section 7.2. The first issue is discussed in Section 7.3. The experimental verification of the convergence rate of FSS procedure is discussed in Section 7.4. Next the scalability of $\mathbf{USSVM_{MN}}$ over $\mathbf{USSVM_{SOCP}}$ is discussed in Section 7.5.

7.1 Data Sets

We created synthetic data sets to test the generalization and robustness properties of the proposed formulations. Additionally we also have empirically tested them on protein structure data. We describe them below.

4. Relevant data and scripts are available at `http://mllab.csa.iisc.ernet.in/~sahely/uncertainkernel.html`.

7.1.1 SYNTHETIC DATA SETS AND KERNEL FUNCTIONS

It is important to evaluate the effect of robustness on wide variety of situations. To this end we use the following data generation mechanism suited for binary classification problems.

Choose $d \sim Unif(2,100)$, where $Unif(n_1,n_2)$ is the uniform distribution over all integers from n_1 to n_2. For such a choice of d create a mixture distribution consisting of 4 Gaussian distributions, $N(\mu,\Sigma)$, with diagonal covariance matrix. The mean of each Gaussian distribution, $\mu \in \mathbb{R}^d$, is determined by independently choosing $\mu_j \sim Uniform(-5,5)$ $j = 1,\ldots,d$. Each diagonal entry of the diagonal matrix Σ is independently drawn from $Uniform(0,5)$. We assigned labels to the centers of each Gaussian distribution according to the $sign(w^\top x)$ where $w \in \mathbb{R}^d$ is a random vector with $\|w\| = 1$. A data set of $2N$ points was generated as follows. First a set of N points corresponding to the positive class was generated by sampling N observations from the Gaussian mixture distribution consisting of positively labeled mixture components. The set of N points were generated from negatively labeled mixture components.

We study the problem of robust classification when the kernel values are not available but are governed by (10). Next we describe the construction of **base** kernels needed in (10). A linear kernel will be very effective for any data set D, created by the data generation process described. Given $D = \{(\mathbf{x}_i, y_i)|i = 1,\ldots,N\}$ we define $\overline{\mathbf{K}} = \mathbf{x}_i^\top \mathbf{x}_j$. Furthermore, L kernels were simulated as follows; $\mathbf{K}_l = \overline{\mathbf{K}} + Z^l Z^{l\top}$, where Z^l_{ij} were generated using: a) **Gaussian (0,1)** b) **Uniform [-1,1]** c) centered **Beta (0.5,0.5)** distributions. After that, the generated values were multiplied by a random $l_{ij} \sim \mathbf{Uniform}(\mathbf{0}, \mathbf{0.05}|\overline{\mathbf{K}}_{\mathbf{ij}}|)$. This leads to L valid positive semidefinite kernels for each of the distribution, namely Gaussian, Uniform, or Beta.

We will denote by $\mathbf{D_G}(S,N,L)$, the set of S data sets, $\{D_1,\ldots,D_S\}$. Each data set was created by the data-generation mechanism discussed earlier and has N examples per class with L kernels generated by the Gaussian distribution. Similarly $\mathbf{D_U}(S,N,L)$ and $\mathbf{D_B}(S,N,L)$ will correspond to the Uniform and Beta distribution.

7.1.2 RESOLUTION-AWARE PROTEIN STRUCTURE CLASSIFICATION

We have used a data set based on the SCOP (Murzin et al., 1995) 40% sequence non-redundant data set taken from Bhadra et al. (2010). The data set has 15 classes (SCOP superfamilies), having 10 structures each. The names of these superfamilies are reported in Appendix D. To study the effect of robustness we studied the classification problem on all possible pairs, which gave rise to 105 data sets in total. Each data set D can be thought of $D = \{P_i, y_i, r_i|i = 1,\ldots,n\}$ where P_i is the nominal structure described in (3) with label y_i. Incorporation of resolution information r_i leads to uncertainty sets $U(P_i)$ (see (4)). Using the kernel function described in Bhattacharya et al. (2007) and assuming that the resultant uncertainty in kernel values obey (10) the kernel functions $\overline{\mathbf{K}}, \mathbf{K}_l$ are computed by the procedure outlined in Section 4.3.

As described in the Section 3, the uncertainty set imposed by **RSVM** maynot always be appropriate. However we still provide a comparison to the robust formulations described in this paper for the sake of completion. In the setting of the paper a set of kernel matrices $\mathcal{K} = \{\mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_L\}$ are specified. The formulation **RSVM** needs support information, (see Table 7), which could be extracted as follows

$$\overline{\mathbf{K}} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{K}_l \;\; a_{ij} = \min_l(\mathbf{K}_{l_{ij}}) \;\; b_{ij} = \max_l(\mathbf{K}_{l_{ij}}).$$

The algorithms **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** and **RSVM** have been implemented in Matlab with the help of Sedumi[5] (Sturm, 1999). We have used libSVM[6] as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.66 GHz processors and 16GB of RAM. The **RSVM** implementation uses a Quasi Newton procedure outlined in Bhadra et al. (2010). As it often gets stuck in a local minima we have used multiple starting points. All results on **RSVM** reported here corresponds to the results of the best starting point among 100 randomly selected starting points according to **RSVM** objective function.

7.2 Comparison of Robustness

We begin by studying the effect of robustness on synthetic data. In the proposed model of uncertainty the parameter κ plays a very important role. When $\kappa = 0$, then there is no uncertainty and as it increases the uncertainty becomes more pronounced. The utility of robust formulations would become clear as κ is increased. One would like to experimentally verify the fact that indeed this is the case. To this end we conducted the following experiment.

We created data sets $\mathbf{D_G}(S,N,L), \mathbf{D_U}(S,N,L), \mathbf{D_B}(S,N,L)$ with $S = 10, N = 250, L = 200$, as described in Section 7.1. We have performed 5-fold cross-validation on all the 10 data set. Here we vary $\kappa \in \{0.1, 0.2, 0.3, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 5\}$ with $R = 100$. In Figure 2, we have plotted the RobustErr (28) averaged over all 10 data sets for various distributions and choices of κ. Though we get similar result for few different values of C here we have reported the results for the value of $C = 100$. Again $\mathbf{D_G}$ will refer to the Gaussian distribution, $\mathbf{D_U}$ refers to the uniform case, and $\mathbf{D_B}$ refers to the Beta distribution.

The results of the experiment were as follows. It can be seen from Figure 2 that at $\kappa = 0$, the RE for both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** are exactly same as that of **Nominal** $-$ **SVM**. It confirms the fact that at $\kappa = 0$, **USSVM$_{\mathbf{SOCP}}$**(**USSVM$_{\mathbf{MN}}$**) is equivalent to **Nominal** $-$ **SVM**, as there is no uncertainty. Figure 2 shows that, with the increase of uncertainty in the test examples, the RobustErr(28) for **Nominal** $-$ **SVM** increases substantially when compared to **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** on all the 3 data sets. This shows that, non-robust classifiers, for example, SVM, are unable to handle uncertainty compared to the proposed robust classifiers. Also as expected both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** are equivalent and so on the test one they exhibit similar performance.

7.3 Comparison of Generalization Error

In this section we compare the error measures, RE (28) and ME (27).

We again use the same data sets described in the previous subsection. For all the metrics, we have performed 5-fold cross-validation on all the 10 data sets corresponding to each distribution. The hyper-parameters (C and ε) for each classifier, were chosen using a grid search mechanism from the set $C = \{0.1,\ 1,\ 5,\ 10,\ 50,\ 100,\ 200,\ 500\}$ and $\varepsilon = \{0.05 + 0.05step | step = 0, \ldots, 9\}$. For each metric, the cross-validation accuracy, $100(1 - ErrorMeasure)\%$, averaged over 10 data sets for various distributions, are reported in Figure 3. Note that $\mathbf{D_G}$ refers to the Gaussian case, $\mathbf{D_U}$ refers to Uniform case and $\mathbf{D_B}$ refers to the Beta case. The parameter κ was set to 1.

The results were as follows. All the formulations achieved an accuracy of 90% when NE was used as a error measure. From Figure 3 we see that both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** beats **RSVM** in terms of RE indicating that **RSVM** is not well suited for the uncertainty sets considered

5. Sedumi can be found at `http://sedumi.ie.lehigh.edu/`.
6. LibSVM can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

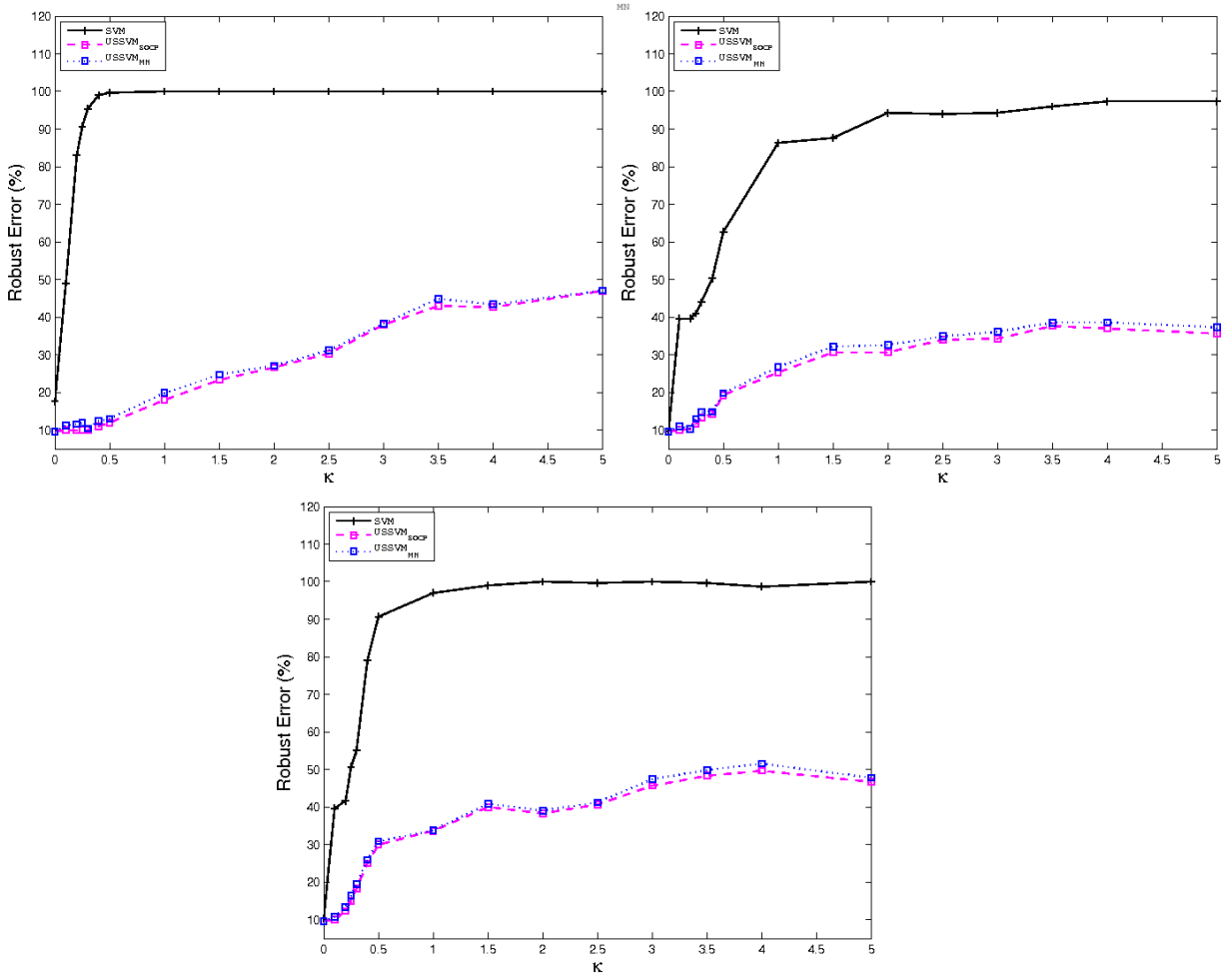


Figure 2: Plot of RE for distributions (clockwise starting from top left)$\mathbf{D_G}, \mathbf{D_U}, \mathbf{D_B}$ with varying κ. Formulations compared are $\mathbf{USSVM_{MN}}, \mathbf{USSVM_{SOCP}}$ and $\mathbf{Nominal-SVM}$. The legend SVM refers to $\mathbf{Nominal-SVM}$

here. When we use ME, which is not as conservative as RE the gap narrows. This experiment demonstrates that in the presence of uncertainty the performance of extremely accurate classifiers suffer drastically but the proposed robust formulations fare much better in handling uncertainty. In addition, Figure 4 shows that the average training time of $\mathbf{USSVM_{SOCP}}$ is same as that of **RSVM** but $\mathbf{USSVM_{MN}}$ is 10 times faster than both of them, even for these small scale data sets (200 training datapoints per class).

7.4 Verification of Convergence of FSS Algorithm

In this section, we have experimentally verified that the proposed saddle point based algorithm has $O(\frac{1}{M_s^2})$ convergence rate (see (20)). Recall that M_s is the actual number of steps, which one can consider as iterations.

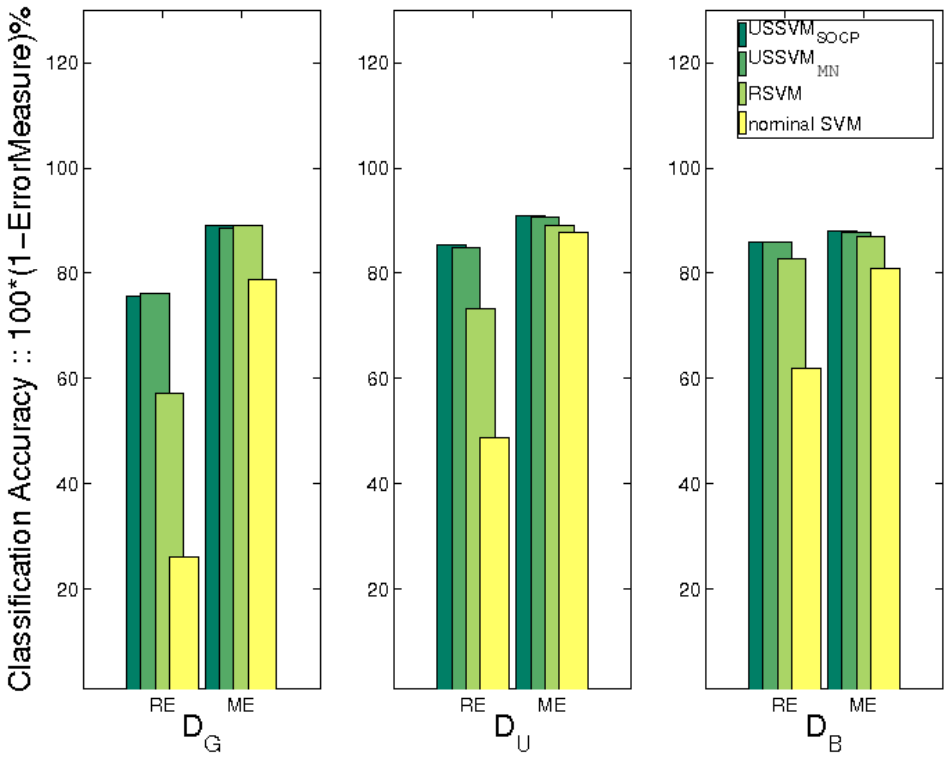


Figure 3: Cross-validation accuracy (%) obtained with $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$, **RSVM** and $\mathbf{Nominal-SVM}$ using RE (28)and ME (27). All values reported here are $100(1-Errormeasure)\%$

We report results on Data Set $\mathbf{D_U}(1,N,5)$ where $N \in \{50,500\}$, and C was chosen to be 1 for $N = 50$ and similarly for $N = 500$ it was fixed to be 10.

Figure 5 shows the convergence rate of the Saddle Point based algorithm for **USSVM** formulation. The x-axis and the y-axis denote $log_{10}(M_s)$ and $log_{10}(\varepsilon_{sad})$ respectively. All the points on the graph indicate the "end" of the s^{th} step and circled points indicate the "end" of the $s*$ step. When $s > s^*$, ideally the graph should be a straight-line with slope less than -2 and one can observe the same in Figure 5. On the other hand, for $s < s*$ rate of decrease in ε_{sad} is much slower than the case in $s > s*$.

7.5 Scalability of $\mathbf{USSVM_{MN}}$

In this section we study the relative performace of $\mathbf{USSVM_{MN}}$ versus $\mathbf{USSVM_{SOCP}}$ on large data sets. We also verify the convergence criteria of proposed $\mathbf{USSVM_{MN}}$.

In the $\mathbf{USSVM_{MN}}$ algorithm the number of stages and the number of iterations inside one stage do not depend on the number of data points, see Section 5.2. In each iteration we need to solve two gradient projection type problems. It appears that they are extremely cheap to compute; in one case there is a closed form solution, while in the other case we could solve it by a line search algorithm. Both $\mathbf{USSVM_{SOCP}}$ and $\mathbf{USSVM_{MN}}$ solves the same problem but the computation required may dif-

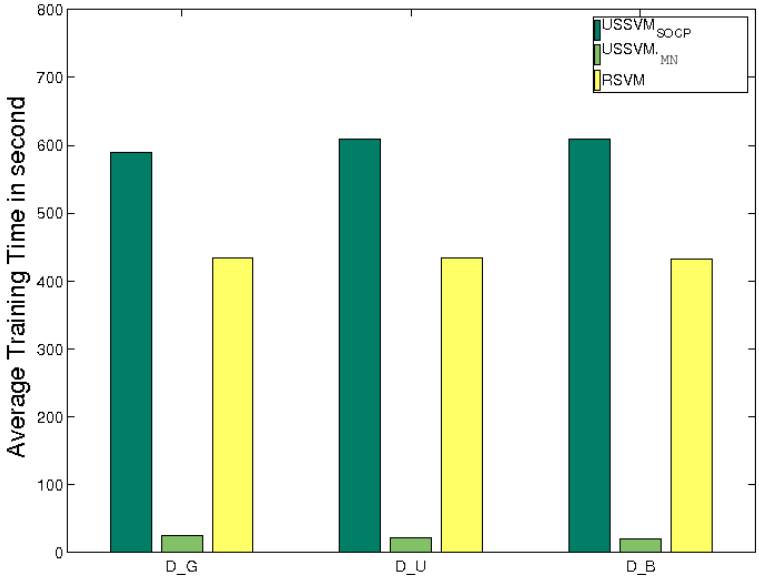


Figure 4: Average Training Time in Seconds obtained with **USSVM$_{\mathbf{SOCP}}$**, **USSVM$_{\mathbf{MN}}$**, and **RSVM**.

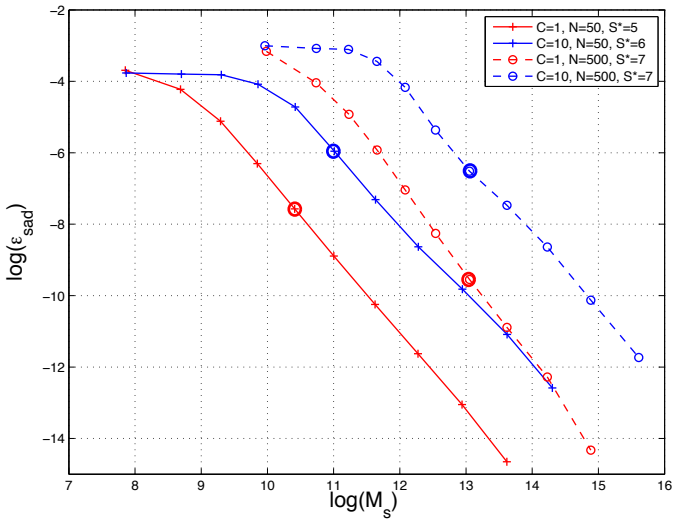


Figure 5: Rate of convergence for Saddle point based algorithm

fer significantly. To this end we have compared the training times for **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** with increasing the number of training data points (N). For this experiment, we have used following data set.

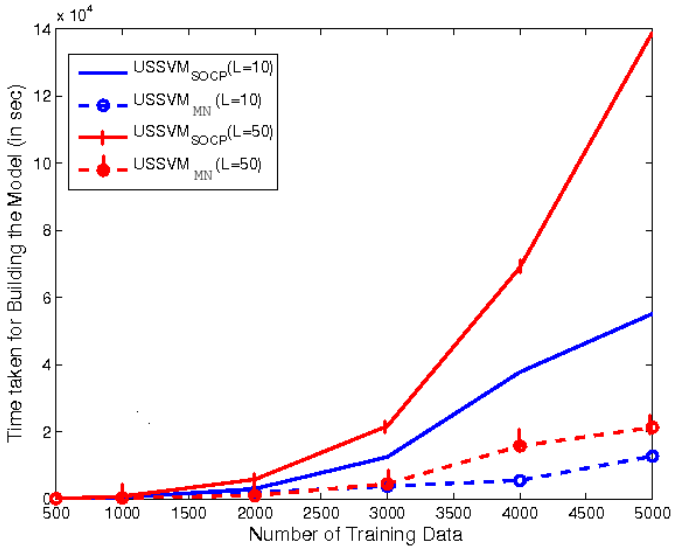


Figure 6: Training time for **USSVM$_{\mathbf{SOCP}}$** **USSVM$_{\mathbf{MN}}$** with $N = [500, 1000, 2000, 3000, 4000, 5000]$ and $L = [10, 50]$

We have used synthetic data set generated similar to D_U (please see Section 7.1) with number of data points in each class are $\{250, 500, 1000, 1500, 2000, 2500\}$, where $L = \{10, 50\}$. The values of $R = 100, C = 10$ and $\kappa = 1$ were used.

Figure 6 shows training time (in sec) for varying N. One can observe that, with the increase of N, the training time for **USSVM$_{\mathbf{SOCP}}$** increases very steeply compared to the training time for **USSVM$_{\mathbf{MN}}$**. As expected, training time for **USSVM** in general increases with the increase of number of uncertain kernels (L). As an example, to build a robust classifier with only 3000 data points, **USSVM$_{\mathbf{SOCP}}$** needs more than **5 hours** while **USSVM$_{\mathbf{MN}}$** completes within **20 minutes**. This concludes that, to build a robust classifier with a medium scale of data (even more than 1000) the saddle point based algorithm is much more effective then a Quadratic Conic Program based formulation.

7.6 Discussion of Experimental Results

The results on the synthetic experiments show that **USSVM$_{\mathbf{SOCP}}$**, **USSVM$_{\mathbf{MN}}$** performs better than **RSVM** in terms of generalization as measured by various error measures. All the three formulations are more robust than **Nominal** $-$ **SVM**. It is also demonstrated that **USSVM$_{\mathbf{MN}}$** is much more scalable than **USSVM$_{\mathbf{SOCP}}$**. Even to build a robust classifier with 3000 data points, **USSVM$_{\mathbf{SOCP}}$** needs more than **5 hours** while **USSVM$_{\mathbf{MN}}$** completes within **20 minutes**.

7.7 Resolution-aware Protein Structure Classification

This section presents experimental results for comparing the robustness performance of the proposed **USSVM**, with the existing **RSVM** formulation.

Error Measure	$\mathbf{USSVM_{SOCP}}$	$\mathbf{USSVM_{MN}}$	**RSVM**	**Nominal − SVM**
Robust Error(RE)	25.14	24.95	20.95	10.38
Majority Error(ME)	85.70	83.36	81.12	71.01
Nominal Error(NE)	80.76	79.67	80.38	71.57

Table 2: Comparison of $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$ **RSVM** and **Nominal − SVM** using accuracy measures, $100(1-ErrorMeasure)\%$, where Error measures are defined in Section 6.2. Table shows average accuracy for all 105 **one-vs-one** classification problems

The data set is described in Section 7.1. The experimental methodology follows "one-vs-one" classification setting with all 15 classes of protein structures. Leave-One-Out (LOO) cross validation using SVM, RSVM and USSVM was performed on all 105 of such classification problem. In all cases we report accuracy, computed as $100(1-ErrorMeasure)\%$.

Let $\mathcal{D} = \{(P_i, r_i, y_i)\}$ be a protein structure data sets where P_i is the set of coordinates of i^{th} protein structure obtained from Astral[7] database, where r_i is the corresponding resolution information obtained from the PDB, and y_i is the class label. Using resolution information, we generated a set of perturbed structures $Q_i = \{P_i^1, \ldots, P_i^L\}$ for each P_i as follows. For each atom p_{i_a} of P_i generated structure P_i^s with coordinates of atoms as $p_{i_a}^l = p_{i_a} + u$ and $u \sim U(\frac{-r_i}{2}, \frac{r_i}{2})$. One can create a set of uncertain kernels, where $K(p,p')$ is a kernel function computed between two protein structures $p \in Q_i$ and $p' \in Q_j$. For our experiments, we have generated a set of kernels consisting of $L = 50$ base kernels. Denoting the kernel matrices by $\{\mathbf{K_1}, \ldots, \mathbf{K_L}\}$ the uncertainty set is defined as $\mathcal{E}(1)$ (see (11)) with $\overline{\mathbf{K}} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{K_l}$ and $\kappa = 1$. Given the base kernels the prediction is implemented, as reported in Section 6, with $R = 100$ and $\kappa = 1$. For the purpose of our comparison, we have used weighted pair-wise distance substructure kernel (Bhattacharya et al., 2007). These kernels are purely based on protein structure (specially position of c^α). Please refer to Appendix C for details. For **RSVM**, we compute the following,

$$\overline{K}_{ij} = \overline{\mathbf{K}}_{ij},\ a_{ij} = \min_{p \in Q_i, p' \in Q_j} K(p,p'), b_{ij} = \max_{p \in Q_i, p' \in Q_j} K(p,p').$$

7.7.1 RESULTS ON PROTEIN STRUCTURE CLASSIFICATION

Table 2 and Table 3 report results for **RSVM**, $\mathbf{USSVM_{MN}}$ and **Nominal − SVM** (SVM with kernels based on *nominal* protein structure reported in PDB files) using both standard and robust error measures defined in section 6.2 in the Leave-One-Out (LOO) procedure. Hyper-parameters (C and/or ε) for **RSVM** and C for $\mathbf{USSVM_{MN}}$ and **Nominal − SVM** were tuned separately using the grid search mechanism. As this is a 15 class classification problem and we followed a "one-vs-one" setting, we have reported average accuracy of all 105 classifiers (Table 2). We have also provided a list of a few individual classes (Table 3) where $USSVM$ performed significantly better than RSVM. The results are presented in the form of a histogram of performance differences (%) of $USSVM$ against $RSVM$ and SVM obtained by using RE (28) in Figure 7.

It is clear that **RSVM**, and **USSVM** perform significantly better than their non-robust counterparts, both in terms of Accuracy (measured by MajorityErr) and Robustness (measured by RobustErr). This result indicates that, the use of resolution information improves the overall classifica-

7. Astral can be found at `http://astral.berkeley.edu`.

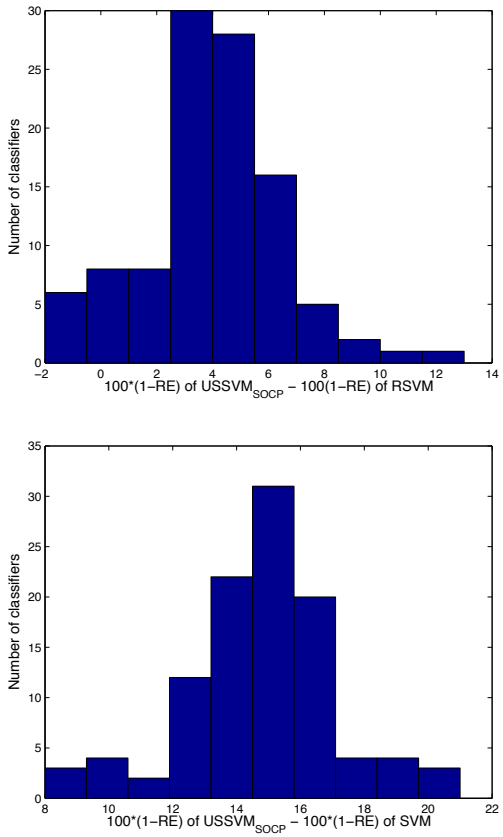


Figure 7: Histogram of performance differences (%) between $\mathbf{USSVM_{SOCP}}$ and **RSVM** is shown in the top figure. The bottom figure corresponds $\mathbf{USSVM_{SOCP}}$ and **SVM**

tion accuracy. In fact, **USSVM** even beats **RSVM** in terms of robustness. Note that, for more than 50% classification problem accuracy of $\mathbf{USSVM_{SOCP}}$ is more than 5% of that of **RSVM** in terms of Robustness. For few classes difference in accuracy was more than 10% (see Table 3). Moreover, this performance difference increases while comparing **USSVM** against $\mathbf{Nominal-SVM}$. For more than 60% classification problem accuracy of $\mathbf{USSVM_{SOCP}}$ is more than 15% of that of $\mathbf{Nominal-SVM}$ in terms of robustness and notably for almost all the cases the margin was more than 10% in term of RobustErr.

8. Conclusion

We studied the problem of designing robust classifiers when the kernel matrices are uncertain. The chance constraint model proposed in Bhadra et al. (2010) made important progress on this problem but it had an important theoretical flaw. It did not constitute a valid model of uncertainty but instead a relaxed version of the original problem. This led to non-convexity and local minima problems. Instead of a chance constraint approach we follow the robust optimization community and advocate a geometric approach. The approach proposed here not only defines a valid model of uncertainty,

Error Measure	$USSVM_{SOCP}$	$USSVM_{MN}$	$RSVM$	**Nominal – SVM**
	c.66.1 vs c.68.1			
Robust Error	31.00	31.00	18.00	11.00
Majority Error	56.00	65.00	50.00	50.00
Nominal Error	50.00	55.00	40.00	50.00
	c.37.1 vs c.55.3			
Robust Error	23.00	23.00	12.00	6.00
Majority Error	60.00	60.00	55.00	51.00
Nominal Error	55.00	55.00	35.00	55.00
	d.58.4 vs c.108.1			
Robust Error	21.00	21.00	12.00	6.00
Majority Error	64.00	70.00	65.00	60.00
Nominal Error	60.00	65.00	60.00	55.00
	c.66.1 vs c.108.1			
Robust Error	27.00	27.00	18.00	12.00
Majority Error	58.00	55.00	60.00	50.00
Nominal Error	55.00	55.00	45.00	55.00
	c.55.1 vs c.2.1			
Robust Error	27.00	27.00	19.00	10.00
Majority Error	70.00	65.00	70.00	60.00
Nominal Error	70.00	55.00	60.00	55.00
	c.66.1 vs d.58.4			
Robust Error	33.00	33.00	25.00	18.00
Majority Error	72.00	65.00	70.00	55.00
Nominal Error	50.00	50.00	65.00	70.00
	b.18.1 vs b.80.1			
Robust Error	23.00	23.00	15.00	6.00
Majority Error	65.00	70.00	63.00	58.00
Nominal Error	60.00	60.00	60.00	50.00
	c.55.3 vs c.55.1			
Robust Error	28.00	25.00	20.00	10.00
Majority Error	88.00	70.00	80.00	65.00
Nominal Error	60.00	60.00	85.00	60.00
	c.66.1 vs d.92.1			
Robust Error	29.00	29.00	22.00	11.00
Majority Error	75.00	75.00	72.00	55.00
Nominal Error	65.00	60.00	70.00	65.00

Table 3: Comparison of $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$ **RSVM**, and **Nominal – SVM** using accuracy measures, $100(1 - ErrorMeasure)\%$, where Error measures are defined in Section 6.2. Table shows accuracy for **one-vs-one** classification problem among few classes. Description of superfamilies of SOCP are in Table 4

but leads to a tractable optimization problem namely a SOCP formulation. However SOCP formulations maynot be well suited for large scale problems. We show that the problem can be equivalently posed as a minmax problem and can be solved by saddle-point algorithm. We adapt the general purpose algorithm of Nemirovski (2004) for solving saddle point procedure to this problem. The algorithm proceeds in *stages*. We propose a novel special case of this algorithm, FSS, where the stepsize remains fixed per stage. The FSS algorithm has same order of $O(1/T^2)$ convergence, T being the number of iterates. This procedure is widely applicable which is a matter of independent study. We demonstrate its applicability to **RSVM** and to the problem at hand. The proposed algorithm, $\mathbf{USSVM_{MN}}$ combines FSS with suitable projection steps and is more scalable than the SOCP formulation, $\mathbf{USSVM_{SOCP}}$. Using a robust optimization based framework we pose the problem of classifier design as a minimax problem. The minimax procedure is solved by a novel FSS procedure which has $O(1/T^2)$ convergence. Empirical results show that $\mathbf{USSVM_{SOCP}}$ is indeed a robust alternative to uncertainty in the kernel matrices both on synthetic and real world data sets. Furthermore experimental results demonstrate that $\mathbf{USSVM_{MN}}$ indeed achieves the theoretical rate of convergence and is a scalable alternative to $\mathbf{USSVM_{SOCP}}$.

Acknowledgments

Chiranjib Bhattacharyya has been supported by a Yahoo! unrestricted Faculty gift. The authors Aharon Ben-Tal and A. Nemirovskii have been supported by the USA-Israel Bi-National Science Foundation(BSF) grant no. 2008302.

Appendix A. Proofs for the Saddle Point Algorithm

In this section we prove the convergence of FSS algorithm.

A.1 An Important Lemma

We start with the following Lemma.

Lemma 6 *Consider stage s of Algorithm FSS, and let*

$$\|(x,y)\| = \sqrt{\frac{\alpha_s}{\Omega_X}\|x\|_{\mathcal{X}}^2 + \frac{\beta_s}{\Omega_{\mathcal{Y}}R_s^2}\|y\|_{\mathcal{Y}}^2} \Rightarrow \|(\xi,\eta)\|_* = \sqrt{\frac{\Omega_X}{\alpha_s}\|\xi\|_{\mathcal{X},*}^2 + \frac{\Omega_{\mathcal{Y}}R_s^2}{\beta_s}\|\eta\|_{\mathcal{Y},*}^2}.$$

Function $\omega_s(\cdot)$ is a d.-g.f. for Z compatible with the norm $\|\cdot\|$, and

$$\begin{array}{ll}(a) & \operatorname{argmin}_Z \omega_s(\cdot) = (x_\omega, y_s) \in Z_s \ \&\ \max_{Z_s}\omega_s(\cdot) - \min_{Z_s}\omega_s(\cdot) \le 1, \\ (b) & \forall (z,z' \in Z) : \|G(z) - G(z')\|_* \le \mathcal{L}_s\|z - z'\|. \end{array} \tag{29}$$

Proof Since $\omega_{\mathcal{Y}}(\cdot)$ is a d.-g.f. for $\mathcal{Y}$, this function is convex and continuously differentiable on the entire $\mathcal{Y}$, It follows that $\omega_s(x,y)$ is continuous convex function on Z, the set $Z^o := \{(x,y) : \partial\omega_s(z) \neq \emptyset\}$ is equal to $\{x : \partial\omega_X(x) \neq \emptyset\} \times Y$, and $\omega_s(x,y)$ admits a continuous on Z^0 selection of subgradient. All we need in order to complete the verification of the fact that ω_s is a d.-g.f. compatible with $\|\cdot\|$ is to verify that ω_s is strongly convex with modulus 1 w.r.t. the latter norm, which is immediate:

with $(x,y),(x',y') \in Z^o$ we have

$$\begin{array}{l}
\langle \omega_s'(x,y) - \omega_s'(x',y'), (x-x', y-y')\rangle \\
= \frac{\alpha_s}{\Omega_X}\langle \omega_X'(x) - \omega_X'(x'), x-x'\rangle + \frac{\beta_s}{\Omega_{\mathcal{Y}}} R_s^{-1}\langle \omega_{\mathcal{Y}}'([y-y_s]/R_s) - \omega_{\mathcal{Y}}'([y'-y_s]/R_s), y-y'\rangle \\
= \frac{\alpha_s}{\Omega_X}\langle \omega_X'(x) - \omega_X'(x'), x-x'\rangle \\
\quad + \frac{\beta_s}{\Omega_{\mathcal{Y}}}\langle \omega_{\mathcal{Y}}'([y-y_s])/R_s) - \omega_{\mathcal{Y}}'([y'-y_s]/R_s), [y-y_s]/R_s - [y'-y_s]/R_s\rangle \\
\geq \frac{\alpha_s}{\Omega_X}\|x-x'\|_X^2 + \frac{\beta_s}{\Omega_{\mathcal{Y}}}(\|y-y'\|_{\mathcal{Y}}^2/R_s^2) = \|(x-x', y-y')\|^2.
\end{array}$$

It remains to prove (29). Relation (29.*a*) is evident. To verify (29.*b*), let $z=(x,y), z'=(x',y') \in Z^o$, let $\Delta x = x'-x$, $\Delta y = y'-y$, $\xi = \|\Delta x\|_X\sqrt{\alpha_s/\Omega_X}$, $\eta = \|\Delta y\|_{\mathcal{Y}}\sqrt{\beta_s/\Omega_{\mathcal{Y}}}R_s^{-1}$, so that $\|z-z'\| = \|[\xi;\eta]\|_2$ by (16). We have

$$\begin{array}{l}
\|G_x(z) - G_x(z')\|_X \leq L_{xy}\|y-y'\|_{\mathcal{Y}},\ \|G_y(z) - G_y(z')\|_{\mathcal{Y}} \leq L_{xy}\|x-x'\|_X + L_{yy}\|y-y'\|_{\mathcal{Y}} \\
\text{[see (15)]} \\
\Rightarrow \|G(z) - G(z')\|_*^2 \leq \frac{\Omega_X}{\alpha_s}L_{xy}^2\|\Delta y\|_{\mathcal{Y}}^2 + \frac{\Omega_{\mathcal{Y}}R_s^2}{\beta_s}\left[L_{xy}\|\Delta x\|_X + L_{yy}\|\Delta y\|_{\mathcal{Y}}\right]^2 \\
= \frac{\Omega_X\Omega_{\mathcal{Y}}R_s^2L_{xy}^2}{\alpha_s\beta_s}\eta^2 + \frac{\Omega_{\mathcal{Y}}R_s^2}{\beta_s}\left[L_{xy}\sqrt{\Omega_X/\alpha_s}\xi + L_{yy}R_s\sqrt{\Omega_{\mathcal{Y}}/\beta_s}\eta\right]^2 \\
= \|M[\xi;\eta]\|_2^2, \\
M = \left[\begin{array}{c|c} & \sqrt{\frac{\Omega_X\Omega_{\mathcal{Y}}}{\alpha_s\beta_s}}R_sL_{xy} \\ \hline \sqrt{\frac{\Omega_X\Omega_{\mathcal{Y}}}{\alpha_s\beta_s}}R_sL_{xy} & \frac{\Omega_{\mathcal{Y}}R_s^2}{\beta_s}L_{yy}\end{array}\right] = \mathcal{L}_sN,\ N = \left[\begin{array}{c|c} & \sqrt{(1-\beta_s)/\beta_s} \\ \hline \sqrt{(1-\beta_s)/\beta_s} & (2\beta_s-1)/\beta_s\end{array}\right],
\end{array}$$

where the relations in the last line are readily given by (16). In view of this computation and the fact that $\|z-z'\| = \|[\xi;\eta]\|_2$, in order to verify (29.*b*) it suffices to show that the spectral norm of the symmetric matrix N is ≤ 1; since N is nonnegative due to $\beta_s \geq 1/2$, see (16), the latter task is exactly the same as verifying positive semidefiniteness of the matrix $I_2 - N$, which is immediate. ■

A.2 Proof of Proposition 2

Consider stage s, assuming that (I_s) take place. For $z \in Z^o$ and $\xi \in \mathcal{Z} := \mathcal{X}\times\mathcal{Y}$, let

$$\begin{array}{l}
V_z^s(u) = \omega_s(u) - \omega_s(z) - \langle \omega_s'(z), u-z\rangle : Z \to \mathbf{R}, \\
\mathrm{Prox}_z(\xi) = \mathop{\mathrm{argmin}}_{u\in Z}\{\langle \xi - \omega_s'(z), u\rangle + \omega_s(u)\} : \mathcal{Z} \to Z;
\end{array}$$

note that $\mathrm{Prox}_z(\xi)$ is well defined due to strong convexity of $\omega_s(\cdot)$. Our basic observation is as follows:

Lemma 7 Nemirovski, 2004, cf. Lemma 3.1 *Given $z \in Z^o$, $\xi,\eta \in E$, let $w = \mathrm{Prox}_z(\xi)$ and $z_+ = \mathrm{Prox}_z(\eta)$. Then for all $u \in Z$ it holds*

$$\begin{array}{ll}
\langle \eta, w-u\rangle \leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta, w-z_+\rangle - V_z^s(z_+) & (a) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta-\xi, w-z_+\rangle - V_z^s(w) - V_w^s(z_+) & (b) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \left[\frac{1}{2}\|\eta-\xi\|_*\|w-z_+\| - \frac{1}{2}\|z-w\|^2 - \frac{1}{2}\|z_+-w\|^2\right] & (c) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \frac{1}{2}[\|\eta-\xi\|_*^2 - \|w-z\|^2] & (d).
\end{array} \tag{30}$$

Proof. By definition of $z_+ = \mathrm{Prox}_z(\eta)$ we have $\langle \eta - \omega_s'(z) + \omega_s'(z_+), u - z_+\rangle \geq 0$; rearranging terms and taking into account the definition of $V_v^s(u)$, we get (a). By definition of $w = \mathrm{Prox}_z(\xi)$ we have $\langle \xi - \omega_s'(z) + \omega_s'(w), z_+ - w\rangle \geq 0$, whence $\langle \eta, w - z_+\rangle \leq \langle \eta - \xi, w - z_+\rangle + \langle \omega_s'(w) - \omega_s'(z), z_+ - w\rangle$; replacing the third term in the right-hand side of (a) with this upper bound and rearranging terms, we get (b). (c) follows from (b) due to the strong convexity of ω_s implying that $V_v^s(u) \geq \frac{1}{2}\|u - v\|^2$, and (d) is an immediate consequence of (c). □

Applying Lemma 7 to $z = z_{t,s}$, $\xi = \tau_s G(z_{t,s})$ (which results in $w = w_{t,s}$) and $\eta = \tau_s G(w_{t,s})$ (which results in $z_+ = z_{t+1,s}$), we obtain due to (30) for all $u \in Z$:

$$\tau_s \langle G(w_{t,s}), w_{t,s} - u\rangle \leq V^s_{z_{t,s}}(u) - V^s_{z_{t+1,s}}(u) + \frac{1}{2}\underbrace{\left[\tau_s^2 \|G(w_{t,s}) - G(z_{t,s})\|_*^2 - \|w_{t,s} - z_{t,s}\|^2\right]}_{\delta_{t,s}}.$$

Observe that $\delta_{t,s} \leq 0$ by $(29.b)$ and by definition of τ_s (see (16)), we arrive at

$$\tau_s \langle G(w_{t,s}), w_{t,s} - u\rangle \leq V^s_{z_{t,s}}(u) - V^s_{z_{t+1,s}}(u)\ \forall u \in Z. \tag{31}$$

Let $u \in Z_s$, and let

$$\overline{\phi}_s(x) = \max_{y \in Y : \|y - y_s\|_{\mathcal{Y}} \leq R_s} \phi(x, y).$$

Summing up (31) over $t = 1, ..., N_s$, taking into account that $V^s_{z_{1,s}}(u) \leq 1$ due to $u \in Z_s$ by $(29.a)$, that $V_z^s(u) \geq 0$, we get

$$\forall u \in Z_s : \frac{1}{N_s}\sum_{t=1}^{N_s} \langle G(w_{t,s}), w_{t,s} - u\rangle \leq A := \frac{1}{\tau_s N_s}.$$

On the other hand, setting $w_{t,s} = (x_{t,s}, y_{t,s})$, $u = (x, y)$ and noting that $z^s = (x^s, y^s) = \frac{1}{N_s}\sum_{t=1}^{N_s} w_{t,s}$ (see (18)), we have

$$\begin{array}{ll}
\frac{1}{N_s}\sum_{t=1}^{N_s} \langle G(w_{t,s}), w_{t,s} - u\rangle & \\
= \frac{1}{N_s}\sum_{t=1}^{N_s} \left[\langle \phi_x'(x_{t,s}, y_{t,s}), x_{t,s} - x\rangle + \langle \phi_y'(x_{t,s}, y_{t,s}), y - y_{t,s}\rangle\right] & \\
\geq \frac{1}{N_s}\sum_{t=1}^{N_s} [[\phi(x_{t,s}, y_{t,s}) - \phi(x, y_{t,s})] + [\phi(x_{t,s}, y) - \phi(x_{t,s}, y_{t,s})]] & (a) \\
= \frac{1}{N_s}\sum_{t=1}^{N_s} [\phi(x_{t,s}, y) - \phi(x, y_{t,s})] & \\
\geq \phi(\frac{1}{N_s}\sum_{t=1}^{N_s} x_t, y) - \phi(x, \frac{1}{N_s}\sum_{t=1}^{N_s} y_t) = \phi(x^s, y) - \phi(x, y^s) & (b)
\end{array} \tag{32}$$

(inequalities in $(a), (b)$ are due to the convexity-concavity of ϕ), so that (32) results in $\phi(x^s, y) - \phi(x, y^s) \leq A$ for all $(x, y) \in Z_s$. Taking supremum in $(x, y) \in Z_s$, we arrive at

$$\overline{\phi}_s(x^s) - \underline{\phi}(y^s) \leq A \leq \frac{\mathcal{L}_s}{N_s}, \tag{33}$$

where the concluding inequality follows from the definition of A due to $\tau_s = 1/\mathcal{L}_s$. Observe that the left-hand side in (33) is $\geq \overline{\phi}_s(x^s) - \mathrm{SadVal}$ (due to $\underline{\phi}(y^s) \leq \mathrm{SadVal}$), while the right-hand side in (33) is $\leq \frac{\theta R_s^2}{32}$ due to $(16.e)$. Thus, (33) implies that

$$\overline{\phi}_s(x^s) - \mathrm{SadVal} \leq \frac{\theta R_s^2}{32}. \tag{34}$$

We claim that in fact $\overline{\phi}_s(x^s) = \overline{\phi}(x^s)$. Indeed, assuming the opposite, let $Y_s = \{y \in Y : \|y - y_s\|_{\mathcal{Y}} \leq R_s\}$, and let $y = \mathrm{argmax}_{y \in Y_s} \phi(x^s, y)$, so that $\overline{\phi}_s(x^s) = \phi(x^s, y)$. Since $\overline{\phi}(x^s) := \max_{y \in Y} \phi(x^s, y) > \overline{\phi}_s(x^s) := \max_{y \in Y_s} \phi(x^s, y)$ and Y_s is cut off Y by the inequality $\|y - y_s\|_{\mathcal{Y}} \leq R_s$, we have $\|y - y_s\|_{\mathcal{Y}} = R_s$, while by (I_s) we have $\|y_* - y_s\|_{\mathcal{Y}} \leq R_s/2$, whence, in particular, $y_* \in Y_s$ and $\|y_* - y_s\|_{\mathcal{Y}} \geq R_s/2$. Since the function $\phi(x^s, y)$ is strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, and attains its maximum in $y \in Y_s$ at y, while $y_* \in Y_s$, we have $\phi(x^s, y_*) \leq \phi(x^s, y) - \frac{\theta}{2}\|y_* - y\|_{\mathcal{Y}}^2 \leq \phi(x^s, y) - \frac{\theta}{8}R_s^2$. It follows that $\mathrm{SadVal} = \min_{x \in X} \phi(x, y_*) \leq \phi(x^s, y_*) \leq \phi(x^s, y) - \frac{\theta}{8}R_s^2 = \overline{\phi}_s(x^s) - \frac{\theta}{8}R_s^2$. The resulting inequality contradicts (34), and this contradiction shows that in fact $\overline{\phi}_s(x^s) = \overline{\phi}(x^s)$. Thus, (33) reads

$$\overline{\phi}(x^s) - \underline{\phi}(y^s) \leq \frac{\mathcal{L}_s}{N_s} \leq \frac{\theta R_s^2}{32}, \tag{35}$$

as required in (19) (recall that by construction $R_s = 2^{-s}R_0$). Finally, (35) implies that

$$\underline{\phi}(y_*) - \underline{\phi}(y^s) = \mathrm{SadVal} - \underline{\phi}(y^s) \leq \overline{\phi}(x^s) - \underline{\phi}(y^s) \leq \frac{\theta R_s^2}{32}; \tag{36}$$

since the function $\underline{\phi}(\cdot)$ is strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, and attains its maximum over $y \in Y$ at y_*, we have $\underline{\phi}(y_*) - \underline{\phi}(y^s) \geq \frac{\theta}{2}\|y_* - y^s\|_{\mathcal{Y}}^2$, which combines with (36) to imply that $\|y_* - y^s\|_{\mathcal{Y}} \leq R_s/4 = R_{s+1}/2$; this is nothing but (I_{s+1}). Thus, we have proved that if (I_s) takes place, then Algorithm FSS ensures (19) and (I_{s+1}). Since (I_0) holds true by assumption, we conclude that (19) and (I_s) take place for all s. All remaining claims in Proposition are now straightforward. □

Appendix B. Projection on the SVM Constraint Set

This appendix discusses the projection step encountered in Section 5.3. We consider the following problem

$$y_+ = \operatorname*{argmin}_{y \in Y} \left\{ -y^T q + \frac{1}{2}(y - y)^T (y - y) \right\},$$

where $Y = \{y \in \mathbb{R}^n | 0 \leq y_i \leq C, \sum_{i=1}^n y_i s_i = 0\}$ is the SVM constraint set and $s_i \in \{1, -1\}$. When $y \in Y$ the optimality conditions yield

$$y_{+i} = \left\{ \begin{array}{ll} 0, & y_i + q_i + \nu s_i \leq 0 \\ C, & y_i + q_i + \nu s_i \geq C \\ y_i + q_i + \nu s_i & otherwise \end{array} \right\}.$$

Furthermore y_+ should satisfy $\sum_i y_{+i} s_i = 0$. It is easy to verify that $min(-\max_{i_+}(y_i + q_i), \min_{i_-}(y_i + q_i) - C) \leq \nu \leq max(C - \min_{i_+}(y_i + q_i), \max_{i_-}(y_i + q_i))$. Since the problem is feasible there exists at least one ν for which $\sum_i y_{+i} s_i = 0$ holds. We compute this by grid search. If there are more than one solution satisfying the constraint $\sum_i y_{+i} s_i = 0$ we choose the solution which yields a lower objective.

Appendix C. Kernel Functions for Protein Structures

Experiments on protein structures have been conducted with **Weighted Pairwise Distance Substructure Kernel** described in Bhattacharya et al. (2007). To make the paper self-contained we describe the kernel function in brief, for more details please see Bhattacharya et al. (2007).

Fix a positive integer l. A substructure N_{i_a} consists of l spatially nearest residues to the ath residue of protein P_i. The substructure kernel between two substructures N_{i_a} and N_{j_b} is defined as

$$K_{pds}(N_{i_a}, N_{j_b}) = \sum_{\pi \in \prod(l)} e^{\frac{-\|d_{i_a} - \pi(d_{j_b})\|^2}{\sigma^2}}$$

where d_{i_a} denotes set of pairwise distance betwen all possible pair of residues in N_{i_a}. Hence $d_i = \{d_i^1, \ldots, d_i^m\}$ where $m = l(l-1)/2$. The distance between any two residues a and b are computed by $\|\mathbf{c}_a - \mathbf{c}_b\|$, see (3), where $\prod(l)$ denote all possible permutations of l residues in the substructure. Finally the kernel function between two protein structures is defined as

$$K(P_i, P_j) = \sum_{a,b=1}^{n_i} \sum_{c,d=1}^{n_j} K_{pds}(N_{i_a}, N_{j_c}) K_{pds}(N_{i_b}, N_{j_d}) K_{norm}(i_a, i_b, j_c, j_d)$$

where $K_{norm}(i_a, i_b, j_c, j_d) = e^{-\frac{(\|\mathbf{c}_{i_a} - \mathbf{c}_{i_b}\| - \|\mathbf{c}_{j_c} - \mathbf{c}_{j_d}\|)^2}{\sigma^2}}$.

Appendix D. List of Superfamilies

We list below the Superfamilies studied in Section 7.7.

Superfamily	Description
b.18.1	Galactose-binding domain-like
b.29.1	Concanavalin A-like lectins/glucanases
b.30.5	Galactose mutarotase-like
b.40.4	Nucleic acid-binding proteins
b.80.1	Pectin lyase-like
c.2.1	NAD(P)-binding Rossmann-fold domains
c.37.1	P-loop containing nucleoside triphosphate hydrolases
c.55.1	Actin-like ATPase domain
c.55.3	Ribonuclease H-like
c.66.1	S-adenosyl-L-methionine-dependent methyltransferases
c.68.1	Nucleotide-diphospho-sugar transferases
c.69.1	alpha/beta-Hydrolases
c.108.1	HAD-like
d.58.4	Dimeric alpha+beta barrel
d.92.1	Metalloproteases (”zincins”), catalytic domain

Table 4: List of Superfamilies

References

A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001. ISBN 0-89871-491-5.

A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

A. Ben-Tal, S. Bhadra, C. Bhattacharyya, and J. S. Nath. Chance constrained uncertain classification via robust optimization. *Math. Program.*, 127(1), 2011.

S. Bhadra, J. Saketha Nath, A. Ben-Tal, and C. Bhattacharyya. Interval data classification under partial information: A chance-constraint approach. In *PAKDD*, pages 208–219, 2009.

S. Bhadra, S. Bhattacharya, C. Bhattacharyya, and A. Ben-Tal. Robust formulations for handling uncertainty in kernel matrices. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 71–78, 2010.

S. Bhattacharya, C. Bhattacharyya, and N. Chandra. Structural alignment based kernels for protein structure classification. In *Proceedings of 24th International Conference on Machine Learning (ICML)*, pages 73–80, 2007. URL `http://doi.acm.org/10.1145/1273496.1273506`.

C. Bhattacharyya, L. R. Grate, M. I. Jordan, L. El Ghaoui, and S. I. Mian. Robust sparse hyperplane classiers: application to uncertain molecular proling data. *Journal of Computational Biology*, 11(6):1073–1089, 2004.

C. Branden and John Tooze. *Introduction to Protein Structure*. Garland Publishing, second edition, 1999. ISBN 0815323050.

L. El Ghaoui, G. R. G. Lanckriet, and G. Natsoulis. Robust Classification with Interval Data. Technical Report UCB/CSD-03-1279, Computer Science Division, University of California, Berkeley, 2003.

L. Holm and C. Sander. Mapping the protein universe. *Science*, 273(5275):595–602, 1996.

A. Juditski and A. Nemirovski. First order methods for large-scale convex optimization. In Sra S., S. Nowozin, and S. Wright, editors, *Optimization for Machine Learning*. The MIT Press, 2011.

J. Mercer. Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations. In *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, volume 209, pages 415–446, 1909.

A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247(4):536–540, April 1995.

A. Nemirovski. Prox-method with rate of convergence $o(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. Optim.*, 15:229–251, 2004.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady, 27*, pages 372–376, 1983.

J. Qiu, M. Hue, A. B.-Hur, J.-P. Vert, and W. S. Noble. A structural alignment kernel for protein structures. *Bioinformatics*, 23(9):1090–1098, 2007.

J. Shawe-Taylor and N. Cristianini. *Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Engineering*, 11(9):739–747, 1998.

P. K. Shivaswamy, C. Bhattacharyya, and A. J. Smola. Second order cone programming approaches for handling missing and uncertain data. *Journal of Machine Learning Research*, 7:1283–1314, 2006.

J. F. Sturm. Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

Journal of Machine Learning Research 13 (2012) 2955-2994 Submitted 7/11; Revised 6/12; Published 10/12

Facilitating Score and Causal Inference Trees for Large Observational Studies

Xiaogang Su XGSU@UAB.EDU
School of Nursing
University of Alabama at Birmingham
1720 2nd Ave S
Birmingham, AL 35294, USA

Joseph Kang JOSEPH-KANG@NORTHWESTERN.EDU
Department of Preventive Medicine
Northwestern University
680 N. Lake Shore, Suite 1410
Chicago, IL 60611, USA

Juanjuan Fan JJFAN@SCIENCES.SDSU.EDU
Richard A. Levine RALEVINE@SCIENCES.SDSU.EDU
Department of Mathematics and Statistics
San Diego State University
5500 Campanile Drive, GMCS 415
San Diego, CA 92182, USA

Xin Yan XIN.YAN@UCF.EDU
Department of Statistics
University of Central Florida
4000 Central Florida Blvd
Orlando, FL 32816, USA

Editor: Peter Spirtes

Abstract

Assessing treatment effects in observational studies is a multifaceted problem that not only involves heterogeneous mechanisms of how the treatment or cause is exposed to subjects, known as propensity, but also differential causal effects across sub-populations. We introduce a concept termed the facilitating score to account for both the confounding and interacting impacts of covariates on the treatment effect. Several approaches for estimating the facilitating score are discussed. In particular, we put forward a machine learning method, called causal inference tree (CIT), to provide a piecewise constant approximation of the facilitating score. With interpretable rules, CIT splits data in such a way that both the propensity and the treatment effect become more homogeneous within each resultant partition. Causal inference at different levels can be made on the basis of CIT. Together with an aggregated grouping procedure, CIT stratifies data into strata where causal effects can be conveniently assessed within each. Besides, a feasible way of predicting individual causal effects (ICE) is made available by aggregating ensemble CIT models. Both the stratified results and the estimated ICE provide an assessment of heterogeneity of causal effects and can be integrated for estimating the average causal effect (ACE). Mean square consistency of CIT is also established. We evaluate the performance of proposed methods with simulations and illustrate their use with the NSW data in Dehejia and Wahba (1999) where the objective is to assess the impact of

a labor training program, the National Supported Work (NSW) demonstration, on post-intervention earnings.

Keywords: CART, causal inference, confounding, interaction, observational study, personalized medicine, recursive partitioning

1. Introduction

Comparative studies that involve evaluation of the effect of an investigational treatment or a putative cause on an outcome variable are fundamental in many application fields. The data may come from either a designed experiment or an observational study. Regardless of the data sources, two major issues exist when assessing the treatment effect: confounding and interaction effects of covariates.

A confounding variable or confounder is an extraneous covariate that relates to both the treatment and the response and hence influences the treatment effect estimation. Controlling or adjusting for confounders can be done in either design or analysis. In designed experiments, randomization, matching, cohort restriction, and stratification are commonly-used ways to effectively control for confounding variables. However, observational studies are often the only available choice due to ethical or practical considerations. Causal inference with observational data is particularly challenging. The main obstacle is the nonrandom treatment assignment mechanism, in which the subjects select a treatment that they believe best serve their interests or are exposed to a treatment according to individual traits. As a result, systematic imbalance or heterogeneity may exist between individuals in the treated group and those in the control group. Thus it is crucial to control for confounders in the analysis stage of such data. Common approaches include analysis of covariance (ANCOVA), propensity score methods (Rosenbaum and Rubin, 1983), and directed acyclic graphs (DAGs; Pearl 2000 and Spirtes, Glymour, and Scheines 2001). Even with randomized experimental data, covariate imbalance can also be revealed when examining data in a multivariate manner. Consider a hypothetical example where m older women and m younger men are assigned to the treated group while m older men and m younger women are assigned to the control group. The data appear to be perfectly balanced in terms of either age or gender, despite the perfect imbalance at their combination levels. When the dimension of covariates gets high, each experimental unit essentially represents an unique individual that is not replicable, which makes randomization less relevant. This partially explains why covariate adjustment is practiced even with randomized experimental data. Associated with variable selection issues, additional challenges present themselves in the form of over- or under- adjustment when confounders are incorrectly identified. For example, under-adjustment occurs when an important confounder is uncollected in the data or excluded from the model. On the other hand, some intermediary outcome variables, often referred to effect-mediators, are important in understanding the mechanism how and why the treatment becomes effective. As an example of over-adjustment, the treatment effect would be under-estimated when a mediator is mistakenly considered as a confounder and included in the model for adjustment. Over-adjustment also may occur when controlling for a collider that correlates with both the treatment and the outcome via an 'M-diagram' (Greenland, 2003).

In terms of influence of covariates on treatment effect assessment, another equally important issue is interaction, also known as effect modification or effect moderation (see, e.g., VanderWeele and Robins 2007 and VanderWeele 2009), which is concerned with differential treatment effects at different levels or values of covariates. An effect modifier is a covariate that interacts with the treatment and changes the direction and/or degree of its causal effect on the outcome. Existence of

interaction complicates model interpretation. Detection of interaction is challenging. While interactions are mostly formulated via cross-product terms in a linear model and restricted to be of the first- or second-order, complex nonlinear or higher-order interactions may exist in reality. It is also important to distinguish between qualitative interactions and quantitative interactions. Qualitative interaction (Gail and Simon, 1985) occurs when there is a directional change in terms of treatment preference, a cause of greater concern to practitioners. Closely related to treatment-by-covariate interactions, subgroup analysis (see, e.g., Lagakos 2006) is an integral part in the analysis of clinical trials. Practitioners and regulatory agencies are keen to know if there are subgroups of trial participants who are more or less likely to be helped or harmed by the intervention under investigation. Subgroup analysis helps explore the heterogeneity of the treatment effect across sub-populations and extract the maximum amount of information from the available data. On the other hand, subgroup analysis is subject to malpractice owing to difficulties in subgroup determination, multiple testings, and lack of power. The new stimulating concept of personalized medicine or personalized treatments (see, e.g., Jain 2009) is intended to refine the traditional medical decisions by capitalizing on results of subgroup analysis or the knowledge of individualized treatment effects. Nevertheless, sorting out differential causal effects often entails large data that are collected at post-trial periods, for example, the Medicare or Medicaid databases.

Assessments of confounding and interaction intervene with each other. First of all, confounding emerges as one primary issue in the assessment of the main effect of treatment, also known as the average causal effect (ACE). However, ACE implicitly assumes homogeneity or unimportant heterogeneity of causal effects. When strong treatment-by-covariate interaction exists, ACE may become less practically useful. This is the case especially when the interaction is qualitative. Suppose, for example, that the treatment effect is δ for half of the data (say, males) and $-\delta$ for the other half (say, females), both having important scientific implications. The ACE in this case is null. When solely based on ACE, one would arrive at the misleading conclusion that the treatment does not have an effect. On the other hand, when the estimation bias caused by inadequately handled confounders gets overwhelming, it may be disguised as differential treatment effects. We shall illustrate more on this point later with simulation in Section 4. Therefore, it is crucial to have both confounding and interaction well addressed in comparative analysis.

Rubin's causal model (Rubin, 1974, 1977, 1978, 2005) provides a general framework for making these assessments, within which the treatment effect is finely calibrated at three different hierarchical levels (i.e., unit, subpopulation, and population) using a counterfactual model and the concept of potential outcomes. In this article, causal inference is explicitly reformulated as a predictive modeling problem within the framework of Rubin's causal model. To approach, we introduce a concept, termed facilitating score, to address both the confounding and interacting impact of extraneous variables on causal inference. Conditional on the facilitating score, homogeneity can be achieved in both the assignment mechanism and and the effect of the treatment. Then we put forward a causal inference tree (CIT) procedure, to approximate the facilitating score with a piecewise constant function. CIT recursively splits data into disjoint groups in such a way that both treatment assignment mechanisms and the treatment effects become more homogeneous within each group. On the basis of CIT, a group of recursive partitioning methods are devised to make causal inference at different levels.

The remainder of this paper is arranged in the following manner. In Section 2, following an outline of Rubin's causal inference framework, the concept of facilitating score is introduced and methods for estimating the facilitating score are discussed. Section 3 presents the CIT methodology

in detail. Section 4 contains simulation studies that are designed to investigate the performance of CIT. An illustration is provided via a real data example in Section 5. In Section 6, we extend the results to situations where the treatment variable is ordinal or nominal. Section 7 ends the article with a brief discussion.

2. Facilitating Scores

We first review Rubin's causal models, then we introduce the facilitating score concept and discuss methods for estimating the facilitating score.

2.1 Causal Inference

In Rubin's causal model (Rubin, 1974, 1977, 1978, 2005), a fine calibration of treatment effect is facilitated by a comparison between the observed outcome on an individual or unit and the potential outcome if the individual had been assigned to the counterfactual treatment group. Adopting his notations, let $\Omega = \{\omega\}$ be a finite population with N units, endowed with a probability measure P that places uniform mass $1/N$ on each unit. Let $T = T(\omega)$ be a binary treatment assignment variable with value 1 if unit ω receives the putative treatment and 0 otherwise. While the term 'treatment assignment' or 'selection' is best suitable for designed experiments, we shall use it throughout this article. In addition, let $\mathbf{X} = \mathbf{X}(\omega)$ be a p-dimensional vector of measured covariates for unit ω.

Let $Y_0 = Y_0(\omega)$ be the response that would have been observed if unit ω were assigned to the control group and let $Y_1 = Y_1(\omega)$ be the response that would have been observed if unit ω received the treatment. These two variables are called *potential outcomes* (Neyman, 1923). In reality, either $Y_0(\omega)$ or $Y_1(\omega)$, but not both, can actually be observed depending on the value of $T(\omega)$, an inherent fact called the *fundamental problem of causal inference* (Holland, 1986). Thus the observed outcome is

$$Y(\omega) = \{1 - T(\omega)\}Y_0(\omega) + T(\omega)Y_1(\omega).$$

Throughout this paper, we consider random sampling from Ω so that $\{\omega_1, ..., \omega_n\}$ forms an independent and identically distributed (iid) sample of size n. The available data $\{(y_i, t_i, \mathbf{x}_i) = (y(\omega_i), t(\omega_i), \mathbf{x}(\omega_i)) : i = 1, \ldots, n\}$ consist of n realizations of Y, T, and $\mathbf{X}$. For the sake of simplicity, we sometimes omit unit ω from the notations.

Causal inference is concerned with the comparison of the two potential outcomes via the observed data. Holland and Rubin (1988) distinguished three levels of causal inferences: unit level, subpopulation level, and population level. The lowest level of causal inference is a comparison of $Y_0(\omega)$ and $Y_1(\omega)$, typically the difference $Y_1(\omega) - Y_0(\omega)$, for each unit ω. Subpopulations can be formed by restricting the values of covariates to a partition of Ω. The causal effect in a subpopulation $\{\omega : \mathbf{X}(\omega) \in B\}$ is $E(Y_1|\mathbf{X} \in B) - E(Y_0|\mathbf{X} \in B)$ for some Borel set B in the predictor space $\mathbb{X}$. The average causal effect (ACE) over the entire population Ω is $E(Y_1) - E(Y_0)$. These three levels form a hierarchy of causal inference in decreasing order of strength, in the sense that knowledge of upper-level causal inferences can be inferred from that of lowered-level causal inferences, but not vice versa. A preponderance of the literature in causal inference is centered on schemes for making the population-level inference or estimating ACE under various scenarios.

Rosenbaum and Rubin (1983) introduced the concept of balancing score to tackle the confounding issue in causal inference. A balancing score $b(\mathbf{x})$ accounts for the dependence between $\mathbf{X}$ and

treatment assignment or selection T; that is

$$\mathbf{X} \perp\!\!\!\perp T \,|\, b(\mathbf{X}).$$

Treated and untreated subjects sharing the same balancing score tend to have the same distribution of covariates. Various covariate adjustment techniques implicitly adjust for an estimated scalar balancing score. They showed that the propensity score

$$e(\mathbf{x}) = P(T = 1|\mathbf{X} = \mathbf{x}),$$

which is defined as the conditional probability of assignment to the treated group given the measured covariates $\mathbf{X}$, is the coarsest balancing score. Namely, $b(\mathbf{x})$ is a balancing score if and only if $b(\mathbf{x})$ is finer than $e(\mathbf{x})$, that is, $e(\mathbf{x})$ is a function of $b(\mathbf{x})$.

Propensity score based matching, stratification (or subclassification), and adjustment have been extensively used to balance the discrepancy in covariates between the treatment groups in the assessment of ACE. In propensity score analysis, the assumption of *strong ignorability* plays a pivotal role. Similar to that of missing at random (MAR) in the missing data literature (Rubin, 1976), this assumption states that $P(T|\mathbf{X}, Y_0, Y_1) = P(T|\mathbf{X})$ or,

$$T \perp\!\!\!\perp (Y_0, Y_1)|\mathbf{X}.$$

It is possible that strong ignorability is violated even there are no unmeasured variables that are direct causes of any pair of measured variables. See, for example, Greenland (2003) for more discussions. It is worth noting that this assumption does not imply that $T \perp\!\!\!\perp Y \,|\, \mathbf{X}$. To illustrate, consider a simple example where the causal effect at the unit level is constant, namely, $Y_1(\omega) - Y_0(\omega) = \delta$ for any ω. Suppose that $Y_0 = f(\mathbf{X}) + \varepsilon$ and $Y_1 = f(\mathbf{X}) + \delta + \varepsilon$, where $\varepsilon \perp\!\!\!\perp \mathbf{X}$ is the error term. It follows that $Y = \delta T + f(\mathbf{X}) + \varepsilon$. The ignorability assumption amounts to $\varepsilon \perp\!\!\!\perp T \,|\, \mathbf{X}$, which, by no means, implies $Y \perp\!\!\!\perp T \,|\, \mathbf{X}$.

Under this assumption of strong ignorability, Rosenbaum and Rubin (1983) established that $(Y_1, Y_0) \perp\!\!\!\perp T|b(\mathbf{X})$ when $0 < e(\mathbf{X}) < 1$.It follows that

$$E(Y_1|b(\mathbf{X}), T = 1) - E(Y_0|b(\mathbf{X}), T = 0) = E(Y_1|b(\mathbf{X})) - E(Y_0|b(\mathbf{X})). \tag{1}$$

Therefore, the population-level causal interpretation may be achieved by averaging over the distribution of $b(\mathbf{X})$,

$$E(Y_1 - Y_0) = E_{b(\mathbf{X})}\{E(Y_1|b(\mathbf{X})) - E(Y_0|b(\mathbf{X}))\}. \tag{2}$$

Equations (1) and (2) provide the basis for propensity score based methods.

2.2 Facilitating Score

Parallel to confounding, interaction is concerned with differential causal effects among units or subpopulations. It is important to note that both Equation (1) and (2) involve a reduction of hierarchy in causal inference, where individual-level inferences are integrated to make subpopulation-level inferences on $\Omega_b = \{\omega : b(\mathbf{X}(\omega)) = b\}$ or sub-population level inferences are reduced to the population-level inference on Ω. Such a reduction may not be taken for granted, because it implicitly assumes homogenous lower-level causal effects. Specifically, if substantial differences in causal effects are present at a lower level of inference, then transition to an upper-level inference may not be plausible

and conclusions based on upper-level causal effects can be misleading. This can be particularly problematic when qualitative interactions exist.

To gain insight, note that with balancing score $b(\mathbf{X})$,

$$\mathbf{X} \not\perp\!\!\!\perp (Y_0, Y_1) \,|\, b(\mathbf{X}).$$

As a result, $\delta_b(\mathbf{X}) = E(Y_1|b(\mathbf{X}) = b) - E(Y_0|b(\mathbf{X}) = b)$ in (2) is not a constant, but a function of $\mathbf{X}$ within the subpopulation Ω_b. If $\delta_b(\mathbf{X})$ varies substantially with $\mathbf{X}$, we say that a treatment-by-covariate interaction exists. In this case, the overall causal effect δ_b in Ω_b becomes less pertinent as it implicitly assumes that $\delta_b(\mathbf{X})$ can be reduced to a constant δ_b. A fine delineation of treatment effect $\delta_b(\mathbf{X})$ at the individual level is desirable in the efforts of advancing personalized medicines. Even if estimating δ_b is of interest, it cannot be summarized by direct comparison of treatment means. Instead, it should be obtained by integrating over the distribution of $\mathbf{X}$ in Ω_b, that is, $\delta_b = \int_{\Omega_b} \delta_b(\mathbf{x})\, d\mu(\mathbf{x})$. Direct comparison of treatment means in Ω_b makes another implicit assumption that, within Ω_b, $\mathbf{X}$ follows a uniform distribution. The same problem remains when using (2) for ACE estimation.

It is therefore critical to take both heterogeneous treatment assignment mechanisms and differential treatment effects into consideration when assessing the treatment effects. We introduce a concept termed facilitating score to address these two issues simultaneously.

Definition 1 *A facilitating score* $\mathbf{a}_0(\mathbf{X})$ *is a* q_0*-dimensional (*$0 < q_0 \leq p$*) function of* $\mathbf{X}$ *such that* $\mathbf{X} \perp\!\!\!\perp (Y_0, Y_1, T) \,|\, \mathbf{a}_0(\mathbf{X})$.

In this definition, the joint independence between $\mathbf{X}$ and (Y_0, Y_1, T) given $\mathbf{a}_0(\mathbf{X})$ can be relaxed as two marginal independence conditions: $\mathbf{X} \perp\!\!\!\perp T \,|\, \mathbf{a}_0(\mathbf{X})$ and $\mathbf{X} \perp\!\!\!\perp (Y_0, Y_1) \,|\, \mathbf{a}_0(\mathbf{X})$, which separately address the confounding effect and the interacting effect of $\mathbf{X}$. But, if strong ignorability, that is, $T \perp\!\!\!\perp (Y_0, Y_1) \,|\, \mathbf{X}$, is further assumed, it follows that $T \perp\!\!\!\perp (Y_0, Y_1) \,|\, \mathbf{a}_0(\mathbf{X})$ and hence the marginal independence implies the joint independence as well. Existence of $\mathbf{a}_0(\mathbf{X})$ is guaranteed, since $\mathbf{X}$ itself can be regarded as a facilitating score.

Nevertheless, Definition 1 places strong requirements on $\mathbf{a_0}(\mathbf{X})$. Estimating the facilitating score essentially involves jointly modeling $\{Y_0, Y_1, T\}$ conditional on $\mathbf{X}$, which is unworkable since (Y_0, Y_1) can not be observed at the same time. To get around this difficulty, we next consider a weaker definition of facilitating score that is more practically useful.

Definition 2 *A weak facilitating score* $\mathbf{a}(\mathbf{X})$ *is a q-dimensional (*$0 < q \leq p$*) function of* $\mathbf{X}$ *such that (i)* $\mathbf{X} \perp\!\!\!\perp T \,|\, \mathbf{a}(\mathbf{X})$ *and (ii)* $E(Y_1 - Y_0|\mathbf{X}) = E(Y_1 - Y_0|\mathbf{a}(\mathbf{X}))$.

By condition (i), a weak facilitating score $\mathbf{a}(\mathbf{X})$ must be a balancing score; by condition (ii), any effect moderation owing to $\mathbf{X}$ can be fully represented by $\mathbf{a}(\mathbf{X})$. Condition (ii) is equivalent to saying that $E(Y_1 - Y_0|\mathbf{a}(\mathbf{X}) = \mathbf{a})$ is independent of $\mathbf{X}$. However, this does not necessarily imply that

$$E(Y_1|\mathbf{X}) = E(Y_1|\mathbf{a}(\mathbf{X})) \;\text{ and }\; E(Y_0|\mathbf{X}) = E(Y_0|\mathbf{a}(\mathbf{X})). \tag{3}$$

There could exist a common function $g(\mathbf{X})$ that has been cancelled out in Condition (ii). Namely,

$$g(\mathbf{X}) = E(Y_1|\mathbf{X}) - E(Y_1|\mathbf{a}(\mathbf{X})) = E(Y_0|\mathbf{X}) - E(Y_0|\mathbf{a}(\mathbf{X})).$$

A facility score must also be a weak facilitating score, but not vice versa. We use the term 'facilitating' because conditioning on $\mathbf{a}(\mathbf{X})$ helps facilitate causal inference, in the sense that causal inference within the sub-population $\Omega_{\mathbf{a}} = \{\omega : \mathbf{a}(\mathbf{X}(\omega)) = \mathbf{a}\}$ can be conveniently obtained via direct comparison of sample mean responses. This is because both propensity and the treatment effect $\delta_{\mathbf{a}}$ become constant within $\Omega_{\mathbf{a}}$.

Since the propensity $e(\mathbf{X})$ is the coarsest balancing score, it follows that $e(\mathbf{X}) = e$ in $\Omega_{\mathbf{a}}$. In some scenarios, $e(\mathbf{X})$ is explicitly a separate component of $\mathbf{a}(\mathbf{X})$, as exemplified by the parametric approach outlined in Section 2.3; but this is not necessarily true in general, as exemplified by the semi-parametric approach outlined in the same section. In terms of stratification, $\Omega_{\mathbf{a}}$ provides additional refinements of $\Omega_e = \{\omega : e(\mathbf{X}(\omega)) = e\}$ in order to achieve homogeneous within-stratum treatment effects.

Theorem 3 *Suppose that the conditional joint density of (Y,T) given $\mathbf{X}$, $f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X})$, can be written as $f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X}) = g(Y,T,\mathbf{h}(\mathbf{X}))$ for some function $g(\cdot)$. In other words, $(Y,T) \perp\!\!\!\perp \mathbf{X}\,|\,\mathbf{h}(\mathbf{X})$. Assuming that treatment assignment is strongly ignorable, $\mathbf{h}(\mathbf{X})$ is a weak facilitating score when $0 < e(\mathbf{X}) < 1$.*

We defer the proof of Theorem 3 to Appendix A, where it is established as a special case of a more general result in Theorem 7. Theorem 3 basically states that both confounding and interacting effect of $\mathbf{X}$ on causal inference with the potential outcomes (Y_1, Y_0) can be handled by working with the observed data $(Y,T,\mathbf{X})$. More specifically, if the joint density of (Y,T) given $\mathbf{X}$ can be accounted for by a vector-valued function $\mathbf{h}(\mathbf{X})$, that is, $(Y,T) \perp\!\!\!\perp \mathbf{X}\,|\,\mathbf{h}(\mathbf{X})$, then $\mathbf{h}(\mathbf{X})$ is a weak facilitating score. Besides, it can be shown that Equation (3) holds for $\mathbf{h}(\mathbf{X})$, that is, $E(Y_1|\mathbf{X}) = E(Y_1|\mathbf{h}(\mathbf{X}))$ and $E(Y_0|\mathbf{X}) = E(Y_0|\mathbf{h}(\mathbf{X}))$. This condition will be relaxed in Section 2.3.

Estimation of $\mathbf{h}(\mathbf{X})$ involves modeling the joint distribution of (Y,T) given $\mathbf{X}$. Searching for a satisfactory $\mathbf{h}(\mathbf{X})$ is not an easy task; we have to look for approximate solutions. On the other hand, it is no longer unattainable as the involved elements $(Y,T,\mathbf{X})$ are all observed. Although $\mathbf{h}(\mathbf{X})$ is generally set as vector-valued, its dimension should be small in order to be practically useful.

2.3 Estimating the Facilitating Score

We shall discuss three proposals for finding useful approximations of $\mathbf{h}(\mathbf{X})$, which are parametric, semiparametric, and nonparametric in nature, respectively. While they are all methodologically interesting, we deem the nonparametric approach most practically useful.

The first method is parametric. Consider

$$\begin{aligned} f(Y,T|\mathbf{X}) &= f(Y|T,\mathbf{X}) \cdot f(T|\mathbf{X}) \\ &= \{f(Y|T=1,\mathbf{X})\}^T \cdot \{f(Y|T=0,\mathbf{X})\}^{1-T} \cdot f(T|\mathbf{X}) \end{aligned} \tag{4}$$

by Bayes' rule. With a parametric approach, we assume a model for each of the terms in (4): propensity score model for $f(T|\mathbf{X})$ and outcome regression models for $f(Y|T=0,\mathbf{X})$ and $f(Y|T=1,\mathbf{X})$. It is convenient to model $T|\mathbf{X}$ with logistic regression and model $Y|(T,\mathbf{X})$ with Gaussian linear regression so that

$$f(Y,T|\mathbf{X}) = \frac{1}{\sigma}\phi\left(\frac{Y-\mu}{\sigma}\right) \cdot \{\pi(h_3(\mathbf{X}))\}^T \{1-\pi(h_3(\mathbf{X}))\}^{1-T}, \tag{5}$$

where σ is the constant error variance;

$$\mu = E(Y|T,\mathbf{X}) = \gamma_0 + \gamma_1 T + h_1(\mathbf{X}) + T \cdot h_2(\mathbf{X}); \tag{6}$$

$\phi(\cdot)$ is the density function of the standard normal $\mathcal{N}(0,1)$ distribution; and $\pi(x) = \exp(x)/(1+\exp(x))$ is the logistic or expit function.

Proposition 4 *Suppose that the propensity model can be specified by $e(\mathbf{X}) = e(h_3(\mathbf{X}))$ as in (5) and the conditional mean response given $(T,\mathbf{X})$ is formulated by (6). Under the assumption of strong ignorability, $\mathbf{h}(\mathbf{X}) = (h_2(\mathbf{X}), h_3(\mathbf{X}))^t$ is a weak facilitating score.*

The proof is provided in Appendix B. Proposition 4 says that $h_1(\mathbf{X})$ is not a necessary component of a weak facilitating score. It holds as long as the conditional mean outcome is specified by (6); in other words, normality is not needed either. Besides, note that Equation (3) is not required with this definition of $\mathbf{h}(\mathbf{X})$.

To continue with the parametric approach, linearity is further enforced so that $h_j(\mathbf{X}) = \beta_j^t \mathbf{X}_j$ for $j = 1,2,3$, where $\mathbf{X}_j$ contains selected components of $\mathbf{X}$. The involved parameters $\theta = \{\beta, \gamma, \sigma\}$ can be estimated via maximum likelihood in a straightforward manner. The likelihood function is

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\sigma} \phi\left(\frac{y_i - \mu_i}{\sigma}\right) \cdot \prod_{i=1}^{n} \left\{\pi(\beta_3^t \mathbf{x}_i)\right\}^{t_i} \left\{1 - \pi(\beta_3^t \mathbf{x}_i)\right\}^{1-t_i} = L_1 \cdot L_2. \tag{7}$$

Clearly there is a variable selection issue involved. Note that (β_1, β_2) are involved only in L_1 for the outcome regression model while β_3 is involved only in L_2 for the propensity score model. This property not only simplifies the likelihood optimization, but also allows for variable selection to be performed separately for the propensity model and outcome regression models.

With an estimated $\widehat{\mathbf{h}}(\mathbf{x}) = (\hat{\beta}_2^t \mathbf{x}, \hat{\beta}_3^t \mathbf{x})^t$, data can be stratified via combined use of the medians or terciles of its components, similar to propensity score subclassification. While this parametric method provides a feasible approach for stratification, there are several difficulties in practice. First of all, it is a two-step approach. The final results rely on correct model specifications. Moreover, the number of strata has to be rather arbitrarily determined. The fact that $\widehat{\mathbf{h}}(\mathbf{x})$ is vector-valued contributes added difficulties to execution. In particular, as the dimension of $\widehat{\mathbf{h}}(\mathbf{x})$ increases, the number of strata grows precipitously. Even with only two categories induced by each component, there are 2^q subclasses for a q-dimensional $\widehat{\mathbf{h}}(\mathbf{x})$.

Another intuitive semi-parametric approach to estimate $\mathbf{h}(\mathbf{X})$ is via dimension reduction techniques. In view of $(Y,T) \perp\!\!\!\perp \mathbf{X} | \mathbf{h}(\mathbf{X})$, if it is further assumed that $\mathbf{h}(\mathbf{X})$ is linear in $\mathbf{X}$ so that $\mathbf{h}(\mathbf{X}) = \mathbf{B}\mathbf{X}$, then the subspace spanned by columns of $\mathbf{B}$, $\mathbb{S}(\mathbf{B})$, is called the dimension-reduction subspace that accounts for the conditional distribution of (Y,T) given $\mathbf{X}$. Let $\mathbb{S}_{(Y,T)|\mathbf{X}}$ denote the intersection of all dimension-reduction subspaces. Under some regular assumptions, $\mathbb{S}_{(Y,T)|\mathbf{X}}$ is also a subspace, termed the central dimension-reduction subspace or central space. Sliced inverse regression (SIR; Li 1991) and its variants can be used to estimate $\mathbb{S}_{(Y,T)|\mathbf{X}}$. While further research efforts are needed in handling the bivariate response (Y,T), there is no additional conceptual complication involved. For example, one convenient approach is to first introduce $(2S)$ slice indicator variables

$$Z_{st} = I\{(y'_{k-1} < Y \le y'_k) \cap (T = t)\},$$

where $s = 0,1,\ldots,S$; $t = 0,1$; and $\{-\infty = y'_0 < y'_1 < \cdots < y'_S = +\infty\}$ are pre-specified grid points that define S slices for Y. Then the sliced regression method (Wang and Xia, 2008) can be applied

to estimate the central mean space of $\mathbf{Z} = (Z_{st})$, which approximates the central space $\mathbb{S}_{(Y,T)|\mathbf{X}}$. Nevertheless, the same above-mentioned difficulties as in the parametric approach remain when it comes to stratification on the estimated linear facilitating scores.

In the next section, we consider yet another recursive partitioning based nonparametric alternative, which seems to provide a more satisfactory solution to the problem. Hereafter, we refer to this method as CIT for causal inference tree. CIT combines estimation of $\mathbf{h}(\mathbf{x})$ and data stratification into one step. On the basis of CIT, we devise methods for making causal inference at different levels.

3. Causal Inference via Recursive Partitioning

Tree-based methods (Morgan and Sonquist 1963 and Breiman et al. 1984) approximate the underlying function of interest with piecewise constants by recursively partitioning the predictor space. At the same time, a tree structure offers natural grouping of data with easily interpretable splitting rules. With an automated algorithmic approach, CIT seeks disjoint groups that have homogeneous joint density of (Y,T) within each. The resultant grouping rules, which are induced by binary splits on the covariates $\mathbf{X}$, are meaningfully interpretable, implying a nonparametric estimation of the facilitating score.

In this section, we first follow the CART (Breiman et al., 1984) convention to construct one single CIT, which consists of three steps: growing a large tree and selecting the optimal subtree via pruning and cross validation. On the basis of CIT, methods for causal inference at different levels are then developed. CIT itself provides a natural stratification of data for subpopulation inference. An aggregated grouping method is introduced in order to enhance its performance. Conditional inference at the individual unit level can also be obtained by combining results from ensemble CIT models. Both stratified and individualized causal effect estimates can help depict variations in propensity and treatment effects and make available a natural evaluation of the plausibility of treatment comparability and ACE assessment. These results can also be integrated for estimating ACE estimates. Finally, we establish the mean square risk consistency of CIT under conditions similar to those in CART (Breiman et al., 1984).

3.1 Causal Inference Trees (CIT)

A tree model can be expressed as a graph with connected nodes, each node corresponding to a subset of the data. We use $\mathcal{T}$ as a generic notation for a tree structure and τ for a node. In tree modeling, the effects of $\mathbf{X}$ are exclusively explained by the splitting rules. To start the tree construction, we consider one single split of data. When restricted to a node τ, the distribution of (Y,T) no longer depends on $\mathbf{X}$, implying a constant propensity and a constant treatment effect. Following decomposition of the joint density $f_\tau(Y,T) = f_\tau(Y|T)f_\tau(T)$ within node τ, it is convenient to impose that

$$T \sim \text{Bernoulli}\,(\pi_\tau) \quad \text{and} \quad Y|T \sim \mathcal{N}\left\{\mu = (1-T)\mu_{\tau 0} + T\mu_{\tau 1},\, \sigma_\tau^2\right\}.$$

We would like to comment that recursive partitioning can be viewed as a localized approach with local optimality achieved at each split. In local areas, the model needs not to be complicated and often employs a parametric form. The procedure starts with splits that are built upon something that is relatively simple and then evolves into a comprehensive model by recursively bisecting.

The resultant tree model is nonparametric in nature and relatively robust to local distributional assumptions.

The associated log-likelihood function becomes

$$l_\tau = -\frac{n_\tau}{2}\ln(2\pi\sigma^2) - \frac{\sum_{i\in\tau}(y_i-\mu_i)^2}{2\sigma^2} + n_{\tau 1}\ln\pi_\tau + n_{\tau 0}\ln(1-\pi_\tau)$$

where $\{n_\tau, n_{\tau 0}, n_{\tau 1}\}$ are the total number of observations in node τ, the number of observations in node τ that are assigned to the control group, and the number of observations in node τ that are assigned to the treatment group, respectively. Maximum likelihood estimates of the involved parameters are explicitly available: $\hat{\pi}_\tau = n_{\tau 1}/n_\tau$, $\hat{\mu}_{\tau 0} = y_{\tau 0}$, $\hat{\mu}_{\tau 1} = y_{\tau 1}$, and $\hat{\sigma}^2 = SSE_\tau/n_\tau$, where

$$SSE_\tau = \sum_{\{i\in\tau:\ t_i=1\}}(y_i - y_{\tau 1})^2 + \sum_{\{i\in\tau:\ t_i=0\}}(y_i - y_{\tau 0})^2,$$

and $\{y_{\tau 0}, y_{\tau 1}\}$ are the sample average responses of the control and treatment groups in node τ, respectively. Up to a constant, the maximized log-likelihood function in node τ is

$$\hat{l}_\tau \propto -\frac{n_\tau}{2}\ln(n_\tau \cdot SSE_\tau) + n_{\tau 1}\ln n_{\tau 1} + n_{\tau 0}\ln n_{\tau 0}. \tag{8}$$

Note that we have assumed a mean-shift Gaussian model with the same constant variance for the causal effect. If different variances are considered, the final form of $\hat{l}_\tau$ would be slightly different.

Without loss of generality, we consider binary splits only. When a split s bisects node τ into the left child node τ_L and the right child node τ_R, the associated likelihood ratio test statistic is

$$LRT(s) = 2\cdot(\hat{l}_{\tau_L} + \hat{l}_{\tau_R} - \hat{l}_\tau), \tag{9}$$

where the maximized log-likelihood score for nodes τ_L and τ_R, $\hat{l}_{\tau_L}$ and $\hat{l}_{\tau_R}$, can be obtained in the same manner as $\hat{l}_\tau$ in (8). The LRT_s can be used as the splitting statistic to select the best split. After removing irrelevant components, we have

$$\begin{aligned} LRT(s) \quad \propto \quad & -n_{\tau_L}/2\cdot\ln(n_{\tau_L}SSE_{\tau_L}) - n_{\tau_R}/2\cdot\ln(n_{\tau_R}SSE_{\tau_R}) + \\ & n_{\tau_L 1}\ln n_{\tau_L 1} + n_{\tau_L 0}\ln n_{\tau_L 0} + n_{\tau_R 1}\ln n_{\tau_R 1} + n_{\tau_R 0}\ln n_{\tau_R 0}. \end{aligned}$$

The best split $s^\star$ is the one that yields the maximum $LRT(s)$ among all allowable splits. Accordingly node τ is split into τ_L and τ_R using $s^\star$. Subsequently, a similar procedure is applied to split either of τ_L and τ_R. We repeat the procedure until some mild stopping rules are satisfied. This process results in a large initial tree, denoted as $\mathcal{T}_0$.

The final tree model is a subtree of $\mathcal{T}_0$. Nevertheless, it is practically infeasible to examine every subtree because the number of subtrees increases rapidly as the number of terminal nodes in $\mathcal{T}_0$ increases. The idea of pruning is to provide a subset of candidate subtrees by iteratively truncating off the 'weakest link' of $\mathcal{T}_0$. There are several pruning algorithms available, including the cost-complexity pruning of CART (Breiman et al., 1984) for trees that are built upon minimizing within-node impurity, the split-complexity pruning of LeBlanc and Crowley (1993) for trees that are built upon maximizing between-node differences, and the AIC pruning of Su, Wang, and Fan (2004) for trees that are built within the maximum likelihood framework. Since CIT is essentially likelihood based, the AIC pruning is adopted for direct use. We shall keep our descriptions concise by referring the reader to appropriate references for greater details.

In the AIC pruning algorithm, the performance of a given tree $\mathcal{T}$ is measured by the Akaike (1974) information criterion:

$$\text{AIC}_{\mathcal{T}} = -2 \cdot \hat{l}_{\mathcal{T}} + \lambda \times (4 \cdot |\widetilde{\mathcal{T}}|)$$

where the associated maximized log-likelihood of $\mathcal{T}$ is

$$\hat{l}_{\mathcal{T}} = \sum_{\tau \in \widetilde{\mathcal{T}}} \hat{l}_{\tau}; \tag{10}$$

$\lambda = 2$ is the penalty parameter for tree complexity; and $|\widetilde{\mathcal{T}}|$ denotes the number of terminal nodes in $\mathcal{T}$, with $\widetilde{\mathcal{T}}$ being the set of all terminal nodes in $\mathcal{T}$ and $|\cdot|$ for cardinality when the argument is a set. Note that each added terminal introduces four more new parameters $\{\pi_\tau, \mu_{\tau 0}, \mu_{\tau 1}, \sigma_\tau\}$. Thus the total number of parameters in tree $\mathcal{T}$ is $4 \cdot |\widetilde{\mathcal{T}}|$. A tree with a smaller AIC is preferable. Alternatively, the Bayesian information criterion (BIC; Schwarz 1978) with $\lambda = \ln(n)$ is another choice in common use. At each step, the algorithm examines all available internal nodes or links in the present tree and truncates the link that results in the subtree with the smallest AIC. The pruning procedure yields a nested sequence of subtrees $\mathcal{T}_0 \succ \mathcal{T}_1 \succ \cdots \mathcal{T}_M$, where $\mathcal{T}_M$ is the null tree structure with root node only and "$\succ$" is read as "has subtree".

The final step of tree size selection entails identifying the optimal tree $\mathcal{T}_\star$ from the subtree sequence. The same AIC or BIC measure can be used for this purpose. However, cross validation is needed to validate $\hat{l}_{\mathcal{T}}$ in Equation (10), which can be done via either the test sample method or resampling methods (V-fold cross-validation or bootstrapping), depending on the available sample size. Again, we refer readers to Su, Wang, and Fan (2004) for details.

Remark Using the parametric approach in Section 2.3, an alternative splitting statistic can be obtained by maximizing the between-node difference. To split node τ, let I_s denote the indicator function corresponding to a permissible split s of τ. Consider model

$$\log \frac{\Pr(T=1|\mathbf{x})}{\Pr(T=0|\mathbf{x})} = \beta_0 + \beta_1 I_s \text{ and}$$
$$y = \gamma_0 + \gamma_1 T + \gamma_2 I_s + \gamma_3 T \cdot I_s + \sigma\varepsilon \text{ with } \varepsilon \sim \mathcal{N}(0,1). \tag{11}$$

In view of Proposition 4, the Wald test statistic for testing $H_0 : \beta_1 = \gamma_3 = 0$ can be used as the splitting statistic. Since the log-likelihood function is separable for β and γ as shown in (7), $\text{cov}(\hat{\beta}, \hat{\gamma}) = \mathbf{0}$. After some algebraic simplification, the Wald test statistic is given by

$$\left(\frac{1}{n_{\tau_L 0}} + \frac{1}{n_{\tau_L 1}} + \frac{1}{n_{\tau_R 0}} + \frac{1}{n_{\tau_R 1}} \right)^{-1} \left[\left(\log \frac{n_{\tau_L 0} n_{\tau_R 1}}{n_{\tau_L 1} n_{\tau_R 0}} \right)^2 + \frac{\{(y_{\tau_L 1} - y_{\tau_L 0}) - (y_{\tau_R 1} - y_{\tau_R 0})\}^2}{\hat{\sigma}^2} \right],$$

where $\hat{\sigma}^2 = \left\{ \sum_{i=1}^n y_i^2 - \left(n_{\tau_L 0} y_{\tau_L 0}^2 + n_{\tau_L 1} y_{\tau_L 1}^2 + n_{\tau_R 0} y_{\tau_R 0}^2 + n_{\tau_R 1} y_{\tau_R 1}^2 \right) \right\} / n$ is the MLE of σ^2 in model (11).

3.2 Aggregated Grouping

Despite easy interpretability, one single tree model is notoriously unstable in the sense that a minor perturbation of the data could result in substantial changes in the final tree structure. In order to get around this problem, we propose an aggregated grouping method to integrate the stratification results from a number of competitive tree models. The key idea of this method is to obtain an $n \times n$

distance or dissimilarity matrix $\mathbf{D}$ with entries that measure how likely each pair of observations is assigned to different strata. Cluster analysis can then be applied for final grouping.

The procedure is described as follows. Let $\mathcal{L}$ denote the whole data set. At each iteration b for $b = 1, \ldots, B$, generate bootstrap sample $\mathcal{L}^{(b)}$ from $\mathcal{L}$. Divide $\mathcal{L}^{(b)}$ into two parts at random with a ratio of 2:1, the learning sample $\mathcal{L}_1^{(b)}$ and the test sample $\mathcal{L}_2^{(b)}$. Using $\mathcal{L}_1^{(b)}$, a large initial CIT $\mathcal{T}_0^{(b)}$ is grown and pruned. With the aid of the test sample $\mathcal{L}_2^{(b)}$, a best-sized tree $\mathcal{T}_\star^{(b)}$ is selected. Let $K_b = |\widetilde{\mathcal{T}}_\star^{(b)}|$ be the number of terminal nodes in $\mathcal{T}_\star^{(b)}$. Then we apply $\mathcal{T}_\star^{(b)}$ to the whole data $\mathcal{L}$ so that each observation in $\mathcal{L}$ falls into one and only one terminal node of $\mathcal{T}_\star^{(b)}$. Next, we define an $n \times n$ pairwise distance matrix $\mathbf{D}_b = \{d_{ii'}\}$ such that

$$d_{ii'} = \begin{cases} 0 & \text{if observations } \{i, i'\} \text{ fall into the same terminal node of } \mathcal{T}_\star^{(b)}; \\ 1 & \text{otherwise,} \end{cases}$$

for $i, i' = 1, \ldots n$. To compute $\mathbf{D}_b$, first obtain an $n \times K_b$ matrix $\mathbf{Z}_b = (z_{ik})$ such that

$$z_{ik} = \begin{cases} 1 & \text{if observation } i \text{ falls into the } k\text{-th terminal node,} \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

for $i = 1, \ldots, n$ and $k = 1, \ldots, K_b$. It follows that

$$\mathbf{D}_b = \mathbf{Z}_b \mathbf{Z}_b^t. \tag{13}$$

Next, the distance matrices are integrated by averaging over B iterations: $\mathbf{D} = \sum_{b=1}^{B} \mathbf{D}_b / B$. It can be seen that the entries in $\mathbf{D}$ satisfy the triangle inequality and other properties that are required for being a legitimate distance measures. Finally, we can apply a clustering algorithm on $\mathbf{D}$ in order to obtain the final data stratification. The number of clusters K can be either determined by the clustering algorithm itself or preset as the mode of K_b's. Other techniques for exploring distance or proximity matrices can also be applied, such as multidimensional scaling (MDS; Torgerson 1958). The whole procedure is outlined in Algorithm 1.

Algorithm 1 Pseudo-Codes for Aggregated Grouping

Set $B \leftarrow$ number of repetitions.
for $b = 1$ to B **do**
— Generate bootstrap sample $\mathcal{L}^{(b)}$.
— Randomly divide data $\mathcal{L}^{(b)}$ into $\{\mathcal{L}_1^{(b)}, \mathcal{L}_2^{(b)}\}$ with a ratio of 2:1.
— Grow a large CIT $\mathcal{T}_0^{(b)}$ using $\mathcal{L}_1^{(b)}$ and prune.
— Select the best tree $\mathcal{T}_\star^{(b)}$ using $\mathcal{L}_2^{(b)}$. Let $K_b = |\widetilde{\mathcal{T}}_\star^{(b)}|$.
— Apply $\mathcal{T}_\star^{(b)}$ to data $\mathcal{L}$ and compute distance matrix $\mathbf{D}_b = (d_{ii'})$ such that $d_{ii'} = 1$ if observation pair $\{i, i'\}$ falls into different nodes of $\mathcal{T}_\star^{(b)}$ and 0 otherwise.
end for
Obtain $\mathbf{D} \leftarrow 1/B \cdot \sum_{b=1}^{B} \mathbf{D}_b$;
Obtain $K \leftarrow \text{mode}\{K_b : b = 1, \ldots, B\}$.
Apply a clustering algorithm on $\mathbf{D}$ with K clusters.

We also suggest an optional alternative for computing the distance matrix $\mathbf{D}_b$, which is motivated by the amalgamation or node merging idea of Ciampi et al. (1988). It is common that non-neighboring terminal nodes in a final tree structure do not show much differences from each other.

This is because similar patterns in treatment assignment and effect may occur to sub-populations with different characteristics. By taking this issue into consideration of the distance matrix $\mathbf{D}_b$ in Algorithm 1, a more effective way of grouping data may be achieved.

To do so, we first obtain a $K_b \times K_b$ pairwise distance matrix $\mathbf{K}_b = \{\kappa\}$ for all the terminal nodes in $\mathcal{T}_\star^{(b)}$, the best-sized tree obtained in the b-th iteration. Here, $\kappa = \kappa(\tau, \tau') \geq 0$ denotes the distance between two terminal nodes $\tau, \tau' \in \widetilde{\mathcal{T}}_\star^{(b)}$, which can be defined as the logworth (i.e., the negative logarithm with base 10) of the p-value obtained from a likelihood ratio test in (9) that compares τ with τ'. That is,

$$\kappa(\tau, \tau') = -\log_{10}(\text{p-value}).$$

The likelihood ratio test can be conducted using all data in $\mathcal{L}$. The smaller the p-value, the larger the difference between τ and τ' is, as reflected by a larger value of $\kappa(\tau, \tau')$. Elements in matrix $\mathbf{D}_b$ are then defined by

$$d_{ii'} = \kappa(\tau(i), \tau(i')),$$

where $\tau(i)$ denotes the terminal node into which the i-th observation falls. In matrix form, $\mathbf{D}_b$ can be computed as

$$\mathbf{D}_b = \mathbf{Z}_b \mathbf{K}_b \mathbf{Z}_b^t, \tag{14}$$

where $\mathbf{Z}_b$ is given by (12). The $\mathbf{D}_b$ in (13) can be viewed as a special case of (14) with $\mathbf{K}_b = \mathbf{I}_b$.

With modified $\mathbf{D}_b$ in (14), there are two immediate consequences: first, the distances $d_{ii'}$ in $\mathbf{D}$ may not necessarily satisfy the triangle inequality; secondly, the number of final clusters K can no longer be suggested by the best tree sizes. Instead, it has to be determined by the clustering algorithm itself. Recent work on automatic determination of the optimal number of clusters is exemplified by Tibshirani, Walther, and Hastie (2001) and Wang (2010). Both methods are computationally demanding.

Compared to one single CIT, the aggregated grouping produces a more accurate and stable grouping of data. Its results can help evaluate the instability of CIT. However, one drawback is loss of interpretability of the stratification rules.

3.3 Summarizing Strata and ACE Estimation

To summarize the final K strata obtained from either one single CIT or the aggregated grouping method, estimated propensity rate $\hat{e}_k$ and the treatment effect $\widehat{\Delta}_k$ can be obtained for each stratum. Such information helps delineate the heterogeneity structures in both assignment mechanisms and effects of the treatment. Strata with extremely low or high propensities may be excluded from causal inference due to lack of comparison basis. One may take a liberal approach when inspecting differential causal effects across K strata. The use of ACE to summarize treatment effects can be tentatively justified unless strong evidence for qualitative interaction exists. This is similar to the common practice in multi-center trials. While the quantitative treatment-by-center interaction is commonly seen, the overall efficacy of an investigational drug can still be established as long as there is no significant directional change in the comparison. An estimate of ACE, $\widehat{\Delta}$ is given by

$$\widehat{\Delta} = \sum_{k=1}^{K} \frac{n_k}{n} \cdot (y_{k1} - y_{k0}) \tag{15}$$

with sampling variance approximated by

$$\sum_{k=1}^{K} \frac{n_k^2}{n^2} \cdot \left(\frac{s_{k1}^2}{n_{k1}} + \frac{s_{k0}^2}{n_{k0}} \right), \tag{16}$$

where (y_{k1}, s_{k1}^2) are the sample mean and variance of observed Y's in the treated group of the k-th stratum and similar definitions apply to other quantities. Additional covariate adjustment within each terminal node can be made and alternative stratification estimates of ACE are available, as summarized and discussed in Lunceford and Davidian (2004).

Propensity score stratification or subclassification seeks subpopulations of form $\Omega_e = \{\omega : e(\mathbf{X}) = e\}$, in which homogeneity of treatment effects, however, can not be guaranteed. Direct comparison of the mean responses could give a distorted estimate of the causal effect in Ω_e. Comparatively, CIT and aggregating grouping offer refined stratification so that the causal effect within each resultant stratum $\Omega_{\mathbf{a}}$ can be correctly captured, which consequently offers improved estimation of ACE. Alternatively, one may try to correct the problem with propensity score stratification by applying additional ANCOVA-typed adjustment within each stratum. It is important to note that ANCOVA does no help with this correction, unless effect modification is incorporated into the model by allowing for treatment-by-covariate interaction terms. This approach would consist of two steps. In the first step, a number of strata are obtained by stratifying propensity scores. In the second step, an extended ANCOVA model that allows for interactions is fit within each stratum. We may adopt an approach explained by Aiken and West (1991) in order to make the overall causal effect in Ω_e appear as a regression coefficient. This approach fits a linear model of form

$$E(Y_i|T_i, \mathbf{X}_i) = \beta_0 + \delta_e T_i + \beta^t \mathbf{x}_i' + T_i \cdot \gamma^t \mathbf{x}_i'. \tag{17}$$

where $\mathbf{x}_i' = \mathbf{x}_i - E(\mathbf{X}|\Omega_e)$ for $i \in \Omega_e$ denotes the centered covariate vector. Then the parameter δ_e in (17) coincides with the overall causal effect in Ω_e. Finally, the ACE is estimated by combining $\hat{\delta}_e$'s via (15). The CIT stratification roughly resembles this two-step approach described above, yet with additional advantages. First, the facilitating score offers a unified setting where these two steps are naturally combined. Secondly, how to specify interaction terms in (17) remains a dazzling task, which, however, can be efficiently handled with recursive partitioning in CIT.

3.4 Predicting Individual Causal Effects (ICE)

With the advent of research with biobanks, molecular profiling technologies have been greatly advanced to allow for collection of a patient's proteomic, genetic, and metabolic information. Given various information collected on a patient, how to customize treatments to the individual's best needs has posed great challenges to players in the field of personalized medicine, including statisticians. A fine delineation of treatment effects plays a critical role in such endeavors.

For this purpose, we define "*individual causal effect*" (ICE) as a conditional expectation $E(Y_1 - Y_0|\mathbf{x})$, given a subject with $\mathbf{X} = \mathbf{x}$. ICE is conceptually different from the unit level causal effect $Y_1(\omega) - Y_0(w)$. Strictly speaking, ICE makes conditional causal inference at the subpopulation level $\{\omega : \mathbf{X}(\omega) = \mathbf{x}\}$. On the other hand, ICE is the best that one could practically do with available information in order to approximate the unit level causal effect. Especially when $\mathbf{X}$ is high-dimensional and has many continuous components, it is likely that each value $\mathbf{x}$ corresponds uniquely to unit ω with $\mathbf{X}(\omega) = \mathbf{x}$. In what follows, we devise a powerful method via ensemble CITs to predict ICE by borrowing ideas from random forests (Breiman, 2001).

To proceed, we first randomly divide data $\mathcal{L}$ into V folds. To ensure similar proportions of individuals in the treatment groups across all folds, stratified sampling with stratification on T can be used. Let $\mathcal{L}_v$ denote the v-th fold and $\mathcal{L}_{(v)} = \mathcal{L} - \mathcal{L}_v$ for the remaining data.

Algorithm 2 Pseudo-Codes for Predicting Personal Causal Effects (ICE)

Set V, B, and m.
Randomly split data $\mathcal{L}$ into V sets $\{\mathcal{L}_1, \ldots, \mathcal{L}_V\}$, with stratification on T.
for $v = 1$ to V **do**
  Set $\mathcal{L}_{(v)} = \mathcal{L} - \mathcal{L}_v$.
  for $b = 1$ to B **do**
    — Generate bootstrap sample $\mathcal{L}_{(v)}^{(b)}$ from $\mathcal{L}_{(v)}$.
    — Grow a CIT $\mathcal{T}_{(v)}^{(b)}$ using $\mathcal{L}_{(v)}^{(b)}$ without pruning. At each split, only m randomly selected variables are used.
    — Estimate causal effects $\hat{\Delta}_\tau$ and propensity $\hat{e}_\tau$ for each $\tau \in \widetilde{\mathcal{T}}_{(v)}^{(b)}$ based on $\mathcal{L}_{(v)}$.
    — Apply $\mathcal{T}_{(v)}^{(b)}$ to data $\mathcal{L}_v$.
    — Compute $\hat{\Delta}_i^{(b)}$ and $\hat{e}_i^{(b)}$ for each $i \in \mathcal{L}_v \cap \tau$, via $\hat{\Delta}_\tau$ and $\hat{e}_\tau$.
  end for
  Obtain $\{\hat{\Delta}_i, \hat{e}_i\}$ as averages of $\{(\hat{\Delta}_i^{(b)}, \hat{e}_i^{(b)}) : b = 1, \ldots, B\}$, for each $i \in \mathcal{L}_v$.
end for
Merge estimated $\{\hat{\Delta}_i, \hat{e}_i\}$ into data $\mathcal{L}$ using ID key.
return $\mathcal{L}$.

We draw B bootstrap samples from $\mathcal{L}_{(v)}$. With each bootstrap sample $\mathcal{L}_{(v)}^{(b)}$, grow a moderately-sized CIT $\mathcal{T}_{(v)}^{(b)}$ without pruning. When constructing $\mathcal{T}_{(v)}^{(b)}$, we adapt the approach in random forests (Breiman, 2001), where only m randomly selected variables and their associated cutoff points are evaluated at each split. This tactic helps improve the predictive performance by de-correlating the tree models in the random forests. For each terminal node $\tau \in \widetilde{\mathcal{T}}_{(v)}^{(b)}$, estimates of the causal effect and propensity,

$$\hat{\Delta}_\tau = y_{\tau 1} - y_{\tau 0} \quad \text{and} \quad \hat{e}_\tau = n_{\tau 1}/n_\tau,$$

are computed using data in $\mathcal{L}_{(v)}$. Then we apply $\widetilde{\mathcal{T}}_{(v)}^{(b)}$ to $\mathcal{L}_{(v)}$ and predict the ICE $\hat{\Delta}_i^{(b)}$ and propensity score $\hat{e}_i^{(b)}$ for each individual $i \in \mathcal{L}_v$. Specifically,

$$\hat{\Delta}_i^{(b)} = \hat{\Delta}_\tau \quad \text{and} \quad \hat{e}_i^{(b)} = \hat{e}_\tau,$$

if the i-th individual falls into the terminal node τ. The final predicted ICE and propensity for the i individual are

$$\hat{\Delta}_i = \frac{1}{B}\sum_{b=1}^{B} \hat{\Delta}_i^{(b)} \quad \text{and} \quad \hat{e}_i = \frac{1}{B}\sum_{b=1}^{B} \hat{e}_i^{(b)}.$$

Their standard errors can also be obtained from the bootstrap repetitions.

The same procedure is repeated for each fold to estimate ICE and propensity scores for all individuals in $\mathcal{L}$. The whole method is described in Algorithm 2. Further exploration can be done with the estimated ICE and propensity scores and some illustrations are given in Section 5. While

we have used a V-fold cross-validation approach in Algorithm 2, the method can be directly applied to an independent future sample for predicting ICE. Other features in random forests such as variable importance ranking and partial dependence plots could also be adopted for causal inference.

Some alternative ways of predicting ICE are discussed below. First of all, the standard method for modeling treatment-by-covariates interaction in many application fields is to fit a linear model with first-order cross-product terms, that is,

$$\begin{aligned} y &= \beta_0 + \beta_1 T + \beta_2^t \mathbf{x} + T \cdot \beta_3^t \mathbf{x} + \varepsilon \\ &= \beta_0 + \beta_2^t \mathbf{x} + (\beta_1 + \beta_3^t \mathbf{x}) \cdot T + \varepsilon. \end{aligned} \tag{18}$$

The ICE is formulated as $(\beta_1 + \beta_3^t \mathbf{x})$, which is also linear in $\mathbf{x}$. While this parametric approach is readily available, it relies heavily on linearity and is subject to a greater risk of model misspecification.

Another convenient way for predicting ICE is to use the 'regression estimation' approach, as described in Schafer and Kang (2008). In this approach, we separately fit a predictive model (possibly using machine learning techniques) for Y_1 using data in the treated group only and a predictive model for Y_0 using data in the untreated group only. Then we apply these models to obtain predicted values $(\hat{y}_{i1}, \hat{y}_{i0})$ for the potential outcomes for every subject in the data. ICE can be estimated as $\hat{\Delta}_i = \hat{y}_{i1} - \hat{y}_{i0}$. Alternatively, the observed response can be used in the calculation so that $\hat{\Delta}_i = y_i - \hat{y}_{i0}$ for the treated group and $\hat{\Delta}_i = \hat{y}_{i1} - y_i$ for the untreated group. Note that this regression estimation method solely involves the outcome models. The underlying rationale is based on the fact that $E(Y_t|\mathbf{X} = \mathbf{x}) = E(Y|\mathbf{X} = \mathbf{x}, T = t)$ for $t = 0, 1$, given strong ignorability and other conditions. However, the prediction across treatment groups heavily involves extrapolation, again due to the imbalance in covariates. When used for ACE estimation, Schafer and Kang (2008) found that it is not among the top performers, but may be possibly improved by incorporating the propensity score into the model.

Estimating ICE also emerges as one intermediate step in some ACE inference procedures including structural nested models introduced by Robins (1989), marginal structural models (see, e.g., Robins 1999), and the targeted maximum likelihood method (see, e.g., van der Laan and Rubin 2006). These procedures are particularly advantageous in dealing with longitudinal observational data where both treatment and covariates are time-varying, but they are also applicable to cross-sectional or 'point treatment' data. Two estimation methods are commonly used in these procedures: the g-computation and the inverse probability of treatment weighting (IPTW). Model (18) is often embedded in either method, for handling effect moderators in IPTW or being used as the Q-model in g-computation (see, e.g., Snowden, Rose, and Mortimer 2011) or targeted maximum likelihood (see, e.g., Rosenblum and van der Laan 2011) to model and predict potential outcomes. With g-computation, it is clear that other semiparametric or nonparametric data adaptive methods (as in 'the 'regression estimation' approach) can be flexibly used for predicting potential outcomes for each observation under each possible treatment regimen. See van der Laan, Polley, and Hubbard (2007) and Austin (2012) for examples.

Yet another method for estimate ICE $E(Y_1 - Y_0|\mathbf{X} = \mathbf{x})$ directly is to relax $\mathbf{X} = \mathbf{x}$ to a neighborhood of $\mathbf{x}$, $\mathcal{N}(\mathbf{x})$. Such a neighborhood of $\mathbf{x}$ can be facilitated using either K-nearest neighbor (KNN) or, more generally, kernel smoothing. If KNN is used, let $\mathcal{N}_K(\mathbf{x})$ denote the corresponding neighborhood of $\mathbf{x}$. A natural estimate of ICE is given by

$$\frac{\sum_{i:\ \mathbf{x}_i \in \mathcal{N}_K(\mathbf{x})} y_i T_i}{\sum_{i:\ \mathbf{x}_i \in \mathcal{N}_K(\mathbf{x})} T_i} - \frac{\sum_{i:\ \mathbf{x}_i \in \mathcal{N}_K(\mathbf{x})} y_i (1 - T_i)}{\sum_{i:\ \mathbf{x}_i \in \mathcal{N}_K(\mathbf{x})} (1 - T_i)}.$$

This KNN approach assigns weight 1 to K observations within $\mathcal{N}_K(\mathbf{x})$ and weight 0 to others. More generally, we may use kernel smoothing to have weights depending on $\| \mathbf{x}_i - \mathbf{x} \|$ for all data points. To make it more robust to non-random treatment assignment mechanism, it might be possible to incorporate propensity score into the weights as well. While this implementation has not been seen in the literature, it has some promising potentials for its nonparametric nature and deserves further research. On the other hand, a neighborhood defined with high-dimensional data could have poor performance and the computation could be demanding. In addition, interpretation with respect to covariates becomes obscure with nearest neighbor approaches.

Comparatively, the essential ingredient in our ensemble CIT approach is stratified causal estimates within subpopuations $\{\mathbf{x} : \mathbf{a}(\mathbf{x}) = \mathbf{a}\}$, which is intermediary in-between ACE and ICE. We have the convenience to either move forward for ICE with ensemble models or move backward for ACE by integrating stratified results. It is natural to use tree methods for extracting strata. Tree-structured methods are nonparametric in nature and hence more robust to model misspecification. Recursive partitioning excels in efficiently handling interactions and categorical variables and provides meaningful interpretations. Besides, ensemble models usually performs better in predictive modeling. With that being said, a comprehensive comparison study of these alternative approaches in predicting ICE would be interesting for future research.

3.5 Consistency

In terms of asymptotic properties of recursive partitioning based estimators, Breiman et al. (1984) provided detailed developments of convergence in rth mean and uniform convergence on compacts. Gordon and Olshen (1984) established the almost sure convergence under certain constraints. In this section, consistency of the CIT based causal effect estimator is provided in the light of Breiman et al. (1984).

Let the predictor space $\mathbb{X} \in \mathbb{R}^p$ be Euclidean. A tree structure $\mathcal{T}$ partitions $\mathbb{X}$ into a number of disjoint sets or terminal nodes $\{\tau : \tau \in \widetilde{\mathcal{T}}\}$. Again, $\tau(\mathbf{x})$ denotes the terminal node where $\mathbf{x}$ falls into. Let $d(\cdot)$ be the diameter of a set, that is, $d_n(\tau) = \sup_{\mathbf{x},\mathbf{x}' \in \tau} \| \mathbf{x} - \mathbf{x}' \|$, where $\| \cdot \|$ is the Euclidean norm. With the observed data of size n, let k_n be nonnegative constants such that, with probability one,

$$n_\tau \geq k_n \log n \quad \text{for any } \tau \in \widetilde{\mathcal{T}},$$

where, same as before, $n_{\tau 1}$ is used to denote the number of subjects in node τ that are assigned to the treated group, that is, $n_{\tau 1} = \sum_{i \in \tau} T_i$, and $n_{\tau 0}$ for the control group. Suppose that $\mathbf{a}(\cdot)$ is a weak facilitating score and $\mathbf{a}_\tau = \sum_{i \in \tau} \mathbf{a}(\mathbf{x}_i)$ denotes its mean vector in node τ. Let $(Y_1, Y_0, Y, T, \mathbf{x}) \in \tau$ represent a new observation that is independent of current data $\{(y_i, T_i, \mathbf{x}_i) : i = 1, \ldots, n\}$. The following theorem establishes the mean square risk consistency for $(y_{\tau 1} - y_{\tau 0})$, the causal effect estimate based on direct comparison of sample means in the terminal node $\tau = \tau(\mathbf{x})$.

Theorem 5 *Suppose that*

$$\max \left\{ E|Y_1|^{2+\varepsilon},\ E|Y_0|^{2+\varepsilon} \right\} \leq M < \infty \text{ for some } \varepsilon > 0 \text{ and } M > 0, \tag{19}$$

$0 < e(\mathbf{x}) < 1$, *and treatment assignment is strongly ignorable. Assume that* $E(Y_1|\mathbf{a})$ *and* $E(Y_0|\mathbf{a})$ *are continuous in* $\mathbf{a}$ *and* $\mathbf{a}(\mathbf{x})$ *is continuous in* $\mathbf{x}$. *Further assume that*

$$\lim_{n \to \infty} k_n = \infty. \tag{20}$$

and

$$\lim_{n\to\infty} d_n(\tau) = 0 \tag{21}$$

in probability. Then

$$\lim_{n\to\infty} E\,|(y_{\tau 1} - y_{\tau 0}) - E\{Y_1 - Y_0 \,|\, \mathbf{a}(\mathbf{x}) = \mathbf{a}_\tau\}|^2 = 0. \tag{22}$$

The results in Theorem 5 can be improved to L_r convergence for any $r \geq 1$ if we change the assumption (19) to

$$E|Y_1|^{r+\varepsilon} \leq M < \infty \text{ and } E|Y_0|^{r+\varepsilon} \leq M < \infty.$$

This can be immediately seen from the proof provided in Appendix C, where all the arguments we have used hold in L_r spaces. Toth and Eltinge (2010) has recently proved asymptotic design L_2 consistency of tree-based estimator when applied to complex survey data, following similar arguments in Gordon and Olshen (1978, 1980). It is worth noting that the Horvitz-Thompson (1952) typed estimator via inverse probability weighting has fundamental use in both causal inference with observational data and in estimation the superpopulation mean with stratified survey data.

These convergence results for recursive partitioning are obtained without dependence on the specifics of the algorithm. Unfortunately, no theoretical justifications have been obtained so far for the splitting rules and pruning algorithms (p. 327; Breiman et al. 1984). Moreover, one of key assumptions for consistency requires that the mesh size of τ goes to 0 when the sample size gets large, as implied by assumption (21). This is an unappealing constrain to practical applications.

4. Simulated Experiments

In this section, simulation experiments are performed to first understand and assess CIT and make comparisons with other methods and then investigate how CIT performs under misspecification.

4.1 Performance of CIT

In terms of applications of tree methods relevant to treatment effect assessment, there have been two major developments serving different purposes: 1) propensity trees (PT) that estimate the propensity score $e(\mathbf{X})$, as studied by McCaffrey, Ridgeway, and Morral (2004) and Lee, Lessler, and Stuart (2010); and 2) interaction trees (IT) for subgroup analysis (Su et al., 2009). An interaction tree explicitly models the treatment-by-covariates interactions for detecting differential treatment effects. However, this method was developed for experimental data and does not take the non-randomized treatment assignment into account. As we shall demonstrate, failure or inadequacy to account for propensity information may lead to misleading interaction results, in that the superficial difference in treatment effects might have been caused merely by heterogenous treatment selection mechanisms.

We generate data with the following steps.

1. Generate $X_1, \ldots X_5$ independently from Unif(0,1) and create threshold variables $Z_j = 1_{\{X_j \leq 0.5\}}$ for $j = 1, \ldots, 5$.

2. Set $\text{logit}(\pi) = a_0 + a_1 Z_1 + a_2 Z_2$ with $\text{logit}(\pi) = \log\{\pi/(1-\pi)\}$. Generate $T \sim \text{Bernoulli}(\pi)$.

3. Set $\mu = b_0 + b_1 T + b_2 Z_2 + b_3 Z_3 + b_4 Z_4 + b_5 T \cdot Z_4$ and generate $Y \sim \mathcal{N}(\mu, \sigma^2)$ with $\sigma = 1$.

In addition to the response variable Y and treatment indicator T, each data set involves five covariates. In the above simulation strategy, covariate X_1 is an exposure or treatment predictor involved in the propensity model only, X_2 is a confounder that relates to both T and Y, X_3 is a response predictor or prognostic factor, X_4 is an effect-modifier, and X_5 is a totally irrelevant covariate. All the covariate values are rounded at the second decimal place.

By applying different values for the coefficients $a_i,\ i = 0,1,2$, and $b_j,\ j = 0,\ldots 5$, we can obtain different model configurations, for example, containing either interaction or confounding terms, both, or neither. We can also investigate how these tree methods handle covariates that play different types of roles in the causal pathway between T and Y. Specifically we consider the following five model configurations:

$$\begin{aligned}
&\text{Model A.} \quad \mathbf{a} = \{a_j\} = (2,0,0)', \quad \mathbf{b} = \{b_j\} = (2,2,0,0,0,0)';\\
&\text{Model B.} \quad \mathbf{a} = \{a_j\} = (2,2,-4)', \quad \mathbf{b} = \{b_j\} = (2,2,2,2,2,2)';\\
&\text{Model C.} \quad \mathbf{a} = \{a_j\} = (2,0,-4)', \quad \mathbf{b} = \{b_j\} = (2,2,2,0,2,2)';\\
&\text{Model D.} \quad \mathbf{a} = \{a_j\} = (2,2,-4)', \quad \mathbf{b} = \{b_j\} = (2,2,2,0,0,0)';\\
&\text{Model E.} \quad \mathbf{a} = \{a_j\} = (2,2,-4)', \quad \mathbf{b} = \{b_j\} = (2,2,0,0,2,2)'.
\end{aligned}$$

Model A is a null model, where the covariates have no influence on the treatment effect. This model helps investigate the size issue or the type I error rate. Model B is equipped with all structures. Nevertheless, a massive tree with 16 terminal nodes is needed in order to fully represent the model structure. Model C also contains both confounding effect of X_2 and interacting effect of X_4, while neither X_1 nor X_3 is involved. In this case, a tree with four terminal nodes is expected. Model D mainly involves the confounder X_2, plus the exposure predictor X_1. Lastly, the active components in Model E are the effect modifier X_4 and the prognostic factor X_3.

For each simulated data set, all three tree methods, CIT, IT, and PT, are applied. Only one sample size is reported and the test sample method is used to select the optimal tree size, with 600 observations for the training sample and 400 observations for the test sample. Both AIC and BIC are used for the tree model selection. For each final tree selected, we record the optimal tree size and the splitting variables involved in the final tree structure. Table 1 presents the summarized results over 200 simulation runs.

We first examine the results from the null Model A. When BIC is used, all three tree methods seem rather conservative in committing Type I errors, implying that unsolicited signals are unlikely to be identified. With AIC, the empirical size, that is, the rate of giving false tree signals, is $(100-90.5)\% = 9.5\%$ for CIT, $(100-88.5)\% = 11.4\%$ for IT, and $(100-98.5)\% = 1.5\%$ for PT.

Next, Model B contains all the components that are related to the treatment and the response. Experimenting with this model provides an overall picture of what patterns each tree method tends to recognize. It can be seen that CIT yields the largest tree models by mostly catching the effects of X_2, X_3, and X_4. The treatment predictor X_1 is completely missed out by BIC and occasionally (32% of the time) selected by AIC. Note that X_1 is neither a confounder nor a modifier to the treatment effect. Due to the smaller penalty for mode complexity, AIC tends to select larger trees than BIC. As expected, the final propensity trees are split by both X_1 and X_2. The average final tree size of IT is 2.92, compared to its expected value 2. It is interesting to note that IT frequently gets confused by the confounding effect of X_2.

Model C contains only the components that actively influence the causal effects, namely, the confounder X_2 and the effect-modifier X_4. Both are perfectly identified by CIT. PT performs well

		Selection	Final Tree Size							Splitting Variables				
Model	Method	Criterion	1	2	3	4	5	6	≥ 7	X_1	X_2	X_3	X_4	X_5
A	CIT	AIC	90.5	5.5	1.5	1.5	1.0	0.0	0.0	4.5	2.0	2.5	2.0	2.0
		BIC	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	IT	AIC	88.5	6.0	4.5	1.0	0.0	0.0	0.0	2.5	4.0	3.5	3.0	3.0
		BIC	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	PT	AIC	98.5	0.5	1.0	0.0	0.0	0.0	0.0	0.5	0.5	0.5	1.0	0.0
		BIC	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B	CIT	AIC	0.0	4.0	0.0	19.5	5.5	39.5	31.5	32.0	96.0	96.0	100.0	1.5
		BIC	0.0	4.0	0.0	26.0	0.0	61.5	8.5	0.0	96.0	96.0	100.0	0.0
	IT	AIC	0.0	6.5	25.0	50.5	13.0	3.0	2.0	7.0	93.5	6.0	100.0	8.0
		BIC	0.0	40.5	28.5	27.0	3.5	0.5	0.0	1.0	59.5	0.5	100.0	1.0
	PT	AIC	0.0	0.5	33.5	61.5	4.0	0.5	0.0	99.5	100.0	0.5	2.0	1.5
		BIC	0.0	1.0	51.5	46.5	1.0	0.0	0.0	99.0	100.0	0.0	0.5	0.0
C	CIT	AIC	0.0	0.0	4.5	90.5	5.0	0.0	0.0	1.0	100.0	2.0	100.0	0.5
		BIC	0.0	0.0	4.5	95.5	0.0	0.0	0.0	0.0	100.0	0.0	100.0	0.0
	IT	AIC	0.0	47.0	35.5	14.0	1.5	1.5	0.5	0.5	52.5	0.5	100.0	2.5
		BIC	0.0	55.5	33.0	100.0	1.0	0.5	0.0	0.0	44.5	0.0	100.0	0.0
	PT	AIC	0.0	97.5	1.0	1.5	0.0	0.0	0.0	1.0	100.0	1.0	0.5	0.5
		BIC	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
D	CIT	AIC	0.0	1.0	83.5	11.0	2.0	1.5	1.0	99.0	100.0	2.0	2.5	4.5
		BIC	0.0	10.5	89.5	0.0	0.0	0.0	0.0	89.5	100.0	0.0	0.0	0.0
	IT	AIC	1.5	43.0	31.5	18.0	4.5	1.0	0.5	5.5	98.5	3.0	4.5	2.0
		BIC	0.2	54.5	28.0	14.0	1.5	0.0	0.0	1.0	98.0	0.5	1.5	0.0
	PT	AIC	0.0	0.0	33.5	62.0	2.5	1.5	0.5	100.0	100.0	2.0	2.5	1.0
		BIC	0.0	1.0	49.5	49.0	0.5	0.0	0.0	99.0	100.0	0.0	0.0	0.0
E	CIT	AIC	0.0	0.0	2.5	89.0	8.5	0.0	0.0	1.5	3.0	100.0	100.0	2.0
		BIC	0.0	0.0	2.5	97.5	0.0	0.0	0.0	0.0	0.0	100.0	100.0	0.0
	IT	AIC	0.0	90.0	9.0	1.0	0.0	0.0	0.0	2.5	1.5	2.5	100.0	3.5
		BIC	1.5	97.5	1.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	98.5	0.5
	PT	AIC	96.5	3.5	0.0	0.0	0.0	0.0	0.0	1.5	0.5	1.0	0.5	0.0
		BIC	98.5	1.5	0.0	0.0	0.0	0.0	0.0	1.0	0.5	0.0	0.0	0.0

Table 1: Simulation Results Based on the Test Sample Method: Relative frequencies (in percentages) of the final tree sizes in 200 runs identified by the causal inference tree (CIT), interaction tree (IT), and propensity tree (PT). Only one set of sample sizes is reported, with 600 observations forming the learning sample and 400 observations for the test sample.

in identifying the confounder X_2 while IT succeeds in recognizing the effect-modifier X_4. The same interesting phenomenon as with Model B occurs again: IT wrongly selects X_2 quite often. This will further be elaborated in Model D.

Model D is basically a propensity model, involving both the exposure predictor X_1 and the confounder X_2 only. In this case, CIT and PT provide equivalent results. Aiming at differential treatment effects, IT is supposed to have a null tree structure. However, we can see that most of time IT ends up with one or more splits on X_2. To gain insight, Figure 1 plots the splitting statistic used in both IT and CIT versus each cutoff point for X_2 in a single split of the data. The splitting statistic used in IT is a squared t test statistic for interaction; thus the best cutoff point corresponds

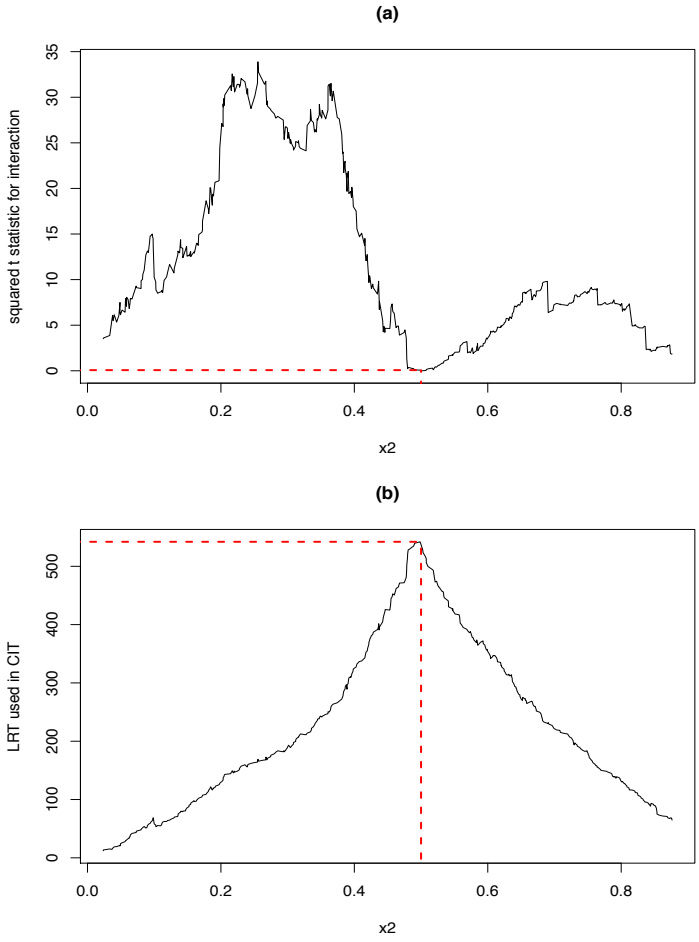


Figure 1: Plot of splitting statistic versus cutoff point on the confounder X_2: (a) t test statistic (squared) for interaction used in IT; (b) likelihood ratio test statistic (up to some constant) used in CIT. Data were generated from Model D in Section 4.

to the maximum of splitting statistics. It is interesting to note in Figure 1(a) that the splitting statistic actually reaches its minimum at $X_2 = 0.5$, the only place where the treatment comparison is unbiased. At other cutoff points, the splitting statistic as a measure of interaction misleadingly inflates due to lack of adjustment for propensity. On the contrary, this does not cause a problem for CIT, which correctly selects the right cutoff point 0.5 as shown in Figure 1(b). Therefore, in order to identify differential causal effects correctly, it is critical to take confounders into consideration; otherwise, the estimation bias owing to imbalance of confounders between treatment groups may become overwhelming and eventually lead to misleading conclusions about the differential causal effects.

Finally, Model E is essentially an outcome regression model, in which both the prognostic factor X_3 and the effect-modifier X_4 are involved. It can be seen that CIT functions similarly to IT in

	Effect	Propensity	Case I		Case II		Case III		Case IV	
Group	Δ_k	e_k	$\widehat{\Delta}_k$	$\hat{e}_k$	$\widehat{\Delta}_k$	$\hat{e}_k$	$\widehat{\Delta}_k$	$\hat{e}_k$	$\widehat{\Delta}_k$	$\hat{e}_k$
1	-1.940	0.156	-2.077	0.189	-3.246	0.502	-2.005	0.171	-2.106	0.201
2	2.067	0.866	1.924	0.861	1.925	0.861	-0.098	0.857	1.916	0.860
3	-1.938	0.843	-1.987	0.829	-2.629	0.676	-1.016	0.840	-2.006	0.824

Table 2: Simulation Results for Assessing Sensitivity of CIT to Misspecification. Four scenarios are considered. In Case I, variables $\{X_1, X_2, X_3, X_4\}$ are used; In Case II, the confounder X_2 is omitted; In Case III, the effect-modifier X_3 is omitted; In Case IV, the collider X_5 is included. The estimated treatment effect and propensity for each group were averaged over 100 runs.

detecting treatment-by-covariate interactions. CIT also identifies splits on the prognostic factor X_3. It comes as no surprise that PT, concerning propensity only, gives a null tree for most of the time.

4.2 Sensitivity under Misspecification

We next investigate how CIT performs under misspecified scenarios where an important confounder or effect-modifier is omitted or when a collider is included. We design an experiment with the following data generation scheme:

1. Generate $X_1, \ldots, X_4$ independently from Unif(0,1) and create threshold variables $Z_j = 1_{\{X_j \le 0.5\}}$ for $j = 1, \ldots, 4$.

2. Generate W_1 and W_2 independently from Bernoulli(0.5) and hence simulate $X_5 \sim \mathcal{N}(2W_1 + 2W_2, 1)$.

3. Set $\text{logit}(\pi) = 0.5 - Z_1 Z_2 + W_1$. Generate $T \sim \text{Bernoulli}(\pi)$.

4. Set $\mu = 2 + 2Z_1 Z_2 - 2T + 4Z_1 Z_3 T + W_2$ and generate $y \sim \mathcal{N}(\mu, 1)$.

The observed data consist of repetitions of $\{Y, X_1, \ldots, X_4\}$. With the above configuration, X_1 is both a confounder and an effect-moderator; X_2 is a confounder; X_3 is a moderator; X_4 is irrelevant; and X_5 is a collider with the M diagram model (see, e.g., Figure 2(a) in Greenland 2003). The data essentially involve three groups with either different propensities or treatment effects. Observations in Group 1 satisfies $Z_1 Z_2 = 1$; Group 2 is characterized by $(1 - Z_1)Z_3 = 1$; and Group 3 contains the others.

In order to assess sensitivity, an independent validation set with $5,000$ observations is first generated. Based on true grouping, the causal effect and propensity for each group are computed and presented in Table 2. Next, a total of 100 simulation runs are considered. For each simulation run, a training set with 600 observations and a test set with 400 observations are generated, on which basis CITs are constructed using different sets of variables. In Case I, variables $\{X_1, X_2, X_3, X_4\}$ are used; Case II uses $\{X_1, X_3, X_4\}$ with confounder X_2 omitted; In Case III, $\{X_1, X_2, X_4\}$ are used by omitting the moderator X_3; In Case IV, $\{X_1, X_2, X_3, X_4, X_5\}$ are used by including the collider X_5. Each final CIT (based on BIC) is applied to the validation set to compute the individual causal effect $\widehat{\Delta}_i$ and propensity $\hat{e}_i$ for each observation in the validation set. The predicted ICEs and propensities are

aggregated for each group, based on the true grouping. The grouped causal effect and propensity estimates are then averaged over 100 simulation runs. The results are also presented in Table 2. It can be seen that, in Case I, CIT does very well in estimating treatment effects and propensities. In both Case II and Case III, substantial bias is present in estimating the treatment effects. The results for Case IV suggest that the collider X_5 also introduces bias. However, compared to the bias from omitting a confounder or moderator, the bias from including a collider is much smaller. This is consistent with the conclusions in Greenland (2003).

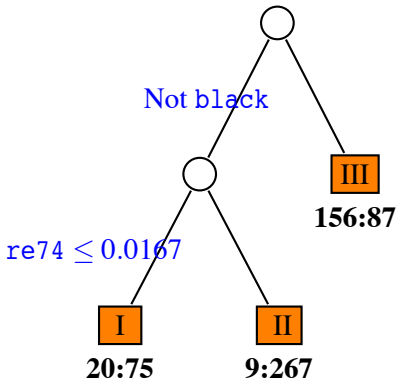
5. Analysis of NSW Data

As an illustration, we revisit the NSW data set extensively analyzed by LaLonde (1986) and Dehejia and Wahba (1999), where the objective is to assess the impact of the National Supported Work (NSW) Demonstration on post-intervention income levels. The NSW demonstration was a labor training program implemented in the mid-1970s to provide work experiences for a period of 6-18 months to individuals facing economic and social difficulties. NSW itself was designed as a randomized controlled study where subjects were randomly assigned to two treatment groups: the NSW-exposed group and the unexposed group.

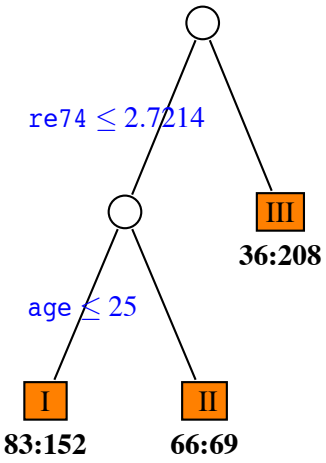
With a rather innovative approach that later on became influential, LaLonde (1986) compiled a composite data set by taking subjects in the NSW-exposed group only and then obtaining the nonexperimental control group from other sources, including the Panel Study of Income Dynamics (PSID) and the Current Population Survey (CPS) databases. His aim was to examine the extent to which nonexperimental estimators can replicate the unbiased experimental estimate of the treatment impact. He concluded that nonexperimental estimators are either inaccurate relative to the experimental benchmark or sensitive to model specification. Since then, the mixed NSW data have been analyzed by various authors with alternative approaches. Among others, Dehejia and Wahba (1999) obtained estimates of the treatment effect that are close to the experimental benchmark estimate or the 'gold' standard using propensity score matching and stratification.

Most of these previous works are focused on estimating the ACE of NSW. Here we shall apply the CIT methods to explore the variabilities of its effects, in addition to dealing with the nonrandom treatment assignments. There are several versions of the data with varying sources for obtaining the control or unexposed group, available from `http://www.nber.org/~rdehejia/nswdata.html`. The data set we use is the one available in the R package `MatchIt` contributed by Ho et al. (2007, 2011). This is a subset restricted to males who had 1974 earnings available, for the reasons explained in Dehejia and Wahba (1999). There are 614 observations (185 treated and 429 control) and 10 variables in the data, which include the treatment assignment indicator. A brief description and some summary statistics of these variables are provided in Table 3. The outcome variable is `re78`, the 1978 earnings. All covariates but `educ` are severely unbalanced between the participants actively exposed to NSW and those in the unexposed group selected from other survey databases.

(a) Propensity Tree



(b) Interaction Tree



(c) Causal Inference Tree

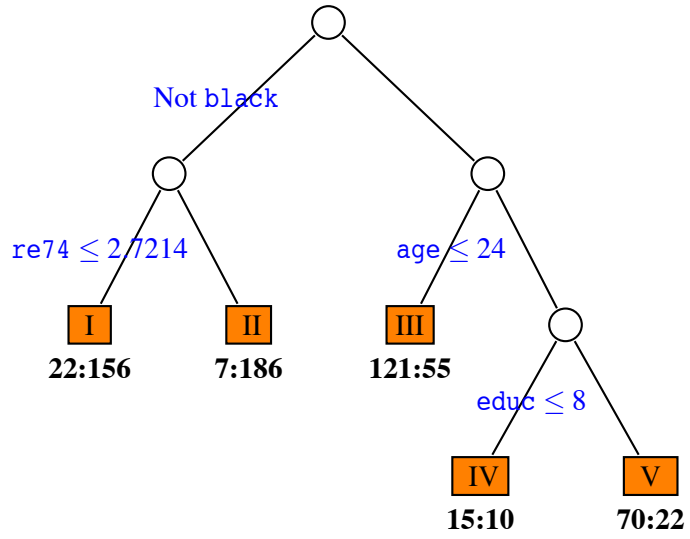


Figure 2: Final Tree Models for the NSW Data: (a) Propensity Tree; (b) Interaction Tree; (c) Causal Inference Tree.

(a) Continuous Variables

Variable		All		NSW Exposed		Unexposed		P-value	
Name	Description	mean	sd	mean	sd	mean	sd	two-sample t	Wilcoxon
`age`	Age in years	27.36	9.88	25.82	7.16	28.03	10.79	0.0107	0.5195
`educ`	Schooling years	10.27	2.63	10.35	2.01	10.24	2.86	0.6330	0.7920
`re74`	1974 earnings	4,557.55	6,477.96	2,095.57	4,886.62	5,619.24	6,788.75	0.0000	0.0000
`re75`	1975 earnings	2,184.94	3,295.68	1,532.06	3,219.25	2,466.48	3,292.00	0.0012	0.0000
`re78`	1978 earnings	6,792.83	7,470.73	6,349.14	7,867.40	6,984.17	7,294.16	0.3342	0.2818

(b) Discrete Variables

Variable		Frequency		P-value	
Name	Description	NSW Exposed	Unexposed	χ^2	Fisher's Exact
`black`	0 - No	29	342	0.0000	0.0000
	1 - African-American	156	87		
`hispan`	0 - No	174	368	0.0053	0.0026
	1 - of Hispanic origin	11	61		
`married`	0 - No	150	209	0.0000	0.0000
	1 - Yes	35	220		
`nodegree`	0 - No	54	173	0.0113	0.0106
	1 - Has a high school degree.	131	256		

Table 3: Variable description and summary statistics for the NSW data set. All earnings are expressed in U.S. dollars.

(a) Propensity Tree

	NSW Group			Unexposed Group			Estimated
Node	size	mean	sd	size	mean	sd	Propensity
I	20	8.1423	6.6646	75	5.2302	6.3981	21.05%
II	9	6.0534	4.9218	267	8.1712	7.6170	3.26%
III	156	6.1363	8.1435	87	4.8534	6.2017	64.20%

(b) Interaction Tree

	NSW Group			Unexposed Group			Treatment Effect	
Node	size	mean	sd	size	mean	sd	estimate	s.e.
I	83	5.0392	5.1160	152	5.2804	5.5401	−0.2412	0.7192
II	66	8.4894	10.3819	69	3.4528	5.8233	5.0366	1.4576
III	36	5.4455	7.0965	208	9.4007	8.0201	−3.9552	1.3070

(c) Causal Inference Tree

	NSW Group			Unexposed Group			Estimated	Treatment Effect	
Node	size	mean	sd	size	mean	sd	Propensity	estimate	s.e.
I	22	8.1431	6.3676	156	4.8438	5.6728	12.35%	3.2993	1.3118
II	7	5.4539	5.3997	186	9.7759	8.0259	3.62%	−4.3221	3.0634
III	71	4.6987	4.8043	55	4.8545	5.9303	56.35%	−0.1558	0.9564
IV	15	3.8662	3.9130	10	1.0999	2.8541	60.00%	2.7663	1.4438
V	70	8.0809	10.7408	22	6.5570	7.3371	76.09%	1.5239	2.4565

Table 4: Summary statistics for the terminal nodes: (a) the final propensity tree (PT); (b) the final interaction tree (IT); and (c) the final causal inference tree (CIT). The means and standard deviations are given in thousand dollars.

We applied three tree procedures to the data: PT, IT, and CIT. The final tree structures, all selected by BIC, are plotted in Figure 2. Considering the moderate sample size, a bootstrap method was used for final tree selection. In Figure 2, the internal nodes are denoted by circles. The splitting rule is given under each internal node. Observations satisfying the rule go to the left child node and observations not satisfying go to the right child node. The terminal nodes are denoted by rectangles and renamed by Roman numerals inside. Underneath each terminal node is the number of exposed subjects versus the number of unexposed subjects within the terminal node. Some summary statistics for the terminal nodes in each final tree are provided in Table 4.

Figure 2(a) gives the final PT structure, which delineates a meaningful heterogeneity in propensity. It is clear that African Americans were more likely to participate in this labor program. PT also identifies a group, terminal node II, with extremely low propensity (3.26%). This group is characterized by people who were not African Americans and had some income in 1974. However, this PT model tells nothing about differential treatment effects.

Figure 2(b) displays the final IT structure. Variables `re74` and `age` stand out as determinants of differential causal effects. Apparently remarkable differential treatment effects seem to exist across the three terminal nodes based on Table 4(b). However, since the method does not adjust

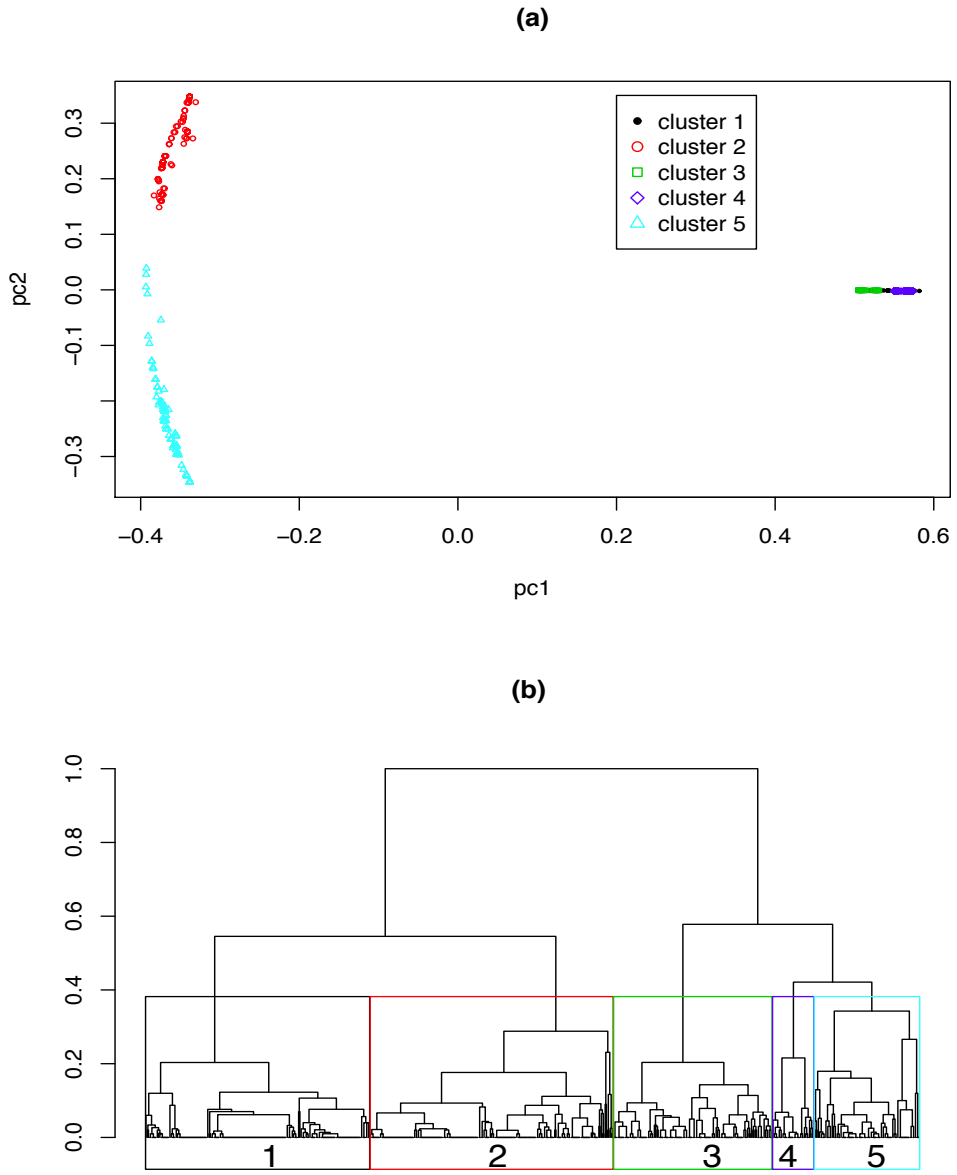


Figure 3: Aggregated Grouping for the NSW Data: (a) Multidimensional scaling (MDS) plot; (b) Dendrogram for hierarchical clustering with average linkage. The distance matrix was computed by aggregating 100 bootstrap samples.

for heterogeneous propensity, the results are not reliable. Hence we make no further attempt in interpreting.

Figure 2(c) presents the final CIT model, which has a more comprehensive structure. It is interesting to see that the left-half of the tree resembles the PT tree in Figure 2(a). In particular, the CIT comes up with a similar terminal node II, which contains non-African Americans with income higher than $2,721 in 1974. Since CIT accounts for both propensity and differential causal effects, it is valid to estimate the NSW effect via direct comparison of sample means within each terminal node. Table 4(c) provides the relevant quantities. CIT also identifies some interesting patterns of differential treatment effects. The surprising comparison occurs to terminal node II, where the NSW-exposed group had a lower average income than the unexposed group with a mean difference of $4,322. However, this should not be a point of great concern due to its very low propensity 3.62%.

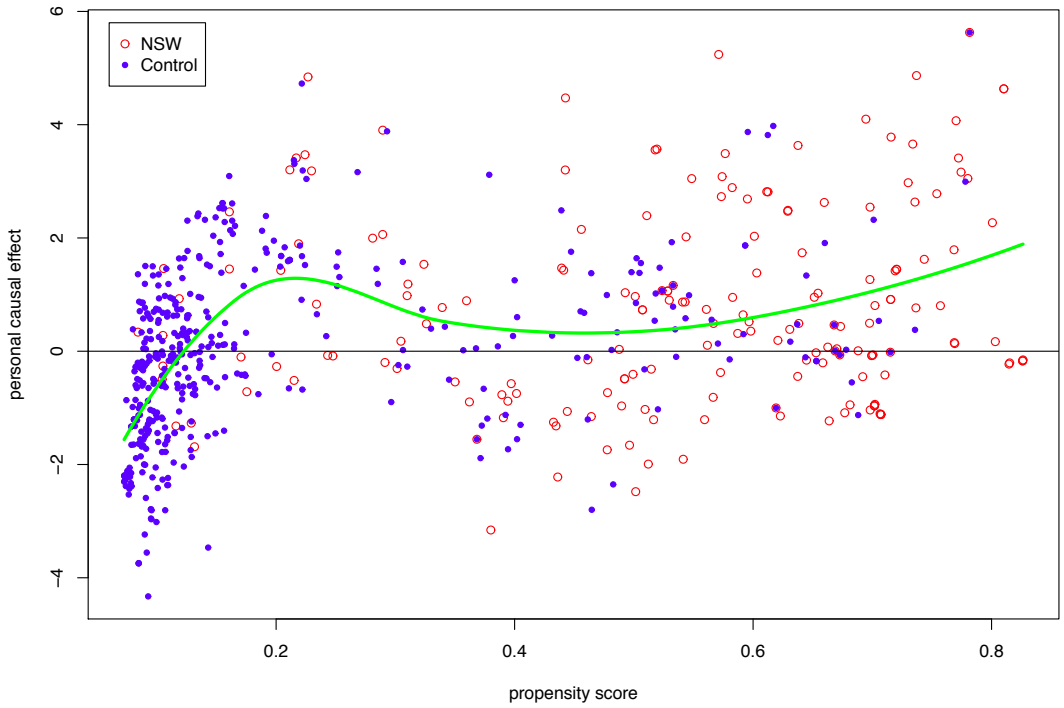


Figure 4: Plot of the Estimated Personal Causal Effects vs. Propensity Scores for the NSW Data. Referring to Algorithm 2, $B = 1000$ bootstrap samples were used in a three-fold cross validation procedure and the parameter m was set as 3.

If it is agreed that terminal node II be excluded from consideration due to lack of comparison basis and the minor negative effect of NSW in terminal node III be ignored, then one may tentatively conclude the absence of qualitative interactions. Using Equations (15)-(16) and information in Table 4(c), the ACE is estimated as $1,845 $\pm$ $809, which is very close to the benchmark randomized

experiment estimate of \$1,794 $\pm$ \$633. As a comparison, the unadjusted estimate is \$635 $\pm$ \$677 and the ANCOVA estimate is \$1,548 $\pm$ \$781 after adjusting for all covariates. It is worth mentioning that the ANCOVA estimate varies dramatically when different sets of variables are included in the model. Using nonparametric matching method (Ho et al., 2007) implemented in the `MatchIt` package, the subclassification estimate (with 5 subclasses) is \$1,237 $\pm$ \$1,196 and the optimal matching (Rosenbaum, 1989) based estimate is \$1,366 $\pm$ \$720.

Next, we applied the aggregated grouping method described in Algorithm 1. We obtained an averaged distance matrix from 100 bootstrap samples. The modal number of optimal tree sizes is 5. The classical MDS (Gower, 1966) was used to explore the the distance matrix. Figure 3(a) provides the resultant plot when the data are represented in a two-dimensional space. Agglomerative hierarchical clustering with average linkage was then used to determine the final clusters. See Figure 3(b) for the dendrogram. The cluster membership specification was also added to the MDS plot in Figure 3(a). It can be seen that Cluster 2 and Cluster 5 are distant from other three clusters. Table 5(a) shows the correspondence between the five clusters and the five CIT terminal nodes. It can be seen that overall they match well, except for minor inconsistency between clusters 4 & 5 and terminal nodes IV & V. This indicates that the CIT structure is relatively stable. The summary statistics for the five clusters are outlined in Table 5(b), showing a pattern similar to Table 4(c). After removing Cluster 2, the estimate of ACE is \$1,897 $\pm$ \$807. We would like to emphasize that the excluded Group II in CIT can be explained by the fact that people who were not black and had some income in 1974 seemed unlikely (with estimated propensity 3.62%) to participate the NSW intervention program. This easy interpretation is no longer available with Cluster 2 obtained from the aggregated grouping procedure.

Finally, ensemble CITs were used to estimate the ICE and propensity score for each individual. Referring to Algorithm 2, three-fold ($V = 3$) cross-validation with $B = 1,000$ bootstrap samples (with stratification on treatment) was used in the analysis; and at each split, $m = 3$ variables were randomly selected as candidate splitting variables. Figure 4 plots the estimated ICE vs. propensity scores. The interpretation for ICE is the difference between what an individual would have earned in 1978 if he had attended NSW, compared to the 1978 earnings if he had not attended. It can be seen that the area with low propensity (below .10) is dominated by subjects in the control and their associated personal effects of NSW are quite mixed. Other than that, the intervention program seem to have an overall positive effect. Figure 5 summarizes the results for each treatment-by-stratum combination, in which the five strata obtained from aggregated grouping are used. It can be seen that both propensity and individual causal effects are reasonably homogeneous within each stratum, even though the individuals were from different treatment groups.

6. Extension to Ordinal/Continuous Treatments

The concept and properties of the facilitating score can be extended to scenarios where the treatment variable is nominal (Lechner, 1999) or ordinal (Imbens, 2000). Suppose that the treatment variable T is allowed to range within $\Im$, where $\Im$ is a discrete set with ordered or unordered values. Let $Y_t = Y_t(\omega)$ denote the potential outcome if unit ω was assigned to the treatment level t. Let $e_t(\mathbf{X}) = \Pr\{T = t|\mathbf{X}\}$ be the generalized propensity score (GPS). A generalized weak facilitating score can be defined as below.

Definition 6 *A generalized weak facilitating score* $\mathbf{a}(\mathbf{X})$ *is a q-dimensional* ($0 < q \leq p$) *function of* $\mathbf{X}$ *such that (i)* $\mathbf{X} \perp\!\!\!\perp T \,|\, \mathbf{a}(\mathbf{X})$ *and (ii)* $E(Y_t - Y_{t'}|\mathbf{a}(\mathbf{X})) = E(Y_t - Y_{t'}|\mathbf{X})$ *for any* $t, t' \in \Im$.

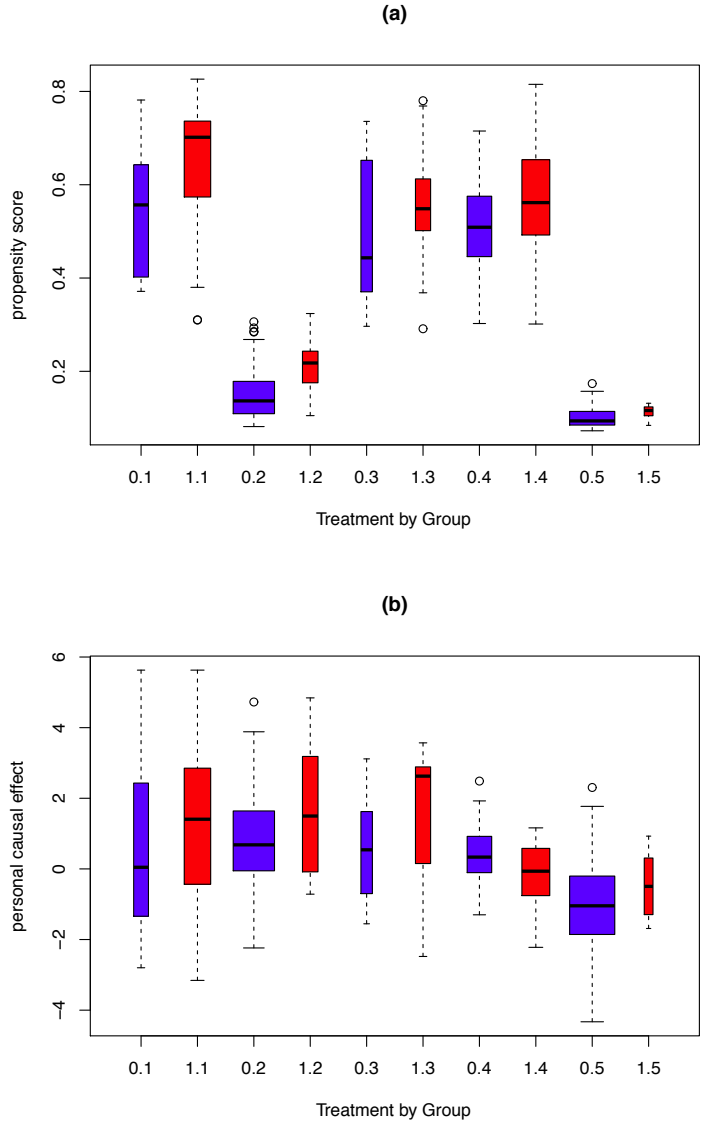


Figure 5: Parallel Box-Plots of (a) the Propensity Scores and (b) the Estimated ICE for Each of the Treatment×Stratum Combinations. The '0.k' combination corresponds to individuals in Stratum k who did not attend the NSW program while '1.k' corresponds to those in Stratum k who did, for $k = 1, \ldots, 5$. The width of each box has been made proportional to the sample size in each combination.

Condition (ii) is equivalent to saying that $E(Y_t - Y_{t'}|\mathbf{a}(\mathbf{X}) = \mathbf{a})$ is independent of $\mathbf{X}$. The following theorem provides a basis for its usage. It shows that, if the joint distribution of (Y, T) can be modeled through a vector-valued function $\mathbf{h}(\mathbf{X})$, then $\mathbf{h}(\mathbf{X})$ is a generalized weak facilitating score and direct estimates of causal effects can be obtained by conditioning on $\mathbf{h}(\mathbf{X}) = \mathbf{h}$.

(a) Correspondence

	Cluster				
	1	2	3	4	5
I	178	0	0	0	0
II	0	193	0	0	0
III	0	0	126	0	0
IV	0	0	0	22	3
V	0	0	0	11	81

(b) Summary of Five Groups

	NSW Group			Unexposed Group			Estimated	Treatment Effect	
Node	size	mean	sd	size	mean	sd	Propensity	estimate	s.e.
1	22	8.1431	6.3676	156	4.8438	5.6728	12.36%	3.2993	1.3118
2	7	5.4539	5.3997	186	9.7759	8.0259	3.63%	−4.3221	3.0634
3	71	4.6987	4.8043	55	4.8545	5.9303	56.35%	−0.1558	0.9564
4	21	4.1514	4.3182	12	2.4027	5.5116	63.64%	1.7487	1.7283
5	64	8.3825	11.0826	20	6.3210	7.1054	76.19%	2.0614	2.6383

Table 5: Results for the five groups obtained from aggregated grouping: (a) correspondence between the obtained groups and the five CIT terminal nodes; (b) summary statistics. The distance matrix was computed from 100 bootstrap samples and hierarchical clustering with average linkage was used for determining the final groups.

Theorem 7 *Assume that the conditional joint density of* (Y,T) *given* $\mathbf{X}$, $f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X})$, *can be written as* $f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X}) = g(Y,T,\mathbf{h}(\mathbf{X}))$ *for some function* $g(\cdot)$. *In other words,* $(Y,T) \perp\!\!\!\perp \mathbf{X}\,|\,\mathbf{h}(\mathbf{X})$. *Further assume that treatment assignment is strongly ignorable so that* $Y_t \perp\!\!\!\perp 1_{\{T=t\}}\,|\,\mathbf{X}$ *for any* $t \in \Im$. *When* $0 < e_t(\mathbf{X}) < 1$, *we have*

(1). $\mathbf{h}(\mathbf{X})$ *is a generalized weak facilitating score.*

(2). Concerning the causal effect in subpopulation $\Omega_{\mathbf{h}} = \{\omega : \mathbf{h}(\mathbf{X}(\omega)) = \mathbf{h}\}$,

$$\begin{aligned} E(Y_t - Y_{t'}|\mathbf{h}(\mathbf{X}) = \mathbf{h}) &= E\{Y_t|T = t, \mathbf{h}(\mathbf{X}) = \mathbf{h}\} - E\{Y_{t'}|T = t', \mathbf{h}(\mathbf{X}) = \mathbf{h}\} \\ &= E\{Y|T = t, \mathbf{h}(\mathbf{X}) = \mathbf{h}\} - E\{Y|T = t', \mathbf{h}(\mathbf{X}) = \mathbf{h}\} \end{aligned}$$

is independent of $\mathbf{X}$.

The proof of Theorem 7 is deferred to Appendix A. As stressed by Lechner (1999) and Imbens (2000), GPS $e_t(\mathbf{X})$ does not have a causal interpretation. However, the reinforced assumption $f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X}) = g(Y,T,\mathbf{h}(\mathbf{X}))$ implies that $e_t(\mathbf{X})$ can be fully characterized by $\mathbf{h}(\mathbf{X})$ or its components. This is analogous to the assumption of *uniquely parameterized propensity function* in Imai and van Dyk (2004), where a parametric form is prescribed for $e_t(\mathbf{X})$. To estimate $\mathbf{h}(\mathbf{X})$, a multinomial or cumulative logit model can be used for propensity and the outcome can be modeled with

multiple linear regression. The above results also can be extended to continuous treatment variables with arguments similar to Hirano and Imbens (2005).

7. Discussion

Embedded in Rubin's causal model, we have introduced a new concept, the facilitating score, to help tackle the heterogeneity in both propensity and causal effects. The facilitating score is a finer balancing score of Rosenbaum and Rubin (1983), plus additional conditions for dealing with differential causal effects. It supplies a framework that promotes joint modeling of (Y,T) for a better understanding of causal effects. Accordingly we have devised recursive partitioning methods to aid in causal inference at different levels.

The facilitating score concept and the CIT methods can be useful in personalized medicine and other similar applications. Medical treatment is traditionally centered on standards of care on the basis of large epidemiological cohort studies or randomized trials that are powered for assessing ACE. These studies however do not account for variabilities of individuals in reacting to the treatments and drug-to-drug interactions. The new medical model of personalized medicine or treatments seeks flexible ways that allow for treatment decisions or practices being tailored to individual by integrating post-trial clinical data and new developments in biotechnology to improve healthcare. The collected covariates are often expanded to a more comprehensive consideration of the patient, including medical measurements, family history, social circumstances, environment and behaviors, and biological variables. As a result, the data are often observational and high-dimensional in nature. As demonstrated in the NSW data example, causal inference in observational studies could be very complex, owing to the confounding and interacting effects complicated by covariates. While personalized medicine is the ultimate goal, stratified medicine has been the current approach. Stratified medicine aims to select the best therapy for groups of patients who share common biological characteristics. The proposed CIT method and aggregated grouping can be used seeking strategies for deploying stratified medicines. Insight into a greater degree of personalized treatment can be gained by studying the personal treatment effects with ensemble CITs.

Some limitations of the proposed methods are listed below. First, despite the usefulness of ICE, assessing ICE entails larger data than assessment of ACE in order to have the same level of precision (or variance). There are many trials in research practice that are only powered to detect ACE. For this reason, the proposed methods are best suitable for moderately-sized or large follow-up data collected in post trial periods or extracted from Medicare or Medicaid databases, in which randomization is not available. Secondly, the recursive partitioning methods are highly adaptive or data-driven in nature and often regarded as exploratory or hypothesis-generating. It is important to interpret the results with caution. In addition, the validity of Theorem 7 relies on the assumption of strong ignorability. Like other methods, CIT performance is vulnerable to violated assumptions and model misspecification. Shpitser and Pearl (2008) examines possibly milder conditions to ensure identifiability and facilitate estimation in causal inference. It would be interesting to investigate how to extend the proposed methods under mild conditions.

In terms of future research, Theorem 7 is readily applied to data with binary outcomes. With further research efforts, both the facilitating score and CIT may be extended to other types of outcomes such as censored survival times or longitudinal measurements. It would also be interesting to extend the proposed methods to scenarios when both treatment and confounders are time-varying, as studied in marginal structural models and structural nested models (Robins, 1999), and when some

confounders are unmeasured but there exist some instrumental variables (IV; Angrist, Imbens, and Rubin 1996) that satisfy the strong ignorability and other conditions. In addition, Robins, Rotnitzky, and Zhao (1994) proposed doubly robust (DB) estimation methods to deal with mis-specification in either the response model or the propensity model. Along similar lines, the targeted maximum likelihood (TML; van der Laan and Rubin 2006) is another newly developed causal inference method that enjoys a favorable theoretical property for being doubly robust and locally efficient, meaning that if at least one of the propensity and outcome models is correctly specified, then the TML estimator is consistent and asymptotically normal; if both models are correctly specified it is also efficient. Similar work with facilitating score modeling could be another avenue for future research.

Acknowledgments

The authors would like to thank the editor and three anonymous referees, whose insightful comments, constructive suggestions, and helpful discussions have greatly improved an earlier version of this manuscript.

Appendix A. Proof of Theorem 7.

We sketch the proof when T is ordinal or nominal. Theorem 3 follows as a special case when $\Im = \{0,1\}$. Some steps are standard arguments in propensity score theories. We include them for the sake of completeness.

First of all, the conditional probability density function of $T|\mathbf{X}$ is

$$f_{T|\mathbf{X}}(T|\mathbf{X}) = \int_Y f_{Y,T|\mathbf{X}}(Y,T|\mathbf{X})dY = \int_Y g(Y,T,\mathbf{h}(\mathbf{X}))dY.$$

Thus the GPS $e_t(\mathbf{X}) = P(T=t|\mathbf{X}) = g_1(\mathbf{h}(\mathbf{X}))$ for some function $g_1(\cdot)$. Namely, $\mathbf{h}(\mathbf{X})$ is a finer function of $e_t(\mathbf{X})$. For this reason, we denote $e_t(\mathbf{X}) = e_t(\mathbf{h}(\mathbf{X}))$.

Next, since $\mathbf{h}(\mathbf{X})$ is measurable with respect to, $\sigma(\mathbf{X})$, the σ-algebra generated by $\mathbf{X}$,

$$\Pr\{T=t|\mathbf{X},\mathbf{h}(\mathbf{X})\} = \Pr\{T=t|\mathbf{X},\} = e_t(\mathbf{X}).$$

Let $\delta_t = I\{T=t\}$ be the indicator function of whether $T=t$. By iterated expectation,

$$\begin{aligned}\Pr\{T=t|\mathbf{h}(\mathbf{X})\} &= E(\delta_t|\mathbf{h}(\mathbf{X})) = E\{E(\delta_t|\mathbf{X},\mathbf{h}(\mathbf{X}))|\mathbf{h}(\mathbf{X})\} \\ &= E\{E(\delta_t|\mathbf{X})|\mathbf{h}(\mathbf{X})\} = E\{e_t(\mathbf{X})|\mathbf{h}(\mathbf{X})\} = e_t(\mathbf{X}).\end{aligned}$$

Namely, $\Pr\{T=t|\mathbf{X},\mathbf{h}(\mathbf{X})\} = e_t(\mathbf{X}) = \Pr\{T=t|\mathbf{h}(\mathbf{X})\}$, which implies $T \perp\!\!\!\perp \mathbf{X}|\mathbf{h}(\mathbf{X})$.

Further assuming the treatment assignment is strongly ignorable given $\mathbf{X}$, it follows that the treatment assignment is ignorable given $\mathbf{h}(\mathbf{X})$, that is, $T \perp\!\!\!\perp Y_t|\mathbf{h}(\mathbf{X})$, which can be established by showing

$$\begin{aligned}\Pr\{T=t'|Y_t,\mathbf{h}(\mathbf{X})\} &= E\{\delta_{t'}|Y_t,\mathbf{h}(\mathbf{X})\} = E\{E(\delta_{t'}|\mathbf{X},Y_t,\mathbf{h}(\mathbf{X}))|Y_t,\mathbf{h}(\mathbf{X})\} \\ &= E\{E(\delta_{t'}|\mathbf{X})|Y_t,\mathbf{h}(\mathbf{X})\} \text{ due to strong ignorability} \\ &= E\{e_{t'}(\mathbf{X})|Y_t,\mathbf{h}(\mathbf{X})\} = e_{t'}(\mathbf{X}) = \Pr\{T=t'|\mathbf{h}(\mathbf{X})\}.\end{aligned}$$

To check condition (ii) in Definition 6, consider $E\{Y_t|\mathbf{h}(\mathbf{X})\}$. Since $Y=\sum_t Y_t\delta_t$ and $\delta_t\delta_{t'}=0$ for $t\neq t'$, we have $Y\delta_t=Y_t\delta_t$. Consider

$$\begin{aligned} E\{Y_t|\mathbf{h}(\mathbf{X})\} &= E\{Y_t|\mathbf{h}(\mathbf{X})\}\cdot E\{\delta_t|\mathbf{h}(\mathbf{X})\}/E\{\delta_t|\mathbf{h}(\mathbf{X})\} \\ &= E\{Y_t\delta_t|\mathbf{h}(\mathbf{X})\}/E\{\delta_t|\mathbf{h}(\mathbf{X})\} \text{ by strong ignorability} \\ &= E\{Y\delta_t|\mathbf{h}(\mathbf{X})\}/e_t(\mathbf{X}). \end{aligned}$$

It can be seen that $\mathbf{h}(\mathbf{x})$ is a finer function of both the numerate and denominator in the above expression. Thus $E\{Y_t|\mathbf{h}(\mathbf{X})=\mathbf{h}\}$ is fully determined by $\mathbf{h}$ and no longer relies on the value of $\mathbf{X}$.

Finally, in order to have available causal inference, it is important to note that, for given t and t', $h(\mathbf{x})=\mathbf{h}$ fully determines both $e_t(\mathbf{h})$ and $e_{t'}(\mathbf{h})$. Therefore,

$$\begin{aligned} E\{Y_t-Y_{t'}|\mathbf{h}(\mathbf{X})=\mathbf{h}\} &= E\{Y_t|\mathbf{h}(\mathbf{X})=\mathbf{h},T=t,e_t(\mathbf{h})\}-E\{Y_{t'}|\mathbf{h}(\mathbf{X})=\mathbf{h},T=t',e_{t'}(\mathbf{h})\} \\ &= E\{Y|\mathbf{h}(\mathbf{X})=\mathbf{h},T=t\}-E\{Y|\mathbf{h}(\mathbf{X})=\mathbf{h},T=t'\} \end{aligned}$$

is independent of $\mathbf{X}$. This justifies the direct use of mean response comparison for causal inference in subpopulation $\Omega_{\mathbf{h}}$. ∎

Appendix B. Proof of Proposition 4.

First of all, condition (i) in Definition 2 holds as $\mathbf{X}\perp\!\!\!\perp T|h_3(\mathbf{X})$. Assuming $(Y_1,Y_0)\perp\!\!\!\perp T\,|\,\mathbf{X}$, it follows that $(Y_1,Y_0)\perp\!\!\!\perp T\,|\,h_3(\mathbf{X})$ under strong ignorability.

Now it suffices to verify condition (ii). Consider

$$\begin{aligned} E\{Y_1\,|\,h_2(\mathbf{X}),h_3(\mathbf{X})\} &= E\{Y_1\,|\,T=1,h_2(\mathbf{X}),h_3(\mathbf{X})\} \\ &= E\{Y\,|\,T=1,h_2(\mathbf{X}),h_3(\mathbf{X})\} \\ &= E\{E(Y|\mathbf{X},T=1)\,|\,h_2(\mathbf{X}),h_3(\mathbf{X})\} \\ &= E\{\gamma_0+\gamma_1+h_1(\mathbf{X})+h_2(\mathbf{X})\,|\,h_2(\mathbf{X}),h_3(\mathbf{X})\} \\ &= \gamma_0+\gamma_1+h_2(\mathbf{X})+E\{h_1(\mathbf{X})\,|\,h_2(\mathbf{X}),h_3(\mathbf{X})\} \end{aligned}$$

Similarly, it can be found that

$$E\{Y_0\,|\,h_2(\mathbf{X}),h_3(\mathbf{X})\}=\gamma_0+E(h_1(\mathbf{X})|h_2(\mathbf{X}),h_3(\mathbf{X})\}.$$

Thus,

$$E\{Y_1-Y_0\,|\,h_2(\mathbf{X})=h_2,\,h_3(\mathbf{X})=h_3\}=\gamma_1+h_2,$$

which is independent of $\mathbf{X}$. ∎

Appendix C. Proof of Theorem 5.

The following lemma (see, e.g., Chapter 9 of Lin and Bai 2011), derived directly from C_r inequality, will be used in the proof.

Lemma 8 *Given a sequence $X_1,\ldots,X_n$ of random variables, $X_n=\sum_{i=1}^n X_i/n$. Then*

$$E|X_n|^r\leq\frac{1}{n}\cdot\sum_{i=1}^n E|X_i|^r \text{ for } r>1.$$

By condition (ii) in Definition 2 of $\mathbf{a}(\mathbf{x})$,

$$\begin{aligned}(y_{\tau 1}-y_{\tau 0})-E(Y_1-Y_0\,|\,\mathbf{a}_\tau) &= (y_{\tau 1}-y_{\tau 0})-E(Y_1-Y_0\,|\mathbf{x}) \\ &\quad +E(Y_1-Y_0|\mathbf{a}(\mathbf{x}))-E(Y_1-Y_0\,|\,\mathbf{a}_\tau) \\ &= \{y_{\tau 1}-E(Y_1|\mathbf{x})+E(Y_1\,|\,\mathbf{a}(\mathbf{x}))-E(Y_1|\mathbf{a}_\tau)\}- \\ &\quad \{y_{\tau 0}-E(Y_0|\mathbf{x})+E(Y_0\,|\,\mathbf{a}(\mathbf{x}))-E(Y_0|\mathbf{a}_\tau)\} \\ &= \zeta_1-\zeta_0\end{aligned}$$

For convenience, we have used $\mathbf{a}_\tau$ as shorthand for the conditioning event $\{\mathbf{a}(\mathbf{x})=\mathbf{a}_\tau\}$. To prove (22), it suffices, by Minkowski's inequality, to verify the mean square or L_2 consistency for ζ_1 and ζ_0 separately.

Consider

$$\zeta_1=\{y_{\tau 1}-E(Y_1|\mathbf{x})\}+\{E(Y_1\,|\,\mathbf{a}(\mathbf{x}))-E(Y_1|\mathbf{a}_\tau)\},$$

which has two terms. We examine the second term $\{E(Y_1\,|\,\mathbf{a}(\mathbf{x}))-E(Y_1|\mathbf{a}_\tau)\}$ first. If assumptions (19), (20), and (21) hold, then $\mathbf{a}_\tau \xrightarrow{p} \mathbf{a}(\mathbf{x})$ by Theorem 12.7 of Breiman et al. (1984, p. 322). Since $E(Y_1|\mathbf{a})$ is assumed continuous in $\mathbf{a}$,

$$E(Y_1|\mathbf{a}_\tau) \xrightarrow{p} E(Y_1|\mathbf{a}(\mathbf{x}))$$

by the continuous mapping theorem. Moreover, since $|E(Y_1|\mathbf{a}_\tau)| \le E(|Y_1||\mathbf{a}_\tau) \le E(|Y_1|) < \infty$, it follows that

$$\lim_n E\,|E(Y_1|\mathbf{a}_\tau)-E(Y_1|\mathbf{a}(\mathbf{x}))|^2=0$$

by the dominated (or bounded) convergence theorem.

Next, consider the first term in ζ_1, $\{y_{\tau 1}-E(Y_1|\mathbf{x})\}$. Rewrite $y_{\tau 1}$ as

$$y_{\tau 1}=\frac{\sum_{i\in\tau} Y_iT_i}{\sum_{i\in\tau} T_i}=\frac{\sum_{i\in\tau} Y_iT_i/n_\tau}{\sum_{i\in\tau} T_i/n_\tau}=\frac{\xi_n}{\rho_n}.$$

which is a ratio estimator. The convergence of ratio estimators in the general form was studied by Cramér (1946). Using Theorem 12.7 of Breiman et al. (1984) again, we have $\xi_n \xrightarrow{p} E(YT|\mathbf{x})$ and $\rho_n \xrightarrow{p} e(\mathbf{x})$ in probability. Thus

$$\frac{\xi_n}{\rho_n} \xrightarrow{p} \frac{E(YT|\mathbf{x})}{e(\mathbf{x})}=E(Y_1|\mathbf{x})$$

in probability as well if $e(\mathbf{x})\neq 0$, under the assumption of strong ignorability. To establish its mean square risk consistency, the necessary and sufficient condition is that the random sequence $\{y_{\tau 1}^2\}_n$ is uniformly integrable, that is,

$$\lim_{c_0\to\infty}\sup_n E\{y_{\tau 1}^2\,I(y_{\tau 1}>c_0)\}=0.$$

A sufficient condition for uniform integrability (**?**) is that

$$\sup_n E|y_{\tau 1}|^{2+\varepsilon}<\infty$$

for some $\varepsilon > 0$. This can be verified because

$$\sup_n E|y_{\tau 1}|^{2+\varepsilon} \leq \sup_n \frac{1}{n_{\tau 1}} \sum_{\{i \in \tau:\, T_i = 1\}} E|Y|^{2+\varepsilon} \leq M < \infty$$

following from (19) and Lemma 8.

Therefore, $\lim_n E|\zeta_1|^2 = 0$ using Minkowski's inequality again. Similar arguments can be used to show $\lim_n E|\zeta_0|^2 = 0$. This completes the proof of Theorem 5. ∎

References

L. S. Aiken and S. G. West. *Multiple Regression: Testing and Interpreting Interactions*. Newbury Park, CA: Sage, 1991.

H. Akaike A new look at model identification. *IEEE Transactions an Automatic Control*, 19: 716–723, 1974.

P. C. Austin Using ensemble-based methods for directly estimating causal effects: An investigation of tree-based g-computation. *Multivariate Behavioral Research*, 47:115-135, 2012.

J. D. Angrist, W. Imbens, and D. B. Rubin. Identification of causal effects using instrumental variables. *Journal of the American Statistical Association*, 91:444–455, 1996.

L. Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Belmont, CA: Wadsworth International Group, 1984.

A. Ciampi, S. A. Hogg, S. McKinney, and J. Thiffault. RECPAM: a computer program for recursive partition and amalgamation for censored survival data and other situations frequently occurring in biostatistics. I. Methods and program features. *Computer Methods and Programs in Biomedicine*, 26(3):239–256, 1988.

H. Cramér. *Mathematical Methods in Statistics*. Princeton, NJ: Princeton University Press, 1946.

R. H. Dehejia and S. Wahba. Causal effects in nonexperimental studies: Re-evaluating the evaluation of training programs. *Journal of the American Statistical Association*, 94:1053–1062, 1999.

M. Gail and R. Simon. Testing for qualitative interactions between treatment effects and patient subsets. *Biometrics*, 41:361–372, 1985.

L. Gordon and R. Olshen. Asymptotically efficient solutions to the classification problem. *The Annals of Statistics*, 6:515–533, 1978.

L. Gordon and R. Olshen. Consistent nonparametric regression from recursive partitioning schemes. *Journal of Multivariate Analysis*, 10:611–627, 1980.

L. Gordon and R. Olshen. Almost surely consistent nonparametric regression from recursive partitioning schemes. *Journal of Multivariate Analysis*, 15:147–163, 1984.

J. C. Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–328, 1966.

S. Greenland. Quantifying biases in causal models: classical confounding vs collider-stratification bias. *Epidemiology*, 14:300-306, 2003.

K. Imai and D. A. van Dyk. Causal inference with general treatment regimes: generalizing the propensity score. *Journal of the American Statistical Association*, 99:854–866, 2004.

K. Hirano and G. W. Imbens. The propensity score with continuous treatments. In *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives: An Essential Journey with Donald Rubin's Statistical Family*. New York, NY: Wiley, 2005.

D. Ho, K. Imai, G. King, and E. Stuart. Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political Analysis*, 15:199–236, 2007.

D. Ho, K. Imai, G. King, and E. Stuart. `Matchit`: Nonparametric preprocessing for parametric causal inference. *Journal of Statistical Software*, 42(8), 2011. `http://gking.harvard.edu/matchit/`.

P. W. Holland. Statistics and causal inference (with discussion). *Journal of the American Statistical Association*, 81:945–970, 1986.

P. W. Holland and D. B. Rubin. Causal inference in retrospective studies. *Evaluation Review*, 12: 203–231, 1988.

D. Horvitz and D. Thompson. A Generalization of sampling without replacement from a finite population. *Journal of the American Statistical Association*, 47:663–685, 1952.

G. Imbens. The role of the propensity score in estimating dose-response functions. *Biometrika*, 87: 706-710, 2000.

K. K. Jain. *Textbook of Personalized Medicine*. New York, NY: Springer, 2009.

S. W. Lagakos. The challenge of subgroup analyses – reporting without distorting. *The New England Journal of Medicine*, 354:1667–1669, 2006.

R. J. LaLonde. Evaluating the econometric evaluations of training programs with experimental data. *American Economic Review*, 76:604–620, 1986.

M. LeBlanc and J. Crowley. Survival trees by goodness of split. *Journal of the American Statistical Association*, 88:457–467, 1993.

M. Lechner. Identification and estimation of causal effects o fmultiple treatments under the conditional independence assumption. In *Econometric Evaluations of Active Labour Market Policies in Europe*, Ed. M. Lechnwe and F. Pfeiffer. Heidelberg: Physica, 1999.

B. K. Lee, J. Lessler, and E. A. Stuart. Improving propensity score weighting using machine learning. *Statistics in Medicine*, 29:337–46, 2010.

K.-C. Li. Sliced inverse regression for dimension reduction (with discussion). *Journal of the American Statistical Association*, 86:316–342, 1991.

Z. Y. Lin and Z. D. Bai. *Probability Inequalities*. Beijing: Sience Press & Berlin Heidelgerg: Springer-Verlag, 2011.

J. K. Lunceford and M. Davidian. Stratification and weighting via the propensity score in estimation of causal treatment effects: a comparative study. *Statistics in Medicine*, 23:2937–2960, 2004.

D. F. McCaffrey, G. Ridgeway, and A. R. Morral. Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological Methods*, 9:403–425, 2004.

J. Morgan and J. Sonquist. Problems in the analysis of survey data and a proposal. *Journal of the American Statistical Association*, 58: 415–434, 1963.

J. Neyman. On the application of probability theory to agricultural experiments: Essay on Principles, Section 9. Translated in *Statistical Science*, 5:465–480, 1923.

J. Pearl. *Casuality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

J. M. Robins. The analysis of randomized and non-randomized AIDS treatment trials using a new approach to causal inference in longitudinal studies. In *Health Service Research Methodology: A Focus on AIDS*. Eds: Sechrest L., Freeman H., Mulley A. Washington, D.C.: U.S. Public Health Service, National Center for Health Services Research, pp. 113–159, 1989.

J. M. Robins. Marginal structural models versus structural nested models as tools for causal inference. In *Statistical Models in Epidemiology: The Environment and Clinical Trials*. Eds: M.E. Halloran and D. Berry, IMA Volume 116, NY: Springer-Verlag, pp. 95-134, 1999.

J. M. Robins, A. Rotnitzky, and L. P. Zhao. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American Statistical Association*, 89:846–866, 1994.

P. R. Rosenbaum. Optimal matching for observational studies. *Journal of the American Statistical Association*, 84:1024–1032, 1989.

P. R. Rosenbaum and D. B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70:41–55, 1983.

M. Rosenblum and M. J. van der Laan. Simple examples of estimating causal effects using targeted maximum likelihood estimation. Johns Hopkins University, Dept. of Biostatistics Working Papers. Working Paper 209, 2011. `http://biostats.bepress.com/jhubiostat/paper209`

D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66:688–701, 1974.

D. B. Rubin. Inference and missing data. *Biometrika*, 63:581–592, 1976.

D. B. Rubin. Assignment of treatment group on the basis of a covariate. *Journal of Educational Statistics*, 2:1–26, 1977.

D. B. Rubin. Bayesian inference for causal effects: The role of randomization. *The Annals of Statistics*, 7:34–58, 1978.

D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100:322–331, 2005.

J. L. Schafer and J. Kang. Average causal effects from nonrandomized studies: a practical guide and simulated example. *Psychological Methods*, 13:279–313, 2008.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6: 461–464, 1978.

R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. New York, NY: Wiley-Interscience, 2001.

I. Shpitser and J. Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9:1941–1979, 2008.

J. M. Snowden, S. Rose, and K. M. Mortimer. Implementation of g-computation on a simulated data set: Demonstration of a causal inference technique. *American Journal of Epidemiology*, 173:731–738, 2011.

P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, Prediction, and Search*. 2nd Edition. The MIT Press, 2001.

X. G. Su, C.-L. Tsai, H. Wang, D. M. Nickerson, and B. G. Li. Subgroup analysis via recursive partitioning. *Journal of Machine Learning Research*, 10: 141–158, 2009.

X. G. Su, M. Wang, and J. Fan. Maximum likelihood regression trees. *Journal of Computational and Graphical Statistics*, 13:586–598, 2004.

R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the GAP statistic. *Journal of the Royal Statistical Society*, *Series B*, 63: 411–423, 2001.

D. Toth and J. L. Eltinge. Building consistent regression trees from complex sample data. Research paper, Office of Survey Methods Research (OSMR), U.S. Bureau of Labor Statistics, 2010. `http://www.bls.gov/osmr/pdf/st100010.pdf`.

W. S. Torgerson. *Theory and Methods of Scaling*. New York: Wiley, 1958.

M. J. van der Laan, E. C. Polley, and A. E. Hubbard. Super learner. *Statistical Applications in Genetics and Molecular Biology*, 6.1, 2007. `http://works.bepress.com/mark_van_der_laan/201`

M. J. van der Laan and D. Rubin. Targeted maximum likelihood learning. U.C. Berkeley Division of Biostatistics Working Paper Series. Working Paper 213, 2006. `http://biostats.bepress.com/ucbbiostat/paper213`.

T. J. VanderWeele and J. M. Robins. Four types of effect modification: A classification based on directed acyclic graphs. *Epidemiology*, 18:561–568, 2007.

T. J. VanderWeele. On the distinction between interaction and effect modification. *Epidemiology*, 20:863–871, 2009.

H. Wang and Y. Xia. Sliced regression for dimension reduction. *Journal of the American Statistical Association*, 103:811–821, 2008.

J. Wang. Consistent selection of the number of clusters via cross validation. *Biometrika*, 97:893–904, 2010.

Journal of Machine Learning Research 13 (2012) 2995-2998 Submitted 9/11; Revised 7/12; Published 10/12

Oger: Modular Learning Architectures For Large-Scale Sequential Processing

David Verstraeten DAVID.VERSTRAETEN@UGENT.BE
Benjamin Schrauwen BENJAMIN.SCHRAUWEN@UGENT.BE
Sander Dieleman SANDER.DIELEMAN@UGENT.BE
Philemon Brakel PHILEMON.BRAKEL@UGENT.BE
Pieter Buteneers PIETER.BUTENEERS@UGENT.BE
Department of Electronics and Information Systems
Ghent University
Ghent, Belgium

Dejan Pecevski DEJAN@IGI.TUGRAZ.AT
Institute for Theoretical Computer Science
Graz University of Technology
Graz, Austria

Editor: Cheng Soon Ong

Abstract

Oger (OrGanic Environment for Reservoir computing) is a Python toolbox for building, training and evaluating modular learning architectures on large data sets. It builds on MDP for its modularity, and adds processing of sequential data sets, gradient descent training, several cross-validation schemes and parallel parameter optimization methods. Additionally, several learning algorithms are implemented, such as different reservoir implementations (both sigmoid and spiking), ridge regression, conditional restricted Boltzmann machine (CRBM) and others, including GPU accelerated versions. Oger is released under the GNU LGPL, and is available from `http://organic.elis.ugent.be/oger`.

Keywords: Python, modular architectures, sequential processing

1. Introduction

The Oger toolbox originated from the need to rapidly implement, investigate and compare complex architectures built from state-of-the-art sequential processing algorithms, focused on but not limited to reservoir computing, and to apply these architectures to large real-world tasks. Reservoir computing (RC) is a learning framework (Verstraeten et al., 2007) whereby a random non-linear dynamical system (usually a recurrent neural network) is left untrained and used as input to a simple learning algorithm such as linear regression. A number of smaller toolboxes for reservoir computing are available, written in C++, Java and Matlab.[1] However, these are generally focused on specific implementations of RC (echo state networks or liquid state machines) and offer less flexibility in creating and evaluating complex architectures.

Rather than contribute yet another toolbox which reimplements many standard algorithms, one of our design choices for Oger was to incorporate existing packages where possible. Because mod-

1. An overview can be found at `http://organic.elis.ugent.be/software`.

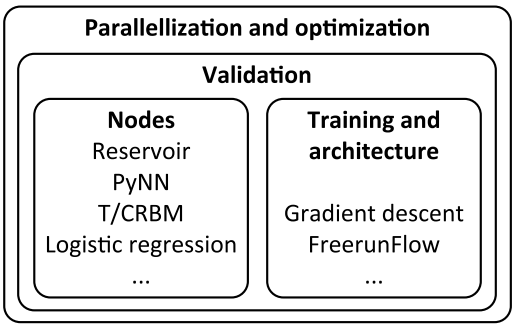


Figure 1: A schematic overview of the structure of Oger. The basic processing blocks (nodes) are combined with methods for constructing and training architectures. These architectures can then be evaluated in a validation and optimization framework.

ularity was one of the key requirements for Oger, it has been based on the well known and widely used Modular Data Processing toolkit (MDP), which provides this modularity in addition to a wide variety of machine learning algorithms (Zito et al., 2008). Oger uses a `Node` as its basic building block: a (optionally trainable) data processing algorithm. These nodes can then be combined into an arbitrary feedforward graph structure called a `Flow`. Much of the error- and type-checking is abstracted away through the object-oriented interface, such that the developer can focus on implementing the actual algorithm.

Python was chosen as the development language because it is a high-level, cross-platform and open-source interpreted language offering flexibility and rapid development, while interfaces to optimized numerical linear algebra packages such as BLAS are provided through the NumPy package so that the speed sacrifice remains limited. Mature and feature-complete packages for plotting (matplotlib) and general scientific computing (SciPy) that in many respects come close to commercial alternatives are available, along with a plethora of smaller libraries providing specific functions.

2. Features

In this section we describe the main features of Oger and give a usage example.

2.1 Algorithms

MDP implements several standard supervised and unsupervised learning methods for operating on stationary inputs, such as principal component analysis, independent component analysis and factor analysis.[2] Oger adds several new methods to this set:

– Several reservoir implementations : a basic reservoir with customizable nonlinear function and weight topologies, a leaky integrator reservoir, and a GPU-optimized reservoir using CUDA.

– Wrappers for creating spiking reservoirs using PyNN-compatible neural network simulators (Davison et al., 2008).

– A logistic regression node trainable with different optimizers such as IRLS, conjugate gradient, BFGS and others.

2. We refer to the MDP website `http://mdp-toolkit.sourceforge.net/` for an exhaustive list.

– A conditional restricted Boltzmann machine: a standard RBM with an additional context vector.

– Several 'utility' signal processing methods: a resampling node, a timeshift node, a winner-take-all node, and others.

Additionally, Oger supports backpropagation training using various methods of gradient descent, such as stochastic gradient descent, RPROP and others. Finally, a FreerunFlow allows easy training and execution of architectures with feedback, for instance for time-series generation tasks (see the usage example below).

2.2 Validation, Optimization and Parallel Execution

Around the data processing algorithms described above, Oger offers functionality for large-scale validation and optimization. The validation automates the process of constructing training and test sets, and the actual training and evaluation. Several standard validation schemes are provided (n-fold, leave-one-out (LOO) cross-validation and others), but this can be customized (for example, if a fixed training and test set is defined).

Oger provides an Optimizer class. This class allows both exploration of a certain parameter space and optimization of a vector of parameters according to a loss function (which can be user-defined, or one of the several provided by Oger). The optimization itself can be done using grid-searching, or using an interface to any of the algorithms in `scipy.optimize` or the Python CMA-ES module (Hansen, 2006). Finally, a variety of error measures and utility classes such as a ConfusionMatrix are included.

Oger allows two modes of parallel execution, both local (multi-threaded or multi-process) and on a computing grid. The first mode is inherited from MDP, where the training and execution of a flow on a data set consisting of different chunks can be done in parallel (if the nodes in the flow support this). The second mode is the parallel evaluation of parameter points for grid-searching and CMA-ES (the `scipy.optimize` functions as yet do not support this). Both modes use runtime overloading of class methods by their parallel versions, which makes the transition from sequential to parallel execution very user-friendly and possible using a couple of lines of code (see the usage example below).

3. Usage Example

As an illustrative example, we construct and train a reservoir and readout setup with output feedback for generating the Mackey-Glass time-series. We refer to the Oger website and the Oger installation package for more usage examples.

```
 1 from scipy import *
 2 import Oger, mdp
 3 signals = Oger.datasets.mackey_glass(n_samples=4, sample_len=3000)
 4 res = Oger.nodes.LeakyReservoirNode(output_dim=400, reset_states=False)
 5 readout = Oger.nodes.RidgeRegressionNode()
 6 flow = Oger.nodes.FreerunFlow([res, readout], freerun_steps=300)
 7 parameters = {res:{'input_scaling':arange(.1, 1, .1), 'bias_scaling':
       arange(0, .5, .1), 'leak_rate':arange(.1,.5,.1)}}
 8 internal_params = {readout:{'ridge_param': 10. ** arange(-4, 0, .5)}}
 9 opt = Oger.evaluation.Optimizer(parameters, Oger.utils.nrmse)
10 opt.scheduler = mdp.parallel.ProcessScheduler(n_processes=None)
11 mdp.activate_extension('parallel')
```

```
12 opt.grid_search([[], signals[:-1]], flow, Oger.evaluation.leave_one_out,
       internal_params)
13 opt_flow = opt.get_optimal_flow(verbose=True)
14 opt_flow.train([[], signals[:-1]])
15 y = opt_flow.execute(signals[-1][0])
```

On line 3, the data set is generated, which in this case consists of four Mackey-Glass time-series generated from different initial states. In the next two lines, a reservoir node and a linear readout node trained with ridge regression are created. Line 6 concatenates these nodes into a FreerunFlow, which provides one-step ahead prediction during training and feeds the output back to the input of the flow during execution. Lines 7 and 8 define a search space for the reservoir parameters and the regularization constant of the readout node which is optimized separately for each set of reservoir parameters. On line 9 an Optimizer object is instantiated which will optimize these parameters using the provided error measure (normalized root mean squared error). Lines 10 and 11 ensure that the optimization is done in parallel, using separate processes. On line 12, the actual optimization is performed using LOO cross-validation on the four time-series, while for each fold the regularization constant for the ridge regression is optimized again using LOO cross-validation. This can take a few minutes. On line 13 the Optimizer is queried to return the optimal flow, which is subsequently trained using all the training signals and applied to an unseen test signal in lines 14 and 15 respectively.

Acknowledgments

This work was funded by the European Commission FP7 project ORGANIC (FP7-231267). Dejan Pecevski has been additionally partially supported by the European Union project FP7-506778 (PASCAL2).

References

A.P. Davison, D. Brüderle, J. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger. Pynn: A common interface for neuronal network simulators. *Frontiers in Neuroinformatics*, 2, 2008.

N. Hansen. The CMA evolution strategy: A comparing review. In *Towards a New Evolutionary Computation*, pages 75–102. Springer, 2006.

D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. A unifying comparison of reservoir computing methods. *Neural Networks*, 20:391–403, 2007.

T. Zito, N. Wilbert, L. Wiskott, and P. Berkes. Modular toolkit for data processing (mdp): A python data processing framework. *Frontiers in Neuroinformatics*, 2, 2008.

Journal of Machine Learning Research 13 (2012) 2999-3039 Submitted 11/10; Revised 5/12; Published 10/12

Multi-Instance Learning with Any Hypothesis Class

Sivan Sabato SIVAN.SABATO@MICROSOFT.COM
Microsoft Research New England
1 Memorial Drive
Cambridge, MA

Naftali Tishby TISHBY@CS.HUJI.AC.IL
School of Computer Science & Engineering
The Hebrew University
Jerusalem 91904, Israel

Editor: Nicolas Vayatis

Abstract

In the supervised learning setting termed Multiple-Instance Learning (MIL), the examples are bags of instances, and the bag label is a function of the labels of its instances. Typically, this function is the Boolean OR. The learner observes a sample of bags and the bag labels, but not the instance labels that determine the bag labels. The learner is then required to emit a classification rule for bags based on the sample. MIL has numerous applications, and many heuristic algorithms have been used successfully on this problem, each adapted to specific settings or applications. In this work we provide a unified theoretical analysis for MIL, which holds for any underlying hypothesis class, regardless of a specific application or problem domain. We show that the sample complexity of MIL is only poly-logarithmically dependent on the size of the bag, for any underlying hypothesis class. In addition, we introduce a new PAC-learning algorithm for MIL, which uses a regular supervised learning algorithm as an oracle. We prove that efficient PAC-learning for MIL can be generated from any efficient non-MIL supervised learning algorithm that handles one-sided error. The computational complexity of the resulting algorithm is only polynomially dependent on the bag size.

Keywords: multiple-instance learning, learning theory, sample complexity, PAC learning, supervised classification

1. Introduction

We consider the learning problem termed Multiple-Instance Learning (MIL), first introduced in Dietterich et al. (1997). MIL is a special type of a supervised classification problem. As in classical supervised classification, in MIL the learner receives a sample of labeled examples drawn i.i.d from an arbitrary and unknown distribution, and its objective is to discover a classification rule with a small expected error over the same distribution. In MIL additional structure is assumed, whereby the examples are received as *bags* of *instances*, such that each bag is composed of several instances. It is assumed that each instance has a true label, however the learner only observes the labels of the bags. In classical MIL the label of a bag is the Boolean OR of the labels of the instances the bag contains. Various generalizations to MIL have been proposed (see, e.g., Raedt, 1998; Weidmann et al., 2003). Here we consider both classical MIL and the more general setting, where a function

other than Boolean OR determines bag labels based on instance labels. This function is known to the learner a-priori. We term the more general setting *generalized MIL*.

It is possible, in principle, to view MIL as a regular supervised classification task, where a bag is a single example, and the instances in a bag are merely part of its internal representation. Such a view, however, means that one must analyze each specific MIL problem separately, and that results and methods that apply to one MIL problem are not transferable to other MIL problems. We propose instead a generic approach to the analysis of MIL, in which the properties of a MIL problem are analyzed as a function of the properties of the matching non-MIL problem. As we show in this work, the connections between the MIL and the non-MIL properties are strong and useful. The generic approach has the advantage that it automatically extends all knowledge and methods that apply to non-MIL problems into knowledge and methods that apply to MIL, without requiring specialized analysis for each specific MIL problem. Our results are thus applicable for diverse hypothesis classes, label relationships between bags and instances, and target losses. Moreover, the generic approach allows a better theoretical understanding of the relationship, in general, between regular learning and multi-instance learning with the same hypothesis class.

The generic approach can also be helpful for the design of algorithms, since it allows deriving generic methods and approaches that hold across different settings. For instance, as we show below, a generic PAC-learning algorithm can be derived for a large class of MIL problems with different hypothesis classes. Other applications can be found in follow-up research of the results we report here, such as a generic bag-construction mechanism (Sabato et al., 2010), and learning when bags have a manifold structure (Babenko et al., 2011). As generic analysis goes, it might be possible to improve upon it in some specific cases. Identifying these cases and providing tighter analysis for them is an important topic for future work. We do show that in some important cases—most notably that of learning separating hyperplanes with classical MIL—our analysis is tight up to constants.

MIL has been used in numerous applications. In Dietterich et al. (1997) the drug design application motivates this setting. In this application, the goal is to predict which molecules would bind to a specific binding site. Each molecule has several possible conformations (shapes) it can take. If at least one of the conformations binds to the binding site, then the molecule is labeled positive. However, it is not possible to experimentally identify which conformation was the successful one. Thus, a molecule can be thought of as a bag of conformations, where each conformation is an instance in the bag representing the molecule. This application employs the hypothesis class of Axis Parallel Rectangles (APRs), and has made APRs the hypothesis class of choice in several theoretical works that we mention below. There are many other applications for MIL, including image classification (Maron and Ratan, 1998), web index page recommendation (Zhou et al., 2005) and text categorization (Andrews, 2007).

Previous theoretical analysis of the computational aspects of MIL has been done in two main settings. In the first setting, analyzed for instance in Auer et al. (1998), Blum and Kalai (1998) and Long and Tan (1998), it is assumed that all the instances are drawn i.i.d from a single distribution over instances, so that the instances in each bag are statistically independent. Under this independence assumption, learning from an i.i.d. sample of bags is as easy as learning from an i.i.d. sample of instances with one-sided label noise. This is stated in the following theorem.

Theorem 1 (Blum and Kalai, 1998) *If a hypothesis class is PAC-learnable in polynomial time from one-sided random classification noise, then the same hypothesis class is PAC-learnable in polynomial time in MIL under the independence assumption. The computational complexity of learning is polynomial in the bag size and in the sample size.*

The assumption of statistical independence of the instances in each bag is, however, very limiting, as it is irrelevant to many applications.

In the second setting one assumes that bags are drawn from an arbitrary distribution *over bags*, so that the instances within a bag may be statistically dependent. This is clearly much more useful in practice, since bags usually describe a complex object with internal structure, thus it is implausible to assume even approximate independence of instances in a bag. For the hypothesis class of APRs and an arbitrary distribution over bags, it is shown in Auer et al. (1998) that if there exists a PAC-learning algorithm for MIL with APRs, and this algorithm is polynomial in both the size of the bag and the dimension of the Euclidean space, then it is possible to polynomially PAC-learn DNF formulas, a problem which is solvable only if $\mathcal{RP} = \mathcal{NP}$ (Pitt and Valiant, 1986). In addition, if it is possible to improperly learn MIL with APRs (that is, to learn a classifier which is not itself an APR), then it is possible to improperly learn DNF formulas, a problem which has not been solved to this date for general distributions. This result implies that it is not possible to PAC-learn MIL on APRs using an algorithm which is efficient in both the bag size and the problem's dimensionality. It does not, however, preclude the possibility of performing MIL efficiently in other cases.

In practice, numerous algorithms have been proposed for MIL, each focusing on a different specialization of this problem. Almost none of these algorithms assume statistical independence of instances in a bag. Moreover, some of the algorithms explicitly exploit presumed dependences between instances in a bag. Dietterich et al. (1997) propose several heuristic algorithms for finding an APR that predicts the label of an instance and of a bag. Diverse Density (Maron and Lozano-Pérez, 1998) and EM-DD (Zhang and Goldman, 2001) employ assumptions on the structure of the bags of instances. DPBoost (Andrews and Hofmann, 2003), mi-SVM and MI-SVM (Andrews et al., 2002), and Multi-Instance Kernels (Gärtner et al., 2002) are approaches for learning MIL using margin-based objectives. Some of these methods work quite well in practice. However, no generalization guarantees have been provided for any of them.

In this work we analyze MIL and generalized MIL in a general framework, independent of a specific application, and provide results that hold for any underlying hypothesis class. We assume a fixed hypothesis class defined over instances. We then investigate the relationship between learning with respect to this hypothesis class in the classical supervised learning setting with no bags, and learning with respect to the same hypothesis class in MIL. We address both sample complexity and computational feasibility.

Our sample complexity analysis shows that for binary hypothesis and thresholded real-valued hypotheses, the distribution-free sample complexity for generalized MIL grows only logarithmically with the maximal bag size. We also provide poly-logarithmic sample complexity bounds for the case of margin learning. We further provide distribution-dependent sample complexity bounds for more general loss functions. These bound are useful when only the average bag size is bounded. The results imply generalization bounds for previously proposed algorithms for MIL. Addressing the computational feasibility of MIL, we provide a new learning algorithm with provable guarantees for a class of bag-labeling functions that includes the Boolean OR, used in classical MIL, as a special case. Given a non-MIL learning algorithm for the desired hypothesis class, which can handle one-sided errors, we improperly learn MIL with the same hypothesis class. The construction is simple to implement, and provides a computationally efficient PAC-learning of MIL, with only a polynomial dependence of the run time on the bag size.

In this work we consider the problem of learning to classify bags using a labeled sample of bags. We do not attempt to learn to classify single instances using a labeled sample of bags. We point out

that it is not generally possible to find a low-error classification rule for instances based on a bag sample. As a simple counter example, assume that the label of a bag is the Boolean OR of the labels of its instances, and that every bag includes both a positive instance and a negative instance. In this case all bags are labeled as positive, and it is not possible to distinguish the two types of instances by observing only bag labels.

The structure of the paper is as follows. In Section 2 the problem is formally defined and notation is introduced. In Section 3 the sample complexity of generalized MIL for binary hypotheses is analyzed. We provide a useful lemma bounding covering numbers for MIL in Section 4. In Section 5 we analyze the sample complexity of generalized MIL with real-valued functions for large-margin learning. Distribution-dependent results for binary learning and real-valued learning based on the average bag size are presented in Section 6. In Section 7 we present a PAC-learner for MIL and analyze its properties. We conclude in Section 8. The appendix includes technical proofs that have been omitted from the text. A preliminary version of this work has been published as Sabato and Tishby (2009).

2. Notations and Definitions

For a natural number k, we denote $[k] \triangleq \{1,\ldots,k\}$. For a real number x, we denote $[x]_+ = \max\{0,x\}$. log denotes a base 2 logarithm. For two vectors $\mathbf{x},\mathbf{y} \in \mathbb{R}^n$, $\langle \mathbf{x},\mathbf{y}\rangle$ denotes the inner product of $\mathbf{x}$ and $\mathbf{y}$. We use the function $\text{sign} : \mathbb{R} \to \{-1,+1\}$ where $\text{sign}(x) = 1$ if $x \geq 0$ and $\text{sign}(x) = -1$ otherwise. For a function $f : A \to B$, we denote by $f_{|C}$ its restriction to a set $C \subseteq A$. For a univariate function f, denote its first and second derivatives by f' and f'' respectively.

Let $\mathcal{X}$ be the input space, also called the domain of instances. A bag is a finite ordered set of instances from $\mathcal{X}$. Denote the set of allowed sizes for bags in a specific MIL problem by $R \subseteq \mathbb{N}$. For any set A we denote $A^{(R)} \triangleq \cup_{n\in R} A^n$. Thus the domain of bags with a size in R and instances from $\mathcal{X}$ is $\mathcal{X}^{(R)}$. A bag of size n is denoted by $\mathbf{x} = (x[1],\ldots,x[n])$ where each $x[j] \in \mathcal{X}$ is an instance in the bag. We denote the number of instances in $\mathbf{x}$ by $|\mathbf{x}|$. For any univariate function $f : A \to B$, we may also use its extension to a multivariate function from sequences of elements in A to sequences of elements in B, defined by $f(a[1],\ldots,a[k]) = (f(a[1]),\ldots,f(a[k]))$.

Let $I \subseteq \mathbb{R}$ an allowed range for hypotheses over instances or bags. For instance, $I = \{-1,+1\}$ for binary hypotheses and $I = [-B,B]$ for real-valued hypotheses with a bounded range. $\mathcal{H} \subseteq I^{\mathcal{X}}$ is a hypothesis class for instances. Every MIL problem is defined by a fixed bag-labeling function $\psi : I^{(R)} \to I$ that determines the bag labels given the instance labels. Formally, every instance hypothesis $h : \mathcal{X} \to I$ defines a bag hypothesis, denoted by $\overline{h} : \mathcal{X}^{(R)} \to I$ and defined by

$$\forall \mathbf{x} \in \mathcal{X}^{(R)}, \quad \overline{h}(\mathbf{x}) \triangleq \psi(h(x[1]),\ldots,h(x[r])).$$

The hypothesis class for bags given $\mathcal{H}$ and ψ is denoted $\overline{\mathcal{H}} \triangleq \{\overline{h} \mid h \in \mathcal{H}\}$. Importantly, the identity of ψ is known to the learner a-priori, thus each ψ defines a different generalized MIL problem. For instance, in classical MIL, $I = \{-1,+1\}$ and ψ is the Boolean OR.

We assume the labeled bags are drawn from a fixed distribution D over $\mathcal{X}^{(R)} \times \{-1,+1\}$, where each pair drawn from D constitutes a bag and its binary label. Given a range $I \subseteq \mathbb{R}$ of possible label predictions, we define a loss function $\ell : \{-1,+1\} \times I \to \mathbb{R}$, where $\ell(y,\hat{y})$ is the loss incurred if the true label is y and the predicted label is $\hat{y}$. The true loss of a bag-classifier $h : \mathcal{X}^{(R)} \to I$ is denoted by $\ell(h,D) \triangleq \mathbb{E}_{(\mathbf{X},Y)\sim D}[\ell(Y,h(\mathbf{X}))]$. We say that a sample or a distribution are *realizable* by $\overline{\mathcal{H}}$ if there is a hypothesis $h \in \overline{\mathcal{H}}$ that classifies them with zero loss.

The MIL learner receives a labeled sample of bags $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m)\} \subseteq \mathcal{X}^{(R)} \times \{-1,+1\}$ drawn from D^m, and returns a classifier $\hat{h} : \mathcal{X}^{(R)} \to I$. The goal of the learner is to return $\hat{h}$ that has a low loss $\ell(\hat{h},D)$ compared to the minimal loss that can be achieved with the bag hypothesis class, denoted by $\ell^*(\overline{\mathcal{H}},D) \triangleq \inf_{h\in\overline{\mathcal{H}}} \ell(h,D)$. The empirical loss of a classifier for bags on a labeled sample S is $\ell(h,S) \triangleq \mathbb{E}_{(X,Y)\sim S}[\ell(Y,h(\mathbf{X}))]$. For an unlabeled set of bags $S = \{\mathbf{x}_i\}_{i\in[m]}$, we denote the multi-set of instances in the bags of S by $S^{\cup} \triangleq \{x_i[j] \mid i \in [m], j \in [|\mathbf{x}_i|]\}$. Since this is a multi-set, any instance which repeats in several bags in S is represented the same amount of time in $S^{\cup}$.

2.1 Classes of Real-Valued bag-functions

In classical MIL the bag function is the Boolean OR over binary labels, that is $I = \{-1,+1\}$ and $\psi = \mathrm{OR} : \{-1,+1\}^{(R)} \to \{-1,+1\}$. A natural extension of the Boolean OR to a function over reals is the max function. We further consider two classes of bag functions over reals, each representing a different generalization of the max function, which conserves a different subset of its properties.

The first class we consider is the class of bag-functions that extend monotone Boolean functions. Monotone Boolean functions map Boolean vectors to $\{-1,+1\}$, such that the map is monotone-increasing in each of the inputs. The set of monotone Boolean functions is exactly the set of functions that can be represented by some composition of AND and OR functions, thus it includes the Boolean OR. The natural extension of monotone Boolean functions to real functions over real vectors is achieved by replacing OR with max and AND with min. Formally, we define extensions of monotone Boolean functions as follows.

Definition 2 *A function from $\mathbb{R}^n$ into $\mathbb{R}$ is* an extension of an n-ary monotone Boolean function *if it belongs to the set $\mathcal{M}_n$ defined inductively as follows, where the input to a function is $\mathbf{z} \in \mathbb{R}^n$:*

$$\begin{array}{ll} (1)\,\forall j \in [n], & \mathbf{z} \mapsto z[j] \in \mathcal{M}_n; \\ (2)\,\forall k \in \mathbb{N}^+, & f_1,\ldots,f_k \in \mathcal{M}_n \Longrightarrow \mathbf{z} \mapsto \max_{j\in[k]}\{f_j(\mathbf{z})\} \in \mathcal{M}_n; \\ (3)\,\forall k \in \mathbb{N}^+, & f_1,\ldots,f_k \in \mathcal{M}_n \Longrightarrow \mathbf{z} \mapsto \min_{j\in[k]}\{f_j(\mathbf{z})\} \in \mathcal{M}_n. \end{array}$$

We say that a bag-function $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ extends monotone Boolean functions if for all $n \in R$, $\psi_{|\mathbb{R}^n} \in \mathcal{M}_n$.

The class of extensions to Boolean functions thus generalizes the max function in a natural way.

The second class of bag functions we consider generalizes the max function by noting that for bounded inputs, the max function can be seen as a variant of the infinity-norm $\|\mathbf{z}\|_\infty = \max|z[i]|$. Another natural bag-function over reals is the average function, defined as $\psi(\mathbf{z}) = \frac{1}{n}\sum_{i\in[n]} z_i$, which can be seen as a variant of the 1-norm $\|\mathbf{z}\|_1 = \sum_{i\in[n]}|z[i]|$. More generally, we treat the case where the hypotheses map into $I = [-1,1]$, and consider the class of bag functions inspired by a p-norm, defined as follows.

Definition 3 *For $p \in [1,\infty)$, the p-*norm bag function $\psi_p : [-1,+1]^{(R)} \to [-1,+1]$ *is defined by:*

$$\forall \mathbf{z} \in \mathbb{R}^n, \quad \psi_p(\mathbf{z}) \triangleq \left(\frac{1}{n}\sum_{i=1}^{n}(z[i]+1)^p\right)^{1/p} - 1.$$

For $p = \infty$, Define $\psi_\infty \equiv \lim_{p\to\infty}\psi_p$.

Since the inputs of ψ_p are in $[-1,+1]$, we have $\psi_p(\mathbf{z}) \equiv n^{-1/p} \cdot \|\mathbf{z}+\mathbf{1}\|_p - 1$ where n is the length of $\mathbf{z}$. Note that the average function is simply ψ_1, and $\psi_\infty \equiv \|\mathbf{z}+\mathbf{1}\|_\infty - 1 \equiv \max$. Other values of p fall between these two extremes: Due to the p-norm inequality, which states that for all $p \in [1,\infty)$ and $\mathbf{x} \in \mathbb{R}^n$, $\frac{1}{n}\|\mathbf{x}\|_1 \leq n^{-1/p}\|\mathbf{x}\|_p \leq \|\mathbf{x}\|_\infty$, we have that for all $\mathbf{z} \in [-1,+1]^n$

$$\text{average} \equiv \psi_1(\mathbf{z}) \leq \psi_p(\mathbf{z}) \leq \psi_\infty(\mathbf{z}) \equiv \max.$$

Many of our results hold when the scale of the output of the bag-function is related to the scale of its inputs. Formally, we consider cases where the output of the bag-function does not change by much unless its inputs change by much. This is formalized in the following definition of a Lipschitz bag function.

Definition 4 *A bag function* $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ *is* c-Lipschitz with respect to the infinity norm *for* $c > 0$ *if*

$$\forall n \in R, \forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^n, \quad |\psi(\mathbf{a}) - \psi(\mathbf{b})| \leq c\|\mathbf{a}-\mathbf{b}\|_\infty.$$

The average bag-function and the max bag functions are 1-Lipschitz. Moreover, all extensions of monotone Boolean functions are 1-Lipschitz with respect to the infinity norm—this is easy to verify by induction on Definition 2. All p-norm bag functions are also 1-Lipschitz, as the following derivation shows:

$$|\psi_p(\mathbf{a}) - \psi_p(\mathbf{b})| = n^{-1/p} \cdot |\,\|\mathbf{a}+1\|_p - \|\mathbf{b}+1\|_p| \leq n^{-1/p} \cdot \|\mathbf{a}-\mathbf{b}\|_p \leq \|\mathbf{a}-\mathbf{b}\|_\infty.$$

Thus, our results for Lipschitz bag-functions hold in particular for the two bag-function classes we have defined here, and in specifically for the max function.

3. Binary MIL

In this section we consider binary MIL. In binary MIL we let $I = \{-1,+1\}$, thus we have a binary instance hypothesis class $\mathcal{H} \subseteq \{-1,+1\}^{\mathcal{X}}$. We further let our loss be the zero-one loss, defined by $\ell_{0/1}(y,\hat{y}) = \mathbb{1}[y \neq \hat{y}]$. The distribution-free sample complexity of learning relative to a binary hypothesis class with the zero-one loss is governed by the VC-dimension of the hypothesis class (Vapnik and Chervonenkis, 1971). Thus we bound the VC-dimension of $\overline{\mathcal{H}}$ as a function of the maximal possible bag size $r = \max R$, and of the VC-dimension of $\mathcal{H}$. We show that the VC-dimension of $\overline{\mathcal{H}}$ is at most logarithmic in r, and at most linear in the VC-dimension of $\mathcal{H}$, for any bag-labeling function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$. It follows that the sample complexity of MIL grows only logarithmically with the size of the bag. Thus MIL is feasible even for quite large bags. In fact, based on the results we show henceforth, Sabato et al. (2010) have shown that MIL can sometimes be used to accelerate even single-instance learning. We further provide lower bounds that show that the dependence of the upper bound on r and on the VC-dimension of $\mathcal{H}$ is imperative, for a large class of Boolean bag-labeling functions. We also show a matching lower bound for the VC-dimension of classical MIL with separating hyperplanes.

3.1 VC-Dimension Upper Bound

Our first theorem establishes a VC-Dimension upper bound for generalized MIL. To prove the theorem we require the following useful lemma.

Lemma 5 *For any $R \subseteq \mathbb{N}$ and any bag function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$, and for any hypothesis class $\mathcal{H} \subseteq \{-1,+1\}^{\mathcal{X}}$ and a finite set of bags $S \subseteq \mathcal{X}^{(R)}$,*

$$\left|\overline{\mathcal{H}}_{|S}\right| \leq \left|\mathcal{H}_{|S^{\cup}}\right|.$$

Proof Let $h_1, h_2 \in \overline{\mathcal{H}}$ be bag hypotheses. There exist instance hypotheses $g_1, g_2 \in \mathcal{H}$ such that $\overline{g}_i = h_i$ for $i = 1,2$. Assume that $h_{1|S} \neq h_{2|S}$. We show that $g_{1|S^{\cup}} \neq g_{2|S^{\cup}}$, thus proving the lemma.

From the assumption it follows that $\overline{g}_{1|S} \neq \overline{g}_{2|S}$. Thus there exists at least one bag $\mathbf{x} \in S$ such that $\overline{g}_2(\mathbf{x}) \neq \overline{g}_2(\mathbf{x})$. Denote its size by n. We have $\psi(g_1(x[1]), \ldots, g_1(x[n])) \neq \psi(g_2(x[1]), \ldots, g_2(x[n]))$. Hence there exists a $j \in [n]$ such that $g_1(x[j]) \neq g_2(x[j])$. By the definition of $S^{\cup}$, $x[j] \in S^{\cup}$. Therefore $g_{1|S^{\cup}} \neq g_{2|S^{\cup}}$. ■

Theorem 6 *Assume that $\mathcal{H}$ is a hypothesis class with a finite VC-dimension d. Let $r \in \mathbb{N}$ and assume that $R \subseteq [r]$. Let the bag-labeling function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$ be some Boolean function. Denote the VC-dimension of $\overline{\mathcal{H}}$ by d_r. We have*

$$d_r \leq \max\{16, 2d\log(2er)\}.$$

Proof For a set of hypotheses $\mathcal{J}$, denote by $\mathcal{J}_{|A}$ the restriction of each of its members to A, so that $\mathcal{J}_A \triangleq \{h_{|A} \mid h \in \mathcal{J}\}$. Since d_r is the VC-dimension of $\overline{\mathcal{H}}$, there exists a set of bags $S \subseteq \mathcal{X}^{(R)}$ of size d_r that is shattered by $\overline{\mathcal{H}}$, so that $|\overline{\mathcal{H}}_{|S}| = 2^{d_r}$. By Lemma 5 $|\overline{\mathcal{H}}_{|S}| \leq |\mathcal{H}_{|S^{\cup}}|$, therefore $2^{d_r} \leq |\mathcal{H}_{|S^{\cup}}|$. In addition, $R \subseteq [r]$ implies $|S^{\cup}| \leq rd_r$. By applying Sauer's lemma (Sauer, 1972; Vapnik and Chervonenkis, 1971) to $\mathcal{H}$ we get

$$2^{d_r} \leq |\mathcal{H}_{|S^{\cup}}| \leq \left(\frac{e|S^{\cup}|}{d}\right)^d \leq \left(\frac{erd_r}{d}\right)^d,$$

Where e is the base of the natural logarithm. It follows that $d_r \leq d(\log(er) - \log d) + d\log d_r$. To provide an explicit bound for d_r, we bound $d\log d_r$ by dividing to cases:

1. Either $d\log d_r \leq \frac{1}{2}d_r$, thus $d_r \leq 2d(\log(er) - \log d) \leq 2d\log(er)$,
2. or $\frac{1}{2}d_r < d\log d_r$. In this case,
   (a) either $d_r \leq 16$,
   (b) or $d_r > 16$. In this case $\sqrt{d_r} < d_r/\log d_r < 2d$, thus $d\log d_r = 2d\log\sqrt{d_r} \leq 2d\log 2d$. Substituting in the implicit bound we get $d_r \leq d(\log(er) - \log d) + 2d\log 2d \leq 2d\log(2er)$.

Combining the cases we have $d_r \leq \max\{16, 2d\log(2er)\}$. ■

3.2 VC-Dimension Lower Bounds

In this section we show lower bounds for the VC-dimension of binary MIL, indicating that the dependence on d and r in Theorem 6 is tight in two important settings.

We say that a bag-function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$ is *r-sensitive* if there exists a number $n \in R$ and a vector $\mathbf{c} \in \{-1,+1\}^n$ such that for at least r different numbers $j_1,\ldots,j_r \in [n]$, $\psi(c[1],\ldots,c[j_i],\ldots,c[n]) \neq \psi(c[1],\ldots,-c[j_i],\ldots,c[n])$. Many commonly used Boolean functions, such as OR, AND, Parity, and all their variants that stem from negating some of the inputs, are r-sensitive for every $r \in R$. Our first lower bound shows if ψ is r-sensitive, the bound in Theorem 6 cannot be improved without restricting the set of considered instance hypothesis classes.

Theorem 7 *Assume that the bag function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$ is r-sensitive for some $r \in \mathbb{N}$. For any natural d and any instance domain $\mathcal{X}$ with $|\mathcal{X}| \geq rd\lfloor\log(r)\rfloor$, there exists a hypothesis class $\mathcal{H}$ with a VC-dimension at most d, such that the VC dimension of $\overline{\mathcal{H}}$ is at least $d\lfloor\log(r)\rfloor$.*

Proof Since ψ is r-sensitive, there are a vector $\mathbf{c} \in \{-1,+1\}^n$ and a set $J \subseteq n$ such that $|J| = r$ and $\forall j \in J, \psi(c[1],\ldots,c[n]) \neq \psi(c[1],\ldots,-c[j],\ldots,c[n])$. Since ψ maps all inputs to $\{-1,+1\}$, it follows that $\forall j \in J, \psi(c[1],\ldots,-c[j],\ldots,c[n]) = -\psi(c[1],\ldots,c[n])$. Denote $a = \psi(c[1],\ldots,c[n])$. Then we have

$$\forall j \in J, y \in \{-1,+1\}, \quad \psi(c[1],\ldots,c[j]\cdot y,\ldots,c[n]) = a \cdot y. \tag{1}$$

For simplicity of notation, we henceforth assume w.l.o.g. that $n = r$ and $J = [r]$.

Let $S \subseteq \mathcal{X}^r$ be a set of $d\lfloor\log(r)\rfloor$ bags of size r, such that all the instances in all the bags are distinct elements of $\mathcal{X}$. Divide S into d mutually exclusive subsets, each with $\lfloor\log(r)\rfloor$ bags. Denote bag p in subset t by $\mathbf{x}_{(p,t)}$. We define the hypothesis class

$$\mathcal{H} \triangleq \{h[k_1,\ldots,k_d] \mid \forall i \in [d], k_i \in [2^{\lfloor\log(r)\rfloor}]\},$$

where $h[k_1,\ldots,k_d]$ is defined as follows (see illustration in Table 1): For $x \in \mathcal{X}$ which is not an instance of any bag in S, $h[k_1,\ldots,k_d] = -1$. For $x = x_{(p,t)}[j]$, let $b_{(p,n)}$ be bit p in the binary representation of the number n, and define

$$h[k_1,\ldots,k_d](x_{(p,t)}[j]) = \begin{cases} c[j]\cdot a(2b_{(p,j-1)}-1) & j = k_t, \\ c[j] & j \neq k_t. \end{cases}$$

We now show that S is shattered by $\overline{\mathcal{H}}$, indicating that the VC-dimension of $\overline{\mathcal{H}}$ is at least $|S| = d\lfloor\log(r)\rfloor$. To complete the proof, we further show that the VC-dimension of $\mathcal{H}$ is no more than d.

First, we show that S is shattered by $\overline{\mathcal{H}}$: Let $\{y_{(p,t)}\}_{p\in\lfloor\log(r)\rfloor, t\in[d]}$ be some labeling over $\{-1,+1\}$ for the bags in S. For each $t \in [d]$ let

$$k_t \triangleq 1 + \sum_{p=1}^{\lfloor\log(r)\rfloor} \frac{y_{(p,t)}+1}{2}\cdot 2^{p-1}.$$

Then by Equation (1), for all $p \in [\lfloor\log(r)\rfloor]$ and $t \in [d]$,

$$\begin{aligned} \overline{h}[k_1,\ldots,k_d](\mathbf{x}_{(p,t)}) &= \psi(c[1],\ldots,c[k_t]\cdot a(2b_{(p,k_t-1)}-1),\ldots,c[r]) \\ &= a^2(2b_{(p,k_t-1)}-1) = 2b_{(p,k_t-1)}-1 = y_{(p,t)}. \end{aligned}$$

t	p	Instance label $h(x_{(p,t)}[r])$								Bag label $\overline{h}(\mathbf{x}_i)$
1	1	−	−	−	+	−	−	−	−	+
	2	−	−	−	+	−	−	−	−	+
	3	−	−	−	−	−	−	−	−	−
2	1	−	−	−	−	−	−	−	+	+
	2	−	−	−	−	−	−	−	+	+
	3	−	−	−	−	−	−	−	+	+
3	1	−	−	−	−	−	−	−	−	−
	2	−	+	−	−	−	−	−	−	+
	3	−	−	−	−	−	−	−	−	−

Table 1: An example of the hypotheses $h = h[4,8,3]$, with $\psi = \text{OR}$ (so that $\mathbf{c}$ is the all -1 vector), $r = 8$, and $d = 3$. Each line represents a bag in S, each column represents an instance in the bag.

Thus $h[k_1,\ldots,k_d]$ labels S according to $\{y_{(p,t)}\}$.

Second, we show that the VC-dimension of $\mathcal{H}$ is no more than d. Let $A \subseteq \mathcal{X}$ of size $d+1$. If there is an element in A which is not an instance in S then this element is labeled -1 by all $h \in \mathcal{H}$, therefore A is not shattered. Otherwise, all elements in A are instances in bags in S. Since there are d subsets of S, there exist two elements in A which are instances of bags in the same subset t. Denote these instances by $x(p_1,t)[j_1]$ and $x(p_2,t)[j_2]$. Consider all the possible labelings of the two elements by hypotheses in $\mathcal{H}$. If A is shattered, there must be four possible labelings for these elements. However, by the definition of $h[k_1,\ldots,k_d]$ it is easy to see that if $j_1 = j_2 = j$ then there are at most two possible labelings by hypotheses in $\mathcal{H}$, and if $j_1 \neq j_2$ then there are at most three possible labelings. Thus A is not shattered by $\mathcal{H}$, hence the VC-dimension of $\mathcal{H}$ is no more than d. ■

Theorem 10 below provides a lower bound for the VC-dimension of MIL for the important case where the bag-function is the Boolean OR and the hypothesis class is the class of separating hyperplanes in $\mathbb{R}^n$. For $\mathbf{w} \in \mathbb{R}^n$, the function $h_{\mathbf{w}} : \mathbb{R}^n \to \{-1,+1\}$ is defined by $h_{\mathbf{w}}(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$. The hypothesis class of linear classifiers is $\mathcal{W}_n \triangleq \{h_{\mathbf{w}} \mid \mathbf{w} \in \mathbb{R}^n\}$. Let $r \in \mathbb{N}$. We denote the VC-dimension of $\overline{\mathcal{W}}_n$ for $R = \{r\}$ and $\psi = \text{OR}$ by $d_{r,n}$. We prove a lower bound for $d_{r,n}$ using two lemmas: Lemma 8 provides a lower bound for $d_{r,3}$, and Lemma 9 links $d_{r,n}$ for small n with $d_{r,n}$ for large n. The resulting general lower bound, which holds for $r = \max R$, is then stated in Theorem 10.

Lemma 8 *Let $d_{r,n}$ be the VC-dimension of $\overline{\mathcal{W}}_n$ as defined above. Then $d_{r,3} \geq \lfloor \log(2r) \rfloor$.*

Proof Denote $L \triangleq \lfloor \log(2r) \rfloor$. We will construct a set S of L bags of size r that is shattered by $\mathcal{W}_3$. The construction is illustrated in Figure 1.

Let $\mathbf{n} = (n_1,\ldots,n_K)$ be a sequence of indices from $[L]$, created by concatenating all the subsets of $[L]$ in some arbitrary order, so that $K = L2^{L-1}$, and every index appears $2^{L-1} \leq r$ times in $\mathbf{n}$. Define a set $A = \{\mathbf{a}_k \mid k \in [K]\} \subseteq \mathbb{R}^3$ where $\mathbf{a}_k \triangleq (\cos(2\pi k/K), \sin(2\pi k/K), 1) \in \mathbb{R}^3$, so that $\mathbf{a}_1,\ldots,\mathbf{a}_K$ are equidistant on a unit circle on a plane embedded in $\mathbb{R}^3$. Define the set of bags $S = \{\mathbf{x}_1,\ldots,\mathbf{x}_L\}$ such that $\mathbf{x}_i = (x_i[1],\ldots,x_i[r])$ where $\{x_i[j] \mid j \in [r]\} = \{a_k \mid n_k = i\}$.

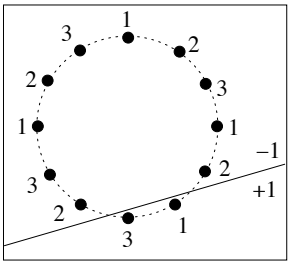


Figure 1: An illustration of the constructed shattered set, with $r = 4$ and $L = \log 4 + 1 = 3$. Each dot corresponds to an instance. The numbers next to the instances denote the bag to which an instance belongs, and match the sequence N defined in the proof. In this illustration bags 1 and 3 are labeled as positive by the bag-hypothesis represented by the solid line.

We now show that S is shattered by $\mathcal{W}_3$: Let $(y_1,\ldots,y_L)$ be some binary labeling of L bags, and let $Y = \{i \mid y_i = +1\}$. By the definition of $\mathbf{n}$, there exist j_1, j_2 such that $Y = \{n_k \mid j_1 \le k \le j_2\}$. Clearly, there exists a hyperplane $\mathbf{w} \in \mathbb{R}^3$ that separates the vectors $\{\mathbf{a}_k \mid j_1 \le k \le j_2\}$ from the rest of the vectors in A. Thus $\text{sign}(\langle \mathbf{w}, \mathbf{a}_k \rangle) = +1$ if and only if $j_1 \le k \le j_2$. It follows that $\overline{h}_{\mathbf{w}}(\mathbf{x}_i) = +1$ if and only if there is a $k \in \{j_1,\ldots,j_2\}$ such that $\mathbf{a}_k$ is an instance in $\mathbf{x}_i$, that is such that $n_k = i$. This condition holds if and only if $i \in Y$, hence $\overline{h}_{\mathbf{w}}$ classifies S according to the given labeling. It follows that S is shattered by $\mathcal{W}_3$, therefore $d_{r,3} \ge |S| = \lfloor \log(2r) \rfloor$. ■

Lemma 9 *Let k, n, r be natural number such that $k \le n$. Then $d_{r,n} \ge \lfloor n/k \rfloor d_{r,k}$.*

Proof For a vector $\mathbf{x} \in \mathbb{R}^k$ and a number $t \in \{0,\ldots,\lfloor n/k \rfloor\}$ define the vector $s(\mathbf{x},t) \triangleq (0,\ldots,0, x[1],\ldots,x[k],0,\ldots,0) \in \mathbb{R}^n$, where $x[1]$ is at coordinate $kt+1$. Similarly, for a bag $\mathbf{x}_i = (\mathbf{x}_i[1],\ldots,\mathbf{x}_i[r]) \in (\mathbb{R}^k)^r$, define the bag $s(\mathbf{x}_i,t) \triangleq (s(\mathbf{x}_i[1],t),\ldots,s(\mathbf{x}_i[r],t)) \in (\mathbb{R}^n)^r$.

Let $S_k = \{\mathbf{x}_i\}_{i \in [d_{r,k}]} \subseteq (\mathbb{R}^k)^r$ be a set of bags with instances in $\mathbb{R}^k$ that is shattered by $\overline{\mathcal{W}_k}$. Define S_n, a set of bags with instances in $\mathbb{R}^n$: $S_n \triangleq \{s(\mathbf{x}_i,t)]\}_{i \in [d_{r,k}], t \in [\lfloor n/k \rfloor]} \subseteq (\mathbb{R}^n)^r$. Then S_n is shattered by $\mathcal{W}_n$: Let $\{y_{(i,t)}\}_{i \in [d_{r,k}], t \in [\lfloor n/k \rfloor]}$ be some labeling for S_n. S_k is shattered by $\mathcal{W}_k$, hence there are separators $\mathbf{w}_1,\ldots,\mathbf{w}_{\lfloor n/k \rfloor} \in \mathbb{R}^k$ such that $\forall i \in [d_{r,k}], t \in \lfloor n/k \rfloor, \quad \overline{h}_{\mathbf{w}_t}(\mathbf{x}_i) = y_{(i,t)}$.

Set $\mathbf{w} \triangleq \sum_{t=0}^{\lfloor n/k \rfloor} s(\mathbf{w}_t, t)$. Then $\langle \mathbf{w}, s(\mathbf{x},t) \rangle = \langle \mathbf{w}_t, \mathbf{x} \rangle$. Therefore

$$\begin{aligned}\overline{h}_{\mathbf{w}}(s(\mathbf{x}_i,t)) &= \text{OR}(\text{sign}(\langle \mathbf{w}, s(\mathbf{x}_i[1],t) \rangle),\ldots,\text{sign}(\langle \mathbf{w}, s(\mathbf{x}_i[r],t) \rangle)) \\ &= \text{OR}(\text{sign}(\langle \mathbf{w}_t, \mathbf{x}_i[1] \rangle),\ldots,\text{sign}(\langle \mathbf{w}_t, \mathbf{x}_i[r] \rangle)) = \overline{h}_{\mathbf{w}_t}(\mathbf{x}_i) = y_{(i,t)}.\end{aligned}$$

S_n is thus shattered, hence $d_{r,n} \ge |S_n| = \lfloor n/k \rfloor d_{r,k}$. ■

The desired theorem is an immediate consequence of the two lemmas above, by noting that whenever $r \in R$, the VC-dimension of $\overline{\mathcal{W}_n}$ is at least $d_{r,n}$.

Theorem 10 *Let $\mathcal{W}_n$ be the class of separating hyperplanes in $\mathbb{R}^n$ as defined above. Assume that the bag function is $\psi = OR$ and the set of allowed bag sizes is R. Let $r = \max R$. Then the VC-dimension of $\overline{\mathcal{W}_n}$ is at least $\lfloor n/3 \rfloor \lfloor \log 2r \rfloor$.*

3.3 Pseudo-dimension for Thresholded Functions

In this section we consider binary hypothesis classes that are generated from real-valued functions using thresholds. Let $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ be a set of real valued functions. The binary hypothesis class of thresholded functions generated by $\mathcal{F}$ is $T_{\mathcal{F}} = \{(x,z) \mapsto \text{sign}(f(x)-z) \mid f \in \mathcal{F}, z \in \mathbb{R}\}$, where $x \in \mathcal{X}$ and $z \in \mathbb{R}$. The sample complexity of learning with $T_{\mathcal{F}}$ and the zero-one loss is governed by the pseudo-dimension of $\mathcal{F}$, which is equal to the VC-dimension of $T_{\mathcal{F}}$ (Pollard, 1984). In this section we consider a bag-labeling function $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$, and bound the pseudo-dimension of $\overline{\mathcal{F}}$, thus providing an upper bound on the sample complexity of binary MIL with $T_{\overline{\mathcal{F}}}$. The following bound holds for bag-labeling functions that extend monotone Boolean functions, defined in Definition 2.

Theorem 11 *Let $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ be a function class with pseudo-dimension d . Let $R \subseteq [r]$, and assume that $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ extends monotone Boolean functions. Let d_r be the pseudo-dimension of $\overline{\mathcal{F}}$. Then*

$$d_r \leq \max\{16, 2d\log(2er)\}.$$

Proof First, by Definition 2, we have that for any ψ which extends monotone Boolean functions, any $n \in R$ and any $\mathbf{y} \in \mathbb{R}^n$,

$$\begin{aligned} \text{sign}(\psi(y[1],\ldots,y[n]) - z) &= \text{sign}(\psi(y[1]-z,\ldots,y[n]-z)) \\ &= \psi(\text{sign}(y[1]-z,\ldots,y[n]-z)). \end{aligned} \tag{2}$$

This can be seen by noting that each of the equalities holds for each of the operations allowed by $\mathcal{M}_n$ for each n, thus by induction they hold for all functions in $\mathcal{M}_n$ and all combinations of them.

For a real-valued function f let $t_f : \mathcal{X} \times \mathbb{R} \to \{-1,+1\}$ be defined by $t_f(y,z) = \text{sign}(f(y)-z)$. We have $T_{\mathcal{F}} = \{t_f \mid f \in \mathcal{F}\}$, and $T_{\overline{\mathcal{F}}} = \{t_{\overline{f}} \mid f \in \mathcal{F}\}$. In addition, for all $f \in \mathcal{F}$, $z \in \mathbb{R}$, $n \in R$ and $\mathbf{x} \in \mathcal{X}^n$, we have

$$\begin{aligned} t_{\overline{f}}(\mathbf{x},z) &= \text{sign}(\overline{f}(\mathbf{x}) - z) = \text{sign}(\psi(f(x[1]),\ldots,f(x[n])) - z) \\ &= \psi(\text{sign}(f(x[1]) - z,\ldots,f(x[n]) - z)) \qquad (3) \\ &= \psi(t_f(x[1],z),\ldots,t_f(x[n],z)) = \overline{t_f}(\mathbf{x},z), \end{aligned}$$

where the equality on line (3) follows from Equation (2). Therefore

$$T_{\overline{\mathcal{F}}} = \{t_{\overline{f}} \mid f \in \mathcal{F}\} = \{\overline{t_f} \mid f \in \mathcal{F}\} = \{\overline{h} \mid h \in T_{\mathcal{F}}\} = \overline{T_{\mathcal{F}}}.$$

The VC-dimension of $T_{\mathcal{F}}$ is equal to the pseudo-dimension of $\mathcal{F}$, which is d. Thus, by Theorem 6 and the equality above, the VC-dimension of $T_{\overline{\mathcal{F}}}$ is bounded by $\max\{16, 2d\log(2er)\}$. The proof is completed by noting that d_r, the pseudo-dimension of $\overline{\mathcal{F}}$, is exactly the VC-dimension of $T_{\overline{\mathcal{F}}}$. ■

This concludes our results for distribution-free sample complexity of Binary MIL. In Section 6 we provide sample complexity analysis for distribution-dependent binary MIL, as a function of the average bag size.

4. Covering Numbers Bounds for MIL

Covering numbers are a useful measure of the complexity of a function class, since they allow bounding the sample complexity of a class in various settings, based on uniform convergence guarantees (see, e.g., Anthony and Bartlett, 1999). In this section we provide a lemma that relates the covering numbers of bag hypothesis classes with those of the underlying instance hypothesis class. We will use this lemma in subsequent sections to derive sample complexity upper bounds for additional settings of MIL. Let $\mathcal{F} \subseteq \mathbb{R}^A$ be a set of real-valued functions over some domain A. A γ-cover of $\mathcal{F}$ with respect to a norm $\|\cdot\|_\circ$ defined on functions is a set of functions $\mathcal{C} \subseteq \mathbb{R}^A$ such that for any $f \in \mathcal{F}$ there exists a $g \in \mathcal{C}$ such that $\|f-g\|_\circ \leq \gamma$. The *covering number* for given $\gamma > 0$, $\mathcal{F}$ and $\circ$, denoted by $\mathcal{N}(\gamma, \mathcal{F}, \circ)$, is the size of the smallest such γ-covering for $\mathcal{F}$.

Let $S \subseteq A$ be a finite set. We consider coverings with respect to the $L_p(S)$ norm for $p \geq 1$, defined by

$$\|f\|_{L_p(S)} \triangleq \left(\frac{1}{|S|} \sum_{s \in S} |f(s)|^p \right)^{1/p}.$$

For $p = \infty$, $L_\infty(S)$ is defined by $\|f\|_{L_\infty(S)} \triangleq \max_{s \in S} |f(S)|$. The covering number of $\mathcal{F}$ for a sample size m with respect to the L_p norm is

$$\mathcal{N}_m(\gamma, \mathcal{F}, p) \triangleq \sup_{S \subseteq A : |S| = m} \mathcal{N}(\gamma, \mathcal{F}, L_p(S)).$$

A small covering number for a function class implies faster uniform convergence rates, hence smaller sample complexity for learning. The following lemma bounds the covering number of bag hypothesis-classes whenever the bag function is Lipschitz with respect to the infinity norm (see Definition 4). Recall that all extensions of monotone Boolean functions (Definition 2) and all p-norm bag-functions (Definition 3) are 1-Lipschitz, thus the following lemma holds for them with $a = 1$.

Lemma 12 *Let $R \subseteq \mathbb{N}$ and suppose the bag function $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ is a-Lipschitz with respect to the infinity norm, for some $a > 0$. Let $S \subseteq \mathcal{X}^{(R)}$ be a finite set of bags, and let r be the average size of a bag in S. For any $\gamma > 0$, $p \in [1, \infty]$, and hypothesis class $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$,*

$$\mathcal{N}(\gamma, \overline{\mathcal{H}}, L_p(S)) \leq \mathcal{N}\left(\frac{\gamma}{a r^{1/p}}, \mathcal{H}, L_p(S^\cup)\right).$$

Proof First, note that by the Lipschitz condition on ψ, for any bag $\mathbf{x}$ of size n and hypotheses $h, g \in \mathcal{H}$,

$$|\overline{h}(\mathbf{x}) - \overline{g}(\mathbf{x})| = |\psi(h(x[1]), \ldots, h(x[n])) - \psi(g(x[1]), \ldots, g(x[n]))| \leq a \max_{x \in \mathbf{x}} |h(x) - g(x)|. \quad (4)$$

Let $\mathcal{C}$ be a minimal γ-cover of $\mathcal{H}$ with respect to the norm defined by $L_p(S^\cup)$, so that $|\mathcal{C}| = \mathcal{N}(\gamma, \mathcal{H}, L_p(S^\cup))$. For every $h \in \mathcal{H}$ there exists a $g \in \mathcal{C}$ such that $\|h - g\|_{L_p(S^\cup)} \leq \gamma$. Assume $p < \infty$.

Then by Equation (4)

$$\begin{aligned}
\|\overline{h}-\overline{g}\|_{L_p(S)} &= \left(\frac{1}{|S|}\sum_{\mathbf{x}\in S}|\overline{h}(\mathbf{x})-\overline{g}(\mathbf{x})|^p\right)^{1/p} \leq \left(\frac{a^p}{|S|}\sum_{\mathbf{x}\in S}\max_{x\in\mathbf{x}}|h(x)-g(x)|^p\right)^{1/p} \\
&\leq \left(\frac{a^p}{|S|}\sum_{\mathbf{x}\in S}\sum_{x\in\mathbf{x}}|h(x)-g(x)|^p\right)^{1/p} = \frac{a}{|S|^{1/p}}\left(\sum_{x\in S^\cup}|h(x)-g(x)|^p\right)^{1/p} \\
&= a\left(\frac{|S^\cup|}{|S|}\right)^{1/p}\left(\frac{1}{|S^\cup|}\sum_{x\in S^\cup}|h(x)-g(x)|^p\right)^{1/p} \\
&= ar^{1/p}\|h-g\|_{L_p(S^\cup)} \leq ar^{1/p}\cdot\gamma.
\end{aligned}$$

It follows that $\overline{\mathcal{C}}$ is a $(ar^{1/p}\gamma)$-covering for $\overline{\mathcal{H}}$. For $p=\infty$ we have

$$\begin{aligned}
\|\overline{h}-\overline{g}\|_{L_\infty(S)} &= \max_{\mathbf{x}\in S}|\overline{h}(\mathbf{x})-\overline{g}(\mathbf{x})| \leq a\max_{\mathbf{x}\in S}\max_{x\in\mathbf{x}}|h(x)-g(x)| \\
&= a\max_{x\in S^\cup}|h(x)-g(x)| = a\|h-g\|_{L_\infty(S^\cup)} \leq a\gamma = a\cdot r^{1/p}\cdot\gamma.
\end{aligned}$$

Thus in both cases, $\overline{\mathcal{C}}$ is a $ar^{1/p}\gamma$-covering for $\overline{\mathcal{H}}$, and its size is $\mathcal{N}(\gamma,\mathcal{H},L_p(S^\cup))$. Thus

$$\mathcal{N}(ar^{1/p}\gamma,\overline{\mathcal{H}},L_p(S^\cup)) \leq \mathcal{N}(\gamma,\overline{\mathcal{H}},L_p(S^\cup)).$$

We get the statement of the lemma by substituting γ with $\frac{\gamma}{ar^{1/p}}$. ■

As an immediate corollary, we have the following bound for covering numbers of a given sample size.

Corollary 13 *Let $r\in\mathbb{N}$, and let $R\subseteq[r]$. Suppose the bag function $\psi:\mathbb{R}^{(R)}\to\mathbb{R}$ is a-Lipschitz with respect to the infinity norm for some $a>0$. Let $\gamma>0, p\in[1,\infty]$, and $\mathcal{H}\in\mathbb{R}^{\mathcal{X}}$. For any $m\geq 0$,*

$$\mathcal{N}_m(\gamma,\overline{\mathcal{H}},p) \leq \mathcal{N}_{rm}\left(\frac{\gamma}{a\cdot r^{1/p}},\mathcal{H},p\right).$$

5. Margin Learning for MIL: Fat-Shattering Dimension

Large-margin classification is a popular supervised learning approach, which has received attention also as a method for MIL. For instance, MI-SVM (Andrews et al., 2002) attempts to optimize an adaptation of the soft-margin SVM objective (Cortes and Vapnik, 1995) to MIL, in which the margin of a bag is the maximal margin achieved by any of its instances. It has not been shown, however, whether minimizing the objective function of MI-SVM, or other margin formulations for MIL, allows learning with a reasonable sample size. We fill in this gap in Theorem 14 below, which bounds the γ-fat-shattering dimension (see, e.g., Anthony and Bartlett 1999) of MIL. The objective of MI-SVM amounts to replacing the hypothesis class $\mathcal{H}$ of separating hyperplanes with the class of bag-hypotheses $\overline{\mathcal{H}}$ where the bag function is $\psi=\max$. Since max is the real-valued extension of OR, this objective function is natural in our MIL formulation. The distribution-free sample complexity of large-margin learning with the zero-one loss is proportional to the fat-shattering dimension (Alon et al., 1997). Thus, we provide an upper bound on the fat-shattering dimension

of MIL as a function of the fat-shattering dimension of the underlying hypothesis class, and of the maximal allowed bag size. The bound holds for any Lipschitz bag-function. Let $\gamma > 0$ be the desired margin. For a hypothesis class H, denote its γ-fat-shattering dimension by $\mathrm{Fat}(\gamma, H)$

Theorem 14 *Let $r \in \mathbb{N}$ and assume $R \subseteq [r]$. Let $B, a > 0$. Let $\mathcal{H} \subseteq [0,B]^{\mathcal{X}}$ be a real-valued hypothesis class and assume that the bag function $\psi : [0,B]^{(R)} \to [0,aB]$ is a-lipschitz with respect to the infinity norm. Then for all $\gamma \in (0, aB]$*

$$\mathrm{Fat}(\gamma, \overline{\mathcal{H}}) \leq \max\left\{33,\ 24\mathrm{Fat}(\frac{\gamma}{64a}, \mathcal{H}) \log^2\left(\frac{6 \cdot 2048 \cdot B^2 a^2}{\gamma^2} \cdot \mathrm{Fat}(\frac{\gamma}{64a}, \mathcal{H}) \cdot r\right)\right\}. \tag{5}$$

This theorem shows that for margin learning as well, the dependence of the bag size on the sample complexity is poly-logarithmic. In the proof of the theorem we use the following two results, which link the covering number of a function class with its fat-shattering dimension.

Theorem 15 (Bartlett et al., 1997) *Let F be a set of real-valued functions and let $\gamma > 0$. For $m \geq \mathrm{Fat}(16\gamma, F)$,*

$$e^{\mathrm{Fat}(16\gamma, F)/8} \leq \mathcal{N}_m(\gamma, F, \infty).$$

The following theorem is due to Anthony and Bartlett (1999) (Theorem 12.8), following Alon et al. (1993).

Theorem 16 *Let F be a set of real-valued functions with range in $[0,B]$. Let $\gamma > 0$. For all $m \geq 1$,*

$$\mathcal{N}_m(\gamma, F, \infty) < 2\left(\frac{4B^2 m}{\gamma^2}\right)^{\mathrm{Fat}(\frac{\gamma}{4}, F)\log(4eBm/\gamma)}. \tag{6}$$

Theorem 12.8 in Anthony and Bartlett (1999) deals with the case $m \geq \mathrm{Fat}(\frac{\gamma}{4}, F)$. Here we only require $m \geq 1$, since if $m \leq \mathrm{Fat}(\frac{\gamma}{4})$ then the trivial upper bound $\mathcal{N}_m(\gamma, \mathcal{H}, \infty) \leq (B/\gamma)^m \leq (B/\gamma)^{\mathrm{Fat}(\frac{\gamma}{4})}$ implies Equation (6).

Proof [of Theorem 14] From Theorem 15 and Lemma 12 it follows that for $m \geq \mathrm{Fat}(16\gamma, \overline{\mathcal{H}})$,

$$\mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) \leq \frac{8}{\log e} \log \mathcal{N}_m(\gamma, \overline{\mathcal{H}}, \infty) \leq 6 \log \mathcal{N}_{rm}(\gamma/a, \mathcal{H}, \infty). \tag{7}$$

By Theorem 16, for all $m \geq 1$, if $\mathrm{Fat}(\gamma/4) \geq 1$ then

$$\begin{aligned}
\forall \gamma \leq \frac{B}{2e}, \quad \log \mathcal{N}_m(\gamma, \mathcal{H}, \infty) &\leq 1 + \mathrm{Fat}(\frac{\gamma}{4}, \mathcal{H}) \log(\frac{4eBm}{\gamma}) \log\left(\frac{4B^2 m}{\gamma^2}\right) \\
&\leq \mathrm{Fat}(\frac{\gamma}{4}, \mathcal{H}) \log(\frac{8eBm}{\gamma}) \log\left(\frac{4B^2 m}{\gamma^2}\right) \qquad (8) \\
&\leq \mathrm{Fat}(\frac{\gamma}{4}, \mathcal{H}) \log^2(\frac{4B^2 m}{\gamma^2}). \qquad (9)
\end{aligned}$$

The inequality in line (8) holds since we have added 1 to the second factor, and the value of the other factors is at least 1. The last inequality follows since if $\gamma \leq \frac{B}{2e}$, we have $8eB/\gamma \leq 4B^2/\gamma^2$. Equation (9) also holds if $\mathrm{Fat}(\gamma/4) < 1$, since this implies $\mathrm{Fat}(\gamma/4) = 0$ and $\mathcal{N}_m(\gamma, \mathcal{H}, \infty) = 1$. Combining Equation (7) and Equation (9), we get that if $m \geq \mathrm{Fat}(16\gamma, \overline{\mathcal{H}})$ then

$$\forall \gamma \leq \frac{aB}{2e}, \quad \mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) \leq 6\mathrm{Fat}(\frac{\gamma}{4a}, \mathcal{H}) \log^2(\frac{4B^2 a^2 rm}{\gamma^2}). \tag{10}$$

Set $m = \lceil \mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) \rceil \leq \mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) + 1$. If $\mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) \geq 1$, we have that $m \geq \mathrm{Fat}(16\gamma, \overline{\mathcal{H}})$ and also $m \leq 2\mathrm{Fat}(16\gamma, \overline{\mathcal{H}})$. Thus Equation (10) holds, and

$$\forall \gamma \leq \frac{aB}{2e}, \quad \mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) \leq 6\mathrm{Fat}(\frac{\gamma}{4a}, \mathcal{H}) \log^2(\frac{4B^2a^2}{\gamma^2} \cdot r \cdot (\mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) + 1))$$
$$\leq 6\mathrm{Fat}(\frac{\gamma}{4a}, \mathcal{H}) \log^2(\frac{8B^2a^2}{\gamma^2} \cdot r \cdot \mathrm{Fat}(16\gamma, \overline{\mathcal{H}})).$$

Now, it is easy to see that if $\mathrm{Fat}(16\gamma, \overline{\mathcal{H}}) < 1$, this inequality also holds. Therefore it holds in general. Substituting γ with $\gamma/16$, we have that

$$\forall \gamma \leq \frac{8aB}{e}, \quad \mathrm{Fat}(\gamma, \overline{\mathcal{H}}) \leq 6\mathrm{Fat}(\frac{\gamma}{64a}, \mathcal{H}) \log^2(\frac{2048B^2a^2}{\gamma^2} \cdot r \cdot \mathrm{Fat}(\gamma, \overline{\mathcal{H}})). \tag{11}$$

Note that the condition on γ holds, in particular, for all $\gamma \leq aB$.

To derive the desired Equation (5) from Equation (11), let $\beta = 6\mathrm{Fat}(\gamma/64a, \mathcal{H})$ and $\eta = 2048B^2a^2/\gamma^2$. Denote $F = \mathrm{Fat}(\gamma, \overline{\mathcal{H}})$. Then Equation (11) can be restated as $F \leq \beta \log^2(\eta r F)$. It follows that $\sqrt{F}/\log(\eta r F) \leq \sqrt{\beta}$, Thus

$$\frac{\sqrt{F}}{\log(\eta r F)} \log\left(\frac{\sqrt{\eta r F}}{\log(\eta r F)}\right) \leq \sqrt{\beta} \log(\sqrt{\beta \eta r}).$$

Therefore

$$\frac{\sqrt{F}}{\log(\eta r F)} (\log(\eta r F)/2 - \log(\log(\eta r F))) \leq \sqrt{\beta} \log(\beta \eta r)/2,$$

hence

$$(1 - \frac{2\log(\log(\eta r F))}{\log(\eta r F)}) \sqrt{F} \leq \sqrt{\beta} \log(\beta \eta r).$$

Now, it is easy to verify that $\log(\log(x))/\log(x) \leq \frac{1}{4}$ for all $x \geq 33 \cdot 2048$. Assume $F \geq 33$ and $\gamma \leq aB$. Then

$$\eta r F = 2048B^2a^2 r F/\gamma^2 \geq 2048F \geq 33 \cdot 2048.$$

Therefore $\log(\log(\eta r F))/\log(\eta r F) \leq \frac{1}{4}$, which implies $\frac{1}{2}\sqrt{F} \leq \sqrt{\beta} \log(\beta \eta r)$. Thus $F \leq 4\beta \log^2(\beta \eta r)$. Substituting the parameters with their values, we get the desired bound, stated in Equation (5). ■

6. Sample Complexity by Average Bag Size: Rademacher Complexity

The upper bounds we have shown so far provide distribution-free sample complexity bounds, which depend only on the maximal possible bag size. In this section we show that even if the bag size is unbounded, we can still have a sample complexity guarantee, if the *average* bag size for the input distribution is bounded. For this analysis we use the notion of Rademacher complexity (Bartlett and Mendelson, 2002). Let A be some domain. The empirical Rademacher complexity of a class of functions $\mathcal{F} \subseteq \mathbb{R}^{A \times \{-1,+1\}}$ with respect to a sample $S = \{(x_i, y_i)\}_{i \in [m]} \subseteq A \times \{-1, +1\}$ is

$$\mathcal{R}(\mathcal{F}, S) \triangleq \frac{1}{m} \mathbb{E}_\sigma[|\sup_{f \in \mathcal{F}} \sum_{i \in [m]} \sigma_i f(x_i, y_i)|],$$

where $\sigma = (\sigma_1, \ldots, \sigma_m)$ are m independent uniform $\{\pm 1\}$-valued variables. The average Rademacher complexity of $\mathcal{F}$ with respect to a distribution D over $A \times \{-1,+1\}$ and a sample size m is

$$\mathcal{R}_m(\mathcal{F}, D) \triangleq \mathbb{E}_{S \sim D^m}[\mathcal{R}(\mathcal{F}, S)].$$

The worst-case Rademacher complexity over samples of size m is

$$\mathcal{R}_m^{\sup}(\mathcal{F}) = \sup_{S \subseteq A^m} \mathcal{R}(\mathcal{F}, S).$$

This quantity can be tied to the fat-shattering dimension via the following result:

Theorem 17 (See, for example, Mendelson, 2002, Theorem 4.11) *Let $m \geq 1$ and $\gamma \geq 0$. If $\mathcal{R}_m^{\sup}(\mathcal{F}) \leq \gamma$ then the γ-fat-shattering dimension of $\mathcal{F}$ is at most m.*

Let $I \subseteq \mathbb{R}$. Assume a hypothesis class $H \subseteq I^A$ and a loss function $\ell : \{-1,+1\} \times I \to \mathbb{R}$. For a hypothesis $h \in H$, we denote by h_ℓ the function defined by $h_\ell(x,y) = \ell(y, h(x))$. Given H and ℓ, we define the function class $H_\ell \triangleq \{h_\ell \mid h \in H\} \subseteq \mathbb{R}^{A \times \{-1,+1\}}$.

Rademacher complexities can be used to derive sample complexity bounds (Bartlett and Mendelson, 2002): Assume the range of the loss function is $[0,1]$. For any $\delta \in (0,1)$, with probability of $1-\delta$ over the draw of samples $S \subseteq A \times \{-1,+1\}$ of size m drawn from D, every $h \in H$ satisfies

$$\ell(h, D) \leq \ell(h, S) + 2\mathcal{R}_m(H_\ell, D) + \sqrt{\frac{8 \ln(2/\delta)}{m}}. \tag{12}$$

Thus, an upper bound on the Rademacher complexity implies an upper bound on the average loss of a classifier learned from a random sample.

6.1 Binary MIL

Our first result complements the distribution-free sample complexity bounds that were provided for binary MIL in Section 3. The average (or expected) bag size under a distribution D over $\mathcal{X}^{(R)} \times \{-1,+1\}$ is $\mathbb{E}_{(\mathbf{X},Y) \sim D}[|\mathbf{X}|]$. Our sample complexity bound for binary MIL depends on the average bag size and the VC dimension of the instance hypothesis class. Recall that the zero-one loss is defined by $\ell_{0/1}(y, \hat{y}) = \mathbb{1}[y \neq \hat{y}]$. For a sample of labeled examples $S = \{(x_i, y_i)\}_{i \in [m]}$, we use S_X to denote the examples of S, that is $S_X = \{x_i\}_{i \in [m]}$.

Theorem 18 *Let $\mathcal{H} \subseteq \{-1,+1\}^{\mathcal{X}}$ be a binary hypothesis class with VC-dimension d. Let $R \subseteq \mathbb{N}$ and assume a bag function $\psi : \{-1,+1\}^{(R)} \to \{-1,+1\}$. Let r be the average bag size under distribution D over labeled bags. Then*

$$\mathcal{R}(\overline{\mathcal{H}}_{\ell_{0/1}}, D) \leq 17 \sqrt{\frac{d \ln(4er)}{m}}.$$

Proof Let S be a labeled bag-sample of size m. Dudley's entropy integral (Dudley, 1967) states that

$$\begin{aligned} \mathcal{R}(\overline{\mathcal{H}}_{\ell_{0/1}}, S) &\leq \frac{12}{\sqrt{m}} \int_0^\infty \sqrt{\ln \mathcal{N}(\gamma, \overline{\mathcal{H}}_{\ell_{0/1}}, L_2(S))}\, d\gamma \\ &= \frac{12}{\sqrt{m}} \int_0^1 \sqrt{\ln \mathcal{N}(\gamma, \overline{\mathcal{H}}_{\ell_{0/1}}, L_2(S))}\, d\gamma. \end{aligned} \tag{13}$$

The second equality holds since for any $\gamma > 1$, $\mathcal{N}(\gamma, \overline{\mathcal{H}}_{\ell_{0/1}}, L_2(S)) = 1$, thus the expression in the integral is zero.

If $\mathcal{C}$ is a γ-cover for $\overline{\mathcal{H}}$ with respect to the norm $L_2(S_X)$, then $\mathcal{C}_{\ell_{0/1}}$ is a $\gamma/2$-cover for $\overline{\mathcal{H}}_{\ell_{0/1}}$ with respect to the norm $L_2(S)$. This can be seen as follows: Let $h_{\ell_{0/1}} \in \overline{\mathcal{H}}_{\ell_{0/1}}$ for some $h \in \overline{\mathcal{H}}$. Let $f \in \mathcal{C}$ such that $\|f-h\|_{L_2(S_X)} \leq \gamma$. We have

$$\begin{aligned}
\|f_{\ell_{0/1}} - h_{\ell_{0/1}}\|_{L_2(S)} &= \left(\frac{1}{m}\sum_{(x,y)\in S} |f_{\ell_{0/1}}(x,y) - h_{\ell_{0/1}}(x,y)|^2\right)^{1/2} \\
&= \left(\frac{1}{m}\sum_{(x,y)\in S} |\ell_{0/1}(y, f(x)) - \ell_{0/1}(y, h(x))|^2\right)^{1/2} \\
&= \left(\frac{1}{m}\sum_{x\in S_X} (\frac{1}{2}|f(x) - h(x)|)^2\right)^{1/2} = \frac{1}{2}\|f-h\|_{L_2(S_X)} \leq \gamma/2.
\end{aligned}$$

Therefore $\mathcal{C}_{\ell_{0/1}}$ is a $\gamma/2$-cover for $L_2(S)$. It follows that we can bound the γ-covering number of $\overline{\mathcal{H}}_{\ell_{0/1}}$ by:

$$\mathcal{N}(\gamma, \overline{\mathcal{H}}_{\ell_{0/1}}, L_2(S)) \leq \mathcal{N}(2\gamma, \overline{\mathcal{H}}, L_2(S_X)). \tag{14}$$

Let $r(S)$ be the average bag size in the sample S, that is $r(S) = |S^{\cup}|/|S|$. By Lemma 12,

$$\mathcal{N}(\gamma, \overline{\mathcal{H}}, L_2(S_X)) \leq \mathcal{N}(\gamma/\sqrt{r(S)}, \mathcal{H}, L_2(S_X^{\cup})). \tag{15}$$

From Equation (13), Equation (14) and Equation (15) we conclude that

$$\mathcal{R}(\overline{\mathcal{H}}_{\ell_{0/1}}, S) \leq \frac{12}{\sqrt{m}} \int_0^1 \sqrt{\ln \mathcal{N}(2\gamma/\sqrt{r(S)}, \mathcal{H}, L_2(S_X^{\cup}))}\, d\gamma.$$

By Dudley (1978), for any $\mathcal{H}$ with VC-dimension d, and any $\gamma > 0$,

$$\ln \mathcal{N}(\gamma, \mathcal{H}, L_2(S_X^{\cup})) \leq 2d \ln\left(\frac{4e}{\gamma^2}\right).$$

Therefore

$$\begin{aligned}
\mathcal{R}(\overline{\mathcal{H}}_{\ell_{0/1}}, S) &\leq \frac{12}{\sqrt{m}} \int_0^1 \sqrt{2d \ln\left(\frac{er(S)}{\gamma^2}\right)}\, d\gamma \\
&\leq 17\sqrt{\frac{d}{m}}\left(\int_0^1 \sqrt{\ln(er(S))}\, d\gamma + \int_0^1 \sqrt{\ln(1/\gamma^2)}\, d\gamma\right) \\
&= 17\sqrt{\frac{d(\ln(er(S)) + \sqrt{\pi/2})}{m}} \leq 17\sqrt{\frac{d\ln(4er(S))}{m}}.
\end{aligned}$$

The function $\sqrt{\ln(x)}$ is concave for $x \geq 1$. Therefore we may take the expectation of both sides of this inequality and apply Jensen's inequality, to get

$$\begin{aligned}
\mathcal{R}_m(\overline{\mathcal{H}}_{\ell_{0/1}}, D) = \mathbb{E}_{S\sim D^m}[\mathcal{R}(\overline{\mathcal{H}}_{\ell_{0/1}}, S)] &\leq \mathbb{E}_{S\sim D^m}\left[17\sqrt{\frac{d\ln(4er(S))}{m}}\right] \\
&\leq 17\sqrt{\frac{d\ln(4e\cdot \mathbb{E}_{S\sim D^m}[r(S)])}{m}} = 17\sqrt{\frac{d\ln(4er)}{m}}.
\end{aligned}$$

■

We conclude that even when the bag size is not bounded, the sample complexity of binary MIL with a specific distribution depends only logarithmically on the average bag size in this distribution, and linearly on the VC-dimension of the underlying instance hypothesis class.

6.2 Real-Valued Hypothesis Classes

In our second result we wish to bound the sample complexity of MIL when using other loss functions that accept real valued predictions. This bound will depend on the average bag size, and on the Rademacher complexity of the instance hypothesis class.

We consider the case where both the bag function and the loss function are Lipschitz. For the bag function, recall that all extensions of monotone Boolean functions are Lipschitz with respect to the infinity norm. For the loss function $\ell : \{-1,+1\} \times \mathbb{R} \to \mathbb{R}$, we require that it is Lipschitz in its second argument, that is, that there is a constant $a > 0$ such that for all $y \in \{-1,+1\}$ and $y_1, y_2 \in \mathbb{R}$, $|\ell(y,y_1) - \ell(y,y_2)| \leq a|y_1 - y_2|$. This property is satisfied by many popular losses. For instance, consider the hinge-loss, which is the loss minimized by soft-margin SVM. It is defined as $\ell_{hl}(y,\hat{y}) = [1 - y\hat{y}]_+$, and is 1-Lipschitz in its second argument.

The following lemma provides a bound on the empirical Rademacher complexity of MIL, as a function of the average bag size in the sample and of the behavior of the worst-case Rademacher complexity over instances. We will subsequently use this bound to bound the average Rademacher complexity of MIL with respect to a distribution. We consider losses with the range $[0,1]$. To avoid degenerate cases, we consider only losses such that there exists at least one labeled bag $(\mathbf{x},y) \subseteq \mathcal{X}^{(R)} \times \{-1,+1\}$ and hypotheses $h, g \in \mathcal{H}$ such that $h_\ell(\mathbf{x},y) = 0$ and $g_\ell(\mathbf{x},y) = 1$. We say that such a loss has a *full range*.

Lemma 19 *Let $\mathcal{H} \subseteq [0,B]^{\mathcal{X}}$ be a hypothesis class. Let $R \subseteq \mathbb{N}$, and let the bag function $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ be a_1-Lipschitz with respect to the infinity norm. Assume a loss function $\ell : \{-1,+1\} \times \mathbb{R} \to [0,1]$, which is a_2-Lipschitz in its second argument. Further assume that ℓ has a full range. Suppose there is a continuous decreasing function $f : (0,1] \to \mathbb{R}$ such that*

$$\forall \gamma \in (0,1], \quad f(\gamma) \in \mathbb{N} \implies \mathcal{R}^{\sup}_{f(\gamma)}(\mathcal{H}) \leq \gamma.$$

Let S be a labeled bag-sample of size m, with an average bag size r. Then for all $\varepsilon \in (0,1]$,

$$\mathcal{R}(\overline{\mathcal{H}}_\ell, S) \leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2a_2^2B^2rm}{\varepsilon^2}\right)\left(1 + \int_\varepsilon^1 \sqrt{f(\frac{\gamma}{4a_1a_2})}\, d\gamma\right).$$

Proof A refinement of Dudley's entropy integral (Srebro et al., 2010, Lemma A.3) states that for all $\varepsilon \in (0,1]$, for all real function classes $\mathcal{F}$ with range $[0,1]$ and for all sets S,

$$\mathcal{R}(\mathcal{F}, S) \leq 4\varepsilon + \frac{10}{\sqrt{m}} \int_\varepsilon^1 \sqrt{\ln \mathcal{N}(\gamma, \mathcal{F}, L_2(S))}\, d\gamma. \tag{16}$$

Since the range of ℓ is $[0,1]$, this holds for $\mathcal{F} = \overline{\mathcal{H}}_\ell$. In addition, for any set S, the $L_2(S)$ norm is bounded from above by the $L_\infty(S)$ norm. Therefore $\mathcal{N}(\gamma, \mathcal{F}, L_2(S)) \leq \mathcal{N}(\gamma, \mathcal{F}, L_\infty(S))$. Thus, by Equation (16) we have

$$\mathcal{R}(\overline{\mathcal{H}}_\ell, S) \leq 4\varepsilon + \frac{10}{\sqrt{m}} \int_\varepsilon^1 \sqrt{\ln \mathcal{N}(\gamma, \overline{\mathcal{H}}_\ell, L_\infty(S))}\, d\gamma. \tag{17}$$

Now, let $h, g \in \mathcal{H}$ and consider $\overline{h}_\ell, \overline{g}_\ell \in \overline{\mathcal{H}_\ell}$. Since ℓ is a_2-Lipschitz, we have

$$\begin{aligned}\|\overline{h}_\ell - \overline{g}_\ell\|_{L_\infty(S)} &= \max_{i\in[m]} |\overline{h}_\ell(\mathbf{x}_i, y_i) - \overline{g}_\ell(\mathbf{x}_i, y_i)| = \max_{i\in[m]} |\ell(y_i, \overline{h}(\mathbf{x}_i)) - \ell(y_i, \overline{g}(\mathbf{x}_i))| \\ &\leq a_2 \max_{i\in[m]} |\overline{h}(\mathbf{x}_i) - \overline{g}(\mathbf{x}_i)| = a_2 \|\overline{h} - \overline{g}\|_{L_\infty(S_X)}.\end{aligned}$$

It follows that if $\mathcal{C} \subseteq \overline{\mathcal{H}}$ is a γ/a_2-cover for $\overline{\mathcal{H}}$ then $\mathcal{C}_\ell \subseteq \overline{\mathcal{H}_\ell}$ is a γ-cover for $\overline{\mathcal{H}_\ell}$. Therefore $\mathcal{N}(\gamma, \overline{\mathcal{H}_\ell}, L_\infty(S)) \leq \mathcal{N}(\gamma/a_2, \overline{\mathcal{H}}, L_\infty(S_X))$. By Lemma 12,

$$\mathcal{N}(\gamma/a_2, \overline{\mathcal{H}}, L_\infty(S_X)) \leq \mathcal{N}(\gamma/a_1 a_2, \mathcal{H}, L_\infty(S_X^{\cup})) \leq \mathcal{N}_{fm}(\gamma/a_1 a_2, \mathcal{H}, \infty).$$

Combining this with Equation (17) it follows that

$$\mathcal{R}(\overline{\mathcal{H}_\ell}, S) \leq 4\varepsilon + \frac{10}{\sqrt{m}} \int_\varepsilon^1 \sqrt{\mathcal{N}_{fm}(\gamma/a_1 a_2, \mathcal{H}, \infty)}\, d\gamma. \tag{18}$$

Now, let $\gamma \in (0,1]$, and let $\gamma_\circ = \sup\{\gamma_\circ \leq \gamma \mid f(\gamma_\circ) \in \mathbb{N}\}$. Since $\mathcal{R}^{\sup}_{f(\gamma_\circ)}(\mathcal{H}) \leq \gamma_\circ$, by Theorem 17 the $\gamma_\circ$-fat-shattering dimension of $\mathcal{H}$ is at most $f(\gamma_\circ)$. It follows that

$$\mathrm{Fat}(\gamma, \mathcal{H}) \leq \mathrm{Fat}(\gamma_\circ, \mathcal{H}) \leq f(\gamma_\circ) \leq 1 + f(\gamma).$$

The last inequality follows from the definition of $\gamma_\circ$, since f is continuous and decreasing. Therefore, by Theorem 16,

$$\begin{aligned}\forall \gamma \leq B, \quad \log \mathcal{N}_m(\gamma, \mathcal{H}, \infty) &\leq 1 + (f(\frac{\gamma}{4}) + 1) \log(\frac{4eBm}{\gamma}) \log\left(\frac{4B^2 m}{\gamma^2}\right) \\ &\leq (f(\frac{\gamma}{4}) + 1) \log(\frac{4eBm}{\gamma}) \log\left(\frac{4eB^2 m}{\gamma^2}\right) \qquad (19) \\ &\leq (f(\frac{\gamma}{4}) + 1) \log^2(\frac{4eB^2 m}{\gamma^2}). \qquad (20)\end{aligned}$$

The inequality in line (19) holds since we have added $\log(e) \geq 1$ to the third factor, and the value of the other factors is at least 1. The last inequality follows since $\gamma \leq B$.

We now show that the assumption $\gamma \leq B$ does not restrict us: By the assumptions on ℓ, there are $h, g \in \mathcal{H}$ and a labeled bag $(\mathbf{x}, y)$ such that $\overline{h}_\ell(\mathbf{x}, y) = 1$ and $\overline{g}_\ell(\mathbf{x}, y) = 0$. Let $n = |\mathbf{x}|$. By the Lipschitz assumptions we have

$$\begin{aligned}1 &= |\overline{h}_\ell(\mathbf{x}, y) - \overline{g}_\ell(\mathbf{x}, y)| = |\ell(y, \overline{h}(\mathbf{x})) - \ell(y, \overline{g}(\mathbf{x}))| \leq a_2 |\overline{h}(\mathbf{x}) - \overline{g}(\mathbf{x})| \\ &= a_2 |\psi(h(x[1]), \ldots, h(x[n])) - \psi(g(x[1]), \ldots, g(x[n]))| \leq a_2 a_1 \max_{j\in[n]} |h(x[j]) - g(x[j])| \leq a_1 a_2 B.\end{aligned}$$

Thus $1 \leq a_1 a_2 B$. It follows that for all $\gamma \in (0,1]$, $\gamma/a_1 a_2 \leq B$. Thus Equation (20) can be combined with Equation (18) to get that for all $\varepsilon \in (0,1]$,

$$\begin{aligned}\mathcal{R}(\overline{\mathcal{H}_\ell}, S) &\leq 4\varepsilon + \frac{10}{\sqrt{m}} \int_\varepsilon^1 \sqrt{\left(f(\frac{\gamma}{4a_1 a_2}) + 1\right) \log^2\left(\frac{4ea_1^2 a_2^2 B^2 rm}{\gamma^2}\right)}\, d\gamma \\ &\leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2 a_2^2 B^2 rm}{\varepsilon^2}\right) \int_\varepsilon^1 \sqrt{f(\frac{\gamma}{4a_1 a_2}) + 1}\, d\gamma \\ &\leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2 a_2^2 B^2 rm}{\varepsilon^2}\right) \left(1 + \int_\varepsilon^1 \sqrt{f(\frac{\gamma}{4a_1 a_2})}\, d\gamma\right).\end{aligned}$$

The last inequality follows from the fact that $\sqrt{a+b} \leq \sqrt{a}+\sqrt{b}$ for non-negative a and b, and from $\int_{\varepsilon}^{1} 1 \leq 1$. ■

Based on Lemma 19, we will now bound the average Rademacher complexity of MIL, as a function of the worst-case Rademacher complexity over instances, and the expected bag size. Since the number of instances in a bag sample of a certain size is not fixed, but depends on the bag sizes in the specific sample, we will need to consider the behavior of $\mathcal{R}_m^{\sup}(\mathcal{H})$ for different values of m. For many learnable function classes, the Rademacher complexity is proportional to $\frac{1}{\sqrt{m}}$, or to $\frac{ln^{\beta}(m)}{\sqrt{m}}$ for some non-negative β. The following theorem bounds the average Rademacher complexity of MIL in all these cases. The resulting bound indicates that here too there is a poly-logarithmic dependence of the sample complexity on the average bag size. Following the proof we show an application of the bound to a specific function class.

Theorem 20 *Let $\mathcal{H} \subseteq [0,B]^{\mathcal{X}}$ be a hypothesis class. Let $R \subseteq \mathbb{N}$, and let the bag function $\psi : \mathbb{R}^{(R)} \to \mathbb{R}$ be a_1-Lipschitz with respect to the infinity norm. Assume a loss function $\ell : \{-1,+1\} \times \mathbb{R} \to [0,1]$, which is a_2-Lipschitz in its second argument. Further assume that ℓ has a full range. Suppose that there are $C, \beta, K \geq 0$ such that for all $m \geq K$,*

$$\mathcal{R}_m^{\sup}(\mathcal{H}) \leq \frac{C \ln^{\beta}(m)}{\sqrt{m}}.$$

Then there exists a number $N \geq 0$ that depends only on C, β and K such that for any distribution D with average bag size r, and for all $m \geq 1$,

$$\mathcal{R}_m(\overline{\mathcal{H}}_{\ell}, D) \leq \frac{4 + 10\log(4ea_1^2a_2^2B^2rm^2)\left(N + \frac{a_1a_2}{\beta+1}C\ln^{\beta+1}(16a_1^2a_2^2m)\right)}{\sqrt{m}}.$$

Proof Let S be a labeled bag sample of size m, and let $\tilde{r}$ be its average bag size. Denote $T(x) = C\ln^{\beta}(x)$, and define $f(\gamma) = \frac{4T^2(1/\gamma^2)}{\gamma^2}$. We will show that $\mathcal{R}_{f(\gamma)}^{\sup} \leq \gamma$, thus allowing the use of Lemma 19. We have $\mathcal{R}_m \leq T(m)/\sqrt{m}$, thus it suffices to show that $T(f(\gamma))/\sqrt{f(\gamma)} \leq \gamma$.

Let $z(\gamma) = \sqrt{f(\gamma)}/T(f(\gamma))$. We will now show that $z(\gamma)T(z^2(\gamma)) \geq \frac{1}{\gamma}T(1/\gamma^2)$. Since the function $xT(x^2) = Cx\ln^{\beta}(x^2)$ is monotonic increasing for $x \geq 1$, we will conclude that $z(\gamma) \geq 1/\gamma$ for all $\gamma \leq 1$.

It is easy to see that for all values of $\beta, C \geq 0$, there is a number $n \geq 0$ such that for all $x \geq n$,

$$C^2\ln^{2\beta}(x) \leq x^{1-2^{-1/\beta}}.$$

For such x we have

$$\begin{aligned} T(x/T^2(x)) &= C\ln^{\beta}\left(\frac{x}{C^2\ln^{2\beta}(x)}\right) = C(\ln(x) - \ln(C^2\ln^{2\beta}(x)))^{\beta} \\ &\geq C(\ln(x) - (1-2^{-1/\beta})\ln(x)))^{\beta} = C\ln^{\beta}(x)/2 = T(x)/2. \end{aligned} \tag{21}$$

Let $\gamma_{\circ} \in (0,1)$ such that $f(\gamma_{\circ}) = k = \max\{n, K\}$. Since $f(\gamma)$ is monotonic decreasing with γ, for all $\gamma \leq \gamma_{\circ}$, $f(\gamma) \geq k$. Therefore, for $\gamma \leq \gamma_{\circ}$,

$$z(\gamma)T(z^2(\gamma)) = \frac{\sqrt{f(\gamma)}}{T(f(\gamma))}T\left(\frac{f(\gamma)}{T^2(f(\gamma))}\right) \geq \frac{1}{2}\frac{\sqrt{f(\gamma)}}{T(f(\gamma))}T(f(\gamma)) = \frac{1}{2}\sqrt{f(\gamma)} = T(1/\gamma^2)/\gamma.$$

The middle inequality follows from Equation (21), and the last equality follows from the definition of $f(\gamma)$. We conclude that $z(\gamma) \geq \frac{1}{\gamma}$. Therefore, for all $\gamma \leq \gamma_\circ$,

$$\mathcal{R}^{\sup}_{f(\gamma)}(\mathcal{H}) \leq \frac{T(f(\gamma))}{\sqrt{f(\gamma)}} = 1/z(\gamma) \leq \gamma.$$

Define $\tilde{f}$ as follows:

$$\tilde{f}(\gamma) = \begin{cases} f(\gamma) & \gamma \leq \gamma_\circ \\ k & \gamma > \gamma_\circ. \end{cases}$$

For $\gamma \leq \gamma_\circ$, clearly $\mathcal{R}^{\sup}_{\tilde{f}(\gamma)}(\mathcal{H}) \leq \gamma$, and for $\gamma > \gamma_\circ$,

$$\mathcal{R}^{\sup}_{\tilde{f}(\gamma)}(\mathcal{H}) = \mathcal{R}^{\sup}_{k}(\mathcal{H}) = \mathcal{R}^{\sup}_{f(\gamma_\circ)}(\mathcal{H}) \leq \gamma_\circ \leq \gamma.$$

Therefore for all $\gamma \in (0,1]$, $\mathcal{R}^{\sup}_{\tilde{f}(\gamma)}(\mathcal{H}) \leq \gamma$. By Lemma 19, for all $\varepsilon \in (0,1]$,

$$\begin{aligned}
\mathcal{R}(\overline{\mathcal{H}}_\ell, S) &\leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2a_2^2B^2\tilde{r}m}{\varepsilon^2}\right)\left(1 + \int_\varepsilon^1 \sqrt{\tilde{f}(\frac{\gamma}{4a_1a_2})}\,d\gamma\right) \\
&= 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2a_2^2B^2\tilde{r}m}{\varepsilon^2}\right)\left(1 + \int_{4a_1a_2\gamma_\circ}^1 \sqrt{k}\,d\gamma + \int_\varepsilon^{4a_1a_2\gamma_\circ} \sqrt{f(\frac{\gamma}{4a_1a_2})}\,d\gamma\right) \\
&\leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2a_2^2B^2\tilde{r}m}{\varepsilon^2}\right)\left(1 + \sqrt{k} + \int_\varepsilon^{4a_1a_2\gamma_\circ} \sqrt{f(\frac{\gamma}{4a_1a_2})}\,d\gamma\right). \qquad (22)
\end{aligned}$$

Denote $N = 1 + \sqrt{k}$. Now, if $\beta > 0$ we have

$$\begin{aligned}
&\int_\varepsilon^{4a_1a_2\gamma_\circ} \sqrt{f(\frac{\gamma}{4a_1a_2})}\,d\gamma \leq \int_\varepsilon^{4a_1a_2\gamma_\circ} \sqrt{f(\frac{\gamma}{4a_1a_2})}\,d\gamma = 2a_1a_2\int_\varepsilon^{4a_1a_2\gamma_\circ} \frac{T(16a_1^2a_2^2/\gamma^2)}{\gamma}\,d\gamma \\
&= 2a_1a_2C\int_\varepsilon^{4a_1a_2\gamma_\circ} \frac{\ln^\beta(16a_1^2a_2^2/\gamma^2)}{\gamma}\,d\gamma = 2a_1a_2C\left[-\ln^{\beta+1}(\frac{16a_1^2a_2^2}{\gamma^2})/(2(\beta+1))\right]_\varepsilon^{4a_1a_2\gamma_\circ} \\
&= \frac{a_1a_2C}{\beta+1}\ln^{\beta+1}(\frac{16a_1^2a_2^2\gamma_\circ^2}{\varepsilon^2}) \leq \frac{a_1a_2C}{\beta+1}\left(\ln^{\beta+1}(\frac{16a_1^2a_2^2}{\varepsilon^2})\right).
\end{aligned}$$

The same inequality holds also for $\beta = 0$, since in that case

$$\begin{aligned}
&\int_\varepsilon^{4a_1a_2\gamma_\circ} \sqrt{f(\frac{\gamma}{4a_1a_2})}\,d\gamma = 2a_1a_2\int_\varepsilon^{4a_1a_2\gamma_\circ} \frac{T(16a_1^2a_2^2/\gamma^2)}{\gamma}\,d\gamma \\
&= 2a_1a_2C\int_\varepsilon^{4a_1a_2\gamma_\circ} \frac{1}{\gamma}\,d\gamma = 2a_1a_2C\left[\ln(\gamma)\right]_\varepsilon^{4a_1a_2\gamma_\circ} = 2a_1a_2C\ln(\frac{4a_1a_2\gamma_\circ}{\varepsilon}) \\
&\leq 2a_1a_2C\ln(\frac{4a_1a_2}{\varepsilon}) = \frac{a_1a_2C}{\beta+1}\left(\ln^{\beta+1}(\frac{16a_1^2a_2^2}{\varepsilon^2})\right).
\end{aligned}$$

Therefore we can further bound Equation (22) to get

$$\mathcal{R}(\overline{\mathcal{H}}_\ell, S) \leq 4\varepsilon + \frac{10}{\sqrt{m}} \log\left(\frac{4ea_1^2a_2^2B^2\tilde{r}m}{\varepsilon^2}\right)\left(N + \frac{a_1a_2C}{\beta+1}\ln^{\beta+1}(\frac{16a_1^2a_2^2}{\varepsilon^2})\right).$$

Setting $\varepsilon = 1/\sqrt{m}$ we get

$$\mathcal{R}(\overline{\mathcal{H}}_\ell, S) \leq \frac{4 + 10\log(4ea_1^2a_2^2B^2\tilde{r}m^2)\left(N + \frac{a_1a_2C}{\beta+1}\ln^{\beta+1}(16a_1^2a_2^2m)\right)}{\sqrt{m}}.$$

Now, for a given sample S denote its average bag size by $\tilde{r}(S)$. We have

$$\begin{aligned}
\mathcal{R}_m(\overline{\mathcal{H}}_\ell, D) &= \mathbb{E}_{S\sim D^m}[\mathcal{R}(\overline{\mathcal{H}}_\ell, S)] \\
&\leq \mathbb{E}\left[\frac{4 + 10\log(4ea_1^2a_2^2B^2\tilde{r}(S)m^2)\left(N + \frac{a_1a_2C}{\beta+1}\ln^{\beta+1}(16a_1^2a_2^2m)\right)}{\sqrt{m}}\right] \\
&\leq \frac{4 + 10\log(4ea_1^2a_2^2B^2rm^2)\left(N + \frac{a_1a_2C}{\beta+1}\ln^{\beta+1}(16a_1^2a_2^2m)\right)}{\sqrt{m}}.
\end{aligned}$$

In the last inequality we used Jensen's inequality and the fact that $\mathbb{E}_{S\sim D^m}[\tilde{r}(S)] = r$. This is the desired bound, hence the theorem is proven. ■

To demonstrate the implications of this theorem, consider the case of MIL with soft-margin kernel SVM. Kernel SVM can operate in a general Hilbert space, which we denote by $\mathcal{T}$. The domain of instances is $\mathcal{X} = \{x \in \mathcal{T} \mid \|x\| \leq 1\}$, and the function class is the class of linear separators with a bounded norm $\mathcal{W}(C) = \{h_w \mid w \in \mathcal{T}, \|w\| \leq C\}$, for some $C > 0$, where $h_w = \langle x, w\rangle$. The loss is the hinge-loss ℓ_{hl} defined above, which is 1-Lipschitz in the second argument. We have (Bartlett and Mendelson, 2002)

$$\mathcal{R}_m^{\sup}(\mathcal{W}(C)_{\ell_{hl}}) \leq \frac{C}{\sqrt{m}} = \frac{C\ln^0(m)}{\sqrt{m}}.$$

Thus we can apply Theorem 20 with $\beta = 0$. Note that $\mathcal{W}(C) \subseteq [-C, C]^{\mathcal{X}}$, thus we can apply the theorem with $B = 2C$ by simply shifting the output of each h_w by C and adjusting the loss function accordingly. By Theorem 20 there exists a number N such that for any 1-Lipschitz bag-function ψ (such as max) and for any distribution D over labeled bags with an average bag size of r, we have

$$\mathcal{R}_m(\overline{\mathcal{H}}_\ell, D) \leq \frac{4 + 10\log(16eC^2rm^2)\,(N + C\ln(16m))}{\sqrt{m}}.$$

We can use this result and apply Equation (12) to get an upper bound on the loss of MIL with soft-margin SVM.

7. PAC-Learning for MIL

In the previous sections we addressed the sample complexity of generalized MIL, showing that it grows only logarithmically with the bag size. We now turn to consider the computational aspect of MIL, and specifically the relationship between computational feasibility of MIL and computational feasibility of the learning problem for the underlying instance hypothesis.

We consider real-valued hypothesis classes $\mathcal{H} \in [-1, +1]^{\mathcal{X}}$, and provide a MIL algorithm which uses a learning algorithm that operates on single instances as an oracle. We show that if the oracle

can minimize error with respect to $\mathcal{H}$, and the bag-function satisfies certain boundedness conditions, then the MIL algorithm is guaranteed to PAC-learn $\overline{\mathcal{H}}$. In particular, the guarantees hold if the bag-function is Boolean OR or max, as in classical MIL and its extension to real-valued hypotheses.

Given an algorithm $\mathcal{A}$ that learns $\mathcal{H}$ from single instances, we provide an algorithm called `MILearn` that uses $\mathcal{A}$ to implement a *weak learner* for bags with respect to $\overline{\mathcal{H}}$. That is, for any weighted sample of bags, `MILearn` returns a hypothesis from $\overline{\mathcal{H}}$ that has some success in labeling the bag-sample correctly. This will allow the use of `MILearn` as the building block in a Boosting algorithm (Freund and Schapire, 1997), which will find a linear combination of hypotheses from $\overline{\mathcal{H}}$ that classifies unseen bags with high accuracy. Furthermore, if $\mathcal{A}$ is efficient then the resulting Boosting algorithm is also efficient, with a polynomial dependence on the maximal bag size.

We open with background on Boosting in Section 7.1. We then describe the weak learner in and analyze its properties in Section 7.2. In Section 7.3 we provide guarantees on a Boosting algorithm that uses our weak leaner, and conclude that the computational complexity of PAC-learning for MIL can be bounded by the computational complexity of agnostic PAC-learning for single instances.

7.1 Background: Boosting with Margin Guarantees

In this section we give some background on Boosting algorithms, which we will use to derive an efficient learning algorithm for MIL. Boosting methods (Freund and Schapire, 1997) are techniques that allow enhancing the power of a *weak learner*—a learning algorithm that achieves error slightly better than chance—to derive a classification rule that has low error on an input sample. The idea is to iteratively execute the weak learner on weighted versions of the input sample, and then to return a linear combination of the classifiers that were emitted by the weak learner in each round.

Let A be a domain of objects to classify, and let $H : [-1,+1]^A$ be the hypothesis class used by the weak learner. A Boosting algorithm receives as input a labeled sample $S = \{(x_i, y_i)\}_{i=1}^m \subseteq A \times \{-1,+1\}$, and iteratively feeds to the weak learner a reweighed version of S. Denote the $m-1$-dimensional simplex by $\Delta_m = \{\mathbf{w} \in \mathbb{R}^m \mid \sum_{i\in[m]} w_i = 1, \forall i \in [m], w[i] \geq 0\}$. For a vector $\mathbf{w} \in \Delta_m$, $S_\mathbf{w} = \{(w[i], x_i, y_i)\}_{i=1}^m$ is the sample S reweighed by $\mathbf{w}$. The Boosting algorithm runs in k rounds. On round t it sets a weight vector $\mathbf{w}_t \in \Delta_m$, calls the weak learner with input $S_{\mathbf{w}_t}$, and receives a hypothesis $h_t \in H$ as output from the weak learner. After k rounds, the Boosting algorithm returns a classifier $f_\circ : A \to [-1,+1]$, which is a linear combination of the hypotheses received from the weak learner: $f_\circ = \sum_{t\in[k]} \alpha_t h_t$, where $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$.

The literature offers plenty of Boosting algorithms with desirable properties. For concreteness, we use the algorithm `AdaBoost`* (Rätsch and Warmuth, 2005), since it provides suitable guarantees on the *margin* of its output classifier. For a labeled example (x,y), the quantity $y f_\circ(x)$ is the margin of $f_\circ$ when classifying x. If the margin is positive, then $\text{sign} \circ f_\circ$ classifies x correctly. The margin of any function f on a labeled sample $S = \{(x_i, y_i)\}_{i=1}^m$ is defined as

$$M(f,S) = \min_{i\in[m]} y_i f(x_i).$$

If $M(f,S)$ is positive, then the entire sample is classified correctly by $\text{sign} \circ f$.

If S is an i.i.d. sample drawn from a distribution on $A \times \{-1,+1\}$, then classification error of $f_\circ$ on the distribution can be bounded based on $M(f_\circ, S)$ and the pseudo-dimension d of the hypothesis class H. The following bound (Schapire and Singer, 1999, Theorem 8) holds with probability $1-\delta$

over the training samples, for any $m \geq d$:

$$\mathbb{P}[Y \cdot f_\circ(X) \leq 0] \leq O\left(\sqrt{\frac{d\ln^2(m/d)/M^2(f_\circ,S)+\ln(1/\delta)}{m}}\right). \tag{23}$$

In fact, inspection of the proof of this bound in Schapire and Singer (1999) reveals that the only property of the hypothesis class H that is used to achieve this result is the following bound, due to Haussler and Long (1995), on the covering number of a hypothesis class H with pseudo-dimension d:

$$\forall \gamma \in (0,1], \quad \mathcal{N}_m(\gamma, \mathcal{H}, \infty) \leq \left(\frac{em}{\gamma d}\right)^d. \tag{24}$$

Thus, Equation (23) holds whenever this covering bound holds—a fact that will be useful to us.

For `AdaBoost`*, a guarantee on the size of the margin of $f_\circ$ can be achieved if one can provide a guarantee on the *edge* of the hypotheses returned by the weak learner. The edge of a hypothesis measures of how successful it is in classifying labeled examples. Let $h : A \to [-1,+1]$ be a hypothesis and let D be a distribution over $A \times \{-1,+1\}$. The edge of h with respect to D is

$$\Gamma(h,D) \triangleq \mathbb{E}_{(X,Y)\sim D}[Y \cdot h(X)].$$

For a weighted and labeled sample $S = \{(w_i,x_i,y_i)\}_{i\in[m]} \subseteq \mathbb{R}_+ \times A \times \{-1,+1\}$,

$$\Gamma(h,S) \triangleq \sum_{i\in[m]} w_i y_i h(x_i).$$

Note that if $h(x)$ is interpreted as the probability of h to emit 1 for input x, then $\frac{1-\Gamma(h,D)}{2}$ is the expected misclassification error of h on D. Thus, a positive edge implies a labeling success of more than chance. For `AdaBoost`*, a positive edge on each of the weighted samples fed to the weak learner suffices to guarantee a positive margin of its output classifier $f_\circ$.

Theorem 21 (Rätsch and Warmuth 2005) *Assume* `AdaBoost`* *receives a labeled sample S of size m as input. Suppose that* `AdaBoost`* *runs for k rounds and returns the classifier $f_\circ$. If for every round $t \in [k]$, $\Gamma(h_t, S_{\mathbf{w}_t}) \geq \rho$, then $M(f_\circ,S) \geq \rho - \sqrt{2\ln m/k}$.*

We present a simple corollary, which we will use when analyzing Boosting for MIL. This corollary shows that `AdaBoost`* can be used to transform a weak learner that approximates the best edge of a weighted sample to a Boosting algorithm that approximates the best margin of a labeled sample. The proof of the corollary employs the following well known result, originally by von Neumann (1928) and later extended (see, e.g., Nash and Sofer, 1996). For a hypothesis class H, denote by $\mathrm{co}(H)$ the set of all linear combinations of hypotheses in H. We say that $H \subseteq [-1,+1]^A$ is compact with respect to a sample $S = \{(x_i,y_i)\}_{i\in[m]} \subseteq A \times \{-1,+1\}$ if the set of vectors $\{(h(x_1),\ldots,h(x_m)) \mid h \in H\}$ is compact.

Theorem 22 (The Strong Min-Max theorem) *If H is compact with respect to S, then*

$$\min_{\mathbf{w}\in\Delta_m} \sup_{h\in H} \Gamma(h,S_{\mathbf{w}}) = \sup_{f\in\mathrm{co}(H)} M(f,S).$$

Corollary 23 *Suppose that* `AdaBoost`* *is executed with an input sample S, and assume that H is compact with respect to S. Assume the weak learner used by* `AdaBoost`* *has the following guarantee: For any $\mathbf{w} \in \Delta_m$, if the weak learner receives $S_\mathbf{w}$ as input, then with probability at least $1-\delta$ it returns a hypothesis $h_\circ$ such that*

$$\Gamma(h_\circ, S_\mathbf{w}) \geq g(\sup_{h \in H} \Gamma(h, S_\mathbf{w})),$$

where $g : [-1,+1] \to [-1,+1]$ is some fixed non-decreasing function. Then for any input sample S, if `AdaBoost`* *runs k rounds, it returns a linear combination of hypotheses $f_\circ = \sum_{t \in [k]} \alpha_t h_t$, such that with probability at least $1-k\delta$*

$$M(f_\circ, S) \geq g(\sup_{f \in \mathrm{co}(H)} M(f,S)) - \sqrt{2\ln m/k}.$$

Proof By Theorem 22, $\min_{\mathbf{w} \in \Delta_m} \sup_{h \in H} \Gamma(h, S_\mathbf{w}) = \sup_{f \in \mathrm{co}(H)} M(f,S)$. Thus, for any vector of weights $\mathbf{w}$ in the simplex, $\sup_{h \in H} \Gamma(h, S_\mathbf{w}) \geq \sup_{f \in \mathrm{co}(H)} M(f,S)$. It follows that in each round, the weak learner that receives $S_{\mathbf{w}_t}$ as input returns a hypothesis h_t such that $\Gamma(h_t, S_{\mathbf{w}_t}) \geq g(\sup_{h \in H} \Gamma(h, S_{\mathbf{w}_t})) \geq g(\sup_{f \in \mathrm{co}(H)} M(f,S))$. By Theorem 21, it follows that $M(f_\circ, S) \geq g(\sup_{f \in \mathrm{co}(H)} M(f,S)) - \sqrt{2\ln m/k}$. ■

7.2 The Weak Learner

In this section we will present our weak learner for MIL and provide guarantees for the edge it achieves. Our guarantees depend on boundedness properties of the bag-function ψ, which we define below. To motivate our definition of boundedness, consider the p-norm bag functions (see Definition 3), defined by $\psi_p(\mathbf{z}) \triangleq \left(\frac{1}{n}\sum_{i=1}^n (z[i]+1)^p\right)^{1/p} - 1$. Recall that this class of functions includes the max function (ψ_∞) and the average function (ψ_1) as two extremes. Assume $R \subseteq [r]$ for some $r \in \mathbb{N}$. It is easy to verify that for any natural n, any sequence $z_1, \ldots, z_n \in [-1,+1]$, and all $p \in [1,\infty]$,

$$\frac{1}{n}\sum_{i \in [n]} z_i \leq \psi_p(z_1, \ldots, z_n) \leq \sum_{i \in [n]} z_i + n - 1.$$

Since $R \subseteq [r]$, it follows that for all $(z_1, \ldots, z_n) \in [-1,+1]^{(R)}$,

$$\frac{1}{r}\sum_{i \in [n]} z_i \leq \psi_p(z_1, \ldots, z_n) \leq \sum_{i \in [n]} z_i + r - 1. \tag{25}$$

We will show that in cases where the bag function is linearly bounded in the sum of its arguments, as in Equation (25), a single-instance learning algorithm can be used to learn MIL. Our weak learner will be parameterized by the boundedness parameters of the bag-function, defined formally as follows.

Definition 24 *A function $\psi : [-1,+1]^{(R)} \to [-1,+1]$ is* (a,b,c,d)-bounded *if for all $(z_1, \ldots, z_n) \in [-1,+1]^{(R)}$,*

$$a \sum_{i \in [n]} z_i + b \leq \psi(z_1, \ldots, z_n) \leq c \sum_{i \in [n]} z_i + d.$$

Thus, for all $p \in [1,\infty)$, ψ_p over bags of size at most r is $(\frac{1}{r}, 0, 1, r-1)$-bounded.

Before listing the weak learner `MILearn`, we introduce some notations. $\mathbf{h}_{\text{pos}}$ denotes a special bag-hypothesis that labels all bags as $+1$: $\forall x \in X^{(R)}, \quad \mathbf{h}_{\text{pos}}(x) = 1$. We denote $\overline{\mathcal{H}}_+ \triangleq \overline{\mathcal{H}} \cup \{\mathbf{h}_{\text{pos}}\}$. Let $\mathcal{A}$ be an algorithm that receives a labeled and weighted instance sample as input, and returns a hypothesis $h \in \mathcal{H}$. The result of running $\mathcal{A}$ with input S is denoted $\mathcal{A}(S) \in \mathcal{H}$.

The algorithm `MILearn`, listed as Algorithm 1 below, accepts as input a bag sample $\overline{S}$ and a bounded bag-function ψ. It also has access to the algorithm $\mathcal{A}$. We sometimes emphasize that `MILearn` uses a specific algorithm $\mathcal{A}$ as an oracle by writing $\texttt{MILearn}^{\mathcal{A}}$. `MILearn` constructs a sample of instances S_I from the instances that make up the bags in $\overline{S}$, labeling each instance in S_I with the label of the bag it came from. The weights of the instances depend on whether the bag they came from was positive or negative, and on the boundedness properties of ψ. Having constructed S_I, `MILearn` calls $\mathcal{A}$ with S_I. It then decides whether to return the bag-hypothesis induced by applying ψ to $\mathcal{A}(S_I)$, or to simply return $\mathbf{h}_{\text{pos}}$.

It is easy to see that the time complexity of `MILearn` is bounded by $O(f(N)+N)$, where N is the total number of instances in the bags of $\overline{S}$, and $f(n)$ is an upper bound on the time complexity of $\mathcal{A}$ when running on a sample of size n. As we presently show, the output of `MILearn` is a bag-hypothesis in $\overline{\mathcal{H}}_+$ whose edge on $\overline{S}$ depends on the best achievable edge for $\overline{S}$.

Algorithm 1: $\texttt{MILearn}^{\mathcal{A}}$

Assumptions:

- $\mathcal{H} \in [-1,+1]^X$
- Algorithm $\mathcal{A}$ receives a weighted instance sample and returns a hypothesis in $\mathcal{H}$.

Input:

- $\overline{S} \triangleq \{(w_i, \mathbf{x}_i, y_i)\}_{i\in[m]}$—a labeled and weighted sample of bags,
- ψ—an (a,b,c,d)-bounded bag-function.

Output: $h_\circ \in \overline{\mathcal{H}}_+$.

$\alpha_{(+1)} \leftarrow a, \alpha_{(-1)} \leftarrow c$.

$S_I \leftarrow \{(\alpha_{y_i} \cdot w_i, x_i[j], y_i)\}_{i\in[m], j\in[r]}$.

$h_I \leftarrow \mathcal{A}(S_I)$.

if $\Gamma(\overline{h}_I, \overline{S}) \geq \Gamma(\mathbf{h}_{\text{pos}}, \overline{S})$ **then**

  $h_\circ \leftarrow \overline{h}_I$,

else

  $h_\circ \leftarrow \mathbf{h}_{\text{pos}}$.

Return $h_\circ$.

The guarantees for $\texttt{MILearn}^{\mathcal{A}}$ depend on the properties of $\mathcal{A}$. We define two properties that we consider for $\mathcal{A}$. The first property is that the edge of the hypothesis $\mathcal{A}$ returns is close to the best possible one on the input sample.

Definition 25 (ε-optimal) *An algorithm $\mathcal{A}$ that accepts a weighted and labeled sample of instances in $\mathcal{X}$ and returns a hypothesis in $\mathcal{H}$ is* ε-optimal *if for all weighted samples $S \subseteq \mathbb{R}_+ \times \mathcal{X} \times \{-1,+1\}$ with total weight W,*

$$\Gamma(\mathcal{A}(S),S) \geq \sup_{h\in\mathcal{H}} \Gamma(h,S) - \varepsilon W.$$

The second property is that the edge of the hypothesis that $\mathcal{A}$ returns is close to the best possible one on the input sample, but only compared to the edges that can be achieved by hypotheses that label all the negative instances of S with -1. For a hypothesis class $\mathcal{H}$ and a distribution D over labeled examples, we denote the set of hypotheses in $\mathcal{H}$ that label all negative examples in D with -1, by

$$\Omega(\mathcal{H},D) = \{h \in \mathcal{H} \mid \mathbb{P}_{(X,Y)\sim D}[h(X) = -1 \mid Y = -1] = 1\}.$$

For a labeled sample S, $\Omega(\mathcal{H},S) \triangleq \Omega(\mathcal{H},U_S)$ where U_S is the uniform distribution over the examples in S.

Definition 26 (one-sided-ε-optimal) *An algorithm $\mathcal{A}$ that accepts a weighted and labeled sample of instances in $\mathcal{X}$ and returns a hypothesis in $\mathcal{H}$ is* one-sided-ε-optimal *if for all weighted samples $S \subseteq \mathbb{R}_+ \times \mathcal{X} \times \{-1,+1\}$ with total weight W,*

$$\Gamma(\mathcal{A}(S),S) \geq \sup_{h\in\Omega(\mathcal{H},S)} \Gamma(h,S) - \varepsilon W.$$

Clearly, any algorithm which is ε-optimal is also one-sided-ε-optimal, thus the first requirement from $\mathcal{A}$ is stronger. In our results below we compare the edge achieved using $\texttt{MILearn}$ to the best possible edge for the sample $\overline{S}$. Denote the best edge achievable for $\overline{S}$ by a hypothesis in $\mathcal{H}$ by

$$\gamma^* \triangleq \sup_{h\in\overline{\mathcal{H}}} \Gamma(h,\overline{S}).$$

We denote by γ^*_+ the best edge that can be achieved by a hypothesis in $\Omega(\overline{\mathcal{H}},\overline{S})$. Formally,

$$\gamma^*_+ \triangleq \sup_{h\in\Omega(\overline{\mathcal{H}},\overline{S})} \Gamma(h,\overline{S}).$$

Denote the weight of the positive bags in the input sample $\overline{S}$ by $W_+ = \sum_{i:y_i=+1} w_i$ and the weight of the negative bags by $W_- = \sum_{i:y_i=-1} w_i$. We will henceforth assume without loss of generality that the total weight of all bags in the input sample is 1, that is $W_+ + W_- = 1$.

Note that for any (a,b,c,d)-bounded ψ, if there exists any sequence $z_1,\ldots,z_n$ such that $\psi(z_1,\ldots,z_n) = -1$, then

$$a\sum_{i\in[n]} z_i + b \leq -1 \leq c\sum_{i\in[n]} z_i + d. \tag{26}$$

This implies

$$\frac{-1-d}{c} \leq \sum_{i\in[n]} z_i \leq \frac{-1-b}{a}.$$

Rearranging, we get $d - \frac{c}{a}b - \frac{c}{a} + 1 \geq 0$, with equality if Equation (26) holds with equalities. The next theorem provides a guarantee for `MILearn` that depends on the tightness of this inequality for the given bag function. As evident from Theorem 21, to guarantee a positive margin for the output of `AdaBoost`* when used with `MILearn` as the weak learner, we need to guarantee that the edge of the hypothesis returned by `MILearn` is always positive. Since the best edge cannot be more than 1, we emphasize in the theorem below that the edge achieved by `MILearn` is positive at least when the best edge is 1 (and possibly also for smaller edges, depending on the parameters). We subsequently show how these general guarantees translate to a specific result for the max function, and other bag functions with the same boundedness properties.

Theorem 27 *Let $r \in \mathbb{N}$ and $R \subseteq [r]$. Let $\psi : [-1,+1]^{(R)} \to [-1,+1]$ be an (a,b,c,d)-bounded bag-function such that $0 < a \leq c$. Let $\varepsilon \in [0, \frac{1}{rc})$, and assume that $d - \frac{c}{a}b - \frac{c}{a} + 1 = \eta$. Denote $Z = \frac{c}{a}$. Consider running the algorithm $\texttt{MILearn}^{\mathcal{A}}$ with a weighted bag sample $\overline{S}$ of total weight 1, and let $h_\circ$ be the hypothesis returned by $\texttt{MILearn}^{\mathcal{A}}$. Then*

1. *If $\mathcal{A}$ is ε-optimal then*

$$\Gamma(h_\circ, \overline{S}) \geq \frac{Z\gamma^* - Z + \frac{1}{Z} - \frac{\eta}{2}(1 + \frac{1}{Z}) - rc\varepsilon}{1 + (1 - \frac{\eta}{2})(1 - \frac{1}{Z})}.$$

   Thus, $\Gamma(h_\circ, \overline{S}) > 0$ whenever

$$\gamma^* > 1 - \frac{1}{Z^2} + \frac{\eta}{2}(\frac{1}{Z} + \frac{1}{Z^2}) + \frac{rc\varepsilon}{Z}.$$

   In particular, if $\eta \leq 2(1 - rc\varepsilon)/(Z+1)$ and $\gamma^* = 1$ then $\Gamma(h_\circ, \overline{S}) > 0$.

2. *If $\mathcal{A}$ is one-sided-ε-optimal, and $\psi(z_1, \ldots, z_n) = -1$ only if $z_1 = \ldots = z_n = -1$, then*

$$\Gamma(h_\circ, \overline{S}) \geq \frac{\gamma^*_+ - \frac{\eta}{2}(Z+1) - rc\varepsilon Z}{2Z - 1 - \frac{\eta}{2}(Z-1)}.$$

   Thus, $\Gamma(h_\circ, \overline{S}) > 0$ whenever

$$\gamma^*_+ > \frac{\eta}{2}(Z+1) + rc\varepsilon Z.$$

   In particular, if $\eta \leq 2(1 - rc\varepsilon Z)/(Z+1)$ and $\gamma^*_+ = 1$ then $\Gamma(h_\circ, \overline{S}) > 0$.

The proof of the theorem is provided in Appendix A. This theorem is stated in general terms, as it holds for any bounded ψ. In particular, if ψ is any function between an average and a max, including any of the p-norm bag functions ψ_p defined in Definition 3, we can simplify the result, as captured by the following corollary.

Corollary 28 *Let $\mathcal{H} \subseteq [-1,+1]^{\mathcal{X}}$. Let $R \subseteq [r]$, and $\varepsilon \in [0, \frac{1}{r})$. Assume a bag function $\psi : [-1,+1]^{(R)} \to [-1,+1]$ such that for any $z_1, \ldots, z_n \in [-1,+1]$,*

$$\frac{1}{n}\sum_{i \in [n]} z_i \leq \psi(z_1, \ldots, z_n) \leq \max_{i \in [n]} z_i.$$

Let $h_\circ$ be the hypothesis returned by $\texttt{MILearn}^{\mathcal{A}}$. Then

1. *If $\mathcal{A}$ is ε-optimal for some $\varepsilon \in [0, 1/r]$, then*

$$\Gamma(h_\circ, \overline{S}) \geq \frac{r^2\gamma^* + 1 - r^2(1+\varepsilon)}{2r-1}.$$

Thus $\Gamma(h_\circ, \overline{S}) > 0$ whenever $\gamma^* \geq 1 - \frac{1}{r^2} + \frac{\varepsilon}{r}$. In particular, if $\gamma^* = 1$ then $\Gamma(h_\circ, \overline{S}) > 0$.

2. *If $\mathcal{A}$ is one-sided-ε-optimal some $\varepsilon \in [0, 1/r^2]$, then*

$$\Gamma(h_\circ, \overline{S}) \geq \frac{\gamma^*_+ - r^2\varepsilon}{2r-1}.$$

Thus $\Gamma(h_\circ, \overline{S}) > 0$ whenever $\gamma^*_+ > r^2\varepsilon$. In particular, if $\gamma^*_+ = 1$ then $\Gamma(h_\circ, \overline{S}) > 0$.

Proof Let $z_1, \ldots, z_n \in [-1, +1]$. We have

$$\max_{i\in[n]} z_i \leq \sum_{i\in[n]} z_i - (n-1)\min(z_i) \leq \sum_{i\in[n]} z_i + n - 1.$$

Therefore, by the assumption on ψ, for any $n \in R$

$$\psi(z_1, \ldots, z_n) \leq \sum_{i\in[n]} z_i + n - 1 \leq \sum_{i\in[n]} z_i + r - 1.$$

In addition

$$\frac{1}{r}\sum_{i\in[n]} z_i \leq \frac{1}{n}\sum_{i\in[n]} z_i \leq \psi(z_1, \ldots, z_n).$$

Therefore ψ is $(\frac{1}{r}, 0, 1, r-1)$-bounded. It follows that $Z = r$ in this case, and $d - Zb - Z + 1 = 0$. Claim (1) follows by applying case (1) of Theorem 27 with $\eta = 0$.

For claim (2) we apply case (2) of Theorem 27. Thus we need to show that if $\psi(z_1, \ldots, z_n) = -1$ and $z_1, \ldots, z_n \in [-1, +1]$, then $z_1 = \ldots = z_n = -1$. We have that

$$-1 \leq \frac{1}{n}\sum_{i\in[n]} z_i \leq \psi(z_1, \ldots, z_n) \leq -1.$$

Therefore $\frac{1}{n}\sum_{i\in[n]} z_i = -1$. Since no z_i can be smaller than -1, $z_1 = \ldots = z_n = -1$. Thus case (2) of Theorem 27 holds. We get our claim (2) directly by subsituting the boundedness parameters of ψ in Theorem 27 case (2). ■

7.3 From Single-Instance Learning to Multi-Instance Learning

In this section we combine the guarantees on `MILearn` with the guarantees on `AdaBoost`*, to show that efficient agnostic PAC-learning of the underlying instance hypothesis $\mathcal{H}$ implies efficient PAC-learning of MIL. For simplicity we formalize the results for the natural case where the bag function is $\psi = \max$. Results for other bounded bag functions can be derived in a similar fashion.

First, we formally define the notions of agnostic and one-sided PAC-learning algorithms. We then show that given an algorithm on instances that satisfies one of these definitions, we can construct an algorithm for MIL which approximately maximizes the margin on an input bag sample.

Specifically, if the input bag sample is realizable by $\overline{\mathcal{H}}$, then the MIL algorithm we propose will find a linear combination of bag hypotheses that classifies the sample with zero error, and with a positive margin. Combining this with the margin-based generalization guarantees mentioned in Section 7.1, we conclude that we have an efficient PAC-learner for MIL.

Definition 29 (Agnostic PAC-learner and one-sided PAC-learner) *Let $\mathcal{B}(\varepsilon,\delta,S)$ be an algorithm that accepts as input $\delta,\varepsilon\in(0,1)$, and a labeled sample $S\in(\mathcal{X}\times\{-1,+1\})^m$, and emits as output a hypothesis $h\in\mathcal{H}$. $\mathcal{B}$ is an* agnostic PAC-learner *for $\mathcal{H}$ with complexity $c(\varepsilon,\delta)$ if $\mathcal{B}$ runs for no more than $c(\varepsilon,\delta)$ steps, and for any probability distribution D over $\mathcal{X}\times\{-1,+1\}$, if S is an i.i.d. sample from D of size $c(\varepsilon,\delta)$, then with probability at least $1-\delta$ over S and the randomization of $\mathcal{B}$,*

$$\Gamma(\mathcal{B}(\varepsilon,\delta,S),D)\geq\sup_{h\in\mathcal{H}}\Gamma(h,D)-\varepsilon.$$

$\mathcal{B}$ is a one-sided PAC-learner *if under the same conditions, with probability at least $1-\delta$*

$$\Gamma(\mathcal{B}(\varepsilon,\delta,S),D)\geq\sup_{h\in\Omega(\mathcal{H},D)}\Gamma(h,D)-\varepsilon.$$

Algorithm 2: $O^{\mathcal{B}}_{\varepsilon,\delta}$

Assumptions:

- $\varepsilon,\delta\in(0,1)$.
- $\mathcal{B}$ receives a labeled instance sample as input and returns a hypothesis in $\mathcal{H}$.
- Algorithm $\mathcal{B}$ is a one-sided (or agnostic) PAC-learning algorithm with complexity $c(\varepsilon,\delta)$.

Input: A labeled and weighted instance sample $S=\{(w_i,x_i,y_i)\}_{i\in[m]}\subseteq\mathbb{R}_+\times\mathcal{X}\times\{-1,+1\}$.
Output: A hypothesis in $\mathcal{H}$

For all $i\in[m]$, $p_i\leftarrow w_i/\sum_{i\in[m]}w_i$.
For each $t\in[c(\varepsilon,\delta)]$, independently draw a random j_t such that $j_t=i$ with probability p_i.
$\tilde{S}\leftarrow\{(x_{j_t},y_{j_t})\}_{t\in[c(\varepsilon,\delta)]}$.
$h\leftarrow\mathcal{B}(\tilde{S})$
Return h.

Given an agnostic PAC-learner $\mathcal{B}$ for $\mathcal{H}$ and parameters $\varepsilon,\delta\in(0,1)$, the algorithm $O^{\mathcal{B}}_{\varepsilon,\delta}$, listed above as Algorithm 2, is an ε-optimal algorithm with probability $1-\delta$. Similarly, if $\mathcal{B}$ is a one-sided PAC-learner, then $O^{\mathcal{B}}_{\varepsilon,\delta}$ is a one-sided-ε-optimal algorithm with probability $1-\delta$. Our MIL algorithm is then simply $\texttt{AdaBoost}^*$ with $\texttt{MILearn}^{O^{\mathcal{B}}_{\varepsilon,\delta}}$ as the (high probability) weak learner. It is easy to see that this algorithm learns a linear combination of hypotheses from $\overline{\mathcal{H}}_+$. We also show below that under certain conditions this linear combination induces a positive margin on the input bag sample with high probability. Given this guaranteed margin, we bound the generalization error of the learning algorithm via Equation (23).

The computational complexity of $O^{\mathcal{B}}_{\varepsilon,\delta}$ is polynomial in $c(\varepsilon,\delta)$ and in the instance-sample size m. Therefore, the computational complexity of $\texttt{MILearn}^{O^{\mathcal{B}}_{\varepsilon,\delta}}$ is polynomial in $c(\varepsilon,\delta)$ and in N, where N is the total number of instances in the input bag sample $\overline{S}$.

For 1-Lipschitz bag functions which have desired boundedness properties, both the sample complexity and the computational complexity of the proposed MIL algorithm are polynomial in the maximal bag size and linear in the complexity of the underlying instance hypothesis class. This is formally stated in the following theorem, for the case of a realizable distribution over labeled bags. Note that in particular, the theorem holds for all the p-norm bag-functions, since they are 1-Lipschitz and satisfy the boundedness conditions.

Theorem 30 *Let $\mathcal{H} \subseteq [-1,+1]^{\mathcal{X}}$ be a hypothesis class with pseudo-dimension d. Let $\mathcal{B}$ be a one-sided PAC-learner for $\mathcal{H}$ with complexity $c(\varepsilon,\delta)$. Let $r \in \mathbb{N}$, and let $R \subseteq [r]$. Assume that the bag function $\psi : [-1,+1]^{(R)} \to [-1,+1]$ is 1-Lipschitz with respect to the infinity norm, and that for any $(z_1,\ldots,z_n) \in [-1,+1]^{(R)}$*

$$\frac{1}{n}\sum_{i\in[n]} z_i \leq \psi(z_1,\ldots,z_n) \leq \max_{i\in[n]} z_i.$$

Assume that $\overline{\mathcal{H}}$ is compact with respect to any sample of size m. Let D be a distribution over $\mathcal{X}^{(R)} \times \{-1,+1\}$ which is realizable by $\overline{\mathcal{H}}$, that is there exists an $h \in \mathcal{H}$ such that $\mathbb{P}_{(\mathbf{X},Y)\sim D}[\overline{h}(\mathbf{X}) = Y] = 1$. Assume $m \geq 10d\ln(er)$, and let $\varepsilon = \frac{1}{2r^2}$ and $k = 32(2r-1)^2\ln(m)$.

For all $\delta \in (0,1)$, if `AdaBoost`* *is executed for k rounds on a random sample $S \sim D^m$, with* `MILearn`$^{O^{\mathcal{B}}_{\varepsilon,\delta/2k}}$ *as the weak learner, then with probability $1-\delta$, the classifier $f_\circ$ returned by* `AdaBoost`* *satisfies*

$$\mathbb{P}_D[Yf(\mathbf{X}) \leq 0] \leq O\left(\sqrt{\frac{dr^2\ln(r)\ln^2(m)+\ln(2/\delta)}{m}}\right). \tag{27}$$

Proof Since $\mathcal{B}$ is a one-sided PAC-learning algorithm, $O^{\mathcal{B}}_{\varepsilon,\delta/2k}$ is one-sided-ε-optimal with probability at least $1-\delta/2k$. Therefore, by case (2) of Corollary 28, if `MILearn`$^{O^{\mathcal{B}}_{\varepsilon,\delta/k}}$ receives a weighted bag sample $S_{\mathbf{w}}$, with probability $1-\delta/2k$ it returns a bag hypothesis $h_\circ \in \overline{\mathcal{H}}_+$ such that

$$\Gamma(h_\circ, S_{\mathbf{w}}) \geq \frac{\sup_{h\in\Omega(\overline{\mathcal{H}},S)}\Gamma(h,S_{\mathbf{w}}) - r^2\varepsilon}{2r-1}.$$

Thus, by Corollary 23, if `AdaBoost`* runs for k rounds then with probability $1-\delta/2$ it returns a linear combination of hypotheses from $\overline{\mathcal{H}}_+$ such that

$$M(f_\circ, S) \geq \frac{\sup_{f\in\mathrm{co}(\Omega(\overline{\mathcal{H}},S))} M(f,S) - r^2\varepsilon}{2r-1} - \sqrt{2\ln m/k}. \tag{28}$$

Due to the realizability assumption for D, there is an $h \in \Omega(\overline{\mathcal{H}}, S)$ that classifies correctly the bag sample S. It follows that for any weighting $\mathbf{w} \in \Delta_m$ of S, $\Gamma(h,S_{\mathbf{w}}) = 1$. It is easy to verify that since $\overline{\mathcal{H}}$ is compact with respect to S, then so is $\Omega(\mathcal{H},S)$. Thus, by Theorem 22, $\sup_{f\in\mathrm{co}(\Omega(\overline{\mathcal{H}},S))} M(f,S) = \min_{\mathbf{w}}\sup_{h\in\Omega(\overline{\mathcal{H}},S)}\Gamma(h,S_{\mathbf{w}}) = 1$. Substituting ε and k with their values, setting $\sup_{f\in\mathrm{co}(\Omega(\overline{\mathcal{H}},S))} M(f,S) = 1$ in Equation (28) and simplifying, we get that with probability $1-\delta/2$

$$M(f_\circ, S) \geq \frac{1}{8r-4}. \tag{29}$$

We would now like to apply the generalization bound in Equation (23), but for this we need to show that Equation (24) holds for $\overline{\mathcal{H}}$. We have the following bound on the covering numbers of $\overline{\mathcal{H}}$, for all $\gamma \in (0,1]$:

$$\mathcal{N}_m(\gamma, \overline{\mathcal{H}}, \infty) \leq \mathcal{N}_{rm}(\gamma, \mathcal{H}, \infty) \leq \left(\frac{erm}{\gamma d}\right)^d.$$

The first inequality is due to Corollary 13 and the fact that ψ is 1-Lipschitz, and the second inequality is due to Haussler and Long (1995) and the pseudo-dimension of $\mathcal{H}$ (see Equation (24) above). This implies

$$\begin{aligned}\mathcal{N}_m(\gamma, \overline{\mathcal{H}}, \infty) &\leq \left(\frac{erm}{\gamma d}\right)^d = \left(\frac{em}{\gamma d}\right)^d \cdot e^{d\ln(r)} = \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^d \cdot (10\ln(er))^d e^{d\ln(r)} \\ &= \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^d \cdot e^{d(\ln(10\ln(er))+\ln(r))}.\end{aligned}$$

Therefore, for $m \geq 10d\ln(er)$

$$\begin{aligned}\mathcal{N}_m(\gamma, \overline{\mathcal{H}}_+, \infty) &\leq 1 + \mathcal{N}_m(\gamma, \overline{\mathcal{H}}, \infty) \leq 1 + \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^d \cdot e^{d(\ln(10\ln(er))+\ln(r))} \\ &\leq \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^d \cdot e^{d(\ln(10\ln(er))+\ln(er))}.\end{aligned}$$

Now, $\ln(10\ln(er))+\ln(er) = \ln(10)+\ln(\ln(er))+\ln(er) \leq \ln(10)+2\ln(er) \leq 3+2\ln(er) \leq 5\ln(er)$. Therefore,

$$\mathcal{N}_m(\gamma, \overline{\mathcal{H}}_+, \infty) \leq \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^d \cdot e^{5d\ln(er)} \leq \left(\frac{e^2 m}{\gamma \cdot 10d\ln(er)}\right)^{5d\ln(er)} \leq \left(\frac{em}{\gamma \cdot 10d\ln(er)}\right)^{10d\ln(er)}.$$

Thus, for $m \geq 10d\ln(er)$, Equation (24) holds for $\overline{\mathcal{H}}_+$ when substituting d with $d_r = 10d\ln(er)$. This means the generalization bound in Equation (23) holds when substituting d with d_r as well. It follows that with probability $1-\delta/2$

$$\mathbb{P}[Yf_\circ(X) \leq 0] \leq O\left(\sqrt{\frac{d_r \ln^2(m/d_r)/M^2(f_\circ, S) + \ln(1/\delta)}{m}}\right).$$

Now, with probability $1-\delta/2$, by Equation (29) we have $M(f_\circ, S) \geq 1/(8r-4)$. Combining the two inequalities and applying the union bound, we have that with probability $1-\delta$

$$\begin{aligned}\mathbb{P}[Yf_\circ(X) \leq 0] &\leq O\left(\sqrt{\frac{d_r(8r-4)^2\ln^2(m/d_r) + \ln(2/\delta)}{m}}\right) \\ &\leq O\left(\sqrt{\frac{10d\ln(er)(8r-4)^2\ln^2(m) + \ln(2/\delta)}{m}}\right).\end{aligned}$$

Due to the O-notation we can simplify the right-hand side to get Equation (27).

■

Similar generalization results for Boosting can be derived for margin-learning as well, using covering-numbers arguments as discussed in Schapire et al. (1998). The theorem above leads to the following conclusion.

Corollary 31 *If there exists a one-sided PAC-learning algorithm for $\mathcal{H}$ with polynomial run-time in $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$, then there exists a PAC-learning algorithm for classical MIL on $\mathcal{H}$, which has polynomial run-time in r,$\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$.*

Corollary 31 is similar in structure to Theorem 1: Both state that if the single-instance problem is solvable with one-sided error, then the realizable MIL problem is solvable. Theorem 1 applies only to bags with statistically independent instances, while Corollary 31 applies to bags drawn from an arbitrary distribution. The assumption of Theorem 1 is similarly weaker, as it only requires that the single-instance PAC-learning algorithm handle random one-sided noise, while Corollary 31 requires that the single-instance algorithm handle arbitrary one-sided noise. Of course, Corollary 31 does not contradict the hardness result provided for APRs in Auer et al. (1998). Indeed, this hardness result states that if there exists a MIL algorithm for d-dimensional APRs which is polynomial in both r and d, then $\mathcal{RP} = \mathcal{NP}$. Our result does not imply that such an algorithm exists, since there is no known agnostic or one-sided PAC-learning algorithm for APRs which is polynomial in d.

7.3.1 EXAMPLE: HALF-SPACES

We have shown a simple and general way, independent of hypothesis class, to create a PAC-learning algorithm for classical MIL from a learning algorithm that runs on single instances. Whenever an appropriate polynomial algorithm exists for the non-MIL learning problem, the resulting MIL algorithm will also be polynomial in r. To illustrate, consider for instance the algorithm proposed in Shalev-Shwartz et al. (2010). This algorithm is an agnostic PAC-learner of fuzzy kernelized half-spaces with an L-Lipschitz transfer function, for some constant $L > 0$. Its time complexity and sample-complexity are at most $\text{poly}((\frac{L}{\varepsilon})^L \cdot \ln(\frac{1}{\delta}))$. Since this complexity bound is polynomial in $1/\varepsilon$ and in $1/\delta$, this algorithm can serve as the algorithm $\mathcal{B}$ in Theorem 30, and Corollary 31 holds. Thus, we can generate an algorithm for PAC-learning MIL with complexity that depends directly on the complexity of this learner, and is polynomial in r, $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$. The full MIL algorithm for fuzzy kernelized half-spaces can thus be described as follows: Run `AdaBoost`* with the weak learner $\texttt{MILearn}^{O^{\mathcal{B}}_{\varepsilon,\delta}}$, where `MILearn` is listed in Algorithm 1, $O^{\mathcal{B}}_{\varepsilon,\delta}$ is listed in Algorithm 2, and $\mathcal{B}$ is the agnostic PAC-learner from Shalev-Shwartz et al. (2010). The input to `AdaBoost`* is a labeled sample of bags, and the output is a real-valued classifier for bags.

More generally, using the construction we proposed here, any advancement in the development of algorithms for agnostic or one-sided learning of any hypothesis class translates immediately to an algorithm for PAC-learning MIL with the same hypothesis class, and with corresponding complexity guarantees.

8. Conclusions

In this work we have provided a new theoretical analysis for Multiple Instance Learning with any underlying hypothesis class. We have shown that the dependence of the sample complexity of

generalized MIL on the number of instances in a bag is only poly-logarithmic, thus implying that the statistical performance of MIL is only mildly sensitive to the size of the bag. The analysis includes binary hypotheses, real-valued hypotheses, and margin learning, all of which are used in practice in MIL applications. Our sample complexity results can be summarized as follows, where d is the VC dimension or pseudo-dimension of the underlying hypothesis class, and r is the maximal/average bag size.

- The VC dimension of binary MIL is $O(d\log(r))$.
- For non-trivial bag functions, there are hypothesis classes such that the VC dimension of binary MIL is $\Omega(d\log(r))$.
- The VC dimension of binary MIL with separating hyperplanes in dimension d is $\Omega(d\log(r))$.
- The pseudo-dimension of binary MIL for bag functions that are extensions of monotone Boolean functions is $O(d\log(r))$.
- Covering numbers for MIL hypotheses with Lipschitz bag functions can be bounded by covering numbers for the single instance hypothesis class.
- The fat-shattering dimension of real-valued MIL with Lipschitz bag-functions is poly-logarithmic in the bag size and quasilinear in the fat shattering dimension of the single instance hypothesis class.
- The Rademacher complexity of binary MIL with a bounded average bag size is $O(\sqrt{d\log(r)/m})$ where m is the sample size.
- The Rademacher complexity of real-valued MIL with a Lipschitz loss function and a Lipschitz bag function is upper bounded by a logarithmic dependence on the average bag size and a quasilinear dependence on the Rademacher complexity of the instance hypothesis class.

For classical MIL, where the bag-labeling function is the Boolean OR, and for its natural extension to max, we have presented a new learning algorithm, that classifies bags by executing a learning algorithm designed for single instances. This algorithm provably PAC-learns MIL. In both the sample complexity analysis and the computational analysis, we have shown tight connections between classical supervised learning and Multiple Instance Learning, which holds regardless of the underlying hypothesis class.

Many interesting open problems remain for the generic analysis of MIL. In particular, our results hold under certain assumptions on the bag functions. An interesting open question is whether these assumptions are necessary, or whether useful results can be achieved for other classes of bag functions. Another interesting question is how additional structure within a bag, such as sparsity, may affect the statistical and computational feasibility of MIL. These interesting problems are left for future research.

Acknowledgments

The authors thank Shai Shalev-Shwartz and Nati Srebro for helpful discussions. Enlightening comments from Phil Long and Sally Goldman on the preliminary version of this work have helped

improve the presentation immensely. During this work Sivan Sabato was supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities. This work is partly supported by the Gatsby Charitable Foundation.

Appendix A. Proof of Theorem 27

The first step in providing a guarantee for the edge achieved by `MILearn`, is to prove a guarantee for the edge achieved on the bag sample by the hypothesis returned by $\mathcal{A}$ in step (3) of the algorithm. This is done in the following lemma.

Lemma 32 *Assume $\psi : [-1,+1]^{(R)} \to [-1,+1]$ is an (a,b,c,d)-bounded bag function with $0 < a \leq c$, and denote $Z = \frac{c}{a}$. Consider running the algorithm `MILearn` with a weighted bag sample $\overline{S}$ of total weight 1. Let h_I be the hypothesis returned by the oracle $\mathcal{A}$ in step (3) of `MILearn`. Let W be the total weight of the sample S_I created in `MILearn`, step (2). Then*

1. *If $\mathcal{A}$ is ε-optimal,*

$$\Gamma(\overline{h}_I, \overline{S}) \geq Z\gamma^* + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d - \varepsilon W.$$

2. *If $\mathcal{A}$ is one-sided-ε-optimal, and $\psi(z_1,\ldots,z_n) = -1$ only if $z_1 = \ldots = z_n = -1$, then*

$$\Gamma(\overline{h}_I, \overline{S}) \geq \frac{1}{Z}\gamma^*_+ + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d + Z - \frac{1}{Z} - \varepsilon W.$$

Proof For all $h \in \mathcal{H}$, and for all $\mathbf{x} = (x_1,\ldots,x_n) \in \mathcal{X}^{(R)}$ we have $\overline{h}(\mathbf{x}) = \psi(h(x_1),\ldots,h(x_n))$. Since ψ is (a,b,c,d)-bounded, it follows that

$$a\sum_{x\in\mathbf{x}} h(x) + b \leq \overline{h}(\mathbf{x}) \leq c\sum_{x\in\mathbf{x}} h(x) + d. \tag{30}$$

In addition, since a and c are positive we also have

$$(\overline{h}(\mathbf{x}) - d)/c \leq \sum_{x\in\mathbf{x}} h(x) \leq (\overline{h}(\mathbf{x}) - b)/a. \tag{31}$$

Assume the input bag sample is $\overline{S} = \{(w_i, \mathbf{x}_i, y_i)\}_{i\in[m]}$. Denote $I_+ = \{i \in [m] \mid y_i = +1\}$ and $I_- = \{i \in [m] \mid y_i = -1\}$. Let $h \in \mathcal{H}$ be a hypothesis. We have

$$\begin{aligned}
\Gamma(\overline{h}, \overline{S}) &= \sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i) - \sum_{i\in I_-} w_i \overline{h}(\mathbf{x}_i) \\
&\geq \sum_{i\in I_+} w_i (a\sum_{x\in\mathbf{x}_i} h(x) + b) - \sum_{i\in I_-} w_i (c\sum_{x\in\mathbf{x}_i} h(x) + d) \qquad (32)\\
&= \sum_{i\in I_+} w_i a \sum_{x\in\mathbf{x}_i} h(x) - \sum_{i\in I_-} w_i c \sum_{x\in\mathbf{x}_i} h(x) + \sum_{i\in I_+} w_i b - \sum_{i\in I_-} w_i d. \qquad (33)
\end{aligned}$$

line (32) follows from Equation (30). As evident by steps (1,2) of `MILearn`, In the sample S_I all instances from positive bags have weight $\alpha(+1) = a$, and all instances from negative bags have weight $\alpha(-1) = c$. Therefore

$$\Gamma(h, S_I) = \sum_{i\in[m]} \sum_{x\in\mathbf{x}_i} w_i y_i \alpha(y_i) h(x) = \sum_{i\in I_+} w_i a \sum_{x\in\mathbf{x}_i} h(x) - \sum_{i\in I_-} w_i c \sum_{x\in\mathbf{x}_i} h(x).$$

Combining this equality with Equation (33) we get

$$\Gamma(\overline{h},\overline{S}) \geq \Gamma(h,S_I) + \sum_{i\in I_+} w_i b - \sum_{i\in I_-} w_i d.$$

Since $\sum_{i\in I_+} w_i = W_+$ and $\sum_{i\in I_-} w_i = W_- = 1 - W_+$, it follows that

$$\Gamma(\overline{h},\overline{S}) \geq \Gamma(h,S_I) + bW_+ - dW_- = \Gamma(h,S_I) + (b+d)W_+ - d. \tag{34}$$

Now, for any hypothesis h we can conclude from Equation (31) that

$$\begin{aligned}
\Gamma(h,S_I) &= \sum_{i\in I_+} aw_i \sum_{x\in\mathbf{x}_i} h(x) - \sum_{i\in I_-} cw_i \sum_{x\in\mathbf{x}_i} h(x) \\
&\geq \sum_{i\in I_+} aw_i(\overline{h}(\mathbf{x}_i) - d)/c - \sum_{i\in I_-} cw_i(\overline{h}(\mathbf{x}_i) - b)/a \\
&= \sum_{i\in I_+} \frac{a}{c} w_i \overline{h}(\mathbf{x}_i) - \sum_{i\in I_-} \frac{c}{a} w_i \overline{h}(\mathbf{x}_i) - \sum_{i\in I_+} adw_i/c + \sum_{i\in I_-} cbw_i/a \\
&= \frac{c}{a}\Gamma(\overline{h},\overline{S}) + (\frac{a}{c} - \frac{c}{a}) \sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i) - \frac{ad}{c} W_+ + \frac{cb}{a} W_- \\
&= \frac{c}{a}\Gamma(\overline{h},\overline{S}) + (\frac{a}{c} - \frac{c}{a}) \sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i) - (\frac{ad}{c} + \frac{cb}{a}) W_+ + \frac{cb}{a}.
\end{aligned}$$

In the last equality we used the fact that $W_- = 1 - W_+$. Since $Z = \frac{c}{a}$, it follows that

$$\Gamma(h,S_I) \geq Z\Gamma(\overline{h},\overline{S}) + (\frac{1}{Z} - Z) \sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i) - (\frac{d}{Z} + Zb) W_+ + Zb. \tag{35}$$

We will now lower-bound the right-hand-side of Equation (35). Note that $\frac{1}{Z} - Z \leq 0$ since $c \geq a$. Therefore we need an upper bound for $\sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i)$. We consider each of the two cases in the statement of the lemma separately.

A.0.2 CASE 1: $\mathcal{A}$ IS ε-OPTIMAL

We have $\sum_{i\in I_+} w_i \overline{h}(\mathbf{x}_i) \leq \sum_{i\in I_+} w_i = W_+$. Therefore, by Equation (35) for any $h \in \mathcal{H}$

$$\Gamma(h,S_I) \geq Z\Gamma(\overline{h},\overline{S}) + (\frac{1}{Z} - Z - \frac{d}{Z} - Zb) W_+ + Zb. \tag{36}$$

For a natural n, set h^n_* such that $\Gamma(\overline{h}^n_*,\overline{S}) \geq \gamma^* - \frac{1}{n}$. We have (see explanations below)

$$\begin{aligned}
\Gamma(\overline{h}_I,\overline{S}) &\geq \Gamma(h_I,S_I) + (b+d)W_+ - d && (37)\\
&\geq \Gamma(h^n_*,S_I) + (b+d)W_+ - d - \varepsilon W && (38)\\
&\geq Z\Gamma(\overline{h}^n_*,\overline{S}) + (\frac{1}{Z} - Z - \frac{d}{Z} - Zb)W_+ + Zb + (b+d)W_+ - d - \varepsilon W && (39)\\
&= Z\Gamma(\overline{h}^n_*,\overline{S}) + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d - \varepsilon W \\
&\geq Z(\gamma^* - \frac{1}{n}) + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d - \varepsilon W.
\end{aligned}$$

Equation (37) is a restatement of Equation (34). Equation (38) follows from the ε-optimality of $\mathcal{A}$. Equation (39) follows from Equation (36). By taking $n \to \infty$, this inequality proves case (1) of the lemma.

A.0.3 CASE 2: $\mathcal{A}$ IS ONE-SIDED-ε-OPTIMAL

We have $\sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) \leq \sum_{i\in I_+} w_i = W_+$. Let $\overline{h} \in \Omega(\overline{\mathcal{H}},\overline{S})$. Then for all $i \in I_-$, $\overline{h}(\mathbf{x}_i) = -1$. Therefore

$$\begin{aligned}\Gamma(\overline{h},\overline{S}) &= \sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) - \sum_{i\in I_-} w_i\overline{h}(\mathbf{x}_i)\\ &= \sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) + \sum_{i\in I_-} w_i\\ &= \sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) + W_-.\end{aligned}$$

Therefore $\sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) = \Gamma(\overline{h},\overline{S}) - W_- = \Gamma(\overline{h},\overline{S}) + W_+ - 1$. Combining this with Equation (35) we get

$$\begin{aligned}\Gamma(h,S_I) &\geq Z\Gamma(\overline{h},\overline{S}) + (\frac{1}{Z} - Z)\sum_{i\in I_+} w_i\overline{h}(\mathbf{x}_i) - (\frac{d}{Z} + Zb)W_+ + Zb\\ &= Z\Gamma(\overline{h},\overline{S}) + (\frac{1}{Z} - Z)(\Gamma(\overline{h},\overline{S}) + W_+ - 1) - (\frac{d}{Z} + Zb)W_+ + Zb.\\ &= \frac{1}{Z}\Gamma(\overline{h},\overline{S}) + (\frac{1}{Z} - Z - \frac{d}{Z} - Zb)W_+ + Zb - \frac{1}{Z} + Z. \qquad (40)\end{aligned}$$

For a natural n, set $\overline{h}^n_+ \in \Omega(\overline{\mathcal{H}},\overline{S})$ such that $\Gamma(\overline{h}^n_+,\overline{S}) \geq \gamma^*_+ - \frac{1}{n}$. For all bags $i \in I_-$, $\overline{h}^n_+(\mathbf{x}_i) = -1$. Thus $\psi(h^n_+(x_i[1]),\ldots,h^n_+(x_i[|\mathbf{x}_i|])) = -1$. By the assumption on ψ in case (2) of the lemma, this implies that for all $i \in I_-$, $j \in [|\mathbf{x}_i|]$, $h^n_+(x_i[j]) = -1$. Therefore $h^n_+ \in \Omega(\mathcal{H},S_I)$. We have (see explanations below)

$$\begin{aligned}\Gamma(\overline{h}_I,\overline{S}) &\geq \Gamma(h_I,S_I) + (b+d)W_+ - d \qquad (41)\\ &\geq \Gamma(h^n_+,S_I) + (b+d)W_+ - d - \varepsilon W \qquad (42)\\ &\geq \frac{1}{Z}\Gamma(\overline{h}^n_+,\overline{S}) + (\frac{1}{Z} - Z - \frac{d}{Z} - Zb)W_+ + Zb - \frac{1}{Z} + Z + (b+d)W_+ - d - \varepsilon W \qquad (43)\\ &= \frac{1}{Z}\Gamma(\overline{h}^n_+,\overline{S}) + (\frac{1}{Z} - Z + (1-\frac{1}{Z})(d - Zb))W_+ + Zb - d + Z - \frac{1}{Z} - \varepsilon W\\ &\geq \frac{1}{Z}(\gamma^*_+ - \frac{1}{n}) + (\frac{1}{Z} - Z + (1-\frac{1}{Z})(d - Zb))W_+ + Zb - d + Z - \frac{1}{Z} - \varepsilon W.\end{aligned}$$

Equation (41) is a restatement of Equation (34). Equation (42) follows from the one-sided-ε-optimality of $\mathcal{A}$ and the fact that $h^n_+ \in \Omega(\mathcal{H},S_I)$. Equation (43) follows from Equation (40). By considering $n \to \infty$, this proves the second part of the lemma. ■

Proof [of Theorem 27] `MILearn` selects the hypothesis with the best edge on $\overline{S}$ between $\overline{h}_I$ and $\mathbf{h}_{\text{pos}}$. Therefore

$$\Gamma(h_\circ,\overline{S}) = \max(\Gamma(\mathbf{h}_{\text{pos}},\overline{S}),\Gamma(\overline{h}_I,\overline{S})).$$

We have

$$\Gamma(\mathbf{h}_{\text{pos}},\overline{S}) = \sum_{i\in[m]} w_i y_i \mathbf{h}_{\text{pos}}(\mathbf{x}_i) = \sum_{i\in[m]} w_i y_i = W_+ - W_- = 2W_+ - 1.$$

Thus

$$\Gamma(h_\circ, \overline{S}) = \max(2W_+ - 1, \Gamma(\overline{h}_I, \overline{S})). \tag{44}$$

We now lower-bound $\Gamma(h_\circ, \overline{S})$ by bounding $\Gamma(\overline{h}_I, \overline{S})$ separately for the two cases of the theorem. Let W be the total weight of S_I. Since $R \subseteq [r]$, $a \leq c$, and $\sum_{i\in[m]} w_i = 1$, we have

$$W = \sum_{i:y_i=+1} \sum_{x\in\mathbf{x}_i} a w_i + \sum_{i:y_i=-1} \sum_{x\in\mathbf{x}_i} c w_i \leq rc \sum_{i\in[m]} w_i = rc \tag{45}$$

A.0.4 CASE 1: $\mathcal{A}$ IS ε-OPTIMAL

From Lemma 32 and Equation (45) we have

$$\begin{aligned}
\Gamma(\overline{h}_I, \overline{S}) &\geq Z\gamma^* + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d - rc\varepsilon \\
&= Z\gamma^* + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(Z - 1 + \eta))W_+ - (Z - 1 + \eta) - rc\varepsilon \\
&= Z\gamma^* + (\eta - 2)(1 - \frac{1}{Z})W_+ + 1 - \eta - Z - rc\varepsilon.
\end{aligned}$$

The second line follows from the assumption $d - Zb - Z + 1 = \eta$. Combining this with Equation (44) we get

$$\Gamma(h_\circ, \overline{S}) \geq \max\{2W_+ - 1, Z\gamma^* + (\eta - 2)(1 - \frac{1}{Z})W_+ + 1 - \eta - Z - rc\varepsilon\}.$$

The right-hand-side is minimal when the two expressions in the maximum are equal. This occurs when

$$W_+ = W_\circ \triangleq \frac{Z\gamma^* + 2 - \eta - Z - rc\varepsilon}{2 + (2 - \eta)(1 - \frac{1}{Z})}.$$

Therefore, for any value of W_+

$$\Gamma(h_\circ, \overline{S}) \geq 2W_\circ - 1 = \frac{Z\gamma^* - Z + \frac{1}{Z} - \frac{\eta}{2}(1 + \frac{1}{Z}) - rc\varepsilon}{1 + (1 - \frac{\eta}{2})(1 - \frac{1}{Z})}.$$

Case 2: $\mathcal{A}$ is one-sided-ε-optimal From Lemma 32 and Equation (45) we have

$$\begin{aligned}
\Gamma(\overline{h}_I, \overline{S}) &\geq \frac{1}{Z}\gamma_+^* + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(d - Zb))W_+ + Zb - d + Z - \frac{1}{Z} - rc\varepsilon \\
&= \frac{1}{Z}\gamma_+^* + (\frac{1}{Z} - Z + (1 - \frac{1}{Z})(Z - 1 + \eta))W_+ - (Z - 1 + \eta) + Z - \frac{1}{Z} - rc\varepsilon \\
&= \frac{1}{Z}\gamma_+^* + (\eta - 2)(1 - \frac{1}{Z})W_+ + 1 - \eta - \frac{1}{Z} - rc\varepsilon.
\end{aligned}$$

The second line follows from the assumption $d - Zb = Z - 1 + \eta$. Combining this with Equation (44) we get

$$\Gamma(h_\circ, \overline{S}) \geq \max\{2W_+ - 1, \frac{1}{Z}\gamma_+^* + (\eta - 2)(1 - \frac{1}{Z})W_+ + 1 - \eta - \frac{1}{Z} - rc\varepsilon\}.$$

The right-hand-side is minimal when the two expressions in the maximum are equal. This occurs when

$$W_+ = W_\circ \triangleq \frac{\gamma_+^* - 1 + (2 - \eta - rc\varepsilon)Z}{2Z + (2 - \eta)(Z - 1)}.$$

Substituting W_+ for $W_\circ$ in the lower bound, we get

$$\Gamma(h_\circ, \overline{S}) \geq 2W_\circ - 1 = \frac{\gamma_+^* - \frac{\eta}{2}(Z+1) - rc\varepsilon Z}{2Z - 1 - \frac{\eta}{2}(Z-1)}.$$

■

References

N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pages 292–301. IEEE, 1993.

N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *J. ACM*, 44(4):615–631, 1997.

S. Andrews. *Learning from Ambiguous Examples*. PhD thesis, Brown University, May 2007.

S. Andrews and T. Hofmann. Multiple-instance learning via disjunctive programming boosting. In *Advances in Neural Information Processing Systems 16*, 2003.

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems 15*, pages 561–568, 2002.

M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

P. Auer, P. M. Long, and A. Srinivasan. Approximating hyper-rectangles: learning and pseudorandom sets. *Journal of Computer and System Sciences*, 57(3):376–388, 1998.

B. Babenko, N. Verma, P. Dollar, and S. Belongie. Multiple instance learning with manifold bags. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 81–88, 2011.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

P. L. Bartlett, S. R. Kulkarni, and S. E. Posner. Covering numbers for real-valued function classes. *IEEE Transactions on Information Theory*, 43(5):1721–1724, 1997.

A. Blum and A. Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30 (1):23–29, 1998.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

R. M. Dudley. Central limit theorems for empirical measures. *The Annals of Probability*, 6(6): 899–929, 1978.

R.M. Dudley. The sizes of compact subsets of hilbert space and continuity of gaussian processes. *Journal of Functional Analysis*, 1(3):290 – 330, 1967.

Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997.

T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 179–186, 2002.

D. Haussler and P. M. Long. A generalization of Sauer's lemma. *Journal of Combinatorial Theory, Series A*, 71(2):219–240, 1995.

P. M. Long and L. Tan. PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples. *Machine Learning*, 30(1):7–21, 1998. ISSN 0885-6125.

O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems 10*, pages 570–576, 1998.

O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 341–349, 1998.

S. Mendelson. Rademacher averages and phase transitions in glivenko-cantelli classes. *IEEE Transactions on Information Theory*, 48(1):251–263, 2002.

S. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York, 1996.

L. Pitt and L. G. Valiant. Computational limitations on learning from examples. Technical report, Harvard University Aiken Computation Laboratory, July 1986.

D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

L. De Raedt. Attribute-value learning versus inductive logic programming: The missing links (extended abstract). In *Proceedings of the 8th International Workshop on Inductive Logic Programming*, pages 1–8, London, UK, 1998. Springer-Verlag.

G. Rätsch and M. K. Warmuth. Efficient margin maximizing with boosting. *Journal of Machine Learning Research*, 6:2131–2152, December 2005.

S. Sabato and N. Tishby. Homogeneous multi-instance learning with arbitrary dependence. In *Proceedings of the Twenty Second Annual Conference on Computational Learning Theory*, 2009.

S. Sabato, N. Srebro, and N. Tishby. Reducing label complexity by learning from bags. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 685–692, 2010.

N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory Series A*, 13:145–147, 1972.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):1–40, 1999.

R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, October 1998.

S. Shalev-Shwartz, O. Shamir, and K. Learning kernel-based halfspaces with the zero-one loss. In *Proceedings of the Twenty Third Annual Conference on Computational Learning Theory*, 2010.

N. Srebro, K. Sridharan, and A. Tewari. Smoothness, low-noise and fast rates. *CoRR*, abs/1009.3896, 2010.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, XVI(2):264–280, 1971.

J. von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

N. Weidmann, E. Frank, and B. Pfahringer. A two-level learning method for generalized multi-instance problems. In *Proceedings of the European conference on machine learning*, pages 468–479, 2003.

Q. Zhang and S.A. Goldman. EM-DD: An improved multiple-instance learning technique. In *Advances in Neural Information Processing Systems 14*, 2001.

Z. Zhou, K. Jiang, and M. Li. Multi-instance learning based web mining. *Applied Intelligence*, 22 (2):135–147, 2005.

Journal of Machine Learning Research 13 (2012) 3041-3074 Submitted 8/10; Revised 11/11; Published 10/12

Finite-Sample Analysis of Least-Squares Policy Iteration

Alessandro Lazaric ALESSANDRO.LAZARIC@INRIA.FR
Mohammad Ghavamzadeh MOHAMMAD.GHAVAMZADEH@INRIA.FR
Rémi Munos REMI.MUNOS@INRIA.FR
INRIA Lille - Nord Europe, Team SequeL
40 Avenue Halley
59650 Villeneuve d'Ascq, France

Editor: Ronald Parr

Abstract

In this paper, we report a performance bound for the widely used least-squares policy iteration (LSPI) algorithm. We first consider the problem of policy evaluation in reinforcement learning, that is, learning the value function of a fixed policy, using the least-squares temporal-difference (LSTD) learning method, and report finite-sample analysis for this algorithm. To do so, we first derive a bound on the performance of the LSTD solution evaluated at the states generated by the Markov chain and used by the algorithm to learn an estimate of the value function. This result is general in the sense that no assumption is made on the existence of a stationary distribution for the Markov chain. We then derive generalization bounds in the case when the Markov chain possesses a stationary distribution and is β-mixing. Finally, we analyze how the error at each policy evaluation step is propagated through the iterations of a policy iteration method, and derive a performance bound for the LSPI algorithm.

Keywords: Markov decision processes, reinforcement learning, least-squares temporal-difference, least-squares policy iteration, generalization bounds, finite-sample analysis

1. Introduction

Least-squares temporal-difference (LSTD) learning (Bradtke and Barto, 1996; Boyan, 1999) is a widely used algorithm for prediction in general, and in the context of reinforcement learning (RL), for learning the value function V^π of a given policy π. LSTD has been successfully applied to a number of problems especially after the development of the least-squares policy iteration (LSPI) algorithm (Lagoudakis and Parr, 2003), which extends LSTD to control by using it in the policy evaluation step of policy iteration. More precisely, LSTD computes the fixed point of the operator $\Pi\mathcal{T}$, where $\mathcal{T}$ is the Bellman operator and Π is the projection operator in a linear function space $\mathcal{F}$. Although LSTD and LSPI have been widely used in the RL community, a finite-sample analysis of LSTD, that is, performance bounds in terms of the number of samples, the space $\mathcal{F}$, and the characteristic parameters of the MDP at hand, is still missing.

Most of the theoretical work analyzing LSTD have been focused on the model-based case, where explicit models of the reward function and the dynamics are available. In particular, Tsitsiklis and Van Roy (1997) showed that the distance between the LSTD solution and the value function V^π is bounded by the distance between V^π and its closest approximation in the linear space, multiplied by a constant which increases as the discount factor approaches 1. In this bound, it is assumed that the Markov chain possesses a stationary distribution ρ^π and the distances are measured according

to ρ^π. Yu (2010) has extended this analysis and derived an asymptotic convergence analysis for off-policy LSTD(λ), that is when the samples are collected following a behavior policy different from the policy π under evaluation. Finally, on-policy empirical LSTD has been analyzed by Bertsekas (2007). His analysis reveals a critical dependency on the inverse of the smallest eigenvalue of the LSTD's A matrix (note that the LSTD solution is obtained by solving a system of linear equations $Ax = b$). Nonetheless, Bertsekas (2007) does not provide a finite-sample analysis of the algorithm. Although these analyses already provide some insights on the behavior of LSTD, asymptotic results do not give a full characterization of the performance of the algorithm when only a finite number of samples is available (which is the most common situation in practice). On the other hand, a finite-sample analysis has a number of important advantages: **1)** unlike in Tsitsiklis and Van Roy (1997), where they assume that model-based LSTD always returns a solution, in a finite-sample analysis we study the characteristics of the actual empirical LSTD fixed point, including its existence, **2)** a finite-sample bound explicitly reveals how the prediction error of LSTD is related to the characteristic parameters of the MDP at hand, such as the discount factor, the dimensionality of the function space $\mathcal{F}$, and the number of samples, **3)** once this dependency is clear, the bound can be used to determine the order of magnitude of the number of samples needed to achieve a desired accuracy.

Recently, several works have been focused on deriving a finite-sample analysis for different RL algorithms. In the following, we review those that are more strictly related to LSTD and to the results reported in this paper. Antos et al. (2008) analyzed the modified Bellman residual (MBR) minimization algorithm for a finite number of samples, bounded function spaces, and a μ-norm that might be different from the norm induced by ρ^π. Although MBR minimization was shown to reduce to LSTD in case of linear spaces, it is not straightforward to extend the finite-sample bounds derived by Antos et al. (2008) to unbounded linear spaces considered by LSTD. Farahmand et al. (2008) proposed a ℓ_2-regularized extension of LSPI and provided finite-sample analysis for the algorithm when the function space is a reproducing kernel Hilbert space (RKHS). In this work, the authors consider the optimization formulation of LSTD (instead of the better known fixed-point formulation) and assume that a generative model of the environment is available. Moreover, the analysis is for ℓ_2-regularized LSTD (LSPI) and also for the case that the function space $\mathcal{F}$ is a RKHS. Pires and Szepesvári (2012) also analyzed a regularized version of LSTD reporting performance bounds for both the on-policy and off-policy case. In this paper, we first report a finite-sample analysis of LSTD. To the best of our knowledge, this is the first complete finite-sample analysis of this widely used algorithm. Our analysis is for a specific implementation of LSTD that we call *pathwise LSTD*. Pathwise LSTD has two specific characteristics: **1)** it takes a single trajectory generated by the Markov chain induced by policy π as input, and **2)** it uses the pathwise Bellman operator (precisely defined in Section 3), which is defined to be a contraction w.r.t. the empirical norm. We first derive a bound on the performance of the pathwise LSTD solution for a setting that we call *Markov design*. In this setting, the performance is evaluated at the points used by the algorithm to learn an estimate of V^π. This bound is general in the sense that no assumption is made on the existence of a stationary distribution for the Markov chain. Then, in the case that the Markov chain admits a stationary distribution ρ^π and is β-mixing, we derive generalization bounds w.r.t. the norm induced by ρ^π. Finally, along the lines of Antos et al. (2008), we show how the LSTD error is propagated through the iterations of LSPI, and under suitable assumptions, derive a performance bound for the LSPI algorithm.

Besides providing a full finite-sample analysis of LSPI, the major insights gained by the analysis in the paper may be summarized as follows. The first result is about the existence of the LSTD

solution and its performance. In Theorem 1 we show that with a slight modification of the empirical Bellman operator $\widehat{\mathcal{T}}$ (leading to the definition of pathwise LSTD), the operator $\widehat{\Pi}\widehat{\mathcal{T}}$ (where $\widehat{\Pi}$ is an empirical projection operator) always has a fixed point $\hat{v}$, even when the sample-based Gram matrix is not invertible and the Markov chain does not admit a stationary distribution. In this very general setting, it is still possible to derive a bound for the performance of the LSTD solution, $\hat{v}$, evaluated at the states of the trajectory used by the algorithm. Moreover, an analysis of the bound reveals a critical dependency on the smallest strictly positive eigenvalue ν_n of the sample-based Gram matrix. Then, in the case in which the Markov chain has a stationary distribution ρ^π, it is possible to relate the value of ν_n to the smallest eigenvalue of the Gram matrix defined according to ρ^π. Furthermore, it is possible to generalize the previous performance bound over the entire state space under the measure ρ^π, when the samples are drawn from a stationary β-mixing process (Theorem 5). It is important to note that the asymptotic bound obtained by taking the number of samples, n, to infinity is equal (up to constants) to the bound in Tsitsiklis and Van Roy (1997) for model-based LSTD. Furthermore, a comparison with the bounds in Antos et al. (2008) shows that we successfully leverage on the specific setting of LSTD: **1)** the space of functions is linear, and **2)** the distribution used to evaluate the performance is the stationary distribution of the Markov chain induced by the policy, and obtain a better bound both in terms of **1)** estimation error, a rate of order $O(1/n)$ instead of $O(1/\sqrt{n})$ for the squared error, and **2)** approximation error, the minimal distance between the value function V^π and the space $\mathcal{F}$ instead of the inherent Bellman errors of $\mathcal{F}$. The extension in Theorem 6 to the case in which the samples belong to a trajectory generated by a fast mixing Markov chain shows that it is possible to achieve the same performance as in the case of stationary β-mixing processes. Finally, the analysis of LSPI reveals the need for several critical assumptions on the stationary distributions of the policies that are greedy w.r.t. to the functions in the linear space $\mathcal{F}$. These assumptions seem unavoidable when an on-policy method is used at each iteration, and whether they can be removed or relaxed in other settings is still an open question. This paper extends and improves over the conference paper by Lazaric et al. (2010) in the following respects: **1)** we report the full proofs and technical tools for all the theoretical results, thus making the paper self-contained, **2)** we extend the LSTD results to LSPI showing how the approximation errors are propagated through iterations.

The rest of the paper is organized as follows. In Section 2, we set the notation used throughout the paper. In Section 3, we introduce pathwise LSTD by a minor modification to the standard LSTD formulation in order to guarantee the existence of at least one solution. In Section 4, we introduce the Markov design setting for regression and report an empirical bound for LSTD. In Section 5, we show how the Markov design bound of Section 4 may be extended when the Markov chain admits a stationary distribution. In Section 6, we analyze how the LSTD error is propagated through the iterations of LSPI and derive a performance bound for the LSPI algorithm. Finally in Section 7, we draw conclusions and discuss some possible directions for future work.

2. Preliminaries

For a measurable space with domain $\mathcal{X}$, we let $\mathcal{S}(\mathcal{X})$ and $\mathcal{B}(\mathcal{X};L)$ denote the set of probability measures over $\mathcal{X}$, and the space of bounded measurable functions with domain $\mathcal{X}$ and bound $0 < L < \infty$, respectively. For a measure $\rho \in \mathcal{S}(\mathcal{X})$ and a measurable function $f : \mathcal{X} \to \mathbb{R}$, we define the $\ell_2(\rho)$-norm of f, $||f||_\rho$, and for a set of n points $X_1, \ldots, X_n \in \mathcal{X}$, we define the empirical norm $||f||_n$

as

$$||f||_\rho^2 = \int f(x)^2 \rho(dx) \qquad \text{and} \qquad ||f||_n^2 = \frac{1}{n}\sum_{t=1}^{n} f(X_t)^2.$$

The supremum norm of f, $||f||_\infty$, is defined as $||f||_\infty = \sup_{x\in\mathcal{X}} |f(x)|$.

We consider the standard RL framework (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998) in which a learning agent interacts with a stochastic environment and this interaction is modeled as a discrete-time discounted Markov decision process (MDP). A discounted MDP is a tuple $\mathcal{M} = \langle \mathcal{X}, \mathcal{A}, r, P, \gamma \rangle$ where the state space $\mathcal{X}$ is a bounded closed subset of the s-dimensional Euclidean space, $\mathcal{A}$ is a finite ($|\mathcal{A}| < \infty$) action space, the reward function $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is uniformly bounded by $R_{\max}$, the transition kernel P is such that for all $x \in \mathcal{X}$ and $a \in \mathcal{A}$, $P(\cdot|x,a)$ is a distribution over $\mathcal{X}$, and $\gamma \in (0,1)$ is a discount factor. A deterministic policy $\pi : \mathcal{X} \to \mathcal{A}$ is a mapping from states to actions. For a given policy π, the MDP $\mathcal{M}$ is reduced to a Markov chain $\mathcal{M}^\pi = \langle \mathcal{X}, R^\pi, P^\pi, \gamma \rangle$ with the reward function $R^\pi(x) = r\big(x, \pi(x)\big)$, transition kernel $P^\pi(\cdot|x) = P\big(\cdot\,|x, \pi(x)\big)$, and stationary distribution ρ^π (if it admits one). The value function of a policy π, V^π, is the unique fixed-point of the Bellman operator $\mathcal{T}^\pi : \mathcal{B}(\mathcal{X}; V_{\max} = \frac{R_{\max}}{1-\gamma}) \to \mathcal{B}(\mathcal{X}; V_{\max})$ defined by

$$(\mathcal{T}^\pi V)(x) = R^\pi(x) + \gamma \int_{\mathcal{X}} P^\pi(dy|x) V(y).$$

We also define the optimal value function V^* as the unique fixed-point of the optimal Bellman operator $\mathcal{T}^* : \mathcal{B}(\mathcal{X}; V_{\max}) \to \mathcal{B}(\mathcal{X}; V_{\max})$ defined by

$$(\mathcal{T}^* V)(x) = \max_{a\in\mathcal{A}} \left[r(x,a) + \gamma \int_{\mathcal{X}} P(dy|x,a) V(y) \right].$$

In the following sections, to simplify the notation, we remove the dependency to the policy π and use R, P, V, ρ, and $\mathcal{T}$ instead of R^π, P^π, V^π, ρ^π, and $\mathcal{T}^\pi$ whenever the policy π is fixed and clear from the context.

To approximate the value function V, we use a linear approximation architecture with parameters $\alpha \in \mathbb{R}^d$ and basis functions $\varphi_i \in \mathcal{B}(\mathcal{X}; L)$, $i = 1, \ldots, d$. We denote by $\phi : \mathcal{X} \to \mathbb{R}^d$, $\phi(\cdot) = \big(\varphi_1(\cdot), \ldots, \varphi_d(\cdot)\big)^\top$ the feature vector, and by $\mathcal{F}$ the linear function space spanned by the basis functions φ_i. Thus $\mathcal{F} = \big\{ f_\alpha \mid \alpha \in \mathbb{R}^d \text{ and } f_\alpha(\cdot) = \phi(\cdot)^\top \alpha \big\}$.

Let $(X_1, \ldots, X_n)$ be a sample path (trajectory) of size n generated by the Markov chain $\mathcal{M}^\pi$. Let $v \in \mathbb{R}^n$ and $r \in \mathbb{R}^n$ be such that $v_t = V(X_t)$ and $r_t = R(X_t)$ be the value vector and the reward vector, respectively. Also, let $\Phi = [\phi(X_1)^\top; \ldots; \phi(X_n)^\top]$ be the feature matrix defined at the states, and $\mathcal{F}_n = \{\Phi\alpha,\ \alpha \in \mathbb{R}^d\} \subset \mathbb{R}^n$ be the corresponding vector space. We denote by $\widehat{\Pi} : \mathbb{R}^n \to \mathcal{F}_n$ the orthogonal projection onto $\mathcal{F}_n$, defined as $\widehat{\Pi} y = \arg\min_{z\in\mathcal{F}_n} ||y - z||_n$, where $||y||_n^2 = \frac{1}{n}\sum_{t=1}^n y_t^2$. Note that the orthogonal projection $\widehat{\Pi} y$ for any $y \in \mathbb{R}^n$ exists and is unique. Moreover, $\widehat{\Pi}$ is a non-expansive mapping w.r.t. the ℓ_2-norm: since the projection is orthogonal and using the Cauchy-Schwarz inequality $||\widehat{\Pi} y - \widehat{\Pi} z||_n^2 = \langle y - z, \widehat{\Pi} y - \widehat{\Pi} z \rangle_n \le ||y - z||_n ||\widehat{\Pi} y - \widehat{\Pi} z||_n$, and thus, we obtain $||\widehat{\Pi} y - \widehat{\Pi} z||_n \le ||y - z||_n$.

3. Pathwise LSTD

Pathwise LSTD (Algorithm 1) is a version of LSTD that takes as input a linear function space $\mathcal{F}$ and a single trajectory $X_1, \ldots, X_n$ generated by following the policy, and returns the fixed-point

Algorithm 1 A pseudo-code for the batch pathwise LSTD algorithm.

Input: Linear space $\mathcal{F} = \text{span}\{\varphi_i, 1 \le i \le d\}$, sample trajectory $\{(x_t, r_t)\}_{t=1}^n$ of the Markov chain

Build the feature matrix $\Phi = [\phi(x_1)^\top; \dots; \phi(x_n)^\top]$
Build the empirical transition matrix $\widehat{P}$: $\widehat{P}_{ij} = \mathbb{I}\{j = i+1, j \neq n\}$
Build matrix $A = \Phi^\top (I - \gamma \widehat{P})\Phi$
Build vector $b = \Phi^\top r$
Return the **pathwise LSTD solution** $\hat{\alpha} = A^+ b$

of the empirical operator $\widehat{\Pi}\widehat{\mathcal{T}}$, where $\widehat{\mathcal{T}} : \mathbb{R}^n \to \mathbb{R}^n$ is the *pathwise Bellman operator* defined as

$$(\widehat{\mathcal{T}} y)_t = \begin{cases} r_t + \gamma y_{t+1} & 1 \le t < n, \\ r_t & t = n. \end{cases}$$

Note that by defining the operator $\widehat{P} : \mathbb{R}^n \to \mathbb{R}^n$ as $(\widehat{P}y)_t = y_{t+1}$ for $1 \le t < n$ and $(\widehat{P}y)_n = 0$, we have $\widehat{\mathcal{T}} y = r + \gamma \widehat{P} y$. The motivation for using the pathwise Bellman operator is that it is γ-contraction in ℓ_2-norm, that is, for any $y, z \in \mathbb{R}^n$, we have

$$||\widehat{\mathcal{T}} y - \widehat{\mathcal{T}} z||_n^2 = ||\gamma \widehat{P}(y - z)||_n^2 \le \gamma^2 ||y - z||_n^2 .$$

Since the orthogonal projection $\widehat{\Pi}$ is non-expansive w.r.t. ℓ_2-norm, from Banach fixed point theorem, there exists a unique fixed-point $\hat{v}$ of the mapping $\widehat{\Pi}\widehat{\mathcal{T}}$, that is, $\hat{v} = \widehat{\Pi}\widehat{\mathcal{T}}\hat{v}$. Since $\hat{v}$ is the unique fixed point of $\widehat{\Pi}\widehat{\mathcal{T}}$, the vector $\hat{v} - \widehat{\mathcal{T}}\hat{v}$ is perpendicular to the space $\mathcal{F}_n$, and thus, $\Phi^\top(\hat{v} - \widehat{\mathcal{T}}\hat{v}) = 0$. By replacing $\hat{v}$ with $\Phi\alpha$, we obtain $\Phi^\top \Phi\alpha = \Phi^\top(r + \gamma \widehat{P}\Phi\alpha)$ and then $\Phi^\top(I - \gamma\widehat{P})\Phi\alpha = \Phi^\top r$. Therefore, by setting $A = \Phi^\top(I - \gamma\widehat{P})\Phi$ and $b = \Phi^\top r$, we recover a $d \times d$ system of equations $A\alpha = b$ similar to the one in the original LSTD algorithm. Note that since the fixed point $\hat{v}$ exists, this system always has at least one solution. We call the solution with minimal norm, $\hat{\alpha} = A^+ b$, where A^+ is the Moore-Penrose pseudo-inverse of A, the pathwise LSTD solution.[1]

Finally, notice that the algorithm reported in Figure 1 may be easily extended to the incremental version of LSTD by incrementally building the inverse of the matrix A as the samples are collected.

4. Markov Design Bound

In Section 3, we defined the pathwise Bellman operator with a slight modification in the definition of the empirical Bellman operator $\widehat{\mathcal{T}}$, and showed that the operator $\widehat{\Pi}\widehat{\mathcal{T}}$ always has a unique fixed point $\hat{v}$. In this section, we derive a bound for the performance of $\hat{v}$ evaluated at the states of the trajectory used by the pathwise LSTD algorithm. We first state the main theorem and we discuss it in a number of remarks. The proofs are postponed at the end of the section.

Theorem 1 *Let $X_1, \dots, X_n$ be a trajectory generated by the Markov chain, and $v, \hat{v} \in \mathbb{R}^n$ be the vectors whose components are the value function and the pathwise LSTD solution at $\{X_t\}_{t=1}^n$, respectively. Then with probability at least $1 - \delta$ (the probability is w.r.t. the random trajectory), we have*

$$||v - \hat{v}||_n \le \frac{1}{\sqrt{1-\gamma^2}} ||v - \widehat{\Pi} v||_n + \frac{1}{1-\gamma} \left[\gamma V_{\max} L \sqrt{\frac{d}{\nu_n}} \left(\sqrt{\frac{8 \log(2d/\delta)}{n}} + \frac{1}{n} \right) \right], \qquad (1)$$

1. Note that whenever the matrix A is invertible $A^+ = A^{-1}$.

where the random variable ν_n is the smallest strictly-positive eigenvalue of the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$.

Remark 1 Theorem 1 provides a bound on the prediction error of the LSTD solution $\hat{v}$ w.r.t. the true value function v on the trajectory $X_1,\ldots,X_n$ used as a training set for pathwise-LSTD. The bound contains two main terms. The first term $||v-\widehat{\Pi}v||_n$ is the *approximation* error and it represents the smallest possible error in approximating v with functions in $\mathcal{F}$. This error cannot be avoided. The second term, of order $O(\sqrt{d/n})$, is the *estimation* error and it accounts for the error due to the use of a finite number of noisy samples and it shows what is the influence of the different elements of the problem (e.g., γ, d, n) on the prediction error and it provides insights about how to tune some parameters. We first notice that the bound suggests that the number of samples n should be significantly bigger than the number of features d in order to achieve a small estimation error. Furthermore, the bound can be used to estimate the number of samples needed to guarantee a desired prediction error ε. In fact, apart from the approximation error, which is unavoidable, we have that $n = O(d/((1-\gamma)^2\varepsilon^2))$ samples are enough to achieve an ε-accurate approximation of the true value function v. We also remark that one might be tempted to reduce the dimensionality d, so as to reduce the sample cost of the algorithm. Nonetheless, this is likely to reduce the approximation capability of $\mathcal{F}$ and thus increase the approximation error.

Remark 2 When the eigenvalues of the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$ are all non-zero, $\Phi^\top\Phi$ is invertible, and thus, $\widehat{\Pi} = \Phi(\Phi^\top\Phi)^{-1}\Phi^\top$. In this case, the uniqueness of $\hat{v}$ implies the uniqueness of $\hat{\alpha}$ since

$$\hat{v} = \Phi\alpha \implies \Phi^\top\hat{v} = \Phi^\top\Phi\alpha \implies \hat{\alpha} = (\Phi^\top\Phi)^{-1}\Phi^\top\hat{v}.$$

On the other hand, when the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$ is not invertible, the system $Ax = b$ may have many solutions. Among all the possible solutions, one may choose the one with minimal norm: $\hat{\alpha} = A^+ b$.

Remark 3 Note that in case there exists a constant $\nu > 0$, such that with probability $1-\delta'$ all the eigenvalues of the sample-based Gram matrix are lower-bounded by ν, Equation 1 (with ν_n replaced by ν) holds with probability at least $1-(\delta+\delta')$ (see Section 5.1 for a case in which such constant ν can be computed and it is related to the smallest eigenvalue of the model based Gram matrix).

Remark 4 Theorem 1 provides a bound without any reference to the stationary distribution of the Markov chain. In fact, the bound of Equation 1 holds even when the chain does not admit a stationary distribution. For example, consider a Markov chain on the real line where the transitions always move the states to the right, that is, $p(X_{t+1} \in dy | X_t = x) = 0$ for $y \le x$. For simplicity assume that the value function V is bounded and belongs to $\mathcal{F}$. This Markov chain is not recurrent, and thus, does not have a stationary distribution. We also assume that the feature vectors $\phi(X_1),\ldots,\phi(X_n)$ are sufficiently independent, so that all the eigenvalues of $\frac{1}{n}\Phi^\top\Phi$ are greater than $\nu > 0$. Then according to Theorem 1, pathwise LSTD is able to estimate the value function at the samples at a rate $O(1/\sqrt{n})$. This may seem surprising because at each state X_t the algorithm is only provided with a noisy estimation of the expected value of the next state. However, the estimates are unbiased conditioned on the current state, and we will see in the proof that using a concentration inequality for martingale, pathwise LSTD is able to learn a good estimate of the value function at a state X_t using noisy pieces of information at other states that may be far away from X_t. In other words, learning the value function at a given state does not require making an average over many samples

close to that state. This implies that LSTD does not require the Markov chain to possess a stationary distribution.

Remark 5 The most critical part of the bound in Equation 1 is the inverse dependency on the smallest positive eigenvalue ν_n. A similar dependency is shown in the LSTD analysis of Bertsekas (2007). The main difference is that here we have a more complete finite-sample analysis with an explicit dependency on the number of samples and the other characteristic parameters of the problem. Furthermore, if the Markov chain admits a stationary distribution ρ, we are able to relate the existence of the LSTD solution to the smallest eigenvalue of the Gram matrix defined according to ρ (see Section 5.1).

In order to prove Theorem 1, we first introduce the regression setting with *Markov design* and then state and prove a lemma about this model. Delattre and Gaïffas (2011) recently analyzed a similar setting in the general case of martingale incremental errors.

Definition 2 *The model of regression with **Markov design** is a regression problem where the data $(X_t, Y_t)_{1\le t\le n}$ are generated according to the following model: $X_1, \dots, X_n$ is a sample path generated by a Markov chain, $Y_t = f(X_t) + \xi_t$, where f is the target function, and the noise term ξ_t is a random variable which is adapted to the filtration generated by $X_1, \dots, X_{t+1}$ and is such that*

$$|\xi_t| \le C \quad \text{and} \quad \mathbb{E}[\xi_t | X_1, \dots, X_t] = 0. \tag{2}$$

The next lemma reports a risk bound for the Markov design setting which is of independent interest.

Lemma 3 (Regression bound for the Markov design setting) *We consider the model of regression with Markov design in Definition 2. Let $\hat{w} \in \mathcal{F}_n$ be the least-squares estimate of the (noisy) values $Y = \{Y_t\}_{t=1}^n$, that is, $\hat{w} = \widehat{\Pi} Y$, and $w \in \mathcal{F}_n$ be the least-squares estimate of the (noiseless) values $Z = \{Z_t = f(X_t)\}_{t=1}^n$, that is, $w = \widehat{\Pi} Z$. Then for any $\delta > 0$, with probability at least $1-\delta$ (the probability is w.r.t. the random sample path $X_1, \dots, X_n$), we have*

$$||\hat{w} - w||_n \le CL \sqrt{\frac{2d \log(2d/\delta)}{n \nu_n}}\,, \tag{3}$$

where ν_n is the smallest strictly-positive eigenvalue of the sample-based Gram matrix $\frac{1}{n}\Phi^\top \Phi$.

Proof [Lemma 3] We define $\xi \in \mathbb{R}^n$ to be the vector with components $\xi_t = Y_t - Z_t$, and $\hat{\xi} = \hat{w} - w = \widehat{\Pi}(Y - Z) = \widehat{\Pi}\xi$. Since the projection is orthogonal we have $\langle \hat{\xi}, \xi \rangle_n = ||\hat{\xi}||_n^2$ (see Figure 1). Since $\hat{\xi} \in \mathcal{F}_n$, there exists at least one $\alpha \in \mathbb{R}^d$ such that $\hat{\xi} = \Phi\alpha$, so by Cauchy-Schwarz inequality we have

$$||\hat{\xi}||_n^2 = \langle \hat{\xi}, \xi \rangle_n = \frac{1}{n} \sum_{i=1}^d \alpha_i \sum_{t=1}^n \xi_t \varphi_i(X_t) \le \frac{1}{n} ||\alpha||_2 \left[\sum_{i=1}^d \Big(\sum_{t=1}^n \xi_t \varphi_i(X_t) \Big)^2 \right]^{1/2}. \tag{4}$$

Now among the vectors α such that $\hat{\xi} = \Phi\alpha$, we define $\hat{\alpha}$ to be the one with minimal ℓ_2-norm, that is, $\hat{\alpha} = \Phi^+ \hat{\xi}$. Let K denote the null-space of Φ, which is also the null-space of $\frac{1}{n}\Phi^\top\Phi$. Then $\hat{\alpha}$ may be decomposed as $\hat{\alpha} = \hat{\alpha}_K + \hat{\alpha}_{K^\perp}$, where $\hat{\alpha}_K \in K$ and $\hat{\alpha}_{K^\perp} \in K^\perp$, and because the decomposition

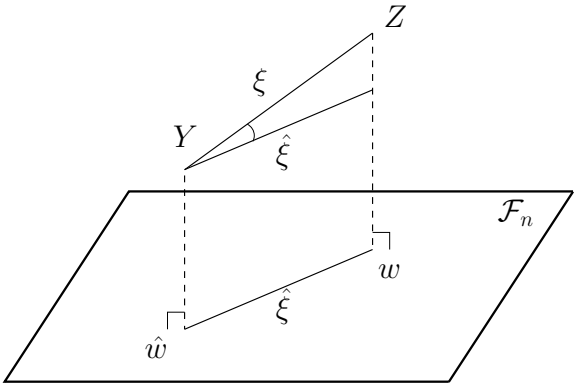


Figure 1: This figure shows the components used in Lemma 3 and its proof such as w, $\hat{w}$, ξ, and $\hat{\xi}$, and the fact that $\langle \hat{\xi}, \xi \rangle_n = ||\hat{\xi}||_n^2$.

is orthogonal, we have $||\hat{\alpha}||_2^2 = ||\hat{\alpha}_K||_2^2 + ||\hat{\alpha}_{K^\perp}||_2^2$. Since $\hat{\alpha}$ is of minimal norm among all the vectors α such that $\hat{\xi} = \Phi\alpha$, its component in K must be zero, thus $\hat{\alpha} \in K^\perp$.

The Gram matrix $\frac{1}{n}\Phi^\top\Phi$ is positive-semidefinite, thus its eigenvectors corresponding to zero eigenvalues generate K and the other eigenvectors generate its orthogonal complement $K^\perp$. Therefore, from the assumption that the smallest strictly-positive eigenvalue of $\frac{1}{n}\Phi^\top\Phi$ is ν_n, we deduce that since $\hat{\alpha} \in K^\perp$,

$$||\hat{\xi}||_n^2 = \frac{1}{n}\hat{\alpha}^\top\Phi^\top\Phi\hat{\alpha} \geq \nu_n\hat{\alpha}^\top\hat{\alpha} = \nu_n||\hat{\alpha}||_2^2. \tag{5}$$

By using the result of Equation 5 in Equation 4, we obtain

$$||\hat{\xi}||_n \leq \frac{1}{n\sqrt{\nu_n}}\left[\sum_{i=1}^{d}\Big(\sum_{t=1}^{n}\xi_t\varphi_i(X_t)\Big)^2\right]^{1/2}. \tag{6}$$

Now, from the conditions on the noise in Equation 2, we have that for any $i = 1,\dots,d$

$$\mathbb{E}[\xi_t\varphi_i(X_t)|X_1,\dots,X_t] = \varphi_i(X_t)\mathbb{E}[\xi_t|X_1,\dots,X_t] = 0,$$

and since $\xi_t\varphi_i(X_t)$ is adapted to the filtration generated by $X_1,\dots,X_{t+1}$, it is a martingale difference sequence w.r.t. that filtration. Thus one may apply Azuma's inequality to deduce that with probability $1-\delta$,

$$\Big|\sum_{t=1}^{n}\xi_t\varphi_i(X_t)\Big| \leq CL\sqrt{2n\log(2/\delta)}\,,$$

where we used that $|\xi_t\varphi_i(X_t)| \leq CL$ for any i and t. By a union bound over all features, we have that with probability $1-\delta$, for all $1 \leq i \leq d$

$$\Big|\sum_{t=1}^{n}\xi_t\varphi_i(X_t)\Big| \leq CL\sqrt{2n\log(2d/\delta)}\,. \tag{7}$$

The result follows by combining Equations 7 and 6. ■

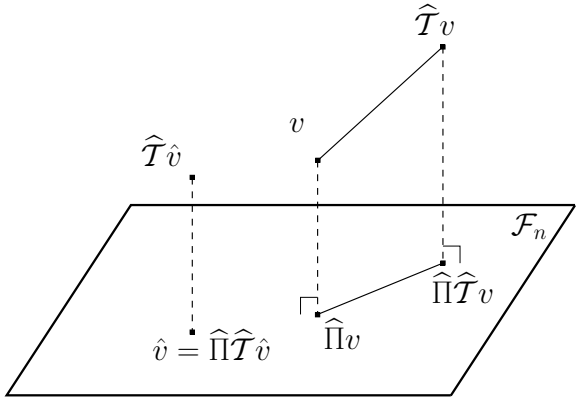


Figure 2: This figure represents the space $\mathbb{R}^n$, the linear vector subspace $\mathcal{F}_n$ and some vectors used in the proof of Theorem 1.

Remark about Lemma 3 Note that this lemma is an extension of the bound for regression with deterministic design in which the states, $\{X_t\}_{t=1}^n$, are fixed and the noise terms, ξ_t's, are independent. In deterministic design, usual concentration results provide high probability bounds similar to Equation 3 (see, e.g., Hsu et al., 2012), but without the dependence on ν_n. An open question is whether it is possible to remove ν_n in the bound for the Markov design regression setting.

In the Markov design model considered in this lemma, states $\{X_t\}_{t=1}^n$ are random variables generated according to the Markov chain and the noise terms ξ_t may depend on the next state X_{t+1} (but should be centered conditioned on the past states $X_1,\ldots,X_t$). This lemma will be used in order to prove Theorem 1, where we replace the target function f with the value function V, and the noise term ξ_t with the temporal difference $r(X_t)+\gamma V(X_{t+1})-V(X_t)$.

Proof [Theorem 1]

Step 1: Using the Pythagorean theorem and the triangle inequality, we have (see Figure 2)

$$||v-\hat{v}||_n^2 = ||v-\widehat{\Pi}v||_n^2 + ||\hat{v}-\widehat{\Pi}v||_n^2 \leq ||v-\widehat{\Pi}v||_n^2 + \big(||\hat{v}-\widehat{\Pi}\widehat{\mathcal{T}}v||_n + ||\widehat{\Pi}\widehat{\mathcal{T}}v-\widehat{\Pi}v||_n\big)^2. \quad (8)$$

From the γ-contraction of the operator $\widehat{\Pi}\widehat{\mathcal{T}}$ and the fact that $\hat{v}$ is its unique fixed point, we obtain

$$||\hat{v}-\widehat{\Pi}\widehat{\mathcal{T}}v||_n = ||\widehat{\Pi}\widehat{\mathcal{T}}\hat{v}-\widehat{\Pi}\widehat{\mathcal{T}}v||_n \leq \gamma||\hat{v}-v||_n, \quad (9)$$

Thus from Equation 8 and 9, we have

$$||v-\hat{v}||_n^2 \leq ||v-\widehat{\Pi}v||_n^2 + \big(\gamma||v-\hat{v}||_n + ||\widehat{\Pi}\widehat{\mathcal{T}}v-\widehat{\Pi}v||_n\big)^2. \quad (10)$$

Step 2: We now provide a high probability bound on $||\widehat{\Pi}\widehat{\mathcal{T}}v-\widehat{\Pi}v||_n$. This is a consequence of Lemma 3 applied to the vectors $Y=\widehat{\mathcal{T}}v$ and $Z=v$. Since v is the value function at the points $\{X_t\}_{t=1}^n$, from the definition of the pathwise Bellman operator, we have that for $1\leq t\leq n-1$,

$$\xi_t = y_t - v_t = r(X_t)+\gamma V(X_{t+1})-V(X_t) = \gamma\big[V(X_{t+1}) - \int P(dy|X_t)V(y)\big],$$

and $\xi_n = y_n - v_n = -\gamma \int P(dy|X_n)V(y)$. Thus, Equation 2 holds for $1 \le t \le n-1$. Here we may choose $C = 2\gamma V_{\max}$ for a bound on ξ_t, $1 \le t \le n-1$, and $C = \gamma V_{\max}$ for a bound on ξ_n. Azuma's inequality may be applied only to the sequence of $n-1$ terms (the n-th term adds a contribution to the bound), thus instead of Equation 7, we obtain

$$\Big| \sum_{t=1}^{n} \xi_t \varphi_i(X_t) \Big| \le \gamma V_{\max} L \big(2\sqrt{2n\log(2d/\delta)} + 1\big),$$

with probability $1-\delta$, for all $1 \le i \le d$. Combining with Equation 6, we deduce that with probability $1-\delta$, we have

$$||\widehat{\Pi}\widehat{\mathcal{T}}v - \widehat{\Pi}v||_n \le \gamma V_{\max} L \sqrt{\frac{d}{\nu_n}} \Big(\sqrt{\frac{8\log(2d/\delta)}{n}} + \frac{1}{n} \Big), \tag{11}$$

where ν_n is the smallest strictly-positive eigenvalue of $\frac{1}{n}\Phi^\top \Phi$. The claim follows by solving Equation 10 for $||v - \hat{v}||_n$ and replacing $||\widehat{\Pi}\widehat{\mathcal{T}}v - \widehat{\Pi}v||_n$ from Equation 11. ■

5. Generalization Bounds

As we pointed out earlier, Theorem 1 makes no assumption on the existence of the stationary distribution of the Markov chain. This generality comes at the cost that the performance is evaluated only at the states visited by the Markov chain and no generalization on other states is possible. However in many problems of interest, the Markov chain has a stationary distribution ρ, and thus, the performance may be generalized to the whole state space under the measure ρ. Moreover, if ρ exists, it is possible to derive a condition for the existence of the pathwise LSTD solution depending on the number of samples and the smallest eigenvalue of the Gram matrix defined according to ρ ; $G \in \mathbb{R}^{d\times d}$, $G_{ij} = \int \varphi_i(x)\varphi_j(x)\rho(dx)$. In this section, we assume that the Markov chain $\mathcal{M}^\pi$ is exponentially fast β-mixing with parameters β, b, κ, that is, its β-mixing coefficients satisfy $\beta_i \le \beta \exp(-bi^\kappa)$ (see Section A.2 in the appendix for a more detailed definition of β-mixing processes).

Before stating the main results of this section, we introduce some notation. If ρ is the stationary distribution of the Markov chain, we define the orthogonal projection operator $\Pi : \mathcal{B}(\mathcal{X}; V_{\max}) \to \mathcal{F}$ as

$$\Pi V = \arg\min_{f\in\mathcal{F}} ||V - f||_\rho \, .$$

Furthermore, in the rest of the paper with a little abuse of notation, we replace the empirical norm $||v||_n$ defined on states $X_1, \ldots, X_n$ by $||V||_n$, where $V \in \mathcal{B}(\mathcal{X}; V_{\max})$ is such that $V(X_t) = v_t$. Finally, we should guarantee that the pathwise LSTD solution $\widehat{V}$ is uniformly bounded on $\mathcal{X}$. For this reason, we move from $\mathcal{F}$ to the truncated space $\widetilde{\mathcal{F}}$ in which for any function $f \in \mathcal{F}$, a truncated function $\tilde{f}$ is defined as

$$\tilde{f}(x) = \begin{cases} f(x) & \text{if } |f(x)| \le V_{\max} \, , \\ \operatorname{sgn}\big(f(x)\big) V_{\max} & \text{otherwise.} \end{cases} \tag{12}$$

In the next sections, we present conditions on the existence of the pathwise LSTD solution and derive generalization bounds under different assumptions on the way the samples $X_1, \ldots, X_n$ are generated.

5.1 Uniqueness of Pathwise LSTD Solution

In this section, we assume that all the eigenvalues of G are strictly positive; that is, we assume the existence of the model-based solution of LSTD, and derive a condition to guarantee that the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$ is invertible. More specifically, we show that if a large enough number of samples (depending on the smallest eigenvalue of G) is available, then the smallest eigenvalue of $\frac{1}{n}\Phi^\top\Phi$ is strictly positive with high probability.

Lemma 4 *Let G be the Gram matrix defined according to the distribution ρ and $\omega > 0$ be its smallest eigenvalue. Let $X_1,\ldots,X_n$ be a trajectory of length n of a stationary β-mixing process with parameters β, b, κ and stationary distribution ρ. If the number of samples n satisfies the following condition*

$$n > \frac{288L^2\Lambda(n,d,\delta)}{\omega}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}, \tag{13}$$

where[2] $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log^+\left(\max\{18(6e)^{2(d+1)},\beta\}\right)$*, then with probability* $1-\delta$*, the family of features $(\varphi_1,\ldots,\varphi_d)$ is linearly independent on the states $X_1,\ldots,X_n$ (i.e., $||f_\alpha||_n = 0$ implies $\alpha = 0$) and the smallest eigenvalue ν_n of the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$ satisfies*

$$\sqrt{\nu_n} \geq \sqrt{\nu} = \frac{\sqrt{\omega}}{2} - 6L\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}} > 0\,. \tag{14}$$

Proof From the definition of the Gram matrix and the fact that $\omega > 0$ is its smallest eigenvalue, for any function $f_\alpha \in \mathcal{F}$, we have

$$||f_\alpha||_\rho^2 = ||\phi^\top\alpha||_\rho^2 = \alpha^\top G\alpha \geq \omega\alpha^\top\alpha = \omega||\alpha||^2. \tag{15}$$

Using the concentration inequality from Corollary 18 in the appendix and the fact that the basis functions φ_i are bounded by L, thus f_α is bounded by $L||\alpha||$, we have $||f_\alpha||_\rho - 2||f_\alpha||_n \leq \varepsilon$ with probability $1-\delta$, where

$$\varepsilon = 12L\|\alpha\|\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}}\,.$$

Thus we obtain

$$2||f_\alpha||_n + \varepsilon \geq \sqrt{\omega}||\alpha||. \tag{16}$$

Let α be such that $||f_\alpha||_n = 0$, then if the number of samples n satisfies the condition of Equation 13, we may deduce from Equation 16 and the definition of ε that $\alpha = 0$. This indicates that given Equation 13, with probability $1-\delta$, the family of features $(\varphi_1,\ldots,\varphi_d)$ is linearly independent on the states $X_1,\ldots,X_n$, and thus, $\nu_n > 0$. The inequality in Equation 14 is obtained by choosing α to be the eigenvector of $\frac{1}{n}\Phi^\top\Phi$ corresponding to the smallest eigenvalue ν_n. For this value of α, we have $||f_\alpha||_n = \sqrt{\nu_n}||\alpha||$. By using the definition of ε in Equation 16 and reordering we obtain

$$2\sqrt{\nu_n}||\alpha|| + 12L||\alpha||\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}} \geq \sqrt{\omega}||\alpha||\,,$$

and the claim follows. ■

2. We define $\log^+ x = \max\{\log x, 0\}$.

Remark 1 In order to make the condition on the number of samples and its dependency on the critical parameters of the problem at hand more explicit, let us consider the case of a stationary process with $b = \beta = \kappa = 1$. Then the condition in Equation 13 becomes (up to constant and logarithmic factors)

$$n \geq \widetilde{O}\left(\frac{288L^2}{\omega}\left((d+1)\log\frac{n}{\delta}\right)^2\right).$$

As can be seen, the number of samples needed to have strictly positive eigenvalues in the sample-based Gram matrix has an inverse dependency on the smallest eigenvalue of G. As a consequence, the more G is ill-conditioned the more samples are needed for the sample-based Gram matrix $\frac{1}{n}\Phi^\top\Phi$ to be invertible.

5.2 Generalization Bounds for Stationary β-mixing Processes

In this section, we show how Theorem 1 may be generalized to the entire state space $\mathcal{X}$ when the Markov chain $\mathcal{M}^\pi$ has a stationary distribution ρ. In particular, we consider the case in which the samples $X_1,\ldots,X_n$ are obtained by following a single trajectory in the stationary regime of $\mathcal{M}^\pi$, that is, when we consider that X_1 is drawn from ρ.

Theorem 5 *Let $X_1,\ldots,X_n$ be a path generated by a stationary β-mixing process with parameters β, b, κ and stationary distribution ρ. Let $\omega > 0$ be the smallest eigenvalue of the Gram matrix defined according to ρ and n satisfy the condition in Equation 13. Let $\widetilde{V}$ be the truncation (using Equation 12) of the pathwise LSTD solution, then*

$$||\widetilde{V} - V||_\rho \leq \frac{2}{\sqrt{1-\gamma^2}}\Big(2\sqrt{2}||V-\Pi V||_\rho + \varepsilon_2\Big) + \frac{2}{1-\gamma}\left[\gamma V_{\max} L\sqrt{\frac{d}{\nu}}\Big(\sqrt{\frac{8\log(8d/\delta)}{n}} + \frac{1}{n}\Big)\right] + \varepsilon_1 \tag{17}$$

with probability $1-\delta$, where ν is a lower-bound on the eigenvalues of the sample-based Gram matrix defined by Equation 14,

$$\varepsilon_1 = 24V_{\max}\sqrt{\frac{2\Lambda_1(n,d,\delta/4)}{n}\max\left\{\frac{\Lambda_1(n,d,\delta/4)}{b},1\right\}^{1/\kappa}},$$

with $\Lambda_1(n,d,\delta/4) = 2(d+1)\log n + \log\frac{4e}{\delta} + \log^+\big(\max\{18(6e)^{2(d+1)},\beta\}\big)$, and

$$\varepsilon_2 = 12\big(V_{\max} + L||\alpha^*||\big)\sqrt{\frac{2\Lambda_2(n,\delta/4)}{n}\max\left\{\frac{\Lambda_2(n,\delta/4)}{b},1\right\}^{1/\kappa}}, \tag{18}$$

with $\Lambda_2(n,\delta/4) = \log\frac{4e}{\delta} + \log\big(\max\{6,n\beta\}\big)$ and α^* is such that $f_{\alpha^*} = \Pi V$.

Proof This result is a consequence of applying generalization bounds to both sides of Equation 1 (Theorem 1). We first bound the left-hand side:

$$2||\widehat{V} - V||_n \geq 2||\widetilde{V} - V||_n \geq ||\widetilde{V} - V||_\rho - \varepsilon_1$$

with probability $1-\delta'$. The first step follows from the definition of the truncation operator, while the second step is a straightforward application of Corollary 17 in the appendix.

We now bound the term $||V - \widehat{\Pi}V||_n$ in Equation 1:

$$||V - \widehat{\Pi}V||_n \leq ||V - \Pi V||_n \leq 2\sqrt{2}||V - \Pi V||_\rho + \varepsilon_2$$

with probability $1-\delta'$. The first step follows from the definition of the operator $\widehat{\Pi}$. The second step is an application of the inequality of Corollary 19 in the appendix for the function $V - \Pi V$.

From Theorem 1, the two generalization bounds, and the lower-bound on ν, each one holding with probability $1-\delta'$, the statement of the Theorem (Equation 17) holds with probability $1-\delta$ by setting $\delta = 4\delta'$. ■

Remark 1 Rewriting the bound in terms of the approximation and estimation error terms (up to constants and logarithmic factors), we obtain

$$||\widetilde{V} - V||_\rho \leq \widetilde{O}\left(\frac{1}{\sqrt{1-\gamma^2}}||V - \Pi V||_\rho + \frac{1}{1-\gamma}\frac{1}{\sqrt{n}}\right).$$

While the first term (*approximation error*) only depends on the target function V and the function space $\mathcal{F}$, the second term (*estimation error*) primarily depends on the number of samples. Thus, when n goes to infinity, the estimation error goes to zero and we obtain the same performance bound (up to a $4\sqrt{2}$ constant) as for the model-based case reported by Tsitsiklis and Van Roy (1997). The additional multiplicative constant $4\sqrt{2}$ in front of the approximation error is the standard cost to have the improved rate bounds for the squared loss and linear spaces (see, e.g., Györfi et al., 2002). In fact, it is possible to derive a bounds with constant 1 but a worse rate $n^{-1/4}$ instead of $n^{-1/2}$. The bound in Theorem 5 is more accurate whenever the approximation error is small and few samples are available.

Remark 2 Antos et al. (2008) reported a sample-based analysis for the modified Bellman residual (MBR) minimization algorithm. They consider a general setting in which the function space $\mathcal{F}$ is bounded and the performance of the algorithm is evaluated according to an arbitrary measure μ (possibly different than the stationary distribution of the Markov chain ρ). Since Antos et al. (2008) showed that the MBR minimization algorithm is equivalent to LSTD when $\mathcal{F}$ is a linearly parameterized space, it would be interesting to compare the bound in Theorem 5 to the one in Lemma 11 of Antos et al. (2008). In Theorem 5, similar to Antos et al. (2008), samples are drawn from a stationary β-mixing process, however, $\mathcal{F}$ is a linear space and ρ is the stationary distribution of the Markov chain. It is interesting to note the impact of these two differences in the final bound. The use of linear spaces has a direct effect on the estimation error and leads to a better convergence rate due to the use of improved functional concentration inequalities (Lemma 16 in the appendix). In fact, while in Antos et al. (2008) the estimation error for the squared error is of order $O(1/\sqrt{n})$, here we achieve a faster convergence rate of order $O(1/n)$. Moreover, although Antos et al. (2008) showed that the solution of MBR minimization coincides with the LSTD solution, its sample-based analysis cannot be directly applied to LSTD. In fact, in Antos et al. (2008) the function space $\mathcal{F}$ is assumed to be bounded, while general linear spaces cannot be bounded. Whether the analysis of Antos et al. (2008) may be extended to the truncated solution of LSTD is an open question that requires further investigation.

5.3 Generalization Bounds for Markov Chains

The main assumption in the previous section is that the trajectory $X_1,\dots,X_n$ is generated by a stationary β-mixing process with stationary distribution ρ. This is possible if we consider samples of a Markov chain during its stationary regime, that is, $X_1 \sim \rho$. However in practice, ρ is not known, and the first sample X_1 is usually drawn from a given initial distribution and the rest of the sequence is obtained by following the Markov chain from X_1 on. As a result, the sequence $X_1,\dots,X_n$ is no longer a realization of a stationary β-mixing process. Nonetheless, under suitable conditions, after $\tilde{n} < n$ steps, the distribution of $X_{\tilde{n}}$ approaches the stationary distribution ρ. In fact, according to the convergence theorem for fast-mixing Markov chains (see, e.g., Proposition 20 in the appendix), for any initial distribution $\lambda \in \mathcal{S}(\mathcal{X})$, we have

$$\left|\left| \int_{\mathcal{X}} \lambda(dx) P^n(\cdot|x) - \rho(\cdot) \right|\right|_{TV} \leq \beta \exp(-b n^{\kappa}).$$

where $||\cdot||_{TV}$ is the total variation.[3]

We now derive a bound for a modification of pathwise LSTD in which the first $\tilde{n}$ samples (that are used to burn the chain) are discarded and the remaining $n-\tilde{n}$ samples are used as training samples for the algorithm.

Theorem 6 *Let $X_1,\dots,X_n$ be a trajectory generated by a β-mixing Markov chain with parameters β, b, κ and stationary distribution ρ. Let $\tilde{n}$ ($1 \leq \tilde{n} < n$) be such that $n-\tilde{n}$ satisfies the condition of Equation 13, and $X_{\tilde{n}+1},\dots,X_n$ be the samples actually used by the algorithm. Let $\omega > 0$ be the smallest eigenvalue of the Gram matrix defined according to ρ and $\alpha^* \in \mathbb{R}^d$ be such that $f_{\alpha^*} = \Pi V$. Let $\widetilde{V}$ be the truncation of the pathwise LSTD solution (using Equation 12), then by setting $\tilde{n} = \left(\frac{1}{b}\log\frac{2e\beta n}{\delta}\right)^{1/\kappa}$, with probability $1-\delta$, we have*

$$||\widetilde{V} - V||_{\rho} \leq \frac{2}{\sqrt{1-\gamma^2}}\left(2\sqrt{2}||V-\Pi V||_{\rho} + \varepsilon_2\right) + \frac{2}{1-\gamma}\left[\gamma V_{\max} L \sqrt{\frac{d}{\nu}}\left(\sqrt{\frac{8\log(8d/\delta)}{n-\tilde{n}}} + \frac{1}{\tilde{n}}\right)\right] + \varepsilon_1, \tag{19}$$

where ε_1 and ε_2 are defined as in Theorem 5 (with $n-\tilde{n}$ as the number of training samples).

The proof of this result is a simple consequence of Lemma 24 in the appendix applied to Theorem 5.

Remark 1 The bound in Equation 19 indicates that in the case of β-mixing Markov chains, a similar performance to the one for stationary β-mixing processes is obtained by discarding the first $\tilde{n} = O(\log n)$ samples.

6. Finite-Sample Analysis of LSPI

In the previous sections we studied the performance of pathwise-LSTD for policy evaluation. Now we move to the analysis of the least-squares policy iteration (LSPI) algorithm (Lagoudakis and Parr, 2003) in which at each iteration k samples are collected by following a single trajectory of the

3. We recall that for any two distributions $\mu_1, \mu_2 \in \mathcal{S}(\mathcal{X})$, the total variation norm is defined as $||\mu_1 - \mu_2||_{TV} = \sup_{X \subseteq \mathcal{X}} |\mu_1(X) - \mu_2(X)|$.

policy under evaluation, π_k, and LSTD is used to compute an approximation of V^{π_k}. In particular, in the next section we report a performance bound by comparing the value of the policy returned by the algorithm after K iterations, V^{π_K}, and the optimal value function, V^*, w.r.t. an arbitrary target distribution σ. In order to achieve this bound we introduce assumptions on the MDP and the linear space $\mathcal{F}$. In Section 6.2 we show that in some cases one of these assumptions does not hold and the performance of LSPI can be arbitrarily bad.

6.1 Generalization Bound for LSPI

In this section, we provide a performance bound for the LSPI algorithm (Lagoudakis and Parr, 2003). We first introduce the *greedy policy* operator $\mathcal{G}$ that maps value functions to their corresponding greedy policies:

$$\big(\mathcal{G}(V)\big)(x) = \arg\max_{a\in\mathcal{A}} \left[r(x,a) + \gamma \int_{\mathcal{X}} P(dy|x,a)V(y) \right].$$

We use $\mathcal{G}(\mathcal{F})$ to refer to the set of all the greedy policies w.r.t. the functions in $\mathcal{F}$. LSPI is a policy iteration algorithm that uses LSTD for policy evaluation at each iteration. It starts with an arbitrary initial value function $V_{-1} \in \widetilde{\mathcal{F}}$ and its corresponding greedy policy π_0. At the first iteration, it approximates V^{π_0} using LSTD and returns a function V_0 whose truncated version $\widetilde{V}_0$ is used to build the policy π_1 for the second iteration.[4] More precisely, π_1 is the greedy policy w.r.t. $\widetilde{V}_0$, that is, $\pi_1 = \mathcal{G}(\widetilde{V}_0)$. So, at each iteration k of LSPI, a function V_{k-1} is computed as an approximation to $V^{\pi_{k-1}}$, and then truncated, $\widetilde{V}_{k-1}$, and used to build the policy $\pi_k = \mathcal{G}(\widetilde{V}_{k-1})$. Note that the MDP model is needed in order to generate the greedy policy π_k. To avoid the need for the model, we could simply move from LSTD to LSTD-Q. The analysis of LSTD in the previous sections may be easily extended to action-value function, and thus, to LSTD-Q.[5] For simplicity we use value function in the paper and report the LSPI bound in terms of the distance to the optimal value function.

It is important to note that in general the measure used to evaluate the final performance of LSPI, $\sigma \in \mathcal{S}(\mathcal{X})$, might be different than the distribution used to generate the samples at each iteration. Moreover, the LSTD performance bounds of Section 5 require the samples to be collected by following the policy under evaluation. Thus, we make the following assumption.

Assumption 1 (Lower-bounding distribution) *There exists a distribution $\mu \in \mathcal{S}(\mathcal{X})$ such that for any policy π that is greedy w.r.t. a function in the truncated space $\widetilde{\mathcal{F}}$, $\mu \le C\rho^\pi$, where $C < \infty$ is a constant and ρ^π is the stationary distribution of policy π.*

Assumption 2 . (Discounted-average Concentrability of Future-State Distribution [Antos et al., 2008]) *Given the target distribution $\sigma \in \mathcal{S}(\mathcal{X})$ and an arbitrary sequence of policies $\{\pi_m\}_{m\ge1}$, let*

$$c_{\sigma,\mu} = \sup_{\pi_1,\dots,\pi_m} \left\| \frac{d(\mu P^{\pi_1} \dots P^{\pi_m})}{d\sigma} \right\|.$$

4. Unlike in the original formulation of LSPI, here we need to explicitly truncate the function so as to prevent unbounded functions.
5. We point out that moving to LSTD-Q requires the introduction of some exploration to the current policy. In fact, in the on-policy setting, if the policy under evaluation is deterministic, it does not provide any information about the value of actions $a \neq \pi(\cdot)$ and the policy improvement step would always fail. Thus, we need to consider stochastic policies where the current policy is perturbed by an $\varepsilon > 0$ randomization which guarantees that any action has a non-zero probability to be selected in any state.

We define the second-order discounted-average concentrability of future-state distributions as

$$C_{\sigma,\mu} = (1-\gamma)^2 \sum_{m\geq 1} m\gamma^{m-1} c_{\sigma,\mu}(m)$$

and we assume that $C_{\sigma,\mu} < \infty$.

We also need to guarantee that with high probability a unique LSTD solution exists at each iteration of the LSPI algorithm, thus, we make the following assumption.

Assumption 3 (Linear independent features) *Let* $\mu \in \mathcal{S}(\mathcal{X})$ *be the lower-bounding distribution from Assumption 1. We assume that the features* $\phi(\cdot)$ *of the function space* $\mathcal{F}$ *are linearly independent w.r.t.* μ. *In this case, the smallest eigenvalue* ω_μ *of the Gram matrix* $G_\mu \in \mathbb{R}^{d\times d}$ *w.r.t.* μ *is strictly positive.*

Lemma 7 *Under Assumption 3, at each iteration k of LSPI, the smallest eigenvalue* ω_k *of the Gram matrix* G_k *defined according to the stationary distribution* $\rho_k = \rho^{\pi_k}$ *is strictly positive and* $\omega_k \geq \frac{\omega_\mu}{C}$.

Proof Similar to Lemma 4, for any function $f_\alpha \in \mathcal{F}$, we have $||\alpha|| \leq \frac{||f_\alpha||_\mu}{\sqrt{\omega_\mu}}$. Using Assumption 1, $||f_\alpha||_\mu \leq \sqrt{C}\,||f_\alpha||_{\rho_k}$, and thus, $||\alpha|| \leq \sqrt{\frac{C}{\omega_\mu}}\,||f_\alpha||_{\rho_k}$. For the α that is the eigenvector of G_k corresponding to ρ_k, we have $||\alpha|| = \frac{||f_\alpha||_{\rho_k}}{\sqrt{\omega_k}}$. For this value of α, we may write $\frac{||f_\alpha||_{\rho_k}}{\sqrt{\omega_k}} \leq \sqrt{\frac{C}{\omega_\mu}}\,||f_\alpha||_{\rho_k}$, and thus, $\omega_k \geq \frac{\omega_\mu}{C}$, which guarantees that ω_k is strictly positive, because ω_μ is strictly positive according to Assumption 3. ■

Finally, we make the following assumption on the stationary β-mixing processes corresponding to the stationary distributions of the policies encountered at the iterations of the LSPI algorithm.

Assumption 4 (Slower β-mixing process) *We assume that there exists a stationary* β*-mixing process with parameters* β, b, κ, *such that for any policy* π *that is greedy w.r.t. a function in the truncated space* $\widetilde{\mathcal{F}}$, *it is slower than the stationary* β*-mixing process with stationary distribution* ρ^π *(with parameters* $\beta_\pi, b_\pi, \kappa_\pi$*). This means that* β *is larger and b and* κ *are smaller than their counterparts* β_π, b_π, *and* κ_π *(see Definition 14).*

Now we may state the main theorem of this section.

Theorem 8 *Let us assume that at each iteration k of the LSPI algorithm, a path of size n is generated from the stationary* β*-mixing process with stationary distribution* $\rho_{k-1} = \rho^{\pi_{k-1}}$. *Let n satisfy the condition in Equation 13 for the slower* β*-mixing process defined in Assumption 4. Let* $V_{-1} \in \widetilde{\mathcal{F}}$ *be an arbitrary initial value function,* $V_0, \ldots, V_{K-1}$ *(*$\widetilde{V}_0, \ldots, \widetilde{V}_{K-1}$*) be the sequence of value functions (truncated value functions) generated by LSPI after K iterations, and* π_K *be the greedy policy w.r.t. the truncated value function* $\widetilde{V}_{K-1}$. *Then under Assumptions 1- 4, with probability* $1-\delta$ *(w.r.t. the random samples), we have*

$$||V^* - V^{\pi_K}||_\sigma \leq \frac{4\gamma}{(1-\gamma)^2}\left\{(1+\gamma)\sqrt{CC_{\sigma,\mu}}\left[\frac{2}{\sqrt{1-\gamma^2}}\left(2\sqrt{2}E_0(\mathcal{F}) + E_2\right)\right.\right.$$
$$\left.\left. + \frac{2}{1-\gamma}\left(\gamma V_{\max} L\sqrt{\frac{d}{\nu_\mu}}\left(\sqrt{\frac{8\log(8dK/\delta)}{n}} + \frac{1}{n}\right)\right) + E_1\right] + \gamma^{\frac{K-1}{2}} R_{\max}\right\},$$

where

1. $E_0(\mathcal{F}) = \sup_{\pi \in \mathcal{G}(\widetilde{\mathcal{F}})} \inf_{f \in \mathcal{F}} ||f - V^\pi||_{\rho^\pi}$,
2. E_1 *is* ε_1 *from Theorem 5 written for the slower* β*-mixing process defined in Assumption 4,*
3. E_2 *is* ε_2 *from Theorem 5 written for the slower* β*-mixing process defined in Assumption 4 and* $||\alpha^*||$ *replaced by* $\sqrt{\frac{C}{\omega_\mu}} \frac{R_{\max}}{1-\gamma}$*, and*
4. ν_μ *is* ν *from Equation 14 in which* ω *is replaced by* ω_μ *defined in Assumption 3, and the second term is written for the slower* β*-mixing process defined in Assumption 4.*

Remark 1 The previous theorem states a bound on the prediction error when LSPI is stopped after a fixed number K of iterations. The structure of the bound resembles the one in Antos et al. (2008). Unlike policy evaluation, the approximation error $E_0(\mathcal{F})$ now depends on how well the space $\mathcal{F}$ can approximate the target functions V^π obtained in the policy improvement step. While the estimation errors are mostly similar to those in policy evaluation, an additional term of order γ^K is introduced. Finally, we notice that the concentrability terms may significantly amplify the prediction error (see also next remark). Farahmand et al. (2010) recently performed a refined analysis of the propagation of the error in approximate policy iteration and have interesting insights on the concentrability terms.

Remark 2 The most critical issue about Theorem 8 is the validity of Assumptions 1–4. The analysis of LSTD explicitly requires that the samples are collected by following the policy under evaluation, π_k, and the performance is bounded according to its stationary distribution ρ_k. Since the performance of LSPI is assessed w.r.t. a target distribution σ, we need each of the policies encountered through the LSPI process to have a stationary distribution which does not differ too much from σ. Furthermore, since the policies are random (at each iteration k the new policy π_k is greedy w.r.t. the approximation $\widetilde{V}_{k-1}$ which is random because of the sampled trajectory), we need to consider the distance of σ and the stationary distribution of any possible policy generated as greedy w.r.t. a function in the truncated space $\widetilde{\mathcal{F}}$, that is, ρ^π, $\pi \in \mathcal{G}(\widetilde{\mathcal{F}})$. Thus in Assumption 1 we first assume the existence of a distribution μ lower-bounding any possible stationary distribution ρ_k. The existence of μ and the value of the constant C depend on the MDP at hand. In Section 6.2, we provide an example in which the constant C is infinite. In this case, we show that the LSPI performance, when the samples at each iteration are generated according to the stationary distribution of the policy under evaluation, can be arbitrarily bad. A natural way to relax this assumption would be the use of off-policy LSTD in which the samples are collected by following a behavior policy. Nonetheless, we are not aware of any finite-sample analysis for such an algorithm. Another critical term appearing in the bound of LSPI, inherited from Theorem 5, is the maximum of $||\alpha_k^*||$ over the iterations, where α_k^* is such that $f_{\alpha_k^*} = \Pi_{\rho_k} V^{\pi_k}$. Each term $||\alpha_k^*||$ can be bounded whenever the features of the space $\mathcal{F}$ are linearly independent according to the stationary distribution ρ_k. Since α_k^* is a random variable, the features $\{\varphi_i\}_{i=1}^d$ of the space $\mathcal{F}$ should be carefully chosen so as to be linearly independent w.r.t. the lower-bounding distribution μ.

We now prove a lemma that is used in the proof of Theorem 8.

Lemma 9 *Let* π_k *be the greedy policy w.r.t.* $\widetilde{V}_{k-1}$*, that is,* $\pi_k = \mathcal{G}(\widetilde{V}_{k-1})$ *and* ρ^{π_k} *be the stationary distribution of the Markov chain induced by* π_k*. We have*

$$||\widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k||_{\rho^{\pi_k}} \leq (1+\gamma)||\widetilde{V}_k - V^{\pi_k}||_{\rho^{\pi_k}} .$$

Proof [Lemma 9] We first show that $\widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k = (I-\gamma P^{\pi_k})(\widetilde{V}_k - V^{\pi_k})$

$$\begin{aligned}(I-\gamma P^{\pi_k})(\widetilde{V}_k - V^{\pi_k}) &= \widetilde{V}_k - V^{\pi_k} - \gamma P^{\pi_k}\widetilde{V}_k + \gamma P^{\pi_k}V^{\pi_k} = \widetilde{V}_k - V^{\pi_k} - \mathcal{T}^{\pi_k}\widetilde{V}_k + \mathcal{T}^{\pi_k}V^{\pi_k} \\ &= \widetilde{V}_k - V^{\pi_k} - \mathcal{T}^{\pi_k}\widetilde{V}_k + V^{\pi_k} = \widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k\,.\end{aligned}$$

For any distribution $\sigma \in \mathcal{S}(\mathcal{X})$, we may write

$$\begin{aligned}||\widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k||_\sigma &= ||(I-\gamma P^{\pi_k})(\widetilde{V}_k - V^{\pi_k})||_\sigma \le ||I-\gamma P^{\pi_k}||_\sigma ||\widetilde{V}_k - V^{\pi_k}||_\sigma \\ &\le \big(1+\gamma||P^{\pi_k}||_\sigma\big)||\widetilde{V}_k - V^{\pi_k}||_\sigma\end{aligned}$$

If σ is the stationary distribution of π_k, that is, $\sigma = \rho^{\pi_k}$, then $||P^{\pi_k}||_\sigma = 1$ and the claim follows. Note that this theorem holds not only for ℓ_2-norm, but for any ℓ_p-norm, $p \ge 1$. ■

Proof [Theorem 8] Rewriting Lemma 12 in Antos et al. (2008) for V instead of Q, we obtain[6]

$$||V^* - V^{\pi_K}||_\sigma \le \frac{4\gamma}{(1-\gamma)^2}\Big(\sqrt{C_{\sigma,\mu}}\max_{0\le k<K}||\widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k||_\mu + \gamma^{\frac{K-1}{2}}R_{\max}\Big). \tag{20}$$

From Assumption 1, we know that $||\cdot||_\mu \le \sqrt{C}||\cdot||_{\rho_k}$ for any $0 \le k < K$ and thus we may rewrite Equation 20 as

$$||V^* - V^{\pi_K}||_\sigma \le \frac{4\gamma}{(1-\gamma)^2}\Big(\sqrt{CC_{\sigma,\mu}}\max_{0\le k<K}||\widetilde{V}_k - \mathcal{T}^{\pi_k}\widetilde{V}_k||_{\rho_k} + \gamma^{\frac{K-1}{2}}R_{\max}\Big). \tag{21}$$

Using the result of Lemma 9, Equation 21 may be rewritten as

$$||V^* - V^{\pi_K}||_\sigma \le \frac{4\gamma}{(1-\gamma)^2}\Big((1+\gamma)\sqrt{CC_{\sigma,\mu}}\max_{0\le k<K}||\widetilde{V}_k - V^{\pi_k}||_{\rho_k} + \gamma^{\frac{K-1}{2}}R_{\max}\Big). \tag{22}$$

We can now use the result of Theorem 5 (which holds with probability δ/K) and replace $||\widetilde{V}_k - V^{\pi_k}||_{\rho_k}$ with its upper-bound. The next step would be to apply the maximum over k to this upper-bound (the right hand side of Equation 17). There are four terms on the r.h.s. of Equation 17 that depend on k and in following we find a bound for each of them.

1. $||V^{\pi_k} - \Pi_{\rho_k}V^{\pi_k}||_{\rho_k}$: This term can be upper-bounded by $E_0(\mathcal{F})$. This quantity, $E_0(\mathcal{F})$, measures the approximation power of the linear function space $\mathcal{F}$.

2. ε_1: This term only depends on the parameters β_k, b_k, κ_k of the stationary β-mixing process with stationary distribution ρ_k. Using Assumption 4, this term can be upper-bounded by E_1, which is basically ε_1 written for the slower β-mixing process from Assumption 4.

3. ε_2: This term depends on the following k-related terms.

6. The slight difference between Equation 20 and the bound in Lemma 12 of Antos et al. (2008) is due to a small error in Equation 26 of Antos et al. (2008). It can be shown that the r.h.s. of Equation 26 in Antos et al. (2008) is not an upper-bound for the r.h.s. of its previous equation. This can be easily fixed by redefining the coefficients α_k while we make sure that they remain positive and still sum to one. This modification causes two small changes in the final bound: the constant 2 in front of the parenthesis becomes 4 and the power of the γ in front of $R_{\max}$ changes from K/p to $(K-1)/p$.

- The term under the root-square in Equation 18: This term depends on the parameters β_k, b_k, κ_k of the stationary β-mixing process with stationary distribution ρ_k. Similar to ε_1, this term can be upper-bounded by rewriting it for the slower β-mixing process from Assumption 4.
- α_k^*: The coefficient vector α_k^* is such that $f_{\alpha_k^*} = \Pi_{\rho_k} V^{\pi_k}$. This term can be upper-bounded as follows:

$$||\alpha_k^*|| \overset{(a)}{\leq} \frac{||f_{\alpha^*}||_\mu}{\sqrt{\omega_\mu}} \overset{(b)}{\leq} \sqrt{\frac{C}{\omega_\mu}} ||f_{\alpha^*}||_{\rho_k} = \sqrt{\frac{C}{\omega_\mu}} ||\Pi_{\rho_k} V^{\pi_k}||_{\rho_k} \overset{(c)}{\leq} \sqrt{\frac{C}{\omega_\mu}} ||V^{\pi_k}||_{\rho_k}$$
$$\leq \sqrt{\frac{C}{\omega_\mu}} ||V^{\pi_k}||_\infty = \sqrt{\frac{C}{\omega_\mu}}\, V_{\max} = \sqrt{\frac{C}{\omega_\mu}}\, \frac{R_{\max}}{1-\gamma}\,.$$

  (a) Similar to Equation 15, this is true for any function $f_\alpha \in \mathcal{F}$.
  (b) This is an immediate application of Assumption 1.
  (c) We use the fact that the orthogonal projection Π_{ρ_k} is non-expansive for norm $||\cdot||_{\rho_k}$.

4. ν_{ρ_k}: This term depends on the following k-related terms.

   - ω_k: This is the smallest eigenvalue of the Gram matrix G_k defined according to the distribution ρ_k. From Lemma 7, this term can be lower-bounded by ω_μ.
   - The second term on the r.h.s. of Equation 14: This term depends on the parameters β_k, b_k, κ_k of the stationary β-mixing process with stationary distribution ρ_k. Similar to ε_1 and ε_2, this term can be upper-bounded by rewriting it for the slower β-mixing process from Assumption 4.

   By replacing the above lower and upper bounds in Equation 14, we obtain ν_μ which is a lower-bound for any ν_{ρ_k}.

The claim follows by replacing the bounds for the above four terms in Equation 22. ■

6.2 A Negative Result for LSPI

In the previous section we analyzed the performance of LSPI when at each iteration the samples are obtained from a trajectory generated by following the policy under evaluation. In order to bound the performance of LSPI in Theorem 8, we made a strong assumption on all possible stationary distributions that can be obtained at the iterations of the algorithm. Assumption 1 states the existence of a lower-bounding distribution μ for the stationary distribution ρ^π of any policy $\pi \in \mathcal{G}(\widetilde{\mathcal{F}})$. If such a distribution does not exist (C is infinite), the LSPI performance can no longer be bounded. In other words, this result states that in some MDPs, even if at each iteration the target function V^{π_k} is perfectly approximated by $\widehat{V}_k$ under ρ_k-norm, that is, $||V^{\pi_k} - \widehat{V}_k||_{\rho_k} = 0$, the LSPI performance could be arbitrarily bad. In this section we show a very simple MDP in which this is actually the case.

Let consider a finite MDP with $\mathcal{X} = \{x_1, x_2, x_3\}$, $\mathcal{A} = \{a, b\}$, and the reward function r and transition model p as illustrated in Figure 3. As it can be noticed only two policies are available in this MDP: π_a which takes action a in state x_1 and π_b which takes action b in this state. It is easy to verify that the stationary distribution ρ^{π_a} assigns probabilities $\frac{\varepsilon}{1+\varepsilon}$, $\frac{1}{1+\varepsilon}$, and 0 to x_1, x_2, and x_3,

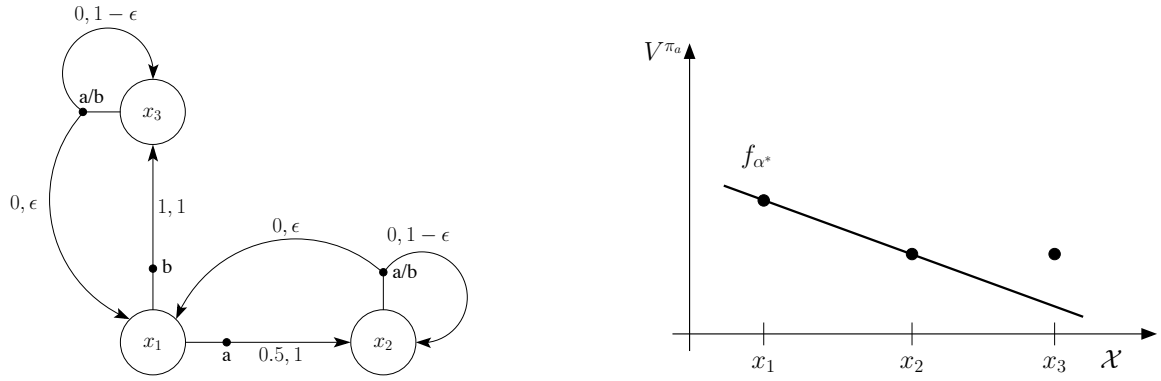


Figure 3: *(left)* The MDP used in the example of Section 6.2 and *(right)* the value function for policy π_a in this MDP.

while ρ^{π_b} has probabilities $\frac{\varepsilon}{1+\varepsilon}, 0$, and $\frac{1}{1+\varepsilon}$. Since ρ^{π_a} and ρ^{π_b} assign a probability 0 to two different states, it is not possible to find a finite constant C such that a distribution μ is lower-bounding both ρ^{π_a} and ρ^{π_b}, thus, $C = \infty$ and according to Theorem 8 LSPI may have an arbitrary bad performance.

Let initialize LSPI with the suboptimal policy π_a. The value function V^{π_a} is shown in Figure 3 (note that the specific values depend on the choice of ε and γ). Let $\mathcal{F} = \{f_\alpha(x) = \alpha_1 x + \alpha_2, \alpha \in \mathbb{R}^2\}$ be the space of lines in dimension 1. Let α^* be the solution to the following minimization problem $\alpha^* = \arg\inf_{\alpha\in\mathbb{R}} ||V^{\pi_a} - f_\alpha||^2_{\rho^{\pi_a}}$ (the projection of V^{π_a} onto space $\mathcal{F}$). Since ρ^{π_a} assigns a probability 0 to state x_3, the f_{α^*} in Figure 3 has a zero loss, that is, $||V^{\pi_a} - f_{\alpha^*}||_{\rho^{\pi_a}} = 0$. Nonetheless, while the greedy policy w.r.t. V^{π_a} is the optimal policy π_b, the policy improvement step w.r.t. f_{α^*} returns the policy π_a. As a result, although at each iteration the function space $\mathcal{F}$ may accurately approximate the value function of the current policy π w.r.t. its stationary distribution ρ^π, LSPI never improves its performance and returns π_a instead of the optimal policy π_b. By properly setting the rewards we could make the performance of π_a arbitrarily worse than π_b.

7. Conclusions

In this paper we presented a finite-sample analysis of the least-squares policy iteration (LSPI) algorithm (Lagoudakis and Parr, 2003). This paper substantially extends the analysis in Lazaric et al. (2010) by reporting all the lemmas used to prove the performance bounds of LSTD in the case of β-mixing and Markov chain processes and by analyzing how the performance of LSTD is propagated through iterations in LSPI.

More in detail, we first studied a version of LSTD, called pathwise LSTD, for policy evaluation. We considered a general setting where we do not make any assumption on the Markov chain. We derived an empirical performance bound that indicates how close the LSTD solution is to the value function at the states along a trajectory generated by following the policy and used by the algorithm. The bound is expressed in terms of the best possible approximation of the value function in the selected linear space (approximation error), and an estimation error which depends on the number of samples and the smallest strictly-positive eigenvalue of the sample-based Gram matrix. We then showed that when the Markov chain possesses a stationary distribution, one may deduce

generalization performance bounds using the stationary distribution of the chain as the generalization measure. In particular, we considered two cases, where the sample trajectory is generated by stationary and non-stationary β-mixing Markov chains, and derived the corresponding bounds. Finally, we considered the whole policy iteration algorithm (LSPI) and showed that under suitable conditions it is possible to bound the error cumulated through the iterations.

The techniques used for the analysis of LSTD have also been recently employed for the development of the finite-sample analysis of a number of novel algorithms such as LSTD with random projections (Ghavamzadeh et al., 2010), LassoTD (Ghavamzadeh et al., 2011), and Classification-based Policy Iteration with a Critic (Gabillon et al., 2011).

Technical issues. From a technical point of view there are two main open issues.

1. *Dependency on ν_n in the bound of Theorem 1*. In Section 4 we introduced the Markov design setting for regression in which the samples are obtained by following a Markov chain and the noise is a zero-mean martingale. By comparing the bound in Lemma 3 with the bounds for least-squares regression in deterministic design (see, e.g., Theorem 11.1 in Györfi et al., 2002), the main difference is the inverse dependency on the eigenvalue ν_n of the empirical Gram matrix. It is not clear whether this dependency is intrinsic in the process generating the samples or whether it can be removed. Abbasi-Yadkori et al. (2011) recently developed improved Azuma's inequalities for self-normalizing process (see also, e.g., de la Peña et al., 2007; de la Peña and Pang, 2009) which suggest that the bound can be improved by removing the dependency from ν_n and, thus, also from the L_∞-norm L of the features.

2. *The $\log n$ dependency in the generalization bounds*. Chaining techniques (Talagrand, 2005) can be successfully applied to remove the $\log n$ dependency in Pollard's inequalities for regression in bounded spaces. An interesting question is whether similar techniques can be applied to the refined analysis for squared losses and linear spaces (see, e.g., Lemma 10) used in our theorems.

Extensions. Some extensions to the current work are possible.

1. *LSTD(λ)*. A popular improvement to LSTD is the use of eligibility traces, thus obtaining LSTD(λ). The extension of the results presented in this paper to this setting does not seem to be straightforward since the regression problem solved in LSTD(λ) does not match the Markov design setting introduced in Definition 2. Hence, it is an open question how a finite-sample analysis of LSTD(λ) could be derived.

2. *Off-policy LSTD*. Yu and Bertsekas (2010) derived new bounds for projected linear equations substituting the $\frac{1}{\sqrt{1-\gamma^2}}$ term in front of the approximation error with a much sharper term depending on the spectral radius of some matrices defined by the problem. An open question is whether these new bounds can be effectively reused in the finite-sample analysis derived in this paper, thus obtaining much sharper bounds.

3. *Joint analysis of BRM and LSTD*. Scherrer (2010) recently proposed a unified view of Bellman residual minimization (BRM) (Schweitzer and Seidmann, 1985; Baird, 1995) and temporal difference methods through the notion of oblique projections. This suggests the possibility that the finite-sample analysis of LSTD could be extended to BRM through this unified view over the two methods.

Acknowledgments

We would like to thank Odalric Maillard for useful discussions. This work was supported by French National Research Agency (ANR) through the projects EXPLO-RA n° ANR-08-COSI-004 and LAMPADA n° ANR-09-EMER-007, by Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council and FEDER through the "contrat de projets état region (CPER) 2007–2013", European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 270327, and by PASCAL2 European Network of Excellence through PASCAL2 Pump Priming Programme.

Appendix A.

In this appendix we report a series of lemmata which are used throughout the paper. In particular, we derive concentration of measures inequalities for linear spaces and squared loss when samples are generated from different stochastic processes. We start with the traditional setting of independent and identically distributed samples in Section A.1, then move to samples generated from mixing processes in Section A.2, and finally consider the more general case of samples obtained by simulating a fast mixing Markov chain starting from an arbitrary distribution in Section A.3.

As a general rule, we use *proposition* to indicate results which are copied from other sources, while *lemma* refers to completely or partially new results.

A.1 IID Samples

Although in the setting considered in the paper the samples are non-i.i.d., we first report functional concentration inequalities for i.i.d. samples which will be later extended to stationary and non-stationary β-mixing processes. We first recall the definition of expected and empirical ℓ_2-norms for a function $f : \mathcal{X} \to \mathbb{R}$

$$\|f\|^2_{X_1^n} = \frac{1}{n}\sum_{t=1}^{n} |f(X_t)|^2 \qquad , \qquad \|f\|^2 = \mathbb{E}\left[|f(X_1)|^2\right].$$

Lemma 10 *Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$ bounded in absolute value by B. Let $X_1^n = \{X_1, \dots, X_n\}$ be a sequence of i.i.d. samples. For any $\varepsilon > 0$*

$$\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X_1^n} > \varepsilon\right] \le 3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{F}, X_1^{2n}\right)\right] \exp\left(-\frac{n\varepsilon^2}{288B^2}\right),$$

and

$$\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\|_{X_1^n} - 2\|f\| > \varepsilon\right] \le 3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{F}, X_1^{2n}\right)\right] \exp\left(-\frac{n\varepsilon^2}{288B^2}\right),$$

where $\mathcal{N}_2(\varepsilon, \mathcal{F}, X_1^n)$ is the (L_2, ε)-cover number of the function space $\mathcal{F}$ on the samples X_1^n (see Györfi et al. 2002).

Proof The first statement is proved in Györfi et al. (2002) and the second one can be proved similarly. ■

Proposition 11 *Let $\mathcal{F}$ be a class of linear functions $f : \mathcal{X} \to \mathbb{R}$ of dimension d and $\widetilde{\mathcal{F}}$ be the class of functions obtained by truncating functions $f \in \mathcal{F}$ at a threshold B. Then for any sample $X_1^n = \{X_1, \dots, X_n\}$ and $\varepsilon > 0$*

$$\mathcal{N}_2\left(\varepsilon, \widetilde{\mathcal{F}}, X_1^n\right) \le 3\left(\frac{3e(2B)^2}{\varepsilon^2}\right)^{2(d+1)}.$$

Proof Using Theorem 9.4. in Györfi et al. (2002) and the fact that the pseudo-dimension of $\widetilde{\mathcal{F}}$ is the same as $\mathcal{F}$, we have

$$\mathcal{N}_2\left(\varepsilon, \widetilde{\mathcal{F}}, X_1^n\right) \le 3\left(\frac{2e(2B)^2}{\varepsilon^2}\log\frac{3e(2B)^2}{\varepsilon^2}\right)^{d+1} \le 3\left(\frac{3e(2B)^2}{\varepsilon^2}\right)^{2(d+1)}.$$

■

We now use Proposition 11 to invert the bound in Lemma 10 for truncated linear spaces.

Corollary 12 *Let $\mathcal{F}$ be a class of linear functions $f : \mathcal{X} \to \mathbb{R}$ of dimension d, $\widetilde{\mathcal{F}}$ be the class of functions obtained by truncating functions $f \in \mathcal{F}$ at a threshold B, and $X_1^n = \{X_1, \dots, X_n\}$ be a sequence of i.i.d. samples. By inverting the bound of Lemma 10, for any $\tilde{f} \in \widetilde{\mathcal{F}}$, we have*

$$\|\tilde{f}\| - 2\|\tilde{f}\|_{X_1^n} \le \varepsilon(\delta),$$
$$\|\tilde{f}\|_{X_1^n} - 2\|\tilde{f}\| \le \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12B\sqrt{\frac{2\Lambda(n,d,\delta)}{n}}, \tag{23}$$

and $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log\left(9(12e)^{2(d+1)}\right)$.

Proof In order to prove the corollary it is sufficient to verify that the following inequality holds for the ε defined in Equation 23

$$3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \widetilde{\mathcal{F}}, X_1^{2n}\right)\right]\exp\left(-\frac{n\varepsilon^2}{288B^2}\right) \le \delta.$$

Using Proposition 11, we bound the first term as

$$\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \widetilde{\mathcal{F}}, X_1^{2n}\right)\right] \le 3\left(\frac{C_1}{\varepsilon^2}\right)^{2(d+1)},$$

with $C_1 = 3456eB^2$. Next we notice that $\Lambda(n,d,\delta) \ge 1$ and thus $\varepsilon \ge \sqrt{1/(nC_2)}$ with $C_2 = (288B^2)^{-1}$. Using these bounds in the original inequality and some algebra we obtain

$$\begin{aligned}
3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \widetilde{\mathcal{F}}, X_1^{2n}\right)\right]\exp\left(-\frac{n\varepsilon^2}{288B^2}\right) &\le 9\left(\frac{C_1}{\varepsilon^2}\right)^{2(d+1)}\exp\left(-nC_2\varepsilon^2\right) \\
&\le 9\left(nC_1C_2\right)^{2(d+1)}\exp\left(-C_2 n\frac{\Lambda(n,d,\delta)}{nC_2}\right) \\
&= 9\left(nC_1C_2\right)^{2(d+1)} n^{-2(d+1)}\frac{\delta}{e}\frac{1}{9(C_1C_2)^{2(d+1)}} \\
&= \frac{\delta}{e} \le \delta.
\end{aligned}$$

■

Non-functional versions of Corollary 12 can be simply obtained by removing the covering number from the statement of Lemma 10.

Corollary 13 *Let $f : \mathcal{X} \to \mathbb{R}$ be a function bounded in absolute value by B and $X_1^n = \{X_1, \dots, X_n\}$ be a sequence of i.i.d. samples. Then*

$$\|f\| - 2\|f\|_{X_1^n} \leq \varepsilon(\delta),$$

$$\|f\|_{X_1^n} - 2\|f\| \leq \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12B\sqrt{\frac{2}{n}\log\frac{3}{\delta}}\,.$$

A.2 Stationary β-mixing Processes

We first introduce β-mixing stochastic processes and β-mixing coefficients.

Definition 14 *Let $\{X_t\}_{t\geq 1}$ be a stochastic process. Let $X_i^j = \{X_i, X_{i+1}, \dots, X_j\}$ and $\sigma(X_i^j)$ denote the sigma-algebra generated by X_i^j. The i-th β-mixing coefficient of the stochastic process is defined by*

$$\beta_i = \sup_{t\geq 1} \mathbb{E}\left[\sup_{B\in\sigma(X_{t+i}^{\infty})} |\mathbb{P}(B|X_1^t) - \mathbb{P}(B)|\right].$$

The process $\{X_t\}_{t\geq 1}$ is said to be β-mixing if $\beta_i \to 0$ as $i \to \infty$. In particular, $\{X_t\}_{t\geq 1}$ mixes at an exponential rate with parameters β, b, κ if $\beta_i \leq \beta\exp(-bi^{\kappa})$. Finally, $\{X_t\}_{t\geq 1}$ is strictly stationary if $X_t \sim \nu$ for any $t > 0$.

Let $X_1, \dots, X_n$ be a sequence of samples drawn from a stationary β-mixing process with coefficients $\{\beta_i\}$. We first introduce the blocking technique of Yu (1994). Let us divide the sequence of samples into blocks of size k_n. For simplicity we assume $n = 2m_n k_n$ with $2m_n$ be the number of blocks.[7] For any $1 \leq j \leq m_n$ we define the set of indexes in an odd and even block respectively as

$$H_j = \{t : 2(j-1)k_n + 1 \leq t \leq (2j-1)k_n\}, \quad \text{and}$$
$$E_j = \{t : (2j-1)k_n + 1 \leq t \leq (2j)k_n\}.$$

Let $H = \cup_{j=1}^{m_n} H_j$ and $E = \cup_{j=1}^{m_n} E_j$ be the set of all indexes in the odd and even blocks, respectively. We use $X(H_j) = \{X_t : t \in H_j\}$ and $X(H) = \{X_t : t \in H\}$. We now introduce a ghost sample X' (the size of the ghost sample X' is equal to the number of samples in each block k_n) in each of the odd blocks such that the joint distribution of $X'(H_j)$ is the same as $X(H_j)$ but independent from any other block. In the following, we also use another ghost sample X'' independently generated from the same distribution as X'.

7. The extension to the general case is straightforward.

Proposition 15 (Yu, 1994) *Let $X_1,\dots,X_n$ be a sequence of samples drawn from a stationary β-mixing process with coefficients $\{\beta_i\}$. Let Q, Q' be the distributions of $X(H)$ and $X'(H)$, respectively. For any measurable function $h : \mathcal{X}^{m_n k_n} \to \mathbb{R}$ bounded by B*

$$\left|\mathbb{E}_Q\left[h(X(H))\right] - \mathbb{E}_{Q'}\left[h(X'(H))\right]\right| \le B m_n \beta_{k_n}.$$

Before moving to the extension of Propsition 10 to β mixing processes, we report this technical lemma.

Lemma 16 *Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$ bounded in absolute value by B and $X_1,\dots,X_n$ be a sequence of samples drawn from a stationary β-mixing process with coefficients $\{\beta_i\}$. For any $\varepsilon > 0$*

$$\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X_1^n} > \varepsilon\right] \le 2\delta(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n}, \tag{24}$$

$$\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\|_{X_1^n} - 2\sqrt{2}\|f\| > \varepsilon\right] \le 2\delta(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n}, \tag{25}$$

where

$$\delta(\varepsilon) = 3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{F}, X'(H)\cup X''(H)\right)\right]\exp\left(-\frac{m_n\varepsilon^2}{288B^2}\right).$$

Proof Similar to Meir (2000), we first introduce $\overline{\mathcal{F}}$ as the class of block functions $f : \mathcal{X}^{k_n} \to \mathbb{R}$ defined as

$$f\big(X(H_j)\big)^2 = \frac{1}{k_n}\sum_{t\in H_j} f(X_t)^2.$$

It is interesting to notice that block functions have exactly the same norms as the functions in $\mathcal{F}$. In fact

$$\|f\|^2_{X(H)} = \frac{1}{m_n}\sum_{j=1}^{m_n}|f(X(H_j))|^2 = \frac{1}{m_n}\sum_{j=1}^{m_n}\frac{1}{k_n}\sum_{t\in H_j}|f(X_t)|^2 = \|f\|_{X(H)}, \tag{26}$$

and

$$\|f\|^2 = \mathbb{E}\left[|f(X(H_1))|^2\right] = \frac{1}{k_n}\sum_{t\in H_1}\mathbb{E}\left[|f(X_t)|^2\right] = \mathbb{E}\left[|f(X_1)|^2\right] = \|f\|, \tag{27}$$

where in Equation 27, we used the fact that the process is stationary. We now focus on Equation 24

$$\begin{aligned}
&\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X_1^n} > \varepsilon\right]\\
&\overset{(a)}{\le} \mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - \big(\|f\|_{X(H)} + \|f\|_{X(E)}\big) > \varepsilon\right]\\
&\overset{(b)}{=} \mathbb{P}\left[\exists f \in \mathcal{F} : \frac{1}{2}\big(\|f\| - 2\|f\|_{X(H)}\big) + \frac{1}{2}\big(\|f\| - 2\|f\|_{X(E)}\big) > \varepsilon\right]\\
&\overset{(c)}{\le} \mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X(H)} > 2\varepsilon\right] + \mathbb{P}\left[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X(E)} > 2\varepsilon\right]\\
&\overset{(d)}{=} 2\mathbb{P}\left[\exists f \in \overline{\mathcal{F}} : \|f\| - 2\|f\|_{X(H)} > 2\varepsilon\right]\\
&\overset{(e)}{\le} 2\left(\mathbb{P}\left[\exists f \in \overline{\mathcal{F}} : \|f\| - 2\|f\|_{X'(H)} > 2\varepsilon\right] + m_n\beta_{k_n}\right)\\
&\overset{(f)}{\le} 2\delta'(2\varepsilon) + 2m_n\beta_{k_n}.
\end{aligned}$$

(a) We used the inequality $\sqrt{a+b} \geq \frac{1}{\sqrt{2}}(\sqrt{a}+\sqrt{b})$ to split the norm $\|f\|_{X_1^n} \geq \frac{1}{2}\big(\|f\|_{X(H)} + \|f\|_{X(E)}\big)$.
(b) Algebra.
(c) Split the probability.
(d) (1) Since the process is stationary the distribution over the even blocks is the same as the distribution over the odd blocks. (2) From Equations 26 and 27.
(e) Using Proposition 15 with h equals to the indicator function of the event inside the bracket, and the fact that the indicator function is bounded by $B = 1$ and its expected value is equal to the probability of the event.
(f) Lemma 10 on space $\overline{\mathcal{F}}$ where

$$\delta'(\varepsilon) = 3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \overline{\mathcal{F}}, \{X'(H_j), X''(H_j)\}_{j=1}^{m_n}\right)\right] \exp\left(-\frac{m_n\varepsilon^2}{288B^2}\right),$$

where X'' is a ghost sample independently generated from the same distribution as X'. Now we relate the ℓ_2-covering number of $\overline{\mathcal{F}}$ to the covering number of $\mathcal{F}$. Using the definition of f we have

$$\begin{aligned}
||f-g||^2_{X(H)} &= \frac{1}{m_n}\sum_{j=1}^{m_n}\Big(f\big(X(H_j)\big) - g\big(X(H_j)\big)\Big)^2 \\
&= \frac{1}{m_n k_n}\sum_{j=1}^{m_n}\left[\Big(\sum_{t\in H_j} f(X_t)^2\Big)^{\frac{1}{2}} - \Big(\sum_{t'\in H_j} g(X_{t'})^2\Big)^{\frac{1}{2}}\right]^2.
\end{aligned}$$

Taking the square and using the Cauchy-Schwarz inequality, each element of the outer summation may be written as

$$\begin{aligned}
&\sum_{t\in H_j}\big(f(X_t)^2 + g(X_t)^2\big) - 2\Big(\sum_{t\in H_j} f(X_t)^2\Big)^{\frac{1}{2}}\Big(\sum_{t'\in H_j} g(X_{t'})^2\Big)^{\frac{1}{2}} \\
&\qquad \leq \sum_{t\in H_j}\big(f(X_t)^2 + g(X_t)^2 - 2f(X_t)g(X_t)\big) = \sum_{t\in H_j}\big(f(X_t) - g(X_t)\big)^2.
\end{aligned}$$

By taking the sum over all the odd blocks we obtain

$$||f-g||^2_{X(H)} \leq ||f-g||^2_{X(H)},$$

which indicates that $\mathcal{N}_2\big(\varepsilon, \overline{\mathcal{F}}, \{X'(H_j), X''(H_j)\}_{j=1}^{m_n}\big) \leq \mathcal{N}_2\big(\varepsilon, \mathcal{F}, X'(H)\cup X''(H)\big)$. Therefore, we have $\delta'(2\varepsilon) \leq \delta(2\varepsilon) \leq \delta(\sqrt{2}\varepsilon)$, which concludes the proof.

With a similar approach, we can prove Equation 25

$$
\begin{aligned}
&\mathbb{P}\left[\exists f \in \mathcal{F} : \|f\|_{X_1^n} - 2\sqrt{2}\|f\| > \varepsilon\right] \\
&\overset{(a)}{\leq} \mathbb{P}\left[\exists f \in \mathcal{F} : \frac{\sqrt{2}}{2}\left(\|f\|_{X(H)} + \|f\|_{X(E)}\right) - 2\sqrt{2}\|f\| > \varepsilon\right] \\
&\overset{(b)}{=} \mathbb{P}\left[\exists f \in \mathcal{F} : \left(\frac{\sqrt{2}}{2}\|f\|_{X(H)} - \sqrt{2}\|f\|\right) + \left(\frac{\sqrt{2}}{2}\|f\|_{X(E)} - \sqrt{2}\|f\|\right) > \varepsilon\right] \\
&\overset{(c)}{\leq} \mathbb{P}\left[\exists f \in \mathcal{F} : \|f\|_{X(H)} - 2\|f\| > \sqrt{2}\varepsilon\right] + \mathbb{P}\left[\exists f \in \mathcal{F} : \|f\|_{X(E)} - 2\|f\| > \sqrt{2}\varepsilon\right] \\
&\overset{(d)}{=} 2\mathbb{P}\left[\exists f \in \overline{\mathcal{F}} : \|f\|_{X(H)} - 2\|f\| > \sqrt{2}\varepsilon\right] \\
&\overset{(e)}{\leq} 2\left(\mathbb{P}\left[\exists f \in \overline{\mathcal{F}} : \|f\|_{X'(H)} - 2\|f\| > \sqrt{2}\varepsilon\right] + m_n\beta_{k_n}\right) \\
&\overset{(f)}{\leq} 2\delta'(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n} \leq 2\delta(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n}.
\end{aligned}
$$

(a) We used the inequality $\sqrt{a+b} \leq (\sqrt{a}+\sqrt{b})$ to split the norm $\|f\|_{X_1^n} \leq \frac{\sqrt{2}}{2}\left(\|f\|_{X(H)} + \|f\|_{X(E)}\right)$. **(b)**-**(f)** use the same arguments as before. ■

Corollary 17 *Let $\mathcal{F}$ be a class of linear functions $f : \mathcal{X} \to \mathbb{R}$ of dimension d, $\widetilde{\mathcal{F}}$ be the class of functions obtained by truncating functions $f \in \mathcal{F}$ at a threshold B, and $X_1^n = \{X_1, \ldots, X_n\}$ be a sequence of samples drawn from a stationary exponentially fast β-mixing process with coefficients $\{\beta_i\}$. By inverting the bound of Lemma 16, for any $\tilde{f} \in \widetilde{\mathcal{F}}$ we have*

$$\|\tilde{f}\| - 2\|\tilde{f}\|_{X_1^n} \leq \varepsilon(\delta),$$

$$\|\tilde{f}\|_{X_1^n} - 2\sqrt{2}\|\tilde{f}\| \leq \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12B\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b}, 1\right\}^{1/\kappa}}, \tag{28}$$

and $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log^+\left(\max\{18(6e)^{2(d+1)}, \beta\}\right)$.

Proof In order to prove the statement, we need to verify that ε in Equation 28 satisfies

$$\delta' = 6\mathbb{E}\left[\mathcal{N}_2\left(\frac{1}{12}\varepsilon, \widetilde{\mathcal{F}}, X'(H) \cup X''(H)\right)\right]\exp\left(-\frac{m_n\varepsilon^2}{144B^2}\right) + 2m_n\beta_{k_n} \leq \delta\,.$$

Using Proposition 11 the covering number can be bounded by

$$\mathbb{E}\left[\mathcal{N}_2\left(\frac{1}{12}\varepsilon, \widetilde{\mathcal{F}}, X'(H) \cup X''(H)\right)\right] \leq 3\left(\frac{1728eB^2}{\varepsilon^2}\right)^{2(d+1)}.$$

By recalling the definition of the β-coefficients $\{\beta_i\}$ and $k_n \geq 1$ we have

$$2m_n\beta_{k_n} \leq \frac{n}{k_n}\beta\exp(-bk_n^\kappa) \leq n\beta\exp(-bk_n^\kappa)\,.$$

From the last two inequalities, $m_n = n/2k_n$, setting $C_1 = 1728eB^2$ and $D = 2(d+1)$ we obtain

$$\delta' \leq 18\left(\frac{C_1}{\varepsilon^2}\right)^D \exp\left(-\frac{n\varepsilon^2}{144B^2}\frac{1}{2k_n}\right) + n\beta\exp(-bk_n^\kappa).$$

By equalizing the arguments of the two exponential we obtain the definition of k_n as

$$k_n = \left\lceil \left(\frac{nC_2\varepsilon^2}{b}\right)^{\frac{1}{\kappa+1}} \right\rceil,$$

where $C_2 = (576B^2)^{-1}$, which implies

$$\max\left\{\left(\frac{nC_2\varepsilon^2}{b}\right)^{\frac{1}{\kappa+1}},1\right\} \leq k_n \leq \max\left\{\left(\frac{2nC_2\varepsilon^2}{b}\right)^{\frac{1}{\kappa+1}},1\right\}.$$

Thus we have the bound

$$\frac{1}{2k_n} \geq \frac{1}{4}\min\left\{\left(\frac{b}{nC_2\varepsilon^2}\right)^{\frac{1}{\kappa+1}},2\right\} \geq \frac{1}{4}\min\left\{\left(\frac{b}{nC_2\varepsilon^2}\right)^{\frac{1}{\kappa+1}},1\right\}.$$

Using the above inequalities, we may write δ' as

$$\delta' \leq 18\left(\frac{C_1}{\varepsilon^2}\right)^D \exp\left(-\min\left\{\frac{b}{nC_2\varepsilon^2},1\right\}^{\frac{1}{\kappa+1}} nC_2\varepsilon^2\right) + n\beta\exp\left(-b\max\left\{\frac{nC_2\varepsilon^2}{b},1\right\}^{\frac{\kappa}{\kappa+1}}\right).$$

The objective now is to make the arguments of the two exponential equal. For the second argument we have

$$b\max\left\{\frac{nC_2\varepsilon^2}{b},1\right\}^{\frac{\kappa}{\kappa+1}} = b\max\left\{\frac{nC_2\varepsilon^2}{b},1\right\}\min\left\{\frac{b}{nC_2\varepsilon^2},1\right\}^{\frac{1}{\kappa+1}} \geq nC_2\varepsilon^2\min\left\{\frac{b}{nC_2\varepsilon^2},1\right\}^{\frac{1}{\kappa+1}}.$$

Thus

$$\delta' \leq \left(18\left(\frac{C_1}{\varepsilon^2}\right)^D + n\beta\right)\exp\left(-\min\left\{\frac{b}{nC_2\varepsilon^2},1\right\}^{\frac{1}{\kappa+1}} nC_2\varepsilon^2\right).$$

Now we plug in ε from Equation 28. Using the fact that $\Lambda \geq 1$, we know that $\varepsilon^2 \geq (nC_2)^{-1}$, and thus

$$\delta' \leq \left(18\,(nC_1C_2)^D + n\beta\right)\exp\left(-\Lambda\right).$$

Using the definition of Λ, we obtain

$$\delta' \leq \left(18\,(nC_1C_2)^D + n\beta\right)n^{-D}\max\{18(C_1C_2)^D,\beta\}^{-1}\frac{\delta}{e} \leq (1+n^{1-D})\frac{\delta}{e} \leq (1+1)\frac{\delta}{e} \leq \delta\,,$$

which concludes the proof. ■

In order to understand better the shape of the estimation error, we consider a simple β-mixing process with parameters $\beta = b = \kappa = 1$. Equation 28 reduces to

$$\varepsilon(\delta) = \sqrt{\frac{288B^2\Lambda(n,d,\delta)}{n}^2},$$

with $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log\left(18(6e)^{2(d+1)}\right)$. It is interesting to notice that the shape of the bound in this case resembles the structure of the bound in Corollary 12 for i.i.d. samples. Finally, we report the non-functional version of the previous corollary.

Corollary 18 *Let $\mathcal{F}$ be a class of linear functions $f : \mathcal{X} \to \mathbb{R}$ of dimension d such that its features $\varphi_i : \mathcal{X} \to \mathbb{R}$ are bounded in absolute value by L for any $i = 1,\dots,d$ and $X_1^n = \{X_1,\dots,X_n\}$ be a sequence of samples drawn from a stationary exponentially fast β-mixing process with coefficients $\{\beta_i\}$. For any $f \in \mathcal{F}$ we have*

$$\|f\| - 2\|f\|_{X_1^n} \le \varepsilon(\delta),$$

$$\|f\|_{X_1^n} - 2\sqrt{2}\|f\| \le \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12||\alpha||L\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}},$$

and $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log^+\left(\max\{18(6e)^{2(d+1)},\beta\}\right)$.

Proof Let $\mathcal{G} = \left\{g_\alpha = \frac{f_\alpha}{L||\alpha||}\right\}$ so that

$$||g_\alpha||_\infty = \frac{1}{L||\alpha||}||f_\alpha||_\infty \le \frac{1}{L||\alpha||}||\alpha||\sup_i||\varphi_i(x)||_\infty \le 1.$$

We can thus apply Lemma 16 to the bounded space $\mathcal{G}$ with $B = 1$. By using a similar inversion as in Corollary 17, we thus obtain that with probability $1-\delta$, for any function $g_\alpha \in \mathcal{G}$

$$\|g_\alpha\| - 2\|g_\alpha\|_{X_1^n} \le \varepsilon(\delta),$$
$$\|g_\alpha\|_{X_1^n} - 2\sqrt{2}\|g_\alpha\| \le \varepsilon(\delta),$$

with

$$\varepsilon(\delta) = 12\sqrt{\frac{2\Lambda(n,d,\delta)}{n}\max\left\{\frac{\Lambda(n,d,\delta)}{b},1\right\}^{1/\kappa}}.$$

Finally, we notice that $||g_\alpha|| = \frac{1}{L||\alpha||}||f_\alpha||$ and $||g_\alpha||_{X_1^n} = \frac{1}{L||\alpha||}||f_\alpha||_{X_1^n}$ and the statement follows. ■

Corollary 19 *Let $f : \mathcal{X} \to \mathbb{R}$ be a linear function, $\tilde{f}$ be its truncation at a threshold B, and $X_1^n = \{X_1, \dots, X_n\}$ be a sequence of samples drawn from a stationary exponentially fast β-mixing process with coefficients $\{\beta_i\}$. Then*

$$\|\tilde{f}\| - 2\|\tilde{f}\|_{X_1^n} \le \varepsilon(\delta),$$
$$\|\tilde{f}\|_{X_1^n} - 2\sqrt{2}\|\tilde{f}\| \le \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12B\sqrt{\frac{2\Lambda(n,\delta)}{n}\max\left\{\frac{\Lambda(n,\delta)}{b}, 1\right\}^{1/\kappa}},$$

$\Lambda(n,\delta) = \log\frac{e}{\delta} + \log\big(\max\{6, n\beta\}\big)$.

Proof The proof follows the same steps as in Corollary 17. We have the following sequence of inequalities

$$\delta' \le 6\exp\left(-\frac{nC_2\varepsilon^2}{k_n}\right) + \frac{n}{k_n}\beta\exp(-bk_n^\kappa) \le (6+n\beta)\exp(-\Lambda)$$
$$= (6+n\beta)\max\{6, n\beta\}^{-1}\frac{\delta}{e} \le (1+1)\frac{\delta}{e} \le \delta,$$

where $C_2 = (576B^2)^{-1}$. ■

A.3 Markov Chains

We first review the conditions for the convergence of Markov chains (Theorem 13.3.3. in Meyn and Tweedie 1993).

Proposition 20 *Let $\mathcal{M}$ be an ergodic and aperiodic Markov chain defined on $\mathcal{X}$ with stationary distribution ρ. If $P(A|x)$ is the transition kernel of $\mathcal{M}$ with $A \subseteq \mathcal{X}$ and $x \in \mathcal{X}$, then for any initial distribution λ*

$$\lim_{i\to\infty}\left|\left|\int_{\mathcal{X}} \lambda(dx)P^i(\cdot|x) - \rho(\cdot)\right|\right|_{TV} = 0,$$

where $||\cdot||_{TV}$ is the total variation norm.

Definition 21 *Let $\mathcal{M}$ be an ergodic and aperiodic Markov chain with stationary distribution ρ. $\mathcal{M}$ is mixing with an exponential rate with parameters β, b, κ, if its β-mixing coefficients $\{\beta_i\}$ satisfy $\beta_i \le \beta\exp(-bi^\kappa)$. Then for any initial distribution λ*

$$\left|\left|\int_{\mathcal{X}} \lambda(dx)P^i(\cdot|x) - \rho(\cdot)\right|\right|_{TV} \le \beta\exp(-bi^\kappa).$$

Lemma 22 *Let $\mathcal{M}$ be an ergodic and aperiodic Markov chain with a stationary distribution ρ. Let $X_1, \dots, X_n$ be a sequence of samples drawn from the stationary distribution of the Markov chain ρ and $X'_1, \dots, X'_n$ be a sequence of samples such that $X'_1 \sim \rho'$ and $X'_{1<t\le n}$ are generated by simulating $\mathcal{M}$ from X'_1. Let η be an event defined on $\mathcal{X}^n$, then*

$$\left|\mathbb{P}\left[\eta(X_1, \dots, X_n)\right] - \mathbb{P}\left[\eta(X'_1, \dots, X'_n)\right]\right| \le \|\rho' - \rho\|_{TV}$$

Proof We prove one side of the inequality. Let Q be the conditional joint distribution of $(X_{1<t\leq n}|X_1 = x)$ and Q' be the conditional joint distribution of $(X'_{1<t\leq n}|X'_1 = x)$. We first notice that Q is exactly the same as Q'. In fact, the first sequence $(X_{1<t\leq n})$ is generated by drawing X_1 from the stationary distribution ρ and then following the Markov chain. Similarly, the second sequence $(X'_{1<t\leq n})$ is obtained following the Markov chain from $X'_1 \sim \rho'$. As a result, the conditional distributions of the two sequences is exactly the same and just depend on the Markov chain. As a result, we obtain the following sequence of inequalities

$$\begin{aligned}
\mathbb{P}\Big[\eta(X_1,\ldots,X_n)\Big] &= \mathbb{E}_{X_1,\ldots,X_n}\left[\mathbb{I}\left\{\eta(X_1,\ldots,X_n)\right\}\right] \\
&= \mathbb{E}_{X_1\sim\rho}\left[\mathbb{E}_{X_2,\ldots,X_n}\left[\mathbb{I}\left\{\eta(X_1,X_2\ldots,X_n)\right\}|X_1\right]\right] \\
&= \mathbb{E}_{X_1\sim\rho}\left[\mathbb{E}_{X'_2,\ldots,X'_n}\left[\mathbb{I}\left\{\eta(X_1,X'_2\ldots,X'_n)\right\}|X_1\right]\right] \\
&\overset{(a)}{\leq} \mathbb{E}_{X_1\sim\rho'}\left[\mathbb{E}_{X'_2,\ldots,X'_n}\left[\mathbb{I}\left\{\eta(X_1,X'_2\ldots,X'_n)\right\}|X_1\right]\right] + \|\rho'-\rho\|_{TV} \\
&\overset{(b)}{=} \mathbb{E}_{X'_1\sim\rho'}\left[\mathbb{E}_{X'_2,\ldots,X'_n}\left[\mathbb{I}\left\{\eta(X'_1,X'_2\ldots,X'_n)\right\}|X'_1\right]\right] + \|\rho'-\rho\|_{TV} \\
&= \mathbb{P}\Big[\eta(X'_1,\ldots,X'_n)\Big] + \|\rho'-\rho\|_{TV}.
\end{aligned}$$

Note that $\mathbb{I}\{\cdot\}$ is the indicator function.
(a) simply follows from

$$\begin{aligned}
\mathbb{E}_{X\sim\rho}[f(X)] - \mathbb{E}_{X\sim\rho'}[f(X)] &= \int_{\mathcal{X}} f(x)\rho(dx) - \int_{\mathcal{X}} f(x)\rho'(dx) \\
&\leq ||f||_\infty \int_{\mathcal{X}} \big(\rho(dx)-\rho'(dx)\big) \leq ||f||_\infty ||\rho-\rho'||_{TV}.
\end{aligned}$$

(b) From the fact that $X_1 = X'_1 = x$. ■

Lemma 23 *Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$ bounded in absolute value by B, $\mathcal{M}$ be a an ergodic and aperiodic Markov chain with a stationary distribution ρ. Let $\mathcal{M}$ be mixing with an exponential rate with parameters β, b, κ. Let λ be an initial distribution over $\mathcal{X}$ and $X_1,\ldots,X_n$ be a sequence of samples such that $X_1 \sim \lambda$ and $X_{1<t\leq n}$ obtained by following $\mathcal{M}$ from X_1. For any $\varepsilon > 0$,*

$$\mathbb{P}\Big[\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X_1^n} > \varepsilon\Big] \leq \|\lambda-\rho\|_{TV} + 2\delta(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n},$$

and

$$\mathbb{P}\Big[\exists f \in \mathcal{F} : \|f\|_{X_1^n} - 2\sqrt{2}\|f\| > \varepsilon\Big] \leq \|\lambda-\rho\|_{TV} + 2\delta(\sqrt{2}\varepsilon) + 2m_n\beta_{k_n},$$

where

$$\delta(\varepsilon) = 3\mathbb{E}\left[\mathcal{N}_2\left(\frac{\sqrt{2}}{24}\varepsilon, \mathcal{F}, X(H)\cup X'(H)\right)\right]\exp\left(-\frac{m_n\varepsilon^2}{288B^2}\right).$$

Proof The proof is an immediate consequence of Lemma 16 and Lemma 22 by defining $\eta(X_1,\ldots,X_n)$ as

$$\eta(X_1,\ldots,X_n) = \{\exists f \in \mathcal{F} : \|f\| - 2\|f\|_{X_1^n} > \varepsilon\},$$

and

$$\eta(X_1,\dots,X_n) = \{\exists f \in \mathcal{F} : \|f\|_{X_1^n} - 2\sqrt{2}\|f\| > \varepsilon\},$$

respectively. ■

Finally, we consider a special case in which out of the n total number of samples, $\tilde{n}$ ($1 \le \tilde{n} < n$) are used to "burn" the chain and $n-\tilde{n}$ are actually used as training samples.

Lemma 24 *Let $\mathcal{F}$ be a class of linear functions $f : \mathcal{X} \to \mathbb{R}$ of dimension d and $\widetilde{\mathcal{F}}$ be the class of functions obtained by truncating functions $f \in \mathcal{F}$ at a threshold B. Let $\mathcal{M}$ be an ergodic and aperiodic Markov chain with a stationary distribution ρ. Let $\mathcal{M}$ be mixing with an exponential rate with parameters β, b, κ. Let μ be the initial distribution and $X_1,\dots,X_n$ be a sequence of samples such that $X_1 \sim \mu$ and $X_{1<t\le n}$ obtained by following $\mathcal{M}$ from X_1. If the first $\tilde{n}$ ($1 \le \tilde{n} < n$) samples are used to burn the chain and $n-\tilde{n}$ are actually used as training samples, by inverting Lemma 23, for any $\tilde{f} \in \widetilde{\mathcal{F}}$, we obtain*

$$\|\tilde{f}\| - 2\|\tilde{f}\|_{X_1^n} \le \varepsilon(\delta),$$

$$\|\tilde{f}\|_{X_1^n} - 2\sqrt{2}\|\tilde{f}\| \le \varepsilon(\delta),$$

with probability $1-\delta$, where

$$\varepsilon(\delta) = 12B\sqrt{\frac{2\Lambda(n-\tilde{n},d,\delta)}{(n-\tilde{n})}\max\left\{\frac{\Lambda(n-\tilde{n},d,\delta)}{b},1\right\}^{1/\kappa}},$$

and $\Lambda(n,d,\delta) = 2(d+1)\log n + \log\frac{e}{\delta} + \log^+\left(\max\{18(6e)^{2(d+1)},\beta\}\right)$, and $\tilde{n} = \left(\frac{1}{b}\log\frac{2e\beta n}{\delta}\right)^{1/\kappa}$.

Proof After $\tilde{n}$ steps, the first sample used in the training set ($X_{\tilde{n}+1}$) is drawn from the distribution $\lambda = \mu P^{\tilde{n}}$. Using Proposition 20 and Definition 21 we have

$$||\lambda - \rho||_{TV} \le \beta\exp(-b\tilde{n}^\kappa). \tag{29}$$

We first substitute the total variation in Lemma 23 with the bound in Equation 29, and then verify that ε in Equation 24 satisfies the following inequality.

$$\begin{aligned}
\delta' &= \|\lambda-\rho\|_{TV} + 2\delta(\sqrt{2}\varepsilon) + 2m_{n-\tilde{n}}\beta_{k_{n-\tilde{n}}} \\
&\le \beta\exp(-b\tilde{n}^\kappa) + 18\left(\frac{C_1}{\varepsilon^2}\right)^D \exp\left(-\frac{(n-\tilde{n})C_2\varepsilon^2}{k_{n-\tilde{n}}}\right) + (n-\tilde{n})\beta\exp(-bk_{n-\tilde{n}}^\kappa) \\
&\le \left(\frac{1}{2n} + 1 + (n-\tilde{n})^{1-D}\right)\frac{\delta}{e} \le \left(\frac{1}{2}+1+1\right)\frac{\delta}{e} \le \delta,
\end{aligned}$$

where $C_1 = 1728eB^2$ and $C_2 = (288B^2)^{-1}$. The above inequality can be verified by following the same steps as in Corollary 17 and by optimizing the bound for $\tilde{n}$. ■

References

Y. Abbasi-Yadkori, D. Pal, and Cs. Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2011.

A. Antos, Cs. Szepesvári, and R. Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning Journal*, 71:89–129, 2008.

L. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 30–37, 1995.

D. Bertsekas. *Dynamic Programming and Optimal Control, volume II*. Athena Scientific, 2007.

D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

J. Boyan. Least-squares temporal difference learning. *Proceedings of the 16th International Conference on Machine Learning*, pages 49–56, 1999.

S. Bradtke and A. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22:33–57, 1996.

V. de la Peña and G. Pang. Exponential inequalities for self-normlized processes with applications. *Electronic Communications in Probability*, 14:372–381, 2009.

V. de la Peña, M. Klass, and T. Leung Lai. Pseudo-maximization and self-normalized processes. *Propability Surveys*, 4:172–192, 2007.

S. Delattre and S. Gaïffas. Nonparametric regression with martingale increment errors. *Stochastic Processes and their Applications*, 121(12):2899 – 2924, 2011.

A. M. Farahmand, M. Ghavamzadeh, Cs. Szepesvári, and S. Mannor. Regularized policy iteration. In *Advances in Neural Information Processing Systems 21*, pages 441–448. MIT Press, 2008.

A. M. Farahmand, R. Munos, and Cs. Szepesvári. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems*, 2010.

V. Gabillon, A. Lazaric, M. Ghavamzadeh, and B. Scherrer. Classification-based policy iteration with a critic. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, pages 1049–1056, 2011.

M. Ghavamzadeh, A. Lazaric, O. Maillard, and R. Munos. Lstd with random projections. In *Advances in Neural Information Processing Systems*, pages 721–729, 2010.

M. Ghavamzadeh, A. Lazaric, R. Munos, and M. Hoffman. Finite-sample analysis of lasso-td. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1177–1184, 2011.

L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A Distribution-free Theory of Nonparametric Regression*. Springer-Verlag, New York, 2002.

D. Hsu, S. Kakade, and T. Zhang. Random design analysis of ridge regression. In *Proceedings of the 25th Conference on Learning Theory*, 2012.

M. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of lstd. In *Proceedings of the 27th International Conference on Machine Learning*, pages 615–622, 2010.

R. Meir. Nonparametric time series prediction through adaptive model selection. *Machine Learning*, 39(1):5–34, April 2000.

S. P. Meyn and R. L. Tweedie. *Markov Chains and Stochastic Stability*. Springer–Verlag, 1993.

B. Ávila Pires and Cs. Szepesvári. Statistical linear estimation with penalized estimators: an application to reinforcement learning. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

B. Scherrer. Should one compute the temporal difference fix point or minimize the bellman residual? the unified oblique projection view. In *Proceedings of the 27th International Conference on Machine Learning*, pages 959–966, 2010.

P. Schweitzer and A. Seidmann. Generalized polynomial approximations in markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110:568–582, 1985.

R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

M. Talagrand. *The Generic Chaining: Upper and Lower Bounds of Stochastic Processes*. Springer–Verlag, 2005.

J. Tsitsiklis and B. Van Roy. An analysis of temporal difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42:674–690, 1997.

B. Yu. Rates of convergence for empirical processes of stationary mixing sequences. *The Annals of Probability*, 22(1):94–116, January 1994.

H. Yu. Convergence of least squares temporal difference methods under general conditions. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1207–1214, 2010.

H. Yu and D. Bertsekas. Error bounds for approximations from projected linear equations. *Math. Oper. Res.*, 35(2):306–329, 2010.

Journal of Machine Learning Research 13 (2012) 3075-3102 Submitted 10/11; Revised 9/12; Published 10/12

Discriminative Hierarchical Part-based Models for Human Parsing and Action Recognition

Yang Wang YWANG@CS.UMANITOBA.CA
Department of Computer Science
University of Manitoba
Winnipeg, MB, R3T 2N2, Canada

Duan Tran DDTRAN2@UIUC.EDU
Zicheng Liao LIAO17@UIUC.EDU
David Forsyth DAF@UIUC.EDU
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

Editor: Isabelle Guyon and Vassilis Athitsos

Abstract

We consider the problem of parsing human poses and recognizing their actions in static images with part-based models. Most previous work in part-based models only considers rigid parts (e.g., torso, head, half limbs) guided by human anatomy. We argue that this representation of parts is not necessarily appropriate. In this paper, we introduce hierarchical poselets—a new representation for modeling the pose configuration of human bodies. Hierarchical poselets can be rigid parts, but they can also be parts that cover large portions of human bodies (e.g., torso + left arm). In the extreme case, they can be the whole bodies. The hierarchical poselets are organized in a hierarchical way via a structured model. Human parsing can be achieved by inferring the optimal labeling of this hierarchical model. The pose information captured by this hierarchical model can also be used as a intermediate representation for other high-level tasks. We demonstrate it in action recognition from static images.

Keywords: human parsing, action recognition, part-based models, hierarchical poselets, max-margin structured learning

1. Introduction

Modeling human bodies (or articulated objects in general) in images is a long-lasting problem in computer vision. Compared with rigid objects (e.g., faces and cars) which can be reasonably modeled using several prototypical templates, human bodies are much more difficult to model due to the wide variety of possible pose configurations.

A promising solution for dealing with the pose variations is to use part-based models. Part-based representations, such as cardboard people (Ju et al., 1996) or pictorial structure (Felzenszwalb and Huttenlocher, 2005), provide an elegant framework for modeling articulated objects, such as human bodies. A part-based model represents the human body as a constellation of a set of rigid parts (e.g., torso, head, half limbs) constrained in some fashion. The typical constraints used are tree-structured kinematic constraints between adjacent body parts, for example, torso-upper half-limb connection,

or upper-lower half-limb connection. Part-based models consist of two important components: (1) part appearances specifying what each body part should look like in the image; (2) configuration priors specifying how parts should be arranged relative to each other. Part-based models have been used extensively in various computer vision applications involving humans, such as human parsing (Felzenszwalb and Huttenlocher, 2005; Ramanan, 2006), kinematic tracking (Ramanan et al., 2005), action recognition (Yang et al., 2010) and human-object interaction (Yao and Fei-Fei, 2010).

Considerable progress has been made to improve part-based models. For example, there has been a line of work on using better appearance models in part-based models. A representative example is the work by Ramanan (2006), who learns color histograms of parts from an initial edge-based model. Ferrari et al. (2008) and Eichner and Ferrari (2009) further improve the part appearance models by reducing the search space using various tricks, for example, the relative locations of part locations with respect to a person detection and the relationship between different part appearances (e.g., upper-arm and torso tend to have the same color), Andriluka et al. (2009) build better edge-based appearance models using the HOG descriptors (Dalal and Triggs, 2005). Sapp et al. (2010b) develop efficient inference algorithm to allow the use of more expensive features. There is also work (Johnson and Everingham, 2009; Mori et al., 2004; Mori, 2005; Srinivasan and Shi, 2007) on using segmentation as a pre-processing step to provide better spatial support for computing part appearances.

Another line of work is on improving configuration priors in part-based models. Most of them focus on developing representations and fast inference algorithms that by-pass the limitations of kinematic tree-structured spatial priors in standard pictorial structure models. Examples include common-factor models (Lan and Huttenlocher, 2005), loopy graphs (Jiang and Martin, 2008; Ren et al., 2005; Tian and Sclaroff, 2010; Tran and Forsyth, 2010), mixtures of trees (Wang and Mori, 2008). There is also work on building spatial priors that adapt to testing examples (Sapp et al., 2010a).

Most of the previous work on part-based models use rigid parts that are anatomically meaningful, for example, torso, head, half limbs. Those rigid parts are usually represented as rectangles (e.g., Andriluka et al. 2009; Felzenszwalb and Huttenlocher 2005; Ramanan 2006; Ren et al. 2005; Sigal and Black 2006; Wang and Mori 2008) or parallel lines (e.g., Ren et al. 2005). However, as pointed out by some recent work (Bourdev and Malik, 2009; Bourdev et al., 2010), rigid parts are not necessarily the best representation since rectangles and parallel lines are inherently difficult to detect in natural images.

In this paper, we introduce a presentation of parts inspired by the early work of Marr (1982). The work in Marr (1982) recursively represents objects as generalized cylinders in a coarse-to-fine hierarchical fashion. In this paper, we extend Marr's idea for two problems in the general area of "looking at people". The first problem is human parsing, also known as human pose estimation. The goal is to find the location of each body part (torso, head, limbs) of a person in a static image. We use a part-based approach for human parsing. The novelty of our work is that our notion of "parts" can range from basic rigid parts (e.g., torso, head, half-limb), to large pieces of bodies covering more than one rigid part (e.g., torso + left arm). In the extreme case, we have "parts" corresponding to the whole body. We propose a new representation called "hierarchical poselets" to capture this hierarchy of parts. We infer the human pose using this hierarchical representation.

The hierarchical poselet also provides rich information about body poses that can be used in other applications. To demonstrate this, we apply it to recognize human action in static images. In this application, we use hierarchical poselets to capture various pose information of the human

body, this information is further used as some intermediate representation to infer the action of the person.

A preliminary version of this work appeared in Wang et al. (2011). We organize the rest of the paper as follows. Section 2 reviews previous work in human parsing and action recognition. Section 3 introduces hierarchical poselet, a new representation for modeling human body configurations. Section 4 describes how to use hierarchical poselets for human parsing. Section 5 develops variants of hierarchical poselets for recognizing human action in static images. We present experimental results on human parsing and action recognition in Section 6 and conclude in Section 7.

2. Previous Work

Finding and understanding people from images is a very active area in computer vision. In this section, we briefly review previous work in human parsing and action recognition that is most related to our work.

Human parsing: Early work related to finding people from images is in the setting of detecting and tracking people with kinematic models in both 2D and 3D. Forsyth et al. (2006) provide an extensive survey of this line of work.

Recent work has examined the problem in static images. Some of these approaches are exemplar-based. For example, Toyama and Blake (2001) track people using 2D exemplars. Mori and Malik (2002) and Sullivan and Carlsson (2002) estimate human poses by matching pre-stored 2D templates with marked ground-truth 2D joint locations. Shakhnarovich et al. (2003) use local sensitive hashing to allow efficient matching when the number of exemplars is large.

Part-based models are becoming increasingly popular in human parsing. Early work includes the cardboard people (Ju et al., 1996) and the pictorial structure (Felzenszwalb and Huttenlocher, 2005). Tree-structured models are commonly used due to its efficiency. But there are also methods that try to alleviate the limitation of tree-structured models, include common-factor models (Lan and Huttenlocher, 2005), loopy graphs (Jiang and Martin, 2008; Ren et al., 2005; Tian and Sclaroff, 2010; Tran and Forsyth, 2010), mixtures of trees (Wang and Mori, 2008).

Many part-based models use discriminative learning to train the model parameters. Examples include the conditional random fields (Ramanan and Sminchisescu, 2006; Ramanan, 2006), max-margin learning (Kumar et al., 2009; Wang et al., 2011; Yang and Ramanan, 2011) and boosting (Andriluka et al., 2009; Sapp et al., 2010b; Singh et al., 2010). Previous approaches have also explored various features, including image segments (superpixels) (Johnson and Everingham, 2009; Mori et al., 2004; Mori, 2005; Sapp et al., 2010a,b; Srinivasan and Shi, 2007), color features (Ramanan, 2006; Ferrari et al., 2008), gradient features (Andriluka et al., 2009; Johnson and Everingham, 2010; Wang et al., 2011; Yang and Ramanan, 2011).

Human action recognition: Most of the previous work on human action recognition focuses on videos. Some work (Efros et al., 2003) uses global template for action recognition. A lot of recent work (Dollár et al., 2005; Laptev et al., 2008; Niebles et al., 2006) uses bag-of-words models. There is also work (Ke et al., 2007; Niebles and Fei-Fei, 2007) using part-based models.

Compared with videos, human action recognition from static images is a relatively less-studied area. Wang et al. (2006) provide one of the earliest examples of action recognition in static images. Recently, template models (Ikizler-Cinbis et al., 2009), bag-of-words models (Delaitre et al., 2010), part-based models (Delaitre et al., 2010; Yang et al., 2010) have all been proposed for static-image action recognition. There is also a line of work on using contexts for action recognition in static

images, including human-object context (Desai et al., 2010; Gupta et al., 2009; Yao and Fei-Fei, 2010) and group context (Lan et al., 2010; Maji et al., 2011).

3. Hierarchical Poselets

Our pose representation is based on the concept of "poselet" introduced in Bourdev and Malik (2009). In a nutshell, poselets refer to pieces of human poses that are tightly clustered in both appearance and configuration spaces. Poselets have been shown to be effective at person detection (Bourdev and Malik, 2009; Bourdev et al., 2010).

In this paper, we propose a new representation called *hierarchical poselets*. Hierarchical poselets extend the original poselets in several important directions to make them more appropriate for human parsing. We start by highlighting the important properties of our representation.

Beyond rigid "parts": Most of the previous work in part-based human modeling are based on the notion that the human body can be modeled as a set of rigid parts connected in some way. Almost all of them use a natural definition of parts (e.g., torso, head, upper/lower limbs) corresponding to body segments, and model those parts as rectangles, parallel lines, or other primitive shapes.

As pointed out by Bourdev and Malik (2009), this natural definition of "parts" fails to acknowledge the fact that rigid parts are not necessarily the most salient features for visual recognition. For example, rectangles and parallel lines can be found as limbs, but they can also be easily confused with windows, buildings, and other objects in the background. So it is inherently difficult to build reliable detectors for those parts. On the other hand, certain visual patterns covering large portions of human bodies, for example, "a torso with the left arm raising up" or "legs in lateral pose", are much more visually distinctive and easier to identify. This phenomenon was observed even prior to the work of poselet and was exploited to detect stylized human poses and build appearance models for kinematic tracking (Ramanan et al., 2005).

Multiscale hierarchy of "parts": Another important property of our representation is that we define "parts" at different levels of hierarchy to cover pieces of human poses at various granularity, ranging from the configuration of the whole body, to small rigid parts. In particular, we define 20 parts to represent the human pose and organize them in a hierarchy shown in Figure 1. To avoid terminological confusion, we will use "part" to denote one of the 20 parts in Figure 1 and use "primitive part" to denote rigid body parts (i.e., torso, head, half limbs) from now on.

In this paper, we choose the 20 parts and the hierarchical structure in Figure 1 manually. Of course, it is possible to define parts corresponding to other combinations of body segments, for example, left part of the whole body. It may also be possible to learn the connectivity of parts automatically from data, for example, using structure learning methods (Koller and Friedman, 2009). We would like to leave these issues as future work.

We use a procedure similar to Yang et al. (2010) to select poselets for each part. First, we cluster the joints on each part into several clusters based on their relative x and y coordinates with respect to some reference joint of that part. For example, for the part "torso", we choose the middle-top joint as the reference and compute the relative coordinates of all the other joints on the torso with respect to this reference joint. The concatenation of all those coordinates will be the vector used for clustering. We run K-means clustering on the vectors collected from all training images and remove clusters that are too small. Similarly, we obtain the clusters for all the other parts. In the end, we obtain 5 to 20 clusters for each part. Based on the clustering, we crop the corresponding patches

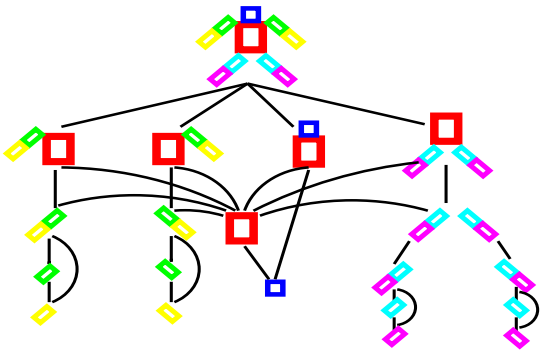


Figure 1: An illustration of the hierarchical pose representation. The black edges indicate the connectivity among different parts.

from the images and form a set of poselets for that part. Figure 2 shows examples of two different poselets for the part "legs".

Our focus is the new representation, so we use standard HOG descriptors (Dalal and Triggs, 2005) to keep the feature engineering to the minimum. For each poselet, we construct HOG features from patches in the corresponding cluster and from random negative patches. Inspired by the success of multiscale HOG features (Felzenszwalb et al., 2010), we use different cell sizes when computing HOG features for different parts. For example, we use cells of 12×12 pixel regions for poselets of the whole body, and cells of 2×2 for poselets of the upper/lower arm. This is motivated by the fact that large body parts (e.g., whole body) are typically well-represented by coarse shape information, while small body parts (e.g., half limb) are better represented by more detailed information. We then train a linear SVM classifier for detecting the presence of each poselet. The learned SVM weights can be thought as a template for the poselet. Examples of several HOG templates for the "legs" poselets are shown as the last columns of Figure 2. Examples of poselets and their corresponding HOG templates for other body parts are shown in Figure 3.

A poselet of a primitive part contains two endpoints. For example, for a poselet of upper-left leg, one endpoint corresponds to the joint between torso and upper-left leg, the other one corresponds to the joint between upper/lower left leg. We record the mean location (with respect to the center of the poselet image patch) of each endpoint. This information will be used in human parsing when we need to infer the endpoints of a primitive part for a test image.

4. Human Parsing

In this section, we describe how to use hierarchical poselets in human parsing. We first develop an undirected graphical model to represent the configuration of the human pose (Section 4.1). We then develop the inference algorithm for finding the best pose configuration in the model (Section 4.2) and the algorithm for learning model parameters (Section 4.3) from training data.

4.1 Model Formulation

We denote the complete configuration of a human pose as $L = \{l_i\}_{i=1}^{K}$, where K is the total number of parts (i.e., $K = 20$ in our case). The configuration of each part l_i is parametrized by $l_i = (x_i, y_i, z_i)$.

Figure 2: Examples of two poselets for the part "legs". Each row corresponds to a poselet. We show several patches from the poselet cluster. The last column shows the HOG template of the poselet.

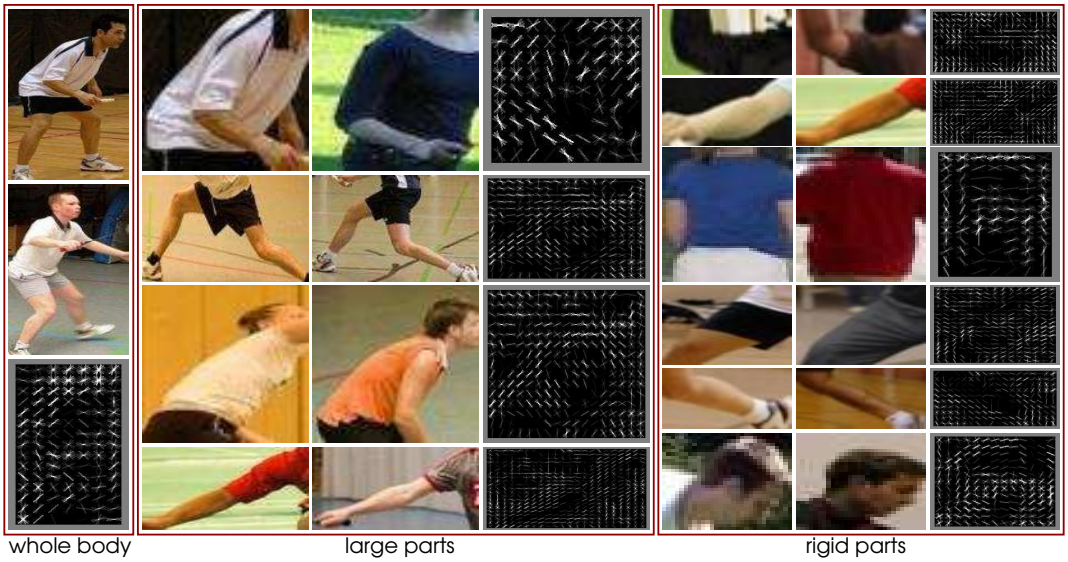


Figure 3: Visualization of some poselets learned from different body parts on the UIUC people data set, including whole body, large parts (top to bottom: torso+left arm, legs, torso+head, left arm), and rigid parts (top to bottom: upper/lower left arm, torso, upper/lower left leg, head). For each poselet, we show two image patches from the corresponding cluster and the learned SVM HOG template.

Here (x_i, y_i) defines the image location, and z_i is the index of the corresponding poselet for this part, that is, $z_i \in \{1, 2, ..., \mathcal{P}_i\}$, where $\mathcal{P}_i$ is the number of poselets for the i-th part. In this paper, we assume the scale of the person is fixed and do not search over multiple scales. It is straightforward to augment l_i with other information, for example, scale and foreshortening.

The complete pose L can be represented by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where a vertex $i \in \mathcal{V}$ denotes a part and an edge $(i, j) \in \mathcal{E}$ captures the constraint between parts i and j. The structure of $\mathcal{G}$ is shown in Figure 1. We define the score of labeling an image I with the pose L as:

$$F(L, I) = \sum_{i \in \mathcal{V}} \phi(l_i; I) + \sum_{(i,j) \in \mathcal{E}} \psi(l_i, l_j). \tag{1}$$

The details of the potential functions in Equation 1 are as follows.

Spatial prior $\psi(l_i, l_j)$: This potential function captures the compatibility of configurations of part i and part j. It is parametrized as:

$$\begin{aligned}\psi(l_i, l_j) &= \alpha_{i;j;z_i;z_j}^{\top} \text{bin}(x_i - x_j, y_i - y_j) \\ &= \sum_{a=1}^{\mathcal{P}_i} \sum_{b=1}^{\mathcal{P}_j} \mathbb{1}_a(z_i) \mathbb{1}_b(z_j) \alpha_{i;j;a;b}^{\top} \text{bin}(x_i - x_j, y_i - y_j).\end{aligned}$$

Similar to Ramanan (2006), the function $\text{bin}(\cdot)$ is a vectorized count of spatial histogram bins. We use $\mathbb{1}_a(\cdot)$ to denote the function that takes 1 if its argument equals a, and 0 otherwise. Here $\alpha_{i;j;z_i;z_j}$ is a model parameter that favors certain relative spatial bins when poselets z_i and z_j are chosen for parts i and j, respectively. Overall, this potential function models the (relative) spatial arrangement and poselet assignment of a pair (i, j) of parts.

Local appearance $\phi(l_i; I)$: This potential function captures the compatibility of placing the poselet z_i at the location (x_i, y_i) of an image I. It is parametrized as:

$$\phi(l_i; I) = \beta_{i;z_i}^{\top} f(I(l_i)) = \sum_{a=1}^{\mathcal{P}_i} \beta_{i;a}^{\top} f(I(l_i)) \cdot \mathbb{1}_a(z_i),$$

where $\beta_{i;z_i}$ is a vector of model parameters corresponding to the poselet z_i and $f(I(l_i))$ is a feature vector corresponding to the image patch defined by l_i. We define $f(I(l_i))$ as a length $\mathcal{P}_i + 1$ vector as:

$$f(I(l_i)) = [f_1(I(l_i)), f_2(I(l_i)), ..., f_{\mathcal{P}_i}(I(l_i)), 1].$$

Each element $f_r(I(l_i))$ is the score of placing poselet z_r at image location (x_i, y_i). The constant 1 appended at the end of vector allows us to learn the model with a bias term. In other words, the score of placing the poselet z_i at image location (x_i, y_i) is a linear combination (with bias term) of the responses all the poselet templates at (x_i, y_i) for part i. We have found that this feature vector works better than the one used in Yang et al. (2010), which defines $f(I(l_i))$ as a scalar of a single poselet template response. This is because the poselet templates learned for a particular part are usually not independent of each other. So it helps to combine their responses as the local appearance model.

We summarize and highlight the important properties of our model and contextualize our research by comparing with related work.

Discriminative "parts": Our model is based on a new concept of "parts" which goes beyond the traditional rigid parts. Rigid parts are inherently difficult to detect. We instead consider parts covering a wide range of portions of human bodies. We use poselets to capture distinctive appearance

patterns of various parts. These poselets have better discriminative powers than traditional rigid part detectors. For example, look at the examples in Figure 2 and Figure 3, the poselets capture various characteristic patterns for large parts, such as the "A"-shape for the legs in the first row of Figure 2.

Coarse-to-fine granularity: Different parts in our model are represented by features at varying levels of details (i.e., cell sizes in HOG descriptors). Conceptually, this multi-level granularity can be seen as providing an efficient coarse-to-fine search strategy. However, it is very different from the coarse-to-fine cascade pruning in Sapp et al. (2010b). The method in Sapp et al. (2010b) prunes the search space of small parts (e.g., right lower arm) at the coarse level using simple features and apply more sophisticated features in the pruned search space. However, we would like to argue that at the coarse level, one should not even consider small parts, since they are inherently difficult to detect or prune at this level. Instead, we should focus on large body parts since they are easy to find at the coarse level. The configurations of large pieces of human bodies will guide the search of smaller parts. For example, an upright torso with arms raising up (coarse-level information) is a very good indicator of where the arms (fine-level details) might be.

Structured hierarchical model: A final important property of our model is that we combine information across different parts in a structured hierarchical way. The original work on poselets (Bourdev and Malik, 2009; Bourdev et al., 2010) uses a simple Hough voting scheme for person detection, that is, each poselet votes for the center of the person, and the votes are combined together. This Hough voting might be appropriate for person detection, but it is not enough for human parsing which involves highly complex and structured outputs. Instead, we develop a structured model that organize information about different parts in a hierarchical fashion. Another work that uses hierarchical models for human parsing is the AND-OR graph in Zhu et al. (2008). But there are two important differences. First, the appearance models used in Zhu et al. (2008) are only defined on sub-parts of body segments. Their hierarchical model is only used to put all the small pieces together. As mentioned earlier, appearance models based on body segments are inherently unreliable. In contrast, we use appearance models associated with parts of varying sizes. Second, the OR-nodes in Zhu et al. (2008) are conceptually similar to poselets in our case. But the OR-nodes in Zhu et al. (2008) are defined manually, while our poselets are learned.

Our work on human parsing can be seen as bridging the gap between two popular schools of approaches for human parsing: part-based methods, and exemplar-based methods. Part-based methods, as explained above, model the human body as a collection of rigid parts. They use local part appearances to search for those parts in an image, and use configuration priors to put these pieces together in some plausible way. But since the configuration priors in these methods are typically defined as pairwise constraints between parts, these methods usually lack any notion that captures what a person should look like as a whole. In contrast, exemplar-based methods (Mori and Malik, 2002; Shakhnarovich et al., 2003; Sullivan and Carlsson, 2002) search for images with similar whole body configurations, and transfer the poses of those well-matched training images to a new image. The limitation of exemplar-based approaches is that they require good matching of the entire body. They cannot handle test images of which the legs are similar to some training images, while the arms are similar to other training images. Our work combines the benefits of both schools. On one hand, we capture the large-scale information of human pose via large parts. On the other hand, we have the flexibility to compose new poses from different parts.

4.2 Inference

Given an image I, the inference problem is to find the optimal pose labeling L^* that maximize the score $F(L,I)$, that is, $L^* = \arg\max_L F(L,I)$. We use the max-product version of belief propagation to solve this problem. We pick the vertex corresponding to part "whole body" as the root and pass messages upwards towards this root. The message from part i to its parent j is computed as:

$$m_i(l_j) = \max_{l_i}(u(l_j) + \psi(l_i, l_j)), \tag{2}$$
$$u(l_j) = \phi(l_j) + \sum_{k \in \text{kids}_j} m_k(l_j).$$

Afterwards, we pass messages downward from the root to other vertices in a similar fashion. This message passing scheme is repeated several times until it converges. If we temporarily ignore the poselet indices z_i and z_j and think of $l_i = (x_i, y_i)$, we can represent the messages as 2D images and pass messages using techniques similar to those in Ramanan (2006). The image $u(l_j)$ is obtained by summing together response images from its child parts $m_k(l_j)$ and its local response image $\phi(l_j)$. $\phi(l_j)$ can be computed in linear time by convolving the HOG feature map with the template of z_j. The maximization in Equation 2 can also be calculated in time linear to the size of $u(l_j)$. In practice, we compute messages on each fixed (z_i, z_j) and enumerate all the possible assignments of (z_i, z_j) to obtain the final message. Note that since the graph structure is not a tree, this message passing scheme does not guarantee to find the globally optimal solution. But empirically, we have found this approximate inference scheme to be sufficient for our application.

The inference gives us the image locations and poselet indices of all the 20 parts (both primitive and non-primitive). To obtain the final parsing result, we need to compute the locations of the two endpoints for each primitive part. These can be obtained from the mean endpoint locations recorded for each primitive part poselet (see Sec. 3).

Figure 4 shows a graphical illustration of applying our model on a test image. For each part in the hierarchy, we show two sample patches and the SVM HOG template corresponding to the poselet chosen for that part.

4.3 Learning

In order to describe the learning algorithm, we first write Equation 1 as a linear function of a single parameter vector w which is a concatenation of all the model parameters, that is:

$$F(L,I) = w^\top \Phi(I,L), \quad \text{where}$$
$$w = [\alpha_{i;j;a;b}; \beta_{i;a}], \ \ \forall i, j, a, b,$$
$$\Phi(I,L) = [\mathbb{1}_a(z_i)\mathbb{1}_b(z_j)\text{bin}(x_i - x_j, y_i - y_j); f(I(l_i))\mathbb{1}_a(z_i)], \ \ \forall i, j, a, b.$$

The inference scheme in Section 4.2 solves $L^* = \arg\max_L w^\top \Phi(I,L)$. Given a set of training images in the form of $\{I^n, L^n\}_{n=1}^N$, we learn the model parameters w using a form of structural SVM (Tsochantaridis et al., 2005) as follows:

$$\min_{w,\xi} \ \frac{1}{2}||w||^2 + C\sum_n \xi^n, \quad \text{s.t. } \forall n, \ \forall L: \tag{3}$$
$$w^\top \Phi(I^n, L^n) - w^\top \Phi(I^n, L) \geq \Delta(L, L^n) - \xi^n. \tag{4}$$

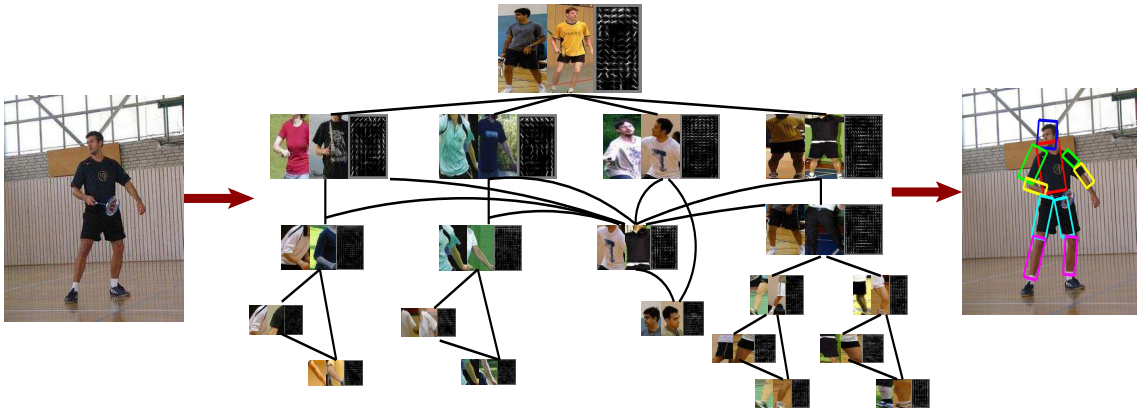


Figure 4: A graphical illustration of applying our model on a test image. For each part (please refer to Figure 1), we show the inferred poselet by visualizing two sample patches from the corresponding poselet cluster and the SVM HOG template.

Consider a training image I^n, the constraint in Equation 4 enforces the score of the true label L^n to be larger than the score of any other hypothesis label L by some margin. The loss function $\Delta(L, L^n)$ measures how incorrect L is compared with L^n. Similar to regular SVMs, ξ_n are slack variables used to handle soft margins. This formulation is often called margin-rescaling in the SVM-struct literature (Tsochantaridis et al., 2005).

We use a loss function that decomposes into a sum of local losses defined on each part $\Delta(L, L^n) = \sum_{i=1}^{K} \Delta_i(L_i, L_i^n)$. If the i-th part is a primitive part, we define the local loss $\Delta_i(L_i, L_i^n)$ as:

$$\Delta_i(L_i, L_i^n) = \lambda \cdot \mathbb{1}(z_i \neq z_i^n) + d((x_i, y_i), (x_i^n, y_i^n)), \tag{5}$$

where $\mathbb{1}(\cdot)$ is an indicator function that takes 1 if its argument is true, and 0 otherwise. The intuition of Equation 5 is as follows. If the hypothesized poselet z_i is the same as the ground-truth poselet z_i^n for the i-th part, the first term of Equation 5 will be zero. Otherwise it will incur a loss λ (we choose $\lambda = 10$ in our experiments). The second term in Equation 5, $d((x_i, y_i), (x_i^n, y_i^n))$, measures the distance (we use l_1 distance) between two image locations (x_i, y_i) and (x_i^n, y_i^n). If the hypothesized image location (x_i, y_i) is the same as the ground-truth image location (x_i^n, y_i^n) for the i-th part, no loss is added. Otherwise a loss proportional to the l_1 distance of these two locations will be incurred.

If the i-th part is not a primitive part, we simply set $\Delta(L_i, L_i^n)$ to be zero. This choice is based on the following observation. In our framework, non-primitive parts only serve as some intermediate representations that help us to search for and disambiguate small primitive parts. The final human parsing results are still obtained from configurations l_i of primitive parts. Even if a particular hypothesized L gets one of its non-primitive part labeling wrong, it should not be penalized as long as the labelings of primitive parts are correct.

The optimization problem in Equations (3,4) is convex and can be solved using the cutting plane method implemented in the SVM-struct package (Joachims et al., 2008). However we opt to use a simpler stochastic subgradient descent method to allow greater flexibility in terms of implementation.

Figure 5: Human actions in static images. We show some sample images and their annotations on the two data sets used in our experiments (see Section 6). Each image is annotated with the action category and joints on the human body. It is clear from these examples that static images convey a lot of information about human actions.

First, it is easy to show that Equations (3,4) can be equivalently written as:

$$\min_w \frac{1}{2}||w||^2 + C\sum_n \mathcal{R}^n(L),$$
$$\text{where} \quad \mathcal{R}^n(L) = \max_L \Big(\Delta(L,L^n) + w^\top \Phi(I^n,L) - w^\top \Phi(I^n,L^n)\Big).$$

In order to do gradient descent, we need to calculate the subgradient $\partial_w \mathcal{R}^n(L)$ at a particular w. Let us define:

$$L^\star = \arg\max_L \Big(\Delta(L,L^n) + w^\top \Phi(I^n,L)\Big). \tag{6}$$

Equation 6 is called loss-augmented inference (Joachims et al., 2008). It can be shown that the subgradient $\partial_w \mathcal{R}^n(L)$ can be computed as $\partial_w \mathcal{R}(L) = \Phi(I^n,L^\star) - \Phi(I^n,L^n)$. Since the loss function $\Delta(L,L^n)$ can be decomposed into a sum over local losses on each individual part, the loss-augmented inference in Equation 6 can be solved in a similar way to the inference problem in Section 4.2. The only difference is that the local appearance model $\phi(l_i;I)$ needs to be augmented with the local loss function $\Delta(L_i,L_i^n)$. Interested readers are referred to Joachims et al. (2008) for more details.

5. Action Recognition

The hierarchical poselet is a representation general enough to be used in many applications. In this section, we demonstrate it in human action recognition from static images.

Look at the images depicted in Figure 5. We can easily perceive the actions of people in those images, even though only static images are given. So far most work in human action recognition has been focusing on recognition from videos. While videos certainly provide useful cues (e.g., motion) for action recognition, the examples in Figure 5 clearly show that the information conveyed by static images is also an important component of action recognition. In this paper, we consider the

problem of inferring human actions from static images. In particular, we are interested in exploiting the human pose as a source of information for action recognition.

Several approaches have been proposed to address the problem of static image action recognition in the literature. The first is a standard pattern classification approach, that is, learning a classifier based on certain image feature representations. For example, Ikizler-Cinbis et al. (2009) learn SVM classifiers based on HOG descriptors. The limitation with this approach is that it completely ignores the pose of a person. Another limitation is that SVM classifiers implicitly assume that images from the same action category can be represented by a canonical prototype (which are captured by the weights of the SVM classifier). However, the examples in Figure 5 clearly show that humans can have very varied appearances when performing the same action, which are hard to characterize with a canonical prototype.

Another approach to static image action recognition is to explicitly recover the human pose, then use the pose as a feature representation for action recognition. For example, Ferrari et al. (2009) estimate the 2D human pose in TV shots. The estimated 2D poses can be used to extract features which in turn can be used to retrieve TV shots containing people with similar poses to a query. As point out in Yang et al. (2010), the problem with this approach is that 2D human pose estimation is still a very challenging problem. The output of the state-of-the-art pose estimation system is typically not reliable enough to be directly used for action recognition.

The work in Yang et al. (2010) is the closest to ours. It uses a representation based on human pose for action recognition. But instead of explicitly recovering the precise pose configuration, it represents the human pose as a set of latent variables in the model. Their method does not require the predicted human pose to be exactly correct. Instead, it learns which components of the pose are useful for differentiating various actions.

The pose representation in Yang et al. (2010) is limited to four parts: upper body, left/right arm, and legs. Learning and inference in their model amounts to infer the best configurations of these four parts for a particular action. A limitation of this representation is that it does not contain pose information about larger (e.g., whole body) or smaller (e.g., half-limbs) parts. We believe that pose information useful for discerning actions can vary depending on different action categories. Some actions (e.g., *running*) have distinctive pose characteristics in terms of both the upper and lower bodies, while other actions (e.g., *pointing*) are characterized by only one arm. The challenge is how to represent the pose information at various levels of details for action recognition.

In this section, we use hierarchical poselets to capture richer pose information for action recognition. While a richer pose representation may offer more pose information (less bias), it must also be harder to estimate accurately (more variance). In this paper, we demonstrate that our rich pose representation (even with higher variance) is useful for action recognition.

5.1 Action-Specific Hierarchical Poselets

Since our goal is action recognition, we choose to use an action-specific variant of the hierarchical poselets. This is similar to the action-specific poselets used in Yang et al. (2010). The difference is that the action-specific poselets in Yang et al. (2010) are only defined in terms of four parts—left/right arms, upper-body, and legs. These four parts are organized in a star-like graphical model. In contrast, our pose representation captures a much wider range of information across various pieces of the human body. So ours is a much richer representation than Yang et al. (2010).

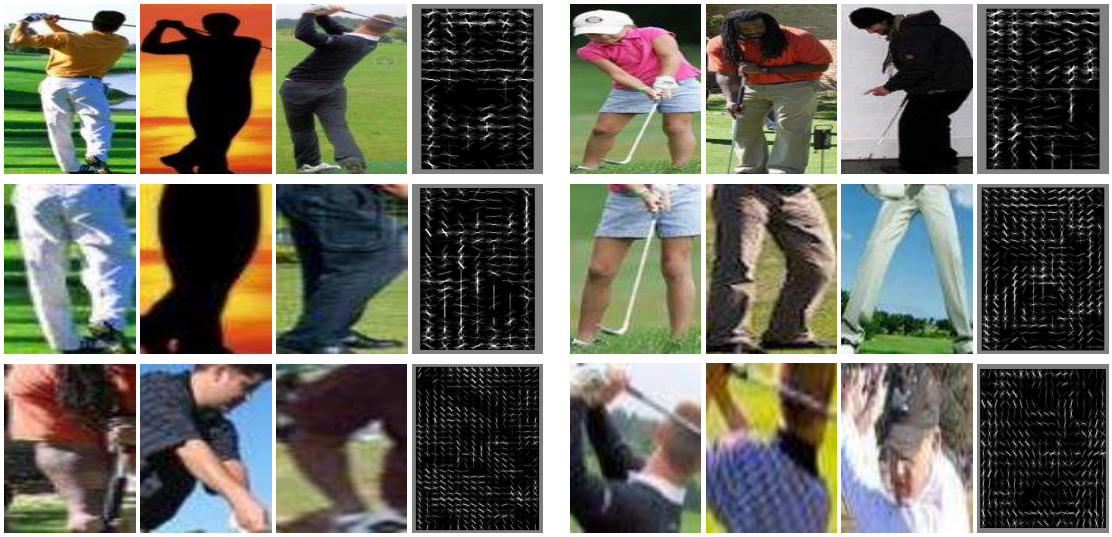


Figure 6: Examples of poselets for "playing golf". For each poselet, we visualize several patches from the corresponding cluster and the SVM HOG template. Notice the multi-scale nature of the poselets. These poselets cover various portions of the human bodies, including the whole body (1st row), both legs (2nd row), one arm (3nd row), respectively.

The training images are labeled with ground-truth action categories and joints on the human body (Figure 5). We use the following procedure to select poselets for a specific part (e.g., *legs*) of a particular action category (e.g., *running*). We first collect training images of that action category (*running*). Then we cluster the joints on the part (*legs*) into several clusters based on their relative (x,y) coordinates with respect to some reference joint. Each cluster will correspond to a "running legs" poselet. We repeat this process for the part in other action categories. In the end, we obtain about 15 to 30 clusters for each part. Figures 6 and 7 show examples of poselets for "playing golf" and "running" actions, respectively.

Similarly, we train a classifier based on HOG features (Dalal and Triggs, 2005) to detect the presence of each poselet. Image patches in the corresponding poselet cluster are used as positive examples and random patches as negative examples for training the classifier. Similar to the model in Sec. 4, we use different cell sizes when constructing HOG features for different parts. Large cell sizes are used for poselets of large body parts (e.g., whole body and torso), while small cell sizes are used for small body parts (e.g., half limbs). Figure 6 and Figure 7 show some examples of the learned SVM weights for some poselets.

5.2 Our Model

Let I be an image containing a person, $Y \in \mathcal{Y}$ be its action label where $\mathcal{Y}$ is the action label alphabet, L be the pose configuration of the person. The complete pose configuration is denoted as $L = \{l_i\}_{i=1}^{K}$ ($K = 20$ in our case), where $l_i = (x_i, y_i, z_i)$ represents the 2D image location and the index of the corresponding poselet cluster for the i-th part. The complete pose L can be represented by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ shown in Figure 1. A vertex $i \in \mathcal{V}$ denotes the i-th part and an edge $(i, j) \in \mathcal{E}$ represents

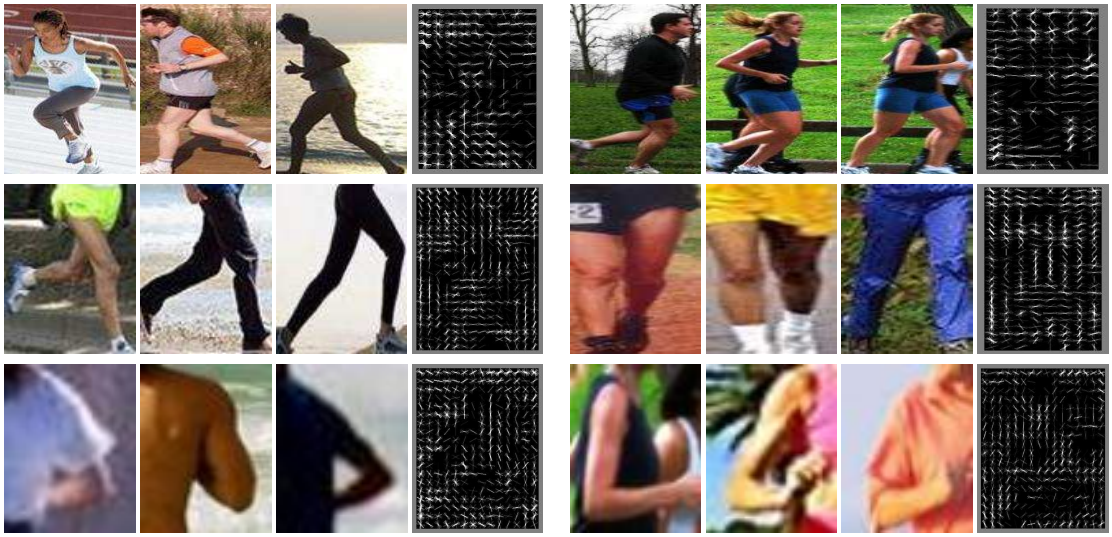


Figure 7: Examples of poselets for "running". For each poselet, we visualize several patches from the corresponding cluster and the SVM HOG template. Similar to Figure 6, these poselets cover various portions of the human bodies

the spatial constraint between the i-th and the j-th parts. We define the following scoring function to measure the compatibility of the triple (I,L,Y):

$$F(I,L,Y) = \omega_Y(I) + \sum_{i \in \mathcal{V}} \phi_Y(I,l_i) + \sum_{i,j \in \mathcal{E}} \psi_Y(l_i,l_j). \tag{7}$$

Here we use the subscript to explicitly emphasize that these functions are specific for a particular action label Y. The details of the potential functions in Equation 7 are as follows.

Root appearance $\omega_Y(I)$*:* This potential function models the compatibility of the action label Y and the global appearance of an image I. It is parametrized as:

$$\omega_Y(I) = \alpha_Y^\top \cdot f(I). \tag{8}$$

Here $f(I)$ is a feature vector extracted from the whole image I without considering the pose. In this paper, we use the HOG descriptor (Dalal and Triggs, 2005) of I as the feature vector $f(I)$. The parameters α_Y can be interpreted as a HOG template for the action category Y. Note that if we only consider this potential function, the parameters $\{\alpha_Y\}_{Y \in \mathcal{Y}}$ can be obtained from the weights of a multi-class linear SVM trained with HOG descriptors $f(I)$ alone without considering the pose information.

Part appearance $\phi_Y(I,l_i)$*:* This potential function models the compatibility of the configuration l_i of the i-th part and the local image patch defined by $l_i = (x_i, y_i, z_i)$, under the assumption that the action label is Y. Since our goal is action recognition, we also enforce that the poselet z_i should comes from the action Y. In other words, if we define $\mathcal{Z}_i^Y$ as the set of poselet indices for the i-th part corresponding to the action category Y, this potential function is parametrized as:

$$\phi_Y(I,l_i) = \begin{cases} \beta_{i,Y}^\top \cdot f(I,l_i) & \text{if } z_i \in \mathcal{Z}_i^Y; \\ -\infty & \text{otherwise.} \end{cases} \tag{9}$$

Here $f(I,l_i)$ is the score of placing the SVM HOG template z_i at location (x_i,y_i) in the image I.

Pairwise part constraint $\psi(l_i,l_j)$*:* This potential function models the compatibility of the configurations between the i-th and the j-th parts, under the assumption that the action label is Y. We parametrize this potential function using a vectorized counts of spatial histogram bins, similar to Ramanan (2006); Yang et al. (2010). Again, we enforce poselets z_i and z_j to come from action Y as follows:

$$\psi_Y(l_i,l_j) = \begin{cases} \gamma_{i,Y}^\top \cdot \text{bin}(l_i - l_j) & \text{if } z_i \in \mathcal{Z}_i^Y, z_j \in \mathcal{Z}_j^Y; \\ -\infty & \text{otherwise.} \end{cases} \tag{10}$$

Here $\text{bin}(\cdot)$ is a vector all zeros with a single one for the occupied bin.

Note that if the potential functions and model parameters in Equations(7,8,9,10) do not depend on the action label Y, the part appearance $\phi(\cdot)$ and pairwise part constraint $\psi(\cdot)$ exactly recover the human parsing model in Section 4.

5.3 Learning and Inference

We define the score of labeling an image I with the action label Y as follows:

$$H(I,Y) = \max_L F(I,L,Y). \tag{11}$$

Given the model parameters $\Theta = \{\alpha,\beta,\gamma\}$, Equation 11 is a standard MAP inference problem in undirected graphical models. We can approximately solve it using message passing scheme similar to that in Section 4.2. The predicted action label Y^* is chosen as $Y^* = \arg\max_Y H(I,Y)$.

We adopt the latent SVM (Felzenszwalb et al., 2010) framework for learning the model parameters. First, it is easy to see that Equation 7 can be written as a linear function of model parameters as $F(I,L,Y) = \Theta^\top \Phi(I,L,Y)$, where Θ is the concatenation of all the model parameters (i.e., α, β and γ) and $\Phi(I,L,Y)$ is the concatenation of the corresponding feature vectors. Given a set of training examples in the form of $\{I^n,L^n,Y^n\}_{n=1}^N$, the model parameters are learned by solving the following optimization problem:

$$\min_{\Theta,\xi} \ \frac{1}{2}||\Theta||^2 + C\sum_n \xi^n, \quad \text{s.t. } \forall n, \ \forall Y: \tag{12}$$

$$H(I^n,Y^n) - H(I^n,Y) \geq \Delta(Y,Y^n) - \xi^n. \tag{13}$$

It is easy to show that Equations (12,13) can be equivalently written as:

$$\min_\Theta \frac{1}{2}||\Theta||^2 + C\sum_n \mathcal{R}^n, \tag{14}$$

$$\text{where } \ \mathcal{R}^n = \max_{Y,L}\Big(\Delta(Y,Y^n) + \Theta^\top \cdot \Phi(I^n,Y)\Big) - \max_L \Theta^\top \cdot \Phi(I^n,L,Y^n).$$

The problem in Equation 14 is not convex, but we can use simple stochastic sub-gradient descent to find a local optimum. Let us define:

$$(Y^*,L^*) = \arg\max_{Y,L}(\Delta(Y,Y^n) + \Theta^\top \cdot \Phi(I^n,L,Y)),$$

$$L' = \arg\max_L(\Theta^\top \cdot \Phi(I^n,L,Y^n)).$$

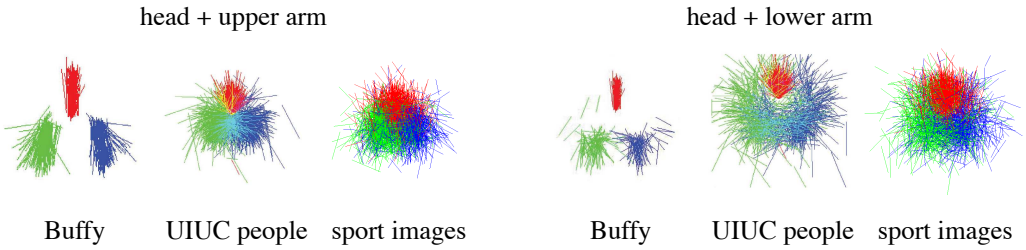


Figure 8: Scatter plots of heads (red) and upper/lower arms (blue and green) with respect to fixed upper body position on three data sets.

Then the gradient of Equation 14 can be computed as:

$$\Theta + C \sum_n \Big(\Phi(I^n, L^*, Y^*) - \Phi(I^n, L', Y^n) \Big).$$

To initialize the parameter learning, we first learn a pose estimation model using the labeled (I^n, L^n) collected from training examples with class label Y. The parameters of these pose estimation models are used to initialize β_Y and γ_Y. The parameters α_Y are initialized from a linear SVM model based on HOG descriptors without considering the poses.

6. Experiments

In this section, we present our experimental results on human parsing (Section 6.1) and action recognition (Section 6.2).

6.1 Experiments on Human Parsing

There are several data sets popular in the human parsing community, for example, Buffy data set (Ferrari et al., 2008), PASCAL stickmen data set (Eichner and Ferrari, 2009). But these data sets are not suitable for us for several reasons. First of all, they only contain upper-bodies, but we are interested in full-body parsing. Second, as pointed out in Tran and Forsyth (2010), there are very few pose variations in those data sets. In fact, previous work has exploited this property of these data sets by pruning search spaces using upper-body detection and segmentation (Ferrari et al., 2008), or by building appearance model using location priors (Eichner and Ferrari, 2009). Third, the contrast of image frames of the Buffy data set is relatively low. This issue suggests that better performance can be achieved by engineering detectors to overcome the contrast difficulties. Please refer to the discussion in Tran and Forsyth (2010) for more details. In our work, we choose to use two data sets[1] containing very aggressive pose variations. The first one is the UIUC people data set introduced in Tran and Forsyth (2010). The second one is a new sport image data set we have collected from the Internet which has been used in Wang et al. (2011). Figure 8 shows scatter plots of different body parts of our data sets compared with the Buffy data set (Ferrari et al., 2008) using a visualization style similar to Tran and Forsyth (2010) . It is clear that the two data sets used in this paper have much more variations.

1. Both data sets can be downloaded from `http://vision.cs.uiuc.edu/humanparse`.

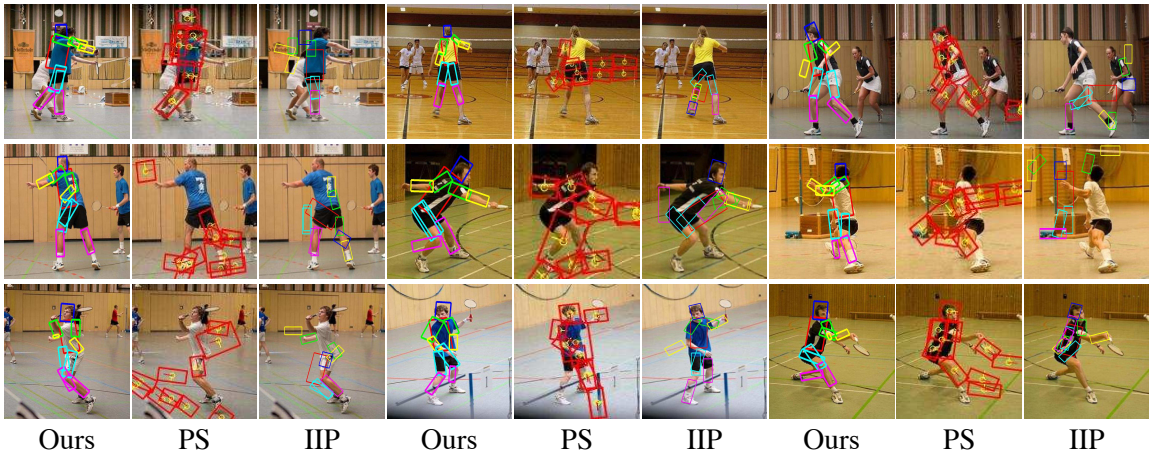


Figure 9: Examples of human body parsing on the UIUC people data set. We compare our method with the pictorial structure (PS) (Andriluka et al., 2009) and the iterative image parsing (IIP) (Ramanan, 2006). Notice the large pose variations, cluttered background, self-occlusions, and many other challenging aspects of the data set.

6.1.1 UIUC PEOPLE DATA SET

The UIUC people data set (Tran and Forsyth, 2010) contains 593 images (346 for training, 247 for testing). Most of them are images of people playing badminton. Some are images of people playing Frisbee, walking, jogging or standing. Sample images and their parsing results are shown in the first three rows of Figure 9. We compare with two other state-of-the-art approaches that do full-body parsing (with published codes): the improved pictorial structure by Andriluka et al. (2009), and the iterative parsing method by Ramanan (2006). The results are also shown in Figure 9.

To quantitatively evaluate different methods, we measure the percentage of correctly localized body parts. Following the convention proposed in Ferrari et al. (2008), a body part is considered correctly localized if the endpoints of its segment lies within 50% of the ground-truth segment length from their true locations. The comparative results are shown in Table 1(a). Our method outperforms other approaches in localizing most of body parts. We also show the result (3rd row, Table 1(a)) of using only the basic-level poselets corresponding to the rigid parts. It is clear that our full model using hierarchical poselets outperforms using rigid parts alone.

Detection and parsing: An interesting aspect of our approach is that it produces not only the configurations of primitive parts, but also the configurations of other larger body parts. These pieces of information can potentially be used for applications (e.g., gesture-based HCI) that do not require precise localizations of body segments. In Figure 10, we visualize the configurations of four larger parts on some examples. Interestingly, the configuration of the whole body directly gives us a person detector. So our model can be seen as a principled way of unifying human pose estimation, person detection, and many other areas related to understanding humans. In the first row of Table 2, we show the results of person detection on the UIUC people data set by running our human parsing model, then picking the bounding box corresponding to the part "whole body" as the detection. We compare with the state-of-the-art person detectors in Felzenszwalb et al. (2010) and Andriluka et al.

Method	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head
Ramanan (2006)	44.1	11.7	7.3	25.5	25.1	11.3	10.9	**25.9**	**25**	30.8
Andriluka et al. (2009)	70.9	37.3	35.6	23.1	22.7	22.3	30.0	9.7	10.5	59.1
Our method (basic-level)	79.4	53.8	53.4	47.8	39.7	17.8	21.1	11.7	16.6	65.2
Our method (full model)	**86.6**	**58.3**	**54.3**	**53.8**	**46.6**	**28.3**	**33.2**	23.1	17.4	**68.8**

(a) UIUC people data set

Method	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head
Ramanan (2006)	28.7	7.4	7.2	17.6	20.8	8.3	6.6	**20.2**	**21**	12.9
Andriluka et al. (2009)	71.5	44.2	43.1	30.7	31	**28**	**29.6**	17.3	15.3	**63.3**
Our method (basic-level)	73.3	45.0	47.6	40.4	39.9	19.4	27.0	13.3	9.9	47.5
Our method (full model)	**75.3**	**50.1**	**48.2**	**42.5**	**36.5**	23.3	27.1	12.2	10.2	47.5

(b) Sport image data set

Table 1: Human parsing results by our method and two comparison methods (Ramanan, 2006; Andriluka et al., 2009) on two data sets. The percentage of correctly localized parts is shown for each primitive part. If two numbers are shown in one cell, they indicate the left/right body parts. As a comparison, we also show the results of using only rigid parts (basic-level).

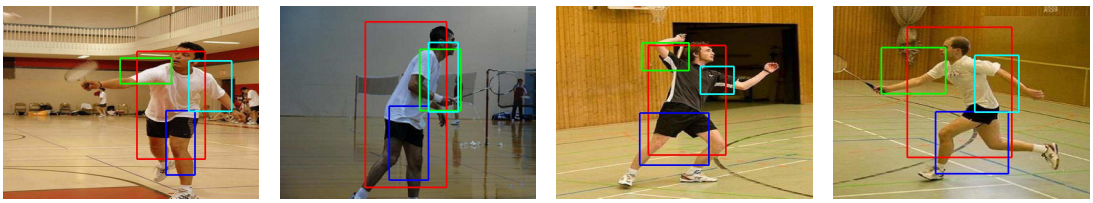


Figure 10: Examples of other information produced by our model. On each image, we show bounding boxes corresponding to the whole body, left arm, right arm and legs. The size of each bounding box is estimated from its corresponding poselet cluster.

(2009). Since most images contain one person, we only consider the detection with the best score on an image for all the methods. We use the metric defined in the PASCAL VOC challenge to measure the performance. A detection is considered correct if the intersection over union with respect to the ground truth bounding box is at least 50%. It is interesting to see that our method outperforms other approaches, even though it is not designed for person detection.

	Our method	Felzenszwalb et al. (2010)	Andriluka et al. (2009)
UIUC people	**66.8**	48.58	50.61
Sport image	**63.94**	45.61	59.94

Table 2: Comparison of accuracies of person detection on both data sets. In our method, the configuration of the poselets corresponding to the whole body can be directly used for person detection.

Figure 11: Examples of human body parsing on the sport image data set. We compare our method with the pictorial structure (PS) (Andriluka et al., 2009) and the iterative image parsing (IIP) (Ramanan, 2006).

6.1.2 SPORT IMAGE DATA SET

The UIUC people data set is attractive because it has very aggressive pose and spatial variations. But one limitation of that data set is that it mainly contains images of people playing badminton. One might ask what happens if the images are more diverse. To answer this question, we have collected a new sport image data set from more than 20 sport categories, including acrobatics, American football, croquet, cycling, hockey, figure skating, soccer, golf and horseback riding. There are in total 1299 images. We randomly choose 649 of them for training and the rest for testing. The last three rows of Figure 9 show examples of human parsing results, together with results of Andriluka et al. (2009) and Ramanan (2006) on this data set. The quantitative comparison is shown in Table 1(b). We can see that our approach outperforms the other two on the majority of body parts.

Similarly, we perform person detection using the poselet corresponding to the whole body. The results are shown in the second row of Table 2. Again, our method outperforms other approaches.

6.1.3 KINEMATIC TRACKING

To further illustrate our method, we apply the model learned from the UIUC people data set for kinematic tracking by independently parsing the human figure in each frame. In Figure 12, we show our results compared with applying the method in Ramanan (2006). It is clear from the results that kinematic tracking is still a very challenging problem. Both methods make mistakes. Interestingly, when our method makes mistakes (e.g., figures with blue arrows), the output still looks like a valid body configuration. But when the method in Ramanan (2006) makes mistakes (e.g., figures with red arrows), the errors can be very wild. We believe this can be explained by the very different representations used in these two methods. In Ramanan (2006), a human body is represented by the set of primitive parts. Kinematic constraints are used to enforce the connectivity of those parts. But these kinematic constraints have no idea what a person looks like as a whole. In the incorrect

Figure 12: Examples of kinematic tracking on the baseball and figure skating data sets. The 1st and 3rd rows are our results. The 2rd and 4th rows are results of Ramanan (2006). Notice how mistakes of our method (blue arrows) still look like valid human poses, while those of Ramanan (2006) (red arrows) can be wild.

results of Ramanan (2006), all the primitive parts are perfectly connected. The problem is their connectivity does not form a reasonable human pose as a whole.

In contrast, our model uses representations that capture a spectrum of both large and small body parts. Even in situations where the small primitive parts are hard to detect, our method can still reason about the plausible pose configuration by pulling information from large pieces of the human bodies.

6.2 Experiments on Action Recognition

We test our approach on two publicly available data sets: the still images data set (Ikizler et al., 2008) and the Leeds sport data set (Johnson and Everingham, 2010). Both data sets contain images of people with ground-truth pose annotations and action labels.

6.2.1 STILL IMAGE DATA SET

We first demonstrate our model on the still image data set collected in Ikizler et al. (2008). This data set contains more than 2000 static images from five action categories: dancing, playing golf, running, sitting, and walking. Sample images are shown in the first two rows of Figure 5. Yang et al. (2010) have annotated the pose with 14 joints on the human body on all the images in the data set. Following Yang et al. (2010), we choose $1/3$ of the images from each category to form the training data, and the remaining ones as the test data.[2]

2. A small number of images/annotations we obtained from the authors of Yang et al. (2010) are somehow corrupted due to some file-system failure. We have removed those images from the data set.

method	overall	avg per-class
Our approach	**65.15**	**70.77**
Yang et al. (2010)*	63.49	68.37
SVM mixtures	62.8	64.05
Linear SVM	60.32	61.5

Table 3: Performance on the still image data set. We report both overall and average per-class accuracies. *The results are based on our own implementation.

Figure 13: Visualization of some inferred poselets on the still image data set. These test images have been correctly recognized by our model. For a test image, we show three poselets that have high responses. Each poselet is visualized by showing several patches from its cluster.

We compare our approach with two baseline method. The first baseline is a multi-class SVM based on HOG features. For the second baseline, we use mixtures of SVM models similar to that in Felzenszwalb et al. (2010). We set the number of mixtures for each class to be the number of whole-body poselets. From Table 3, we can see that our approach outperforms the baseline by a large margin. Our performance is also better than the reported results in Yang et al. (2010). However, the accuracy numbers are not directly comparable since the training/testing data sets and features are not completely identical. In order to do a fair comparison, we re-implemented the method in Yang et al. (2010) by only keeping the parts used in Yang et al. (2010). Our full model performs better.

In Figure 13, we visualize several inferred poselets on some examples whose action categories are correctly classified. Each poselet is visualized by showing several patches from the corresponding poselet cluster.

Figure 14: Visualization of some inferred poselets on the Leeds sport data set. These test images have been correctly recognized by our model. For a test image, we show three poselets that have high responses. Each poselet is visualized by showing several patches from its cluster.

method	overall	avg per-class
Our approach	**54.6**	**54.6**
SVM mixtures	52.7	49.13
Linear SVM	52.7	52.93

Table 4: Performance on the Leeds sport data set. We report both overall and average per-class accuracies.

6.2.2 LEEDS SPORT DATA SET

The Leeds sport data set (Johnson and Everingham, 2010) contains 2000 images from eight different sports: athletics, badminton, baseball, gymnastics, parkour, soccer, tennis, volleyball. Each image in the data set is labeled with 14 joints on the human body. Sample images and the labeled joints are shown in the last four rows of Figure 5. This data set is very challenging due to very aggressive pose variations.

We choose half of the images for training, and the other half for testing. The performance is shown in Table 4. Again, we compare with the HOG-based SVM and SVM mixtures as the baselines. We can see that our method still outperforms the baseline. Similarly, we visualize the inferred poselets on some examples in Figure 14.

American football→dancing croquet→playing golf field hockey→running

Figure 15: Visualization of inferred poses on unseen actions. Here the actions of the test images (*American football*, *croquet* and *field hockey*) are not available during training. Our model recognizes these examples as *dancing*, *playing golf*, *running*, respectively. Some of the results (e.g., *croquet→ golfing*) make intuitive sense. Others (e.g., *football→dancing*) might not be intuitive at first. But if we examine the poselets carefully, we can see that various pieces of the football player are very similar to those found in the dancing action.

6.2.3 UNSEEN ACTIONS

An interesting aspect of our model is that it outputs not only the predicted action label, but also some rich intermediate representation (i.e., action-specific hierarchical poselets) about the human pose. This information can potentially be exploited in various contexts. As an example, we apply the model learned from the still image data set to *describe* images from sports categories not available during training. In Figure 15, we show examples of applying the model learned from the still image data set to images with unseen action categories. The action categories (*American football*, *croquet* and *field hockey*) for the examples in Figure 15 are disjoint from the action categories of the still image data set. In this situation, our model obviously cannot correctly predict the action labels (since they are not available during training). Instead, it classifies those images using the action labels it has learned. For example, it classifies "American football" as "dancing", "croquet" as "playing golf", "field hockey" as "running". More importantly, our model outputs poselets for various parts which support its prediction. From these information, we can say a lot about "American football" even though the predicted action label is wrong. For example, we can say it is closer to "dancing" than "playing golf" because the pose of the football player in the image is similar to certain type of dancing legs, and certain type of dancing arms.

7. Conclusion and Future Work

We have presented hierarchical poselets, a new representation for modeling human poses. Different poselets in our representation capture human poses at various levels of granularity. Some poselets correspond to the rigid parts typically used in previous work. Others can correspond to large pieces of the human bodies. Poselets corresponding to different parts are organized in a structured hierarchical model. The advantage of this representation is that it infers the human pose by pulling information across various levels of details, ranging from the coarse shape of the whole body, to the fine-detailed information of small rigid parts. We have demonstrate the applications of this rep-

resentation in human parsing and human action recognition from static images. Recently, similar ideas (Sun and Savarese, 2011) have been applied in other applications, such as object detection.

As future work, we would like to explore how to automatically construct the parts and the hierarchy using data-driven methods. This will be important in order to extend hierarchical poselets to other objects (e.g., birds) that do not have obvious kinematic structures. We also like to apply the hierarchical poselet representation to other vision tasks, such as segmentation.

Acknowledgments

This work was supported in part by NSF under IIS-0803603 and IIS-1029035, and by ONR under N00014-01-1-0890 and N00014-10-1-0934 as part of the MURI program. Yang Wang was also supported in part by an NSERC postdoc fellowship when the work was done. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of NSF, ONR, or NSERC.

References

Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.

Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors training using 3d human pose annotations. In *IEEE International Conference on Computer Vision*, 2009.

Lubomir Bourdev, Subhransu Maji, Thomas Brox, and Jitendra Malik. Detecting people using mutually consistent poselet activations. In *European Conference on Computer Vision*, 2010.

Navneet Dalal and Bill Triggs. Histogram of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

Vincent Delaitre, Ivan Laptev, and Josef Sivic. Recognizing human actions in still images: a study of bag-of-features and part-based representations. In *British Machine Vision Conference*, 2010.

Chaitanya Desai, Deva Ramanan, and Charless Fowlkes. Discriminative models for static human-object interactions. In *Workshop on Structured Models in Computer Vision*, 2010.

Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *ICCV'05 Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005.

Alexei A. Efros, Alexander C. Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *IEEE International Conference on Computer Vision*, pages 726–733, 2003.

Marcin Eichner and Vittorio Ferrari. Better appearance models for pictorial structures. In *British Machine Vision Conference*, 2009.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, January 2005.

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1672–1645, 2010.

Vittorio Ferrari, Manuel Marín-Jiménez, and Andrew Zisserman. Progressive search space reduction for human pose estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

Vittorio Ferrari, Manuel Marín-Jiménez, and Andrew Zisserman. Pose search: retrieving people using their pose. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.

David A. Forsyth, Okan Arikan, Leslie Ikemoto, James O'Brien, and Deva Ramanan. Computational studies of human motion: Part 1, tracking and motion synthesis. *Foundations and Trends in Computer Graphics and Vision*, 1(2/3):77–254, July 2006.

Abhinav Gupta, Aniruddha Kembhavi, and Larry S. Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1775–1789, 2009.

Nazli Ikizler, R. Gokberk Cinbis, Selen Pehlivan, and Pinar Duygulu. Recognizing actions from still images. In *International Conference on Pattern Recognition*, 2008.

Nazli Ikizler-Cinbis, R. Gokberk Cinbis, and Stan Sclaroff. Learning actions from the web. In *IEEE International Conference on Computer Vision*, 2009.

Hao Jiang and David R. Martin. Globel pose estimation using non-tree models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

Thorsten Joachims, Thomas Finley, and Chun-Nam Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 2008.

Sam Johnson and Mark Everingham. Combining discriminative appearance and segmentation cues for articulated human pose estimation. In *International Workshop on Machine Learning for Vision-based Motion Analysis*, 2009.

Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *British Machine Vision Conference*, 2010.

Shanon X. Ju, Michael J. Black, and Yaser Yaccob. Cardboard people: A parameterized model of articulated image motion. In *International Conference on Automatic Face and Gesture Recognition*, pages 38–44, 1996.

Yan Ke, Rahul Sukthankar, and Martial Hebert. Event detection in crowded videos. In *IEEE International Conference on Computer Vision*, 2007.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Pawan Kumar, Andrew Zisserman, and Philip H. S. Torr. Efficient discriminative learning of parts-based models. In *IEEE International Conference on Computer Vision*, 2009.

Tian Lan, Yang Wang, Weilong Yang, and Greg Mori. Beyond actions: Discriminative models for contextual group activities. In *Advances in Neural Information Processing Systems*. MIT Press, 2010.

Xiangyang Lan and Daniel P. Huttenlocher. Beyond trees: Common-factor models for 2d human pose recovery. In *IEEE International Conference on Computer Vision*, volume 1, pages 470–477, 2005.

Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

Subhransu Maji, Lubomir Bourdev, and Jitendra Malik. Action recognition from a distributed representation of pose and appearance. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.

David Marr. *A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman, 1982.

Greg Mori. Guiding model search using segmentation. In *IEEE International Conference on Computer Vision*, volume 2, pages 1417–1423, 2005.

Greg Mori and Jitendra Malik. Estimating human body configurations using shape context matching. In *European Conference on Computer Vision*, volume 3, pages 666–680, 2002.

Greg Mori, Xiaofeng Ren, Alyosha Efros, and Jitendra Malik. Recovering human body configuration: Combining segmentation and recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 326–333, 2004.

Juan Carlos Niebles and Li Fei-Fei. A hierarchical model of shape and appearance for human action classification. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *British Machine Vision Conference*, volume 3, pages 1249–1258, 2006.

Deva Ramanan. Learning to parse images of articulated bodies. In *Advances in Neural Information Processing Systems*, volume 19, pages 1129–1136, 2006.

Deva Ramanan and Cristian Sminchisescu. Training deformable models for localization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 206–213, 2006.

Deva Ramanan, David A. Forsyth, and Andrew Zisserman. Strike a pose: Tracking people by finding stylized poses. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 271–278, 2005.

Xiaofeng Ren, Alexander Berg, and Jitendra Malik. Recovering human body configurations using pairwise constraints between parts. In *IEEE International Conference on Computer Vision*, volume 1, pages 824–831, 2005.

Benjamin Sapp, Chris Jordan, and Ben Taskar. Adaptive pose priors for pictorial structures. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010a.

Benjamin Sapp, Alexander Toshev, and Ben Taskar. Cascaded models for articulated pose estimation. In *European Conference on Computer Vision*, 2010b.

Greg Shakhnarovich, Paul Viola, and Trevor Darrell. Fast pose estimation with parameter sensitive hashing. In *IEEE International Conference on Computer Vision*, volume 2, pages 750–757, 2003.

Leonid Sigal and Michael J. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2041–2048, 2006.

Vivek Kumar Singh, Ram Nevatia, and Chang Huang. Efficient inference with multiple heterogenous part detectors for human pose estimation. In *European Conference on Computer Vision*, 2010.

Praveen Srinivasan and Jianbo Shi. Bottom-up recognition and parsing of the human body. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

Josephine Sullivan and Stefan Carlsson. Recognizing and tracking human action. In *European Conference on Computer Vision LNCS 2352*, volume 1, pages 629–644, 2002.

Min Sun and Silvio Savarese. Articulated part-base model for joint object detection and pose estimation. In *IEEE International Conference on Computer Vision*, 2011.

Tai-Peng Tian and Stan Sclaroff. Fast globally optimal 2d human detection with loopy graph models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

Kentaro Toyama and Andrew Blake. Probabilistic exemplar-based tracking in a metric space. In *IEEE International Conference on Computer Vision*, volume 2, pages 50–57, 2001.

Duan Tran and David Forsyth. Improved human parsing with a full relational model. In *European Conference on Computer Vision*, 2010.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

Yang Wang and Greg Mori. Multiple tree models for occlusion and spatial constraints in human pose estimation. In *European Conference on Computer Vision*, 2008.

Yang Wang, Hao Jiang, Mark S. Drew, Ze-Nian Li, and Greg Mori. Unsupervised discovery of action classes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

Yang Wang, Duan Tran, and Zicheng Liao. Learning hierarchical poselets for human parsing. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.

Weilong Yang, Yang Wang, and Greg Mori. Recognizing human actions from still images with latent poses. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.

Bangpeng Yao and Li Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

Long Zhu, Yuanhao Chen, Yifei Lu, Chenxi Lin, and Alan Yuille. Max margin AND/OR graph learning for parsing the human body. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

Journal of Machine Learning Research 13 (2012) 3103-3131 Submitted 2/11; Revised 4/12; Published 10/12

Breaking the Curse of Kernelization: Budgeted Stochastic Gradient Descent for Large-Scale SVM Training

Zhuang Wang * ZHUANG.WANG@SIEMENS.COM
Corporate Technology
Siemens Corporation
755 College Road East
Princeton, NJ 08540, USA

Koby Crammer KOBY@EE.TECHNION.AC.IL
Department of Electrical Engineering
The Technion
Mayer Bldg
Haifa, 32000, Israel

Slobodan Vucetic VUCETIC@TEMPLE.EDU
Department of Computer and Information Sciences
Temple University
1805 N Broad Street
Philadelphia, PA 19122, USA

Editor: Tong Zhang

Abstract

Online algorithms that process one example at a time are advantageous when dealing with very large data or with data streams. Stochastic Gradient Descent (SGD) is such an algorithm and it is an attractive choice for online Support Vector Machine (SVM) training due to its simplicity and effectiveness. When equipped with kernel functions, similarly to other SVM learning algorithms, SGD is susceptible to the curse of kernelization that causes unbounded linear growth in model size and update time with data size. This may render SGD inapplicable to large data sets. We address this issue by presenting a class of Budgeted SGD (BSGD) algorithms for large-scale kernel SVM training which have constant space and constant time complexity per update. Specifically, BSGD keeps the number of support vectors bounded during training through several budget maintenance strategies. We treat the budget maintenance as a source of the gradient error, and show that the gap between the BSGD and the optimal SVM solutions depends on the model degradation due to budget maintenance. To minimize the gap, we study greedy budget maintenance methods based on removal, projection, and merging of support vectors. We propose budgeted versions of several popular online SVM algorithms that belong to the SGD family. We further derive BSGD algorithms for multi-class SVM training. Comprehensive empirical results show that BSGD achieves higher accuracy than the state-of-the-art budgeted online algorithms and comparable to non-budget algorithms, while achieving impressive computational efficiency both in time and space during training and prediction.

Keywords: SVM, large-scale learning, online learning, stochastic gradient descent, kernel methods

1. Introduction

Computational complexity of machine learning algorithms becomes a limiting factor when one is faced with very large amounts of data. In an environment where new large scale problems are emerging in various disciplines and pervasive computing applications are becoming common, there is a real need for machine learning algorithms that are able to process increasing amounts of data efficiently. Recent advances in large-

∗. Zhuang Wang was with the Department of Computer and Information Sciences at Temple University while most of the presented research was performed.

scale learning resulted in many algorithms for training SVMs (Cortes and Vapnik, 1995) using large data (Vishwanathan et al., 2003; Zhang, 2004; Bordes et al., 2005; Tsang et al., 2005; Joachims, 2006; Hsieh et al., 2008; Bordes et al., 2009; Zhu et al., 2009; Teo et al., 2010; Chang et al., 2010b; Sonnenburg and Franc, 2010; Yu et al., 2010; Shalev-Shwartz et al., 2011). However, while most of these algorithms focus on linear classification problems, the area of large-scale kernel SVM training remains less explored. SimpleSVM (Vishwanathan et al., 2003), LASVM (Bordes et al., 2005), CVM (Tsang et al., 2005) and parallel SVMs (Zhu et al., 2009) are among the few successful attempts to train kernel SVM from large data. However, these algorithms do not bound the model size and, as a result, they typically have quadratic training time in the number of training examples. This limits their practical use on large-scale data sets.

A promising avenue to SVM training from large data sets and from data streams is to use online algorithms. Online algorithms operate by repetitively receiving a labeled example, adjusting the model parameters, and discarding the example. This is opposed to offline algorithms where the whole collection of training examples is at hand and training is accomplished by batch learning. SGD is a recently popularized approach (Shalev-Shwartz et al., 2011) that can be used for online training of SVM, where the objective is cast as an unconstrained optimization problem. Such algorithms proceed by iteratively receiving a labeled example and updating the model weights through gradient decent over the corresponding instantaneous objective function. It was shown that SGD converges toward the optimal SVM solution as the number of examples grows (Shalev-Shwartz et al., 2011). In its original non-kernelized form SGD has constant update time and constant space.

To solve nonlinear classification problems, SGD and related algorithms, including the original perceptron (Rosenblatt, 1958), can be easily kernelized combined with Mercer kernels, resulting in prediction models that require storage of a subset of observed examples, called the Support Vectors (SVs).[1] While kernelization allows solving highly nonlinear problems, it also introduces heavy computational burden. The main reason is that on noisy data the number of SVs tends to grow with the number of training examples. In addition to the danger of exceeding the physical memory, this also implies a linear growth in both model update and prediction time with data size. We refer to this property of kernel online algorithms as *the curse of kernelization*. To solve the problem, budgeted online SVM algorithms (Crammer et al., 2004) that limit the number of SVs were proposed to bound the number of SVs. In practice, the assigned budget depends on the specific application requirements, such as memory limitations, processing speed, or data throughput.

In this paper we study a class of BSGD algorithms for online training of kernel SVM. The main contributions of this paper are as follows. First, we propose a budgeted version of the kernelized SGD for SVM that has constant update time and constant space. This is achieved by controlling the number of SVs through one of the several budget maintenance strategies. We study the impact of budget maintenance on SGD optimization and show that, in the limit, the gap between the loss of BSGD and the loss of the optimal solution is upper-bounded by the average model degradation induced by budget maintenance. Second, we develop a multi-class version of BSGD based on the multi-class SVM formulation by Crammer and Singer (2001). The resulting multi-class BSGD has similar algorithmic structure as its binary relative and inherits its theoretical properties. Having shown that the quality of BSGD directly depends on the quality of budget maintenance, our final contribution is exploring computationally efficient methods to maintain an accurate low-budget classifier. In this work we consider three major budget maintenance strategies: removal, projection, and merging. In case of removal, we show that it is optimal to remove the smallest SV. Then, we show that optimal projection of one SV to the remaining ones is achieved by minimizing the accumulated loss of multiple sub-problems for each class, which extends the results by Csató and Opper (2001), Engel et al. (2002) and Orabona et al. (2009) to the multi-class setting. In case of merging, when Gaussian kernel is used, we show that the new SV is always on the line connecting two merged SVs, which generalizes the result by Nguyen and Ho (2005) to the multi-class setting. Both space and update time of BSGD scale quadratically with the budget size when projection is used and linearly when merging or removal are used. We show exper-

1. In this paper, Support Vectors refer to the examples that contribute to the online classifier at a given stage of online learning, which differs slightly from the standard terminology where Support Vector refers to the examples with non-zero coefficients in the dual form of the final classifier.

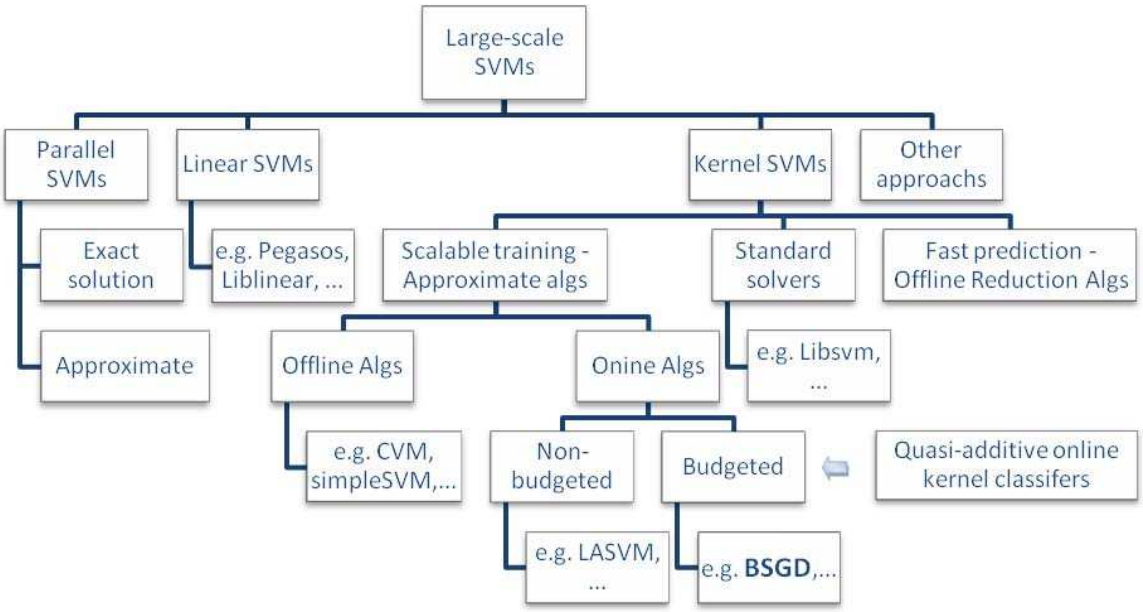


Figure 1: A hierarchy of large-scale SVMs

imentally that BSGD with merging is the most attractive because it is computationally efficient and results in highly accurate classifiers.

The structure of the paper is as follows: related work is given in Section 2; a framework for the proposed algorithms is presented in Section 3; the impact of budget maintenance on SGD optimization is studied in Section 4, which motivates the budget maintenance strategies that are presented in Section 6; the extension to the multi-class setting is described in Section 5; in Section 7, the proposed algorithms are comprehensively evaluated; and, finally, the paper is concluded in Section 8.

2. Related Work

In this section we summarize related work to ours. Figure 1 provides a view at the hierarchy of large-scale SVM training algorithms discussed below.

2.1 Algorithms for Large-Scale SVM Training

LIBSVM (Chang and Lin, 2001) is a widely used SVM solver which is scalable to hundreds of thousands of examples. LIBSVM uses the SMO decomposition technique (Platt, 1998) to solve SVM Quadratic Programming (QP). LASVM (Bordes et al., 2005) is another scalable SMO-based algorithm that approximates the SVM solution by incrementally updating the model. In order to speed up training, LASVM performs only several SMO iterations during each model update and it occasionally removes examples from the training set that are deemed unlikely to become SVs. SimpleSVM (Vishwanathan et al., 2003) is a fast iterative training algorithm that uses greedy working set selection to identify SVs to be incrementally updated. CVM (Tsang et al., 2005) scales up kernel SVM by reformulating SVM's QP as a minimum enclosing ball problem and it applies an efficient approximation algorithm to obtain a near-optimal solution. BVM (Tsang et al., 2007) is a simpler version of CVM that reduces the minimum enclosing ball problem to the enclosing ball problem and thus solves a simpler problem. Experimentally, these approximate algorithms have been demonstrated to have relatively fast training times, result in sparser models, and achieve a slightly reduced accuracy.

Recent research in large-scale linear SVM resulted in many successful algorithms (Zhang, 2004; Joachims, 2006; Shalev-Shwartz et al., 2011; Hsieh et al., 2008; Bordes et al., 2009; Teo et al., 2010) with an impressive scalability and able to train with millions of examples in a matter of minutes on standard PCs. Recently, linear SVM algorithms have been employed for nonlinear classification by explicitly expressing the feature space as a set of attributes and training a linear SVM on the transformed data set (Rahimi and Rahimi, 2007; Sonnenburg and Franc, 2010; Yu et al., 2010). However, this type of approaches is only applicable with special types of kernels (e.g., the low degree polynomial kernels, string kernels or shift invariant kernels) or on very sparse or low dimensional data sets. More recently, Zhang et al. (2012) proposed a low-rank linearization approach that is general to any PSD kernel. The proposed algorithm LLSVM transforms a non-linear SVM to a linear one via an approximate empirical kernel map computed from low-rank approximation of kernel matrices. Taking an advantage of the fast training of linear classifiers, Wang et al. (2011) proposed to use multiple linear classifiers to capture non-linear concepts. A common property of the above linear-classifier-based algorithms is that they usually have low space footprint and are initially designed for offline learning but can also be easily converted to online algorithms by accepting a slight decrease in accuracy. Recent research in training large-scale SVM with the popular Gaussian kernel focuses on parallelizing training on multiple cores or machines. Either optimal (e.g., Graf et al., 2005) or approximate (e.g., Zhu et al., 2009) solutions can be obtained by this type of methods. Other attempts to large-scale kernel SVM learning include a method that modifies the SVM loss function (Collobert et al., 2006), preprocessing methods such as pre-clustering and training on the high-quality summarized data (Li et al., 2007), and a method (Chang et al., 2010a) that decomposes data space and trains multiple SVMs on the decomposed regions.

2.2 Algorithms for SVM Model Reduction

SVM classifier can be thought of as composed of a subset of training examples known as SVs, whose number typically grows linearly with the number of training examples on noisy data (Steinwart, 2003). Bounding the space complexity of SVM classifiers has been an active research since the early days of SVM. SVM reduced set methods (Burges, 1996; Schölkopf et al., 1999) start by training a standard SVM on the complete data and then find a sparse approximation by minimizing Euclidean distance between the original and the approximated SVM. A limitation of reduced set methods is that they require training a full-scale SVM, which can be computationally infeasible on large data. Another line of work (Lee and Mangasarian, 2001; Wu et al., 2005; Dekel and Singer, 2006) is to directly train a reduced classifier from scratch by reformulating the optimization problem. The basic idea is to train SVM with minimal risk on the complete data under a constraint that the model weights are spanned by a small number of examples. A similar method to build reduced SVM classifier based on forward selection was proposed by Keerthi et al. (2006). This method proceeds in an iterative fashion that greedily selects an example to be added to the model so that the risk on the complete data is decreased the most. Although SVM reduction methods can generate a classifier with a fixed size, they require multiple passes over training data. As such, they can be infeasible for online learning.

2.3 Online Algorithms for SVM

Online SVM algorithms were proposed to incrementally update the model weights upon receiving a single example. IDSVM (Cauwenberghs and Poggio, 2000) maintains the optimal SVM solution on all previously seen examples throughout the whole training process by using matrix manipulation to incrementally update the KKT conditions. The high computational cost due to the desire to guarantee an optimum makes it less practical for large-scale learning. As an alternative, LASVM (Bordes et al., 2005) was proposed to trade the optimality with scalability by using an SMO like procedure to incrementally update the model. However, LASVM still does not bound the number of SVs and a potential unlimited growth in their number limits its use for truly large learning tasks. Both IDSVM and LASVM solve SVM optimization by casting it as a QP problem and working on the KKT conditions.

Gradient-based methods are an appealing alternative to the QP based methods for SVM training. SGD for SVM training was first studied by Kivinen et al. (2002), where SVM training is cast as an unconstrained

problem and model weights are updated through gradient decent over an instantaneous objective function. Pegasos (Shalev-Shwartz et al., 2011) is an improved stochastic gradient method, by employing a more aggressively decreasing learning rate and projection. Iterative nature of stochastic gradient makes it suitable for online SVM training. In practice, it is often run in epochs, by scanning the data several times to achieve a convergence to the optimal solution. Recently, Bordes et al. (2009) explored the use of 2nd order information to calculate the gradient in the SGD algorithms. Although the SGD-based methods show impressive training speed for linear SVMs, when equipped with kernel functions, they suffer from the curse of kernelization.

TVM (Wang and Vucetic, 2010b) is a recently proposed budgeted online SVM algorithm which has constant update time and constant space. The basic idea of TVM is to upper bound the number of SVs during the whole learning process. Examples kept in memory (called prototypes) are used both as SVs and as summaries of local data distribution. This has been achieved by positioning the prototypes near the decision boundary, which is the most informative region of the input space. An optimal SVM solution is guaranteed over the set of prototypes at any time. Upon removal or addition of a prototype, IDSVM is employed to update its model.

2.4 Budgeted Quasi-additive Online Algorithms

The Perceptron (Rosenblatt, 1958) is a well-known online algorithm which is updated by simply adding misclassified examples to the model weights. Perceptron belongs to a wider class of quasi-additive online algorithms that updates a model in a greedy manner by using only the last observed example. Popular recent members of this family of algorithms include ALMA (Gentile, 2001), ROMMA (Li and Long, 2002), MIRA (Crammer and Singer, 2003), PA (Crammer et al., 2006), ILK (Cheng et al., 2007), the SGD based algorithms (Kivinen et al., 2002; Zhang, 2004; Shalev-Shwartz et al., 2011), and the Greedy Projection algorithm (Zinkevich, 2003). These algorithms are straightforwardly kernelized. To prevent the curse of kernelization, several budget maintenance strategies for the kernel perceptron have been proposed in recent work. The common property of the methods summarized below is that the number of SVs (the budget) is fixed to a pre-specified value.

Stoptron is a truncated version of kernel perceptron that terminates when number of SVs reaches budget B. This simple algorithm is useful for benchmarking (Orabona et al., 2009).

Budget Perceptron (Crammer et al., 2004) removes the SV that would be predicted correctly and with the largest confidence after its removal. While this algorithm performs well on relatively noise-free data it is less successful on noisy data. This is because in the noisy case this algorithm tends to remove well-classified points and accumulate noisy examples, resulting in a gradual degradation of accuracy.

Random Perceptron employs a simple removal procedure that removes a random SV. Despite its simplicity, this algorithm often has satisfactory performance and its convergence has been proven under some mild assumptions (Cesa-Bianchi and Gentile, 2006).

Forgetron removes the oldest SV. The intuition is that the oldest SV was created when the quality of perceptron was the lowest and that its removal would be the least hurtful. Under some mild assumptions, convergence of the algorithm has also been proven (Dekel et al., 2008). It is worth mentioning that a unified analysis of the convergence of Random Perceptron and Forgetron under the framework of online convex programming was studied by Sutskever (2009) after slightly modifying the two original algorithms.

Tighter Perceptron. The budget maintenance strategy proposed by Weston et al. (2005) is to evaluate accuracy on validation data when deciding which SV to remove. Specifically, the SV whose removal would have the least validation error is selected for removal. From the perspective of accuracy estimation, it is ideal that the validation set consists of all observed examples. Since it can be too costly, a subset of examples can be used for validation. In the extreme, only SVs from the model might be used, but the drawback is that the SVs are not representative of the underlying distribution that could lead to misleading accuracy estimation.

Tightest Perceptron is a modification of Tighter Perceptron that improves how the SV set is used both for model representation and for estimation (Wang and Vucetic, 2009). In particular, instead of using the actual labels of SVs, the Tightest learns distribution of labels in the neighborhood of each SV and uses this information for improved accuracy estimation.

Algorithms	Budget maintenance	Update time	Space
BPA_{NN}	projection	$O(B)$	$O(B)$
$BSGD+removal$	removal	$O(B)$	$O(B)$
$BSGD+project$	projection	$O(B^2)$	$O(B^2)$
$BSGD+merge$	merging	$O(B)$	$O(B)$
$Budget$	removal	$O(B)$	$O(B)$
$Forgetron$	removal	$O(B)$	$O(B)$
$Projectron++$	projection	$O(B^2)$	$O(B^2)$
$Random$	removal	$O(B)$	$O(B)$
$SILK$	removal	$O(B)$	$O(B)$
$Stoptron$	stop	$O(1)$	$O(B)$
$Tighter$	removal	$O(B^2)$	$O(B)$
$Tightest$	removal	$O(B^2)$	$O(B)$
TVM	merging	$O(B^2)$	$O(B^2)$

Table 1: Comparison of different budgeted online algorithms (B is a pre-specified budget equal to the number of SVs; Update time includes both model update time and budget maintenance time; Space corresponds to space needed to store the model and perform model update and budget maintenance.)

Projectron maintains a sparse representation by occasionally projecting an SV onto remaining SVs (Orabona et al., 2009). The projection is designed to minimize the model weight degradation caused by removal of an SV, which requires updating the weights of the remaining SVs. Instead of enforcing a fixed budget, the original algorithm adaptively increases it according to a pre-defined sparsity parameter. It can be easily converted to the budgeted version by projecting when the budget is exceeded.

SILK discards the example with the lowest absolute coefficient value once the budget is exceeded (Cheng et al., 2007).

BPA. Unlike the previously described algorithms that perform budget maintenance only after the model is updated, Wang and Vucetic (2010a) proposed a Budgeted online Passive-Aggressive (BPA) algorithm that does budget maintenance and model updating jointly by introducing an additional constraint into the original Passive-Aggressive (PA) (Crammer et al., 2006) optimization problem. The constraint enforces that the removed SV is projected onto the space spanned by the remaining SVs. The optimization leads to a closed-form solution.

The properties of budgeted online algorithms described in this subsection as well as and the BSGD algorithms presented in following sections are summarized in Table 1. It is worth noting that although (budgeted) online algorithms are typically trained by a single pass through training data, they are also able to perform multiple passes that can lead to improved accuracy.

3. Budgeted Stochastic Gradient Descent (BSGD) for SVMs

In this section, we describe an algorithmic framework of BSGD for SVM training.

3.1 Stochastic Gradient Descent (SGD) for SVMs

Consider a binary classification problem with a sequence of labeled examples $S = \{(\mathbf{x}_i, y_i), i = 1, ..., N\}$, where instance $\mathbf{x}_i \in R^d$ is a d-dimensional input vector and $y_i \in \{+1, -1\}$ is the label. Training an SVM classifier[2] $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ using S, where $\mathbf{w}$ is a vector of weights associated with each input, is formulated as

2. We study the case where the bias term is set to zero.

Algorithms	λ	η_t
Pegasos	> 0	$1/(\lambda t)$
Norma	> 0	$\eta/\sqrt{t}$
Margin Perceptron	0	η

Table 2: A summary of three SGD algorithms (η is a constant.)

solving the following optimization problem

$$\min P(\mathbf{w}) = \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{N}\sum_{t=1}^{N} l(\mathbf{w};(\mathbf{x}_t, y_t)), \tag{1}$$

where $l(\mathbf{w};(\mathbf{x}_t,\mathbf{y}_t)) = \max(0, 1 - y_t \mathbf{w}^T \mathbf{x}_t)$ is the *hinge loss* function and $\lambda \geq 0$ is a regularization parameter used to control model complexity.

SGD works iteratively. It starts with an initial guess of the model weight $\mathbf{w}_1$, and at t-th round it updates the current weight $\mathbf{w}_t$ as

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \nabla_t, \tag{2}$$

where $\nabla_t = \nabla_{\mathbf{w}_t} P_t(\mathbf{w}_t)$ is the (sub)gradient of the *instantaneous loss* function $P_t(\mathbf{w})$ defined only on the latest example,

$$P_t(\mathbf{w}) = \frac{\lambda}{2}||\mathbf{w}||^2 + l(\mathbf{w};(\mathbf{x}_t, y_t)), \tag{3}$$

at $\mathbf{w}_t$, and $\eta_t > 0$ is a learning rate. Thus, (2) can be rewritten as

$$\mathbf{w}_{t+1} \leftarrow (1 - \lambda\eta_t)\mathbf{w}_t + \beta_t \mathbf{x}_t, \tag{4}$$

where

$$\beta_t \leftarrow \begin{cases} \eta_t y_t, & \text{if } y_t \mathbf{w}_t^T \mathbf{x}_t < 1 \\ 0, & \text{otherwise.} \end{cases}$$

Several learning algorithms are based on (or can be viewed as) SGD for SVM. In Table 2, Pegasos[3] (Shalev-Shwartz et al., 2011), Norma (Kivinen et al., 2002), and Margin Perceptron[4] (Duda and Hart, 1973) are viewed as the SGD algorithms. They share the same update rule (4), but have different scheduling of learning rate. In addition, Margin Perceptron differs because it does not contain the regularization term in (3).

3.2 Kernelization

SGD for SVM can be used to solve non-linear problems when combined with Mercer kernels. After introducing a nonlinear function Φ that maps $\mathbf{x}$ from the input to the feature space and replacing $\mathbf{x}$ with $\Phi(\mathbf{x})$, $\mathbf{w}_t$ can be described as

$$\mathbf{w}_t = \sum_{j=1}^{t} \alpha_j \Phi(\mathbf{x}_j),$$

where

$$\alpha_j = \beta_j \prod_{k=j+1}^{t} (1 - \eta_k \lambda). \tag{5}$$

3. In this paper we study the Pegasos algorithm without the optional projecting step (Shalev-Shwartz et al., 2011). It is worth to note that we can both cases (with or without the optional projecting step) allow similar analysis. We focus on this version since it has closer connection to the other two algorithms we study.

4. Margin Perceptron is a robust variant of the classical perceptron (Rosenblatt, 1958), by changing the update criterion from $y\mathbf{w}^T\mathbf{x} < 0$ to $y\mathbf{w}^T\mathbf{x} < 1$.

Algorithm 1 BSGD

1: **Input**: data S, kernel k, regularization parameter λ, budget B;
2: **Initialize**: $b = 0, \mathbf{w}_1 = \mathbf{0}$;
3: **for** $i = t, 2, \ldots$ **do**
4: receive $(\mathbf{x}_t, y_t)$;
5: $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t \lambda)\mathbf{w}_t$
6: **if** $l(\mathbf{w}_t; (\mathbf{x}_t, y_t)) > 0$ **then**
7: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1} + \Phi(\mathbf{x}_t)\beta_t$; // add an SV
8: $b \leftarrow b + 1$;
9: **if** $b > B$ **then**
10: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1} - \Delta_t$; // **budget maintenance**
11: $b \leftarrow b - 1$;
12: **end if**
13: **end if**
14: **end for**
15: **Output**: $f_{t+1}(\mathbf{x})$

From (5), it can be seen that an example $(\mathbf{x}_t, y_t)$ whose hinge loss was zero at time t has zero value of α and can therefore be ignored. Examples with nonzero values α are called the Support Vectors (SVs). We can now represent $f_t(\mathbf{x})$ as the kernel expansion

$$f_t(\mathbf{x}) = \mathbf{w}_t^T \Phi(\mathbf{x}) = \sum_{j \in I_t} \alpha_j k(\mathbf{x}_j, \mathbf{x}),$$

where k is the Mercer kernel induced by Φ and I_t is the set of indexes of all SVs in $\mathbf{w}_t$. Rather than explicitly calculating $\mathbf{w}$ by using $\Phi(\mathbf{x})$, that might be infinite-dimensional, it is more efficient to save SVs to implicitly represent $\mathbf{w}$ and to use kernel function k when calculating prediction $\mathbf{w}^T \Phi(\mathbf{x})$. This is known as the *kernel trick*. Therefore, an SVM classifier is completely described by either a weight vector $\mathbf{w}$ or by an SV set $\{(\alpha_i, \mathbf{x}_i), i \in I_t\}$. From now on, depending on the convenience of presentation, we will use either the $\mathbf{w}$ notation or the α notation interchangeably.

3.3 Budgeted SGD (BSGD)

To maintain a fixed number of SVs, BSGD executes a budget maintenance step whenever the number of SVs exceeds a pre-defined budget B (i.e., $|I_{t+1}| > B$). It reduces the size of I_{t+1} by one, such that $\mathbf{w}_{t+1}$ is only spanned by B SVs. This results in degradation of the SVM classifier. We present a generic BSGD algorithm for SVM in Algorithm 1. Here, we denote by Δ_t the weight degradation caused by budget maintenance at t-th round, which is defined[5] as the difference between model weights before and after budget maintenance (Line 10 of Algorithm 1). We note that all budget maintenance strategies mentioned in Section 2.4, except BPA, can be represented as Line 10 of Algorithm 1.

Budget maintenance is a critical design issue. We describe several budget maintenance strategies for BSGD in Section 6. In the next section, we motivate different strategies by studying in the next section how budget maintenance influences the performance of SGD.

4. Impact of Budget Maintenance on SGD

This section provides an insight into the impact of budget maintenance on SGD. In the following, we quantify the optimization error introduced by budget maintenance on three known SGD algorithms. Without loss of generality, we assume $||\Phi(\mathbf{x})|| \leq 1$.

5. The formal definition for different strategies is presented in Algorithm 2.

First, we analyze Budgeted Pegasos (BPegasos), a BSGD algorithm using the Pegasos style learning rate from Table 2.

Theorem 1 *Let us consider BPegasos (Algorithm 1 using the Pegasos learning rate, see Table 2) running on a sequence of examples S. Let $\mathbf{w}^*$ be the optimal solution of Problem (1). Define the gradient error $E_t = \Delta_t/\eta_t$ and assume $||E_t|| \leq 1$. Define the average gradient error as $E = \sum_{t=1}^{N} ||E_t||/N$. Let*

$$U = \begin{cases} 2/\lambda, & if\ \lambda \leq 4 \\ 1/\sqrt{\lambda}, & otherwise. \end{cases} \tag{6}$$

Then, the following inequality holds,

$$\frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}_t) - \frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}^*) \leq \frac{(\lambda U + 2)^2(\ln(N)+1)}{2\lambda N} + 2UE. \tag{7}$$

■

The proof is in Appendix A. Remarks on Theorem 1:

- In Theorem 1 we quantify how budget maintenance impacts the quality of SGD optimization. Observe that as N grows, the first term in the right side of inequalities (7) converges to zero. Therefore, the averaged instantaneous loss of BSGD converges toward the averaged instantaneous loss of optimal solution $\mathbf{w}^*$, and the gap between these two is upper bounded by the averaged gradient error E. *The results suggest that an optimal budget maintenance should attempt to minimize E. To minimize E in the setting of online learning, we propose a greedy procedure that minimizes $||E_t||$ at each round.*

- The assumption $||E_t|| \leq 1$ is not restrictive. Let us assume the *removal*-based budget maintenance method, where, at round t, SV with index t is removed. Then, the weight degradation is $\Delta_t = \alpha_{t'}\Phi(\mathbf{x}_{t'})$, where t' is the index of any SV in the budget. By using (5) it can be seen that $||E_t||$ is not larger than 1,

$$\begin{aligned} ||E_t|| &\leq \left\| \frac{\alpha_{t'}}{\eta_t} \right\| = \lambda t \left\{ \eta_{t'} \prod_{j=t'+1}^{t} (1-\eta_j\lambda) \right\} \\ &= \lambda t \left\{ \eta_{t'} \cdot \frac{t'}{t'+1} \cdot \frac{t'+1}{t'+2} \cdot \ldots \cdot \frac{t-2}{t-1} \cdot \frac{t-1}{t} \right\} = 1. \end{aligned}$$

  Since our proposed budget maintenance strategy is to minimize $||E_t||$ at each round, $||E_t|| \leq 1$ holds.

Next, we show a similar theorem for Budgeted Norma (BNorma), a BSGD algorithm using the Norma style update rule from Table 1.

Theorem 2 *Let us consider BNorma (Algorithm 1 using the Norma learning rate from Table 2) running on a sequence of examples S. Let $\mathbf{w}^*$ be the optimal solution of Problem (1). Assume $||E_t|| \leq 1$. Let U be defined as in (6). Then, the following inequality holds,*

$$\frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}_t) - \frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}^*) \leq \frac{(2U^2/\eta + \eta(\lambda U + 2)^2)\sqrt{N}}{N} + 2UE. \tag{8}$$

■

The proof is in Appendix B. The remarks on Theorem 1 also hold[6] for Theorem 2.

Next, we show the result for Budgeted Margin Perceptron (BMP). The update rule of Margin Perceptron (MP) summarized in Table 2 does not bound growth of the weight vector. We add a projection step to MP after the SGD update to guarantee an upper bound on the norm of the weight vector.[7] More specifically, the new update rule is $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \nabla_t) \equiv \phi_t(\mathbf{w}_t - \nabla_t)$ where C is the closed convex set with radius U and $\prod_C(\mathbf{u})$

6. The assumption $||E_t|| \leq 1$ holds when budget maintenance is achieved by removing the smallest SV, that is, $t' = \arg\min_{j \in I_{t'+1}} ||\alpha_j \Phi(\mathbf{x}_j)||$.

7. The projection step (Zinkevich, 2003; Shalev-Shwartz and Singer, 2007; Sutskever, 2009) is a widely used technical operation needed for the convergence analysis.

defines the closest point to $\mathbf{u}$ in C. We can replace the projection operator with $\phi_t = \min\{1, U/||\mathbf{w}_t - \nabla_t||\}$. It is worth to note that, although the MP with the projection step solves an un-regularized SVM problem (i.e., $\lambda = 0$ in (1)), the projection to a ball with radius U does introduce the regularization by enforcing the weight with bounded norm U. The vector U should be treated as a hyper-parameter and smaller U values enforce simpler models.

After this modification, the resulting BMP algorithm can be described with Algorithm 1, where an additional projection step $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_{t+1})$ is added at the end of each iteration (after Line 12 of Algorithm 1).

Theorem 3 *Let C be a closed convex set with a pre-specified radius U. Let BMP (Algorithm 1 using the PMP learning rate from Table 2 and the projection step) run on a sequence of examples S. Let $||\mathbf{w}^*||$ be the optimal solution to Problem (1) with $\lambda = 0$ and subject to the constraint $||\mathbf{w}^*|| \leq U$. Assume $||E_t|| \leq 1$ Then, the following inequality holds,*

$$\frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}_t) - \frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}^*) \leq \frac{2U^2}{N\eta} + 2\eta + 2UE. \tag{9}$$

■

The proof is in Appendix C. The remarks on Theorem 1 also hold for Theorem 3.

5. BSGD for Multi-Class SVM

Algorithm 1 can be extended to the multi-class setting. In this section we show that the resulting multi-class BSGD inherits the same algorithmic structure and theoretical properties of its binary counterpart.

Consider a sequence of examples $S = \{(\mathbf{x}_i, y_i), i = 1, ..., N\}$, where instance $\mathbf{x}_i \in R^d$ is a d-dimensional input vector and the multi-class label y_i belongs to the set $Y = \{1, ..., c\}$. We consider the multi-class SVM formulation by Crammer and Singer (2001). Let us define the multi-class model $f^M(\mathbf{x})$ as

$$f^M(\mathbf{x}) = \arg\max_{i \in Y}\{f^{(i)}(\mathbf{x})\} = \arg\max_{i \in Y}\{(\mathbf{w}^{(i)})^T\mathbf{x}\},$$

where $f^{(i)}$ is the i-th class-specific predictor and $\mathbf{w}^{(i)}$ is its corresponding weight vector. By adding all class-specific weight vectors, we construct $\mathbf{W} = [\mathbf{w}^{(1)}...\mathbf{w}^{(c)}]$ as the $d \times c$ weight matrix of $f^M(\mathbf{x})$. The predicted label of $\mathbf{x}$ is the class of the weight vector that achieves the maximal value $(\mathbf{w}^{(i)})^T\mathbf{x}$. Given this setup, training a multi-class SVM on S consists of solving the optimization problem

$$\min_{\mathbf{W}} P^M(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{N}\sum_{t=1}^{N} l^M(\mathbf{W}; (\mathbf{x}_t, y_t)), \tag{10}$$

where the binary hinge loss is replaced with the *multi-class* hinge loss defined as

$$l^M(\mathbf{W}; (\mathbf{x}_t, y_t)) = \max(0, 1 + f^{(r_t)}(\mathbf{x}_t) - f^{(y_t)}(\mathbf{x}_t)), \tag{11}$$

where $r_t = \arg\max_{i \in Y, i \neq y_t} f^{(i)}(\mathbf{x}_t)$, and the norm of the weight matrix $\mathbf{W}$ is

$$||\mathbf{W}||^2 = \sum_{i \in Y} ||\mathbf{w}^{(i)}||^2.$$

The subgradient matrix ∇_t of the multi-class instantaneous loss function,

$$P_t^M(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + l^M(\mathbf{W}; (\mathbf{x}_t, y_t)),$$

at $\mathbf{W}_t$ is defined as $\nabla_t = [\nabla_t^{(1)}...\nabla_t^{(c)}]$, where $\nabla_t^{(i)} = \nabla_{\mathbf{w}^{(i)}} P_t^M(\mathbf{W})$ is a column vector. If loss (11) is equal to zero then $\nabla_t^{(i)} = \lambda\mathbf{w}_t^{(i)}$. If loss (11) is above zero, then

$$\nabla_t^{(i)} = \begin{cases} \lambda\mathbf{w}_t^{(i)} - \mathbf{x}_t, & \text{if } i = y_t \\ \lambda\mathbf{w}_t^{(i)} + \mathbf{x}_t, & \text{if } i = r_t \\ \lambda\mathbf{w}_t^{(i)}, & \text{otherwise.} \end{cases}$$

Thus, the update rule for the multi-class SVM becomes

$$\mathbf{W}_{t+1} \leftarrow \mathbf{W}_t - \eta_t \nabla_t = (1 - \eta_t \lambda)\mathbf{W}_t + \mathbf{x}_t \beta_t,$$

where β_t is a row vector, $\beta_t = [\beta_t^{(1)} ... \beta_t^{(c)}]$. If loss (11) is equal to zero, then $\beta_t = \mathbf{0}$; otherwise,

$$\beta_t^{(i)} = \begin{cases} \eta_t, & \text{if } i = y_t \\ -\eta_t, & \text{if } i = r_t \\ 0, & \text{otherwise.} \end{cases}$$

When used in conjunction with kernel, $\mathbf{w}_t^{(i)}$ can be described as

$$\mathbf{w}_t^{(i)} = \sum_{j=1}^{t} \alpha_j^{(i)} \Phi(\mathbf{x}_j),$$

where

$$\alpha_j^{(i)} = \beta_j^{(i)} \prod_{k=j+1}^{t} (1 - \eta_k \lambda).$$

The budget maintenance step can be achieved as

$$\mathbf{W}_{t+1} \leftarrow \mathbf{W}_{t+1} - \Delta_t \Rightarrow \mathbf{w}_{t+1}^{(i)} \leftarrow \mathbf{w}_{t+1}^{(i)} - \Delta_t^{(i)},$$

where $\Delta_t = [\Delta_t^{(1)} \ ... \ \Delta_t^{(c)}]$ and the column vectors $\Delta_t^{(i)}$ are the coefficients for the i-th class-specific weight, such that $\mathbf{w}_{t+1}^{(i)}$ is spanned only by B SVs.

Algorithm 1 can be applied to the multi-class version after replacing scalar β_t with vector β_t, vector $\mathbf{w}_t$ with matrix $\mathbf{W}_t$ and vector Δ_t with matrix Δ_t.

The analysis of the gap between BSGD and SGD optimization for the multi-class version is similar to that provided for its binary version, presented in Section 4. If we assume $||\Phi(\mathbf{x})||^2 \leq 0.5$, then the resulting multi-class counterparts of Theorems 1, 2, and 3 become identical to their binary variants by simply replacing the text Problem (1) with Problem (10).

6. Budget Maintenance Strategies

The analysis in Sections 4 and 5 indicates that budget maintenance should attempt to minimize the averaged gradient error E. To minimize E in the setting of online learning, we propose a greedy procedure that minimizes the gradient error $||E_t||$ at each round. From the definition of $||E_t||$ in Theorem 1, minimizing $||E_t||$ is equivalent to minimizing the weight degradation $||\Delta_t||$,

$$\min ||\Delta_t||^2. \tag{12}$$

In the following, we address Problem (12) through three budget maintenance strategies: removing, projecting and merging of SV(s). We discuss our solutions under the multi-class setting, and consider the binary setting as a special case. Three styles of budget maintenance update rules are summarized in Algorithm 2 and discussed in more detail in the following three subsections.

6.1 Budget Maintenance through Removal

If budget maintenance removes the p-th SV, then

$$\Delta_t = \Phi(\mathbf{x}_p)\alpha_p,$$

where the row vector $\alpha_p = [\alpha_p^{(1)} ... \alpha_p^{(c)}]$ contains the c class-specific coefficients of j-th SV. The optimal solution of (12) is removal of SV with the smallest norm,

$$p = \arg \min_{j \in I_{t+1}} ||\alpha_j||^2 k(\mathbf{x}_j, \mathbf{x}_j).$$

Algorithm 2 Budget maintenance

Removal:
1. select some p;
2. $\Delta_t = \Phi(\mathbf{x}_p)\alpha_p$;

Projection:
1. select some p;
2. $\Delta_t = \Phi(\mathbf{x}_p)\alpha_p - \sum_{j \in I_{t+1}-p} \Phi(\mathbf{x}_j)\Delta\alpha_j$;

Merging
1. select some m and n;
2. $\Delta_t = \Phi(\mathbf{x}_m)\alpha_m + \Phi(\mathbf{x}_n)\alpha_n - \Phi(\mathbf{z})\alpha_z$;

Let us consider the class of translation invariant kernels where $k(\mathbf{x},\mathbf{x}') = \tilde{k}(\mathbf{x}-\mathbf{x}')$, which encompasses the Gaussian kernel. Let us assume, without loss of generality, that $k(\mathbf{x},\mathbf{x}) = 1$. In this case, the best SV to remove is the one with the smallest $||\alpha_p||$. Note:

- In BPegasos with SV removal with Gaussian kernel, $||E_t|| = 1$. Thus, from the perspective of (12), all removal strategies are equivalent.
- In BNorma, the SV with the smallest norm depends on the specific choice of λ and η parameters. Therefore, the decision of which SV to remove should be made during runtime. It is worth noting that removal of the smallest SV was the strategy used by Kivinen et al. (2002) and Cheng et al. (2007) to truncate model weight for Norma.
- In BMP, $||\alpha_p||^2 k(\mathbf{x}_p,\mathbf{x}_p) = (\eta\prod_{i=p+1}^{t}\phi_i)^2 k(\mathbf{x}_p,\mathbf{x}_p)$, because of the projection operation. Knowing that $\phi_i \leq 1$ the optimal removal will select the oldest SV. We note that removal of the oldest SV is the strategy used in Forgetron (Dekel et al., 2008).

Let us now briefly discuss other kernels, where $k(\mathbf{x},\mathbf{x})$ in general depends on $\mathbf{x}$. In this case, the SV with the smallest norm needs to be found at runtime. How much of computational overhead this would produce depends on the particular algorithm. In case of BPegasos, this would entail finding SV with the smallest $k(\mathbf{x}_p,\mathbf{x}_p)$, while in case of Norma and BMP, it would be SV with the smallest $||\alpha_p||^2 k(\mathbf{x}_p,\mathbf{x}_p)$ value.

6.2 Budget Maintenance through Projection

Let us consider budget maintenance through projection. In this case, before the p-th SV is removed from the model, it is projected to the remaining SVs to minimize the weight degradation. By considering the multi-class case, projection can be defined as the solution of the following optimization problem,

$$\min_{\Delta\alpha} \sum_{i \in Y} \left\| \alpha_p^{(i)} \Phi(\mathbf{x}_p) - \sum_{j \in I_{t+1}-p} \Delta\alpha_j^{(i)} \Phi(\mathbf{x}_j) \right\|^2, \tag{13}$$

where $\Delta\alpha_j^{(i)}$ are coefficients of the projected SV to each of the remaining SVs. After setting the gradient of (13) with respect to the class-specific column vector of coefficients $\Delta\alpha^{(i)}$ to zero, one can obtain the optimal solution as

$$\forall i \in Y, \Delta\alpha^{(i)} = \alpha_p^{(i)} \mathbf{K}_p^{-1} \mathbf{k}_p, \tag{14}$$

where $\mathbf{K}_p = [k_{ij}], \forall i,j \in I_{t+1}-p$ is the kernel matrix, $k_{ij} = k(\mathbf{x}_i,\mathbf{x}_j)$, and $\mathbf{k}_p = [k_{pj}]^T, \forall j \in I_{t+1}-p$ is the column vector. It should be observed that inverting $\mathbf{K}_p$ can be accomplished in $O(B^2)$ time if Woodbury formula (Cauwenberghs and Poggio, 2000) is applied to reuse the results of inversion from previous projections. Finally, upon removal of the p-th SV, $\Delta\alpha$ is added to α of the remaining SVs.

The remaining issue is finding the best among $B+1$ candidate SVs for projection. After plugging (14) into (13) we can observe that the minimal weight degradation of projecting equals

$$\min ||\Delta_t||^2 = \min_{p \in I_{t+1}} ||\alpha_p||^2 \left(k_{pp} - \mathbf{k}_p^T (\mathbf{K}_p^{-1} \mathbf{k}_p)\right). \tag{15}$$

Considering there are $B+1$ SVs, evaluation of (15) requires $O(B^3)$ time for each budget maintenance step. As an efficient approximation, we propose a simplified solution that always projects the smallest SV, $p = \arg\min_{j \in I_{t+1}} ||\alpha_j||^2 k(\mathbf{x}_j, \mathbf{x}_j)$. Then, the computation is reduced to $O(B^2)$. We should also note that the space requirement of projection is $O(B^2)$, which is needed to store the kernel matrix and its inverse. Unlike the recently proposed projection method for multi-class perceptron (Orabona et al., 2009), that projects an SV only onto the SVs assigned to the same class, our method solves a more general case by projecting an SV onto all the remaining SVs, thus resulting in smaller weight degradation.

It should be observed that, by selecting the smallest SV to project, it can be guaranteed that weight degradation of projection is upper bounded by weight degradation of removal for any t, for all three BSGD variants from Table 1. Therefore, Theorems 1, 2, and 3 remain valid for projection. Since weight degradation for projection is expected to be, on average, smaller than that for removal, it is expected that the average error E would be smaller too, thus resulting in smaller gap in the average instantaneous loss .

6.3 Budget Maintenance through Merging

Problem (12) can also be solved by merging two SVs to a newly created one. The justification is as follows. For the i-th class weight, if $\Phi(\mathbf{x}_m)$ and $\Phi(\mathbf{x}_n)$ are replaced by

$$M^{(i)} = \left(\alpha_m^{(i)} \Phi(\mathbf{x}_m) + \alpha_n^{(i)} \Phi(\mathbf{x}_n)\right) / (\alpha_m^{(i)} + \alpha_n^{(i)}),$$

(assuming $\alpha_m^{(i)} + \alpha_n^{(i)} \neq 0$) and the coefficient of $M^{(i)}$ is set to $\alpha_m^{(i)} + \alpha_n^{(i)}$, then the weight remains unchanged. The difficulty is that $M^{(i)}$ cannot be used directly because the pre-image of $M^{(i)}$ may not exist. Moreover, even if the pre-images existed, since every class results in different $M^{(i)}$, it is not clear what M would be the best overall choice. To resolve these issues, we define the merging problem as finding an input space vector $\mathbf{z}$ whose image $\Phi(\mathbf{z})$ is at the minimum distance from the class-specific $M^{(i)}$'s,

$$\min_{\mathbf{z}} \sum_{i \in Y} ||M^{(i)} - \Phi(\mathbf{z})||^2. \tag{16}$$

Let us assume a radial kernel,[8] $k(\mathbf{x}, \mathbf{x}') = \tilde{k}(||\mathbf{x} - \mathbf{x}'||^2)$, is used. Problem (16) can then be reduced to

$$\max_{\mathbf{z}} \sum_{i \in Y} (M^{(i)})^T \Phi(\mathbf{z}). \tag{17}$$

Setting the gradient of (17) with respect to $\mathbf{z}$ to zero, leads to solution

$$\mathbf{z} = h\mathbf{x}_m + (1-h)\mathbf{x}_n, \text{where } h = \frac{\sum_{i \in Y} m^{(i)} \tilde{k}'(||\mathbf{x}_m - \mathbf{z}||^2)}{\sum_{i \in Y} \left(m^{(i)} \tilde{k}'(||\mathbf{x}_m - \mathbf{z}||^2) + (1 - m^{(i)}) \tilde{k}'(||\mathbf{x}_n - \mathbf{z}||^2)\right)}, \tag{18}$$

where $m^{(i)} = \alpha_m^{(i)} / (\alpha_m^{(i)} + \alpha_n^{(i)})$, and $\tilde{k}'(x)$ is the first derivative of $\tilde{k}$. (18) indicates that $\mathbf{z}$ lies on the line connecting $\mathbf{x}_m$ and $\mathbf{x}_n$. Plugging (18) into (17), the merging problem is simplified to finding

$$\max_h \sum_{i \in Y} \left(m^{(i)} k_{1-h}(\mathbf{x}_m, \mathbf{x}_n) + (1 - m^{(i)}) k_h(\mathbf{x}_m, \mathbf{x}_n)\right),$$

where we denoted $k_h(\mathbf{x}, \mathbf{x}') = k(h\mathbf{x}, h\mathbf{x}')$. We can use any efficient line search method (e.g., the golden search) to find the optimal h, which takes $O(log(1/\varepsilon))$time, where ε is the accuracy of the solution. After that, the optimal $\mathbf{z}$ can be calculated using (18).

8. Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp(-\sigma||\mathbf{x} - \mathbf{x}'||^2)$is a radial kernel.

After obtaining the optimal solution $\mathbf{z}$, the optimal coefficient $\alpha_z^{(i)}$ for approximating $\alpha_m^{(i)}\Phi(\mathbf{x}_m)+\alpha_n^{(i)}\Phi(\mathbf{x}_n)$ by $\alpha_z^{(i)}\Phi(\mathbf{z})$ is obtained by minimizing the following objective function

$$||\Delta_t||^2 \equiv \min_{\alpha_z^{(i)}} \sum_{i\in Y} \left\| \alpha_m^{(i)}\Phi(\mathbf{x}_m)+\alpha_n^{(i)}\Phi(\mathbf{x}_n)-\alpha_z^{(i)}\Phi(\mathbf{z}) \right\|^2. \tag{19}$$

The optimal solution of (19) is

$$\alpha_z^{(i)} = \alpha_m^{(i)}k(\mathbf{x}_m,\mathbf{z})+\alpha_n^{(i)}k(\mathbf{x}_n,\mathbf{z}).$$

The remaining question is what pair of SVs leads to the smallest weight degradation. The optimal solution can be found by evaluating merging of all $B(B-1)/2$ pairs of SVs, which would require $O(B^2)$ time. To reduce the computational cost, we use the same simplification as in projection (Section 6.2), by fixing m as the SV with the smallest value of $||\alpha_m||^2$. Thus, the computation is reduced to $O(B)$. We should observe that the space requirement is only $O(B)$ because there is no need to store the kernel matrix.

It should be observed that, by selecting the smallest SV to merge, it can be guaranteed that weight degradation of merging is upper bounded by weight degradation of removal for any t, for all three BSGD variants from Table 1. Therefore, Theorems 1, 2, and 3 remain valid for merging. Using the same argument as for projection, E is expected to be smaller than that of removal.

6.4 Relationship between Budget and Weight Degradation

When budget maintenance is achieved by projection and merging, there is an additional impact of budget size on E. As budget size B grows, the density of SVs is expected to grow. As a result, the weight degradation of projection and merging is expected to decrease, thus leading to decrease in E. The specific amount depends on the specific data set and specific kernel. We evaluate the impact of B on E experimentally in Table 6 .

7. Experiments

In this section, we evaluate BSGD[9] and compare it to related algorithms on 14 benchmark data sets.

7.1 Experimental Setting

We first describe the data sets and the evaluated algorithms.

7.2 Data Sets

The properties (training size, dimensionality, number of classes) of 14 benchmark data sets[10] are summarized in the first row of Tables 3, 4 and 5. Gauss data was generated as a mixture of 2 two-dimensional Gaussians: one class is from $N((0,0),I)$ and another is from $N((2,0),4I)$. Checkerboard data was generated as a uniformly distributed two-dimensional 4×4 checkerboard with alternating class assignments. Attributes in all data sets were scaled to mean 0 and standard deviation 1.

7.3 Algorithms

We evaluated several budget maintenance strategies for BSGD algorithms BPegasos, BNorma, and BMP. Specifically, we explored the following budgeted online algorithms:

- BPegasos+remove: multi-class BPegasos with arbitrary SV removal;[11]

9. Our implementation of BSDG algorithms is available at `www.dabi.temple.edu/~vucetic/BSGD.html`.
10. Adult, Covertype, DNA, IJCNN, Letter, Satimage, Shuttle and USPS are available at `www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`, Banana is available at `ida.first.fhg.de/projects/bench/benchmarks.htm`, and Waveform data generator and Pendigits are available at `archive.ics.uci.edu/ml/datasets.html`.
11. Arbitrary removal is equivalent to removing the smallest one, as discussed in Section 6.1.

- BPegasos+project: multi-class BPegasos with projection of the smallest SV;
- BPegasos+merge: multi-class BPegasos with merging of the smallest SV;
- BNorma+merge: multi-class BNorma with merging of the smallest SV;
- BMP+merge: multi-class BPMP with merging of the smallest SV.

These algorithms were compared to the following offline, online, and budgeted online algorithms:
Offline algorithms:

- LIBSVM: state-of-art offline SVM solver (Chang and Lin, 2001); we used the 1 vs rest method as the default setting for the multi-class tasks.

Online algorithms:

- IDSVM: online SVM algorithm which achieves the optimal solution (Cauwenberghs and Poggio, 2000);
- Pegasos: non-budgeted kernelized Pegasos (Shalev-Shwartz et al., 2011);
- Norma: non-budgeted stochastic gradient descent for kernel SVM (Kivinen et al., 2002);
- MP: non-budgeted margin perceptron algorithm (Duda and Hart, 1973) equipped with a kernel function.

Budgeted online algorithms:

- TVM: SVM-based budgeted online algorithm (Wang and Vucetic, 2010b);
- BPA: budgeted Passive-Aggressive algorithm that uses the projection of an SV to its nearest neighbor to maintain the budget (the BPA_{NN} version in Wang and Vucetic, 2010a);
- MP+stop: margin perceptron algorithm that stops training when the budget is exceeded;
- MP+random: margin perceptron algorithm that removes a random SV when the budget is exceeded;
- Projectron++: margin perceptron that projects an SV only if the weight degradation is below the threshold; otherwise, budget is increased by one SV (Orabona et al., 2009). In our experiments, we set the Projectron++ threshold such that the number of SVs equals B of the budgeted algorithms at the end of training.

Gaussian kernel was used in all experiments. For Norma and BNorma, the learning rate parameter η was set either to 1 (as used by Kivinen et al., 2002) or to $0.5(2+0.5N^{-0.5})^{0.5}$ (as used by Shalev-Shwartz et al., 2011), whichever resulted in higher cross-validation accuracy. The hyper-parameters (kernel width σ, λ for Pegasos and Norma, U for BMP, C for LIBSVM, IDSVM, TVM) were selected by 10 fold cross-validation for each combination of data, algorithm, and budget. We repeated all the experiments five times, where at each run the training examples were shuffled differently. Mean and standard deviation of the accuracies of each set of experiments are reported. For Adult, DNA, IJCNN, Letter, Pendigit, Satimage, Shuttle and USPS data, we used the common training-test split. For other data sets, we randomly selected a fixed number of examples as the test data in each repetition. We trained all online (budgeted and non-budgeted) algorithms using a single pass over the training data. All experiments were run on a 3G RAM, 3.2 GHz Pentium Dual Core. Our proposed algorithms were implemented in MATLAB.

7.4 Experimental Results

The accuracy of different algorithms on test data is reported in Table 3, 4 and 5.[12]

12. For IDSVM, TVM and BPA, only results on the binary data sets are reported since only binary classification versions of these algorithms are available.

Algorithms/Data	Banana (4.3K, 2, 2)	Gauss (10K, 2, 2)	Adult (21K,123,2)	IJCNN (50K, 21, 2)	Checkerb (10M, 2, 2)
Offline:					
LIBSVM *Acc*:	90.70±0.06	81.62±0.40	84.29±0.0	98.72±0.10	99.87±0.02
(#SVs):	(1.1K)	(4.0K)	(8.5K)	(4.9K)	($2.6K_{100K}$)
Online(one pass):					
IDSVM	90.65±0.04	81.67±0.40	83.93±0.03	98.51±0.03	99.40±0.02
	(1.1K)	(4.0K)	($4.0K_{8.3K}$)	($3.6K_{33K}$)	($7.5K_{51K}$)
Pegasos	90.48±0.78	81.54±0.25	84.02±0.14	98.76±0.09	99.35±0.04
	(1.7K)	(6.4K)	(9K)	(16K)	($41K_{916K}$)
Norma	90.23±1.04	81.54±0.06	83.65±0.11	93.41±0.15	99.32±0.09
	(2.1K)	(5.2K)	(10K)	(33K)	($128K_{730K}$)
MP	89.40±0.57	78.45±2.18	82.61±0.61	98.61±0.10	99.43±0.11
	(1K)	(3.4K)	(8K)	(11K)	($22K_{1M}$)
Budgeted(one pass):					
1-st line: $B = 100$:					
2-nd line: $B = 500$:					
TVM	90.03±0.96	**81.56±0.16**	82.77±0.00	***97.20±0.19***	98.90±0.09
	91.13±0.68	***81.39±0.50***	***83.82±0.04***	***98.32±0.14***	**99.94±0.03**
BPA	90.35±0.37	80.75±0.24	83.38±0.56	93.01±0.53	99.01±0.04
	91.30±1.18	**81.67±0.42**	***83.58±0.30***	96.20±0.35	99.70±0.01
Projection++	88.36±1.52	76.06±2.25	77.86±3.45	92.36±1.15	96.92±0.45
	86.76±1.27	75.17±4.02	79.80±2.11	94.73±1.95	98.24±0.34
MP+stop	88.07±1.38	74.10±3.00	80.00±1.61	91.13±0.18	86.39±1.12
	89.77±0.25	79.68±1.19	81.68±0.90	94.60±0.96	95.43±0.43
MP+random	87.54±1.33	75.68±3.68	79.78±0.88	90.22±1.69	84.24±1.39
	88.36±0.99	77.26±1.16	80.40±1.03	91.86±1.39	93.12±0.56
BPegasos+remove	85.63±1.25	79.13±1.40	78.84±0.76	90.73±0.31	83.02±2.12
	89.92±0.66	80.70±0.61	81.67±0.44	93.30±0.57	91.82±0.22
BPegasos+project	***90.21±1.61***	***81.25±0.34***	83.88±0.33	96.48±0.44	97.27±0.72
	90.40±0.47	***81.33±0.40***	***83.84±0.07***	97.52±0.62	98.08±0.27
BPegasos+merge	90.17±0.61	***81.22±0.40***	**84.55±0.17**	**97.27±0.72**	**99.55±0.12**
	89.46±0.81	***81.34±0.38***	**83.93±0.41**	***98.08±0.27***	99.83±0.08
BNorma+merge	**91.53±1.14**	***81.27±0.37***	84.11±0.25	92.69±0.19	99.16±0.23
	90.65±1.28	***81.37±0.25***	***83.80±0.21***	91.35±0.13	99.72±0.05
BMP+merge	89.37±1.31	79.57±0.90	83.34±0.36	***96.67±0.35***	98.24±0.13
	89.46±0.50	79.38±0.82	82.97±0.26	***98.10±0.41***	98.79±0.08

Table 3: Comparison of offline, online, and budgeted online algorithms on 5 benchmark binary classification data sets. Online algorithms (IDSVM, Pegasos, Norma and MP) were early stopped after 10,000 seconds and the number of examples being learned at the time of the early stopping was recorded and shown in the subscript within the #SV parenthesis. LIBSVM was trained on a subset of 100K examples on Checkerboard, Covertype and Waveform due to computational issues. Among the budgeted online algorithms, for each combination of data set and budget, the best accuracy is in bold, while the accuracies that were not significantly worse (with $p > 0.05$ using the one-sided t-test) are in bold and italic.

7.5 Comparison of Non-budgeted Algorithms

On the non-budgeted algorithm side, as expected, the exact SVM solvers LIBSVM and IDSVM have the highest accuracy and are followed by Pegasos, MP and Norma, algorithms trained by a single pass of the training data. The dual-form based LIBSVM and IDSVM have sparser models than the primal-form based

Algorithms/Data	DNA (4.3K,180, 3)	Satimage (4.4K, 36, 6)	USPS (7.3K,256,10)	Pen (7.5K,16, 10)	Letter (16K, 16, 26)
Offline:					
LIBSVM	95.32±0.00 (1.3K)	91.55±0.00 (2.5K)	95.27±0.00 (1.9K)	98.23±0.00 (0.8K)	97.62±0.00 (8.1K)
Online(one pass):					
Pegasos	92.87±0.81 (0.7K)	91.29±0.15 (2.9K)	94.41±0.11 (4.9K)	97.86±0.27 (1.4K)	96.28±0.15 (8.2K)
Norma	86.15±0.67 (2.0K)	90.28±0.35 (4.4K)	93.40±0.33 (6.6K)	95.86±0.27 (7.0K)	95.21±0.09 (15K)
MP	93.36±0.93 (0.8K)	91.23±0.54 (1.6K)	94.37±0.04 (2.2K)	98.02±0.11 (1.9K)	96.41±0.24 (8.2K)
Budgeted(one pass):					
1-st line: $B = 100$:					
2-nd line: $B = 500$:					
Projection++	82.94±3.73	84.47±1.75	81.40±1.26	93.33±0.96	47.23±0.99
	90.11±2.11	88.66±0.66	92.02±0.59	95.78±0.75	75.90±0.76
MP+stop	73.56±7.59	82.34±2.43	79.11±2.15	88.27±1.56	41.89±1.16
	91.23±0.78	88.68±0.60	90.78±0.58	***97.78±0.20***	67.32±1.53
MP+random	73.87±4.93	82.51±1.34	78.06±2.01	87.77±2.96	40.93±2.31
	87.84±4.84	87.25±1.07	90.10±0.97	**97.20±0.68**	68.23±1.14
BPegasos+remove	78.63±2.03	81.09±3.21	80.16±1.15	91.84±1.27	41.50±1.49
	91.48±1.65	86.77±1.01	89.44±1.05	97.6±0.21	71.97±1.04
BPegasos+ project	86.53±2.03	***87.69±0.62***	89.67±0.42	96.19±0.85	74.49±1.89
	92.26±1.20	88.86±0.2	92.61±0.32	***97.58±0.49***	87.85±0.49
BPegasos+merge	***93.13±1.49***	***87.53±0.72***	***91.76±0.24***	97.06±0.19	***73.63±1.72***
	92.42±1.24	89.77±0.14	***92.91±0.19***	***97.63±0.14***	**89.68±0.61**
BNorma+merge	75.72±0.25	85.61±0.54	87.44±0.45	90.82±0.42	61.79±1.58
	76.25±3.27	86.33±0.40	89.51±0.24	94.60±0.22	75.84±0.35
BMP+merge	**93.76±0.31**	**88.33±0.90**	**92.31±0.57**	**97.35±0.16**	**74.99±1.08**
	93.84±0.64	**90.41±0.22**	**93.10±0.36**	**97.86±0.33**	88.22±0.36

Table 4: Comparison of offline, online, and budgeted online algorithms on 5 benchmark multi-class data sets

Pegasos, MP, and Norma. Pegasos and MP achieve similar accuracy on most data sets, while Pegasos significantly outperforms MP on the two noisy data sets Gauss and Waveform. Norma is generally less accurate than Pegasos and MP, and the gap is larger on IJCNN, Checkerboard, DNA, and Covertype. Additionally, Norma generates more SVs than its two siblings.

7.6 Comparison of Budgeted Algorithms

On the budgeted side, BPegasos+merge and BMP+merge are the two most accurate algorithms and their accuracies are comparable on most data sets. Considering that BPegasos+merge largely outperforms BMP+merge on Phoneme and Covertype and also the additional computational cost of the projection step in BMP, BPegasos+merge is clearly the winner of this category. The accuracy of BPegasos+project is highly competitive to the above two algorithms, but we should note that projection is costlier than merging. Accuracy of TVM and BPA is comparable to BPegasos+merge on the binary data sets (with exception of the lower BPA accuracy on IJCNN). Accuracies of Projectron++, MP+stop, and MP+random are significantly lower. In this subgroup, Projectron++ is the most successful, showing the benefits of projecting as compared to removal. Consistent with this result, BSGD algorithms using removal fared significantly worse than those using projection and merging.

Algorithms/Data	Shuttle (43K, 9, 2)	Phoneme (84K,41,48)	Covertype (0.5M, 54, 7)	Waveform (2M, 21, 3)
Offline:				
LIBSVM	99.90±0.00 (0.3K)	78.24±0.05 (69K)	89.69±0.15 ($36K_{100K}$)	85.83±0.06 ($32K_{100K}$)
Online(one pass):				
Pegasos	99.90±0.00 (1.2K)	79.62±0.16 ($80K_{80K}$)	87.73±0.31 ($47K_{136K}$)	86.50±0.10 ($74K_{192K}$)
Norma	99.79±0.01 (8K)	79.86±0.09 (84K)	82.80±0.33 ($92K_{92K}$)	86.29±0.15 ($111K_{189K}$)
MP	99.89±0.02 ($0.4K_{44K}$)	79.80±0.12 ($78K_{78K}$)	88.84±0.06 ($56K_{160K}$)	84.36±0.36 ($83K_{310K}$)
Budgeted(one pass):				
1-st line: $B = 100$:				
2-nd line: $B = 500$:				
Projection++	99.55±0.16	21.20±1.24	62.54±3.14	80.75±0.81
	99.85±0.08	32.32±1.97	67.32±2.93	83.56±0.54
MP+stop	99.39±0.35	24.86±2.10	56.96±1.59	81.04±2.61
	99.90±0.01	33.76±1.01	61.93±1.56	83.76±0.71
MP+random	98.67±0.07	23.29±1.39	55.56±1.37	79.94±1.12
	99.90±0.01	31.37±1.91	60.47±1.70	81.61±1.51
BPegasos+remove	99.26±0.54	24.39±1.48	55.64±1.82	78.43±1.79
	99.89±0.02	32.10±0.85	62.97±0.55	84.38±0.53
BPegasos+ project	**99.81±0.05**	43.60±0.10	70.84±0.59	85.63±0.07
	99.89±0.02	48.87±0.07	74.94±0.22	86.18±0.06
BPegasos+merge	99.63±0.02	**46.49±0.78**	**74.10±0.30**	**86.71±0.38**
	99.89±0.02	**51.57±0.30**	**76.89±0.51**	**86.63±0.28**
BNorma+merge	99.48±0.01	39.66±0.66	71.54±0.53	***86.60±0.12***
	99.80±0.01	45.13±0.43	72.81±0.46	82.03±0.53
BMP+merge	98.99±0.55	42.18±1.94	67.28±3.86	86.02±0.22
	99.91±0.01	47.02±0.98	72.31±0.75	86.03±0.17

Table 5: Comparison of offline, online, and budgeted online algorithms on 4 benchmark multi-class data sets

7.7 Best Budgeted Algorithm vs Non-budgeted Algorithms

Comparing the best budgeted algorithm BPegasos+merge with modest budgets of $B = 100$ and 500 with the non-budgeted Pegasos and LIBSVM, we can see that it achieves very competitive accuracy. Interestingly, its accuracy is even larger than the two non-budgeted algorithms on two largest data sets Checkerboard and Waveform. This indicates noise reduction capability of SV merging. This result is even more significant as BPegasos+merge has faster training time and learns a much smaller model. On Covertype, Phoneme and Letter data, the accuracy gap between budget $B = 500$ and non-budgeted algorithms remained large and it can be explained by the complexity of these problems; for example, 30% of Covertype examples, 50% of Letter examples, and 100% of Phoneme examples became SVs in Pegasos. In addition, Letter had 26 class labels and Phoneme 48. In all 3 data sets, the accuracy clearly improved from $B = 100$ to 500, which indicates that extra budget is needed for comparable accuracy. To better illustrate the importance of budget size on some data sets, Figure 2 shows that on Letter and Covertype, the accuracy of BPegasos+merge approaches that of Pegasos as the larger budget is used. Interestingly, while 16K examples appear to be sufficient for convergence on Letter data set, it is evident that Covertype could benefit from a much larger budget than the available half million labeled examples.

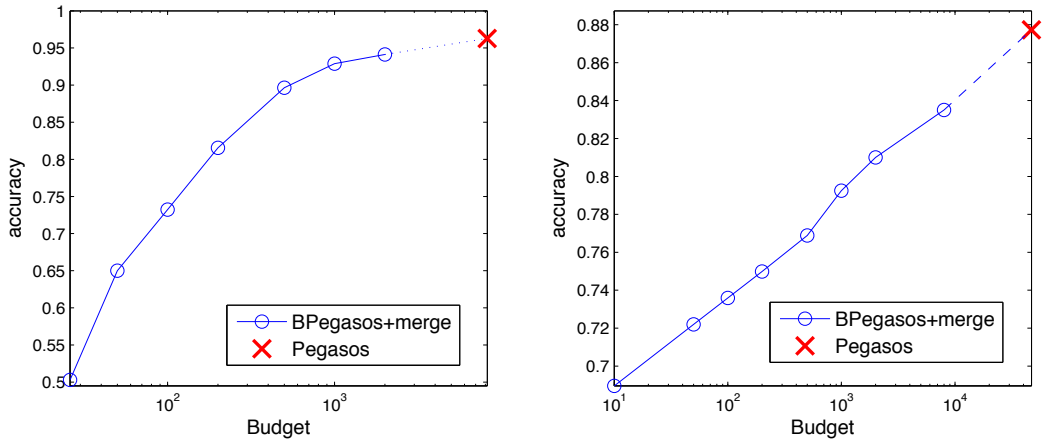


Figure 2: The accuracy of BPegasos as a function of the budget size

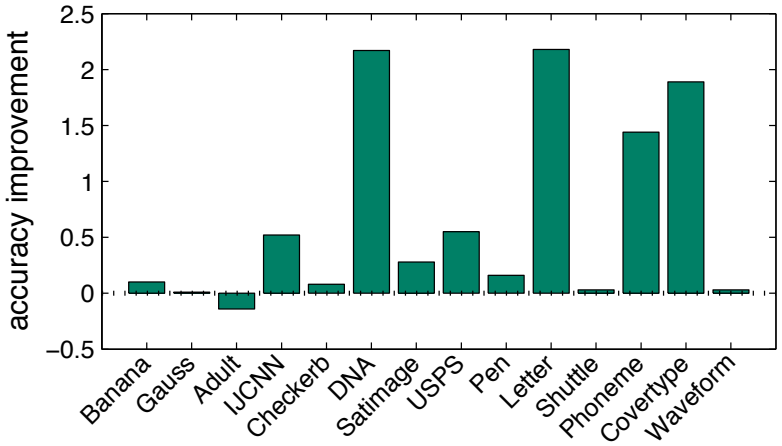


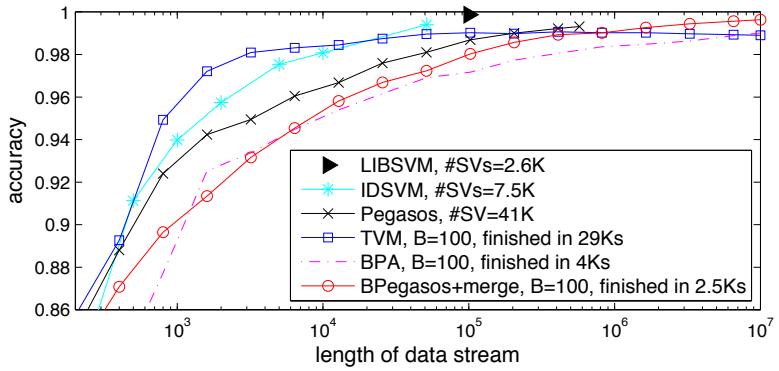
Figure 3: BPegasos+merge ($B = 500$): difference in accuracy between a model trained with 5-passes and a model trained with a single-pass of the data.

7.8 Multi-epoch vs Single-pass Training

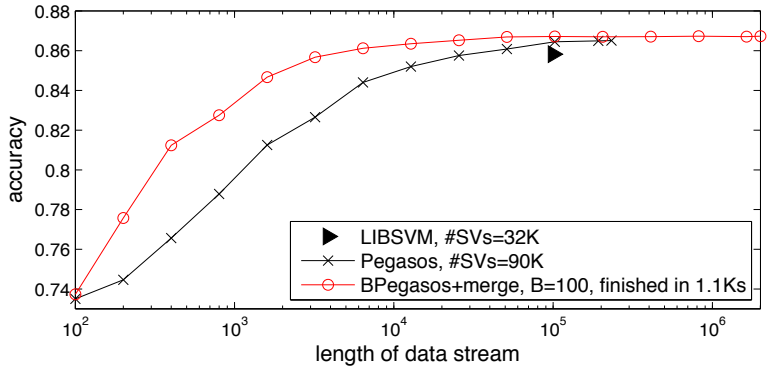
For the most accurate budgeted online algorithm BPegasos+merge, we also report its accuracy after allowing it to make 5 passes through training data. In this scenario, BPegasos should be treated as an offline algorithm. The accuracy improvement as compared to the single-pass version is reported in Figure 3. We can observe that multi-epoch training improves the accuracy of BPegasos on most data sets. This result suggests that, if the training time is not of major concern, multiple accesses to the training data should be used.

7.9 Accuracy Evolution Curves

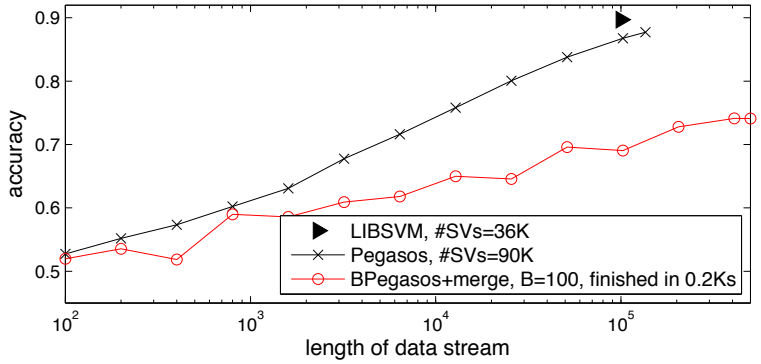
In Figure 4 we show evolution of accuracy as a function of the number of observed examples on the three largest data sets. By comparing BPegasos+merge with non-budgeted SVMs and several other budgeted algorithms from Tables 3, 4, and 5, we observe that the accuracy of BPegasos+merge consistently increases with



(a) Checkerboard



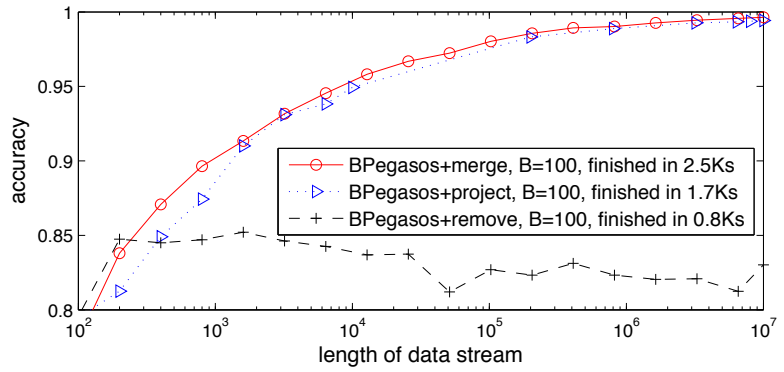
(b) Waveform



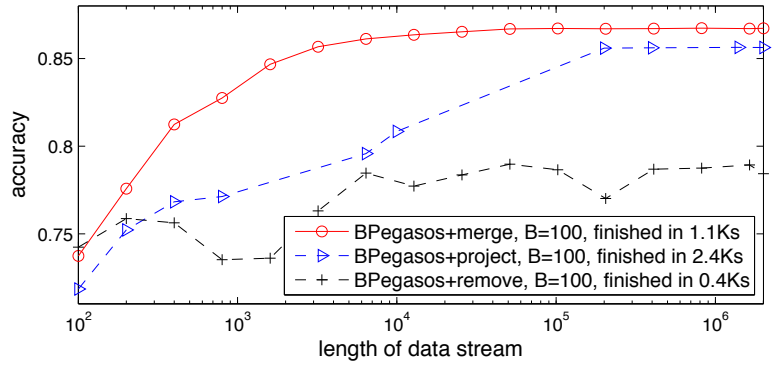
(c) Covertype

Figure 4: Comparison of accuracy evolution curves

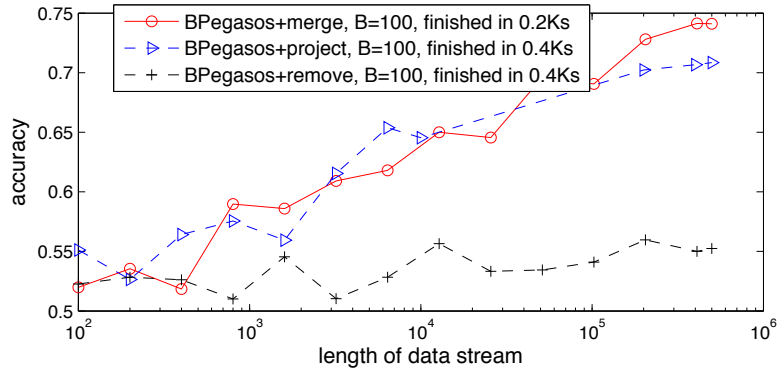
data stream size. On Checkerboard, its accuracy closely follows Pegasos and eventually surpasses it after Pegasos had to be early stopped. IDSVM and its budgeted version TVM exhibit faster accuracy growth initially, but are surpassed by BPegasos+merge as more training examples become available. On waveform, the accuracy of BPegasos grows faster than the original non-budgeted version. This behavior can be attributed to the noise-reduction property of merging. Finally, on Covertype, BPegasos+merge significantly trails its non-budgeted cousin, and this behavior is consistent with Figure 2.b.



(a) Checkerboard



(b) Waveform



(c) Covertype

Figure 5: Accuracy evolution curves of BPegasos for different budget maintenance strategies

In Figure 5 we compare evolution of accuracy of BPegasos for 3 proposed budget maintenance strategies. As could be seen, removal is inferior to projection and merging on all 3 data sets. On Checkerboard, removal even causes a gradual drop in accuracy after an initial moderate increase, while on the other two sets the accuracy fluctuates around a very small value close to that achieved by training on 100 initial examples. On the other hand, projection and merging result in strong and consistent accuracy increase with data stream size. Interestingly, on Waveform data, merging significantly outperforms projection, which may point to its robustness to noise.

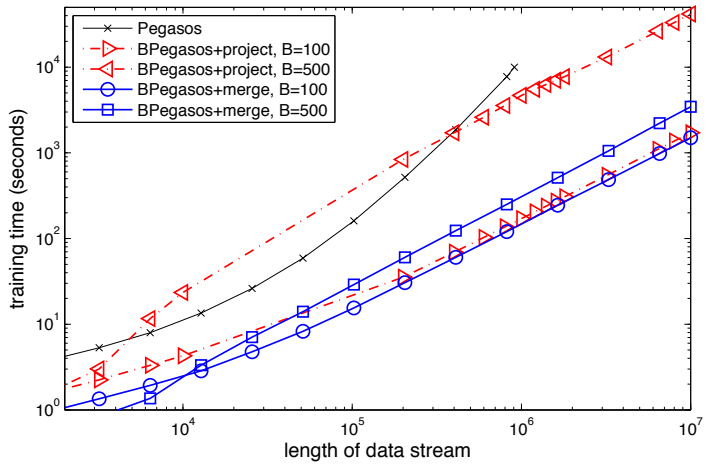


Figure 6: Training time curves on Checkerboard data

Legends in both Figures 4 and 5 list total training time of budgeted algorithms at the end of the data stream (non-budgeted algorithms were early stopped after 10K seconds). Considering that our implementation of all algorithms except LIBSVM was in Matlab and on a 3GB RAM, 3.2GHz Pentium Dual Core 2 PC, Figure 4 indicates a rather impressive speed of the budgeted algorithms. From Figure 5, it can be seen that merging and projection are very fast and are comparable to removal.

7.10 Training Time Scalability

Figure 6 presents log-log plot of the training time versus the data stream length on Checkerboard data set with 10 million examples. Excluding the initial stage, Pegasos had the fastest increase in training time, confirming the expected $O(N^2)$ runtime. On the budgeted side, the runtime time of BPegasos with merging and projecting increases linearly with data size. However, it is evident that BPegasos with projecting grow much faster with the budget size than costs of BPegasos with merging. This confirms the expected $O(B)$ scaling of the merging and $O(B^2)$ scaling of the projection version.

7.11 Weight Degradation

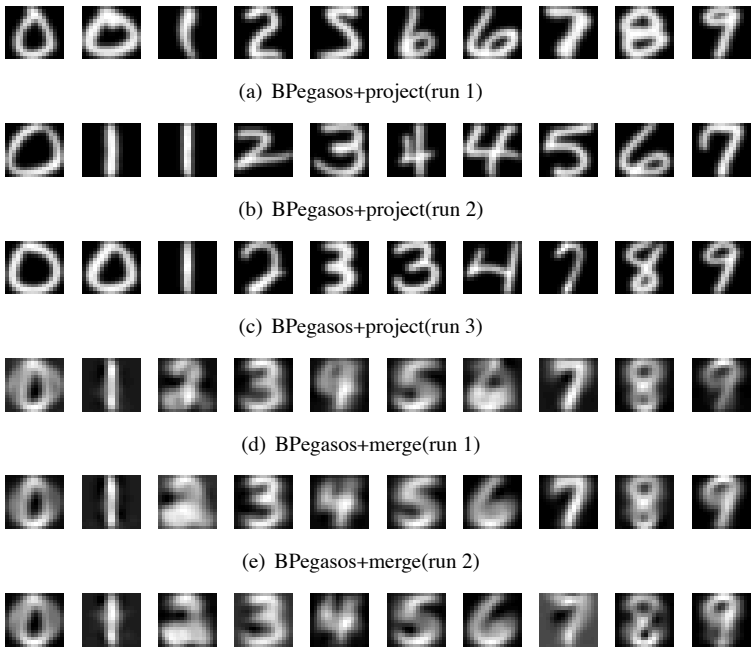
Theorems 1, 2, and 3 indicate that lower E leads to lower gap between the optimal solution and the budgeted solution. We also argued that E decreases with budget size through three mechanisms. In Table 6, we show how the value of E on Checkerboard data is being influenced by the budget B and, in turn, how the change in E influences accuracy. From the comparison of three strategies for two B values (100 and 500), we see as B gets larger, E is getting smaller. The results also show that projection and merging achieve significantly lower value than removal and that lower E indeed results in higher accuracy.

7.12 Merged vs Projected SVs

In order to gain further insight into the projection and merging-based budget maintenance, in Figure 7 we compare final SVs generated by these two strategies on the USPS data where classes are 10 digits. We used budget $B = 10$ for BPegasos to explore how successful the algorithm was in revealing the 10 digits. Comparing Figures 7.a-c and 7.d-e we can observe that SVs generated by 3 different runs of BPegasos+project did not represent all 10 digits (e.g., in Run 1, digits 0 and 6 appear twice, while digits 3 and 4 are not represented). It should be noted that the 10 SVs obtained using projection are identical to 10 actual training examples of USPS. On the other hand, SVs obtained by merging in all 3 runs represent all 10 digits. The appearance of

	$B = 100$		$B = 500$	
	E	Acc	E	Acc
BPegasos+remove	1.402±0.000	79.19±3.05	1.401±0.000	90.32±0.40
BPegasos+project	0.052±0.007	99.25±0.06	0.007±0.001	99.66±0.10
BPegasos+merge	0.037±0.006	99.55±0.14	0.002±0.001	99.74±0.08

Table 6: Comparison of accuracy and averaged weight degradation for three versions of BPegasos as a function of budget size B on 10M Checkerboard examples, using same parameters ($\lambda = 10^{-4}$, kernel width $\sigma = 0.0625$).



(a) BPegasos+project(run 1)

(b) BPegasos+project(run 2)

(c) BPegasos+project(run 3)

(d) BPegasos+merge(run 1)

(e) BPegasos+merge(run 2)

(f) BPegasos+merge(run 3)

Figure 7: The plot of SVs on USPS data. (Each row corresponds to a different run. 3 different runs using BPegasos+project ($B = 10$) with average accuracy 74%; 3 different runs using BPegasos+merge ($B = 10$) with average accuracy 86%.)

each SV in Figure 7.d-e is blurred and is a result of many mergings of the original labeled examples. This example is a useful illustration of the main difference between projection and merging, and it can be helpful in selecting the appropriate budget maintenance strategy for a particular learning task.

8. Conclusion

We proposed a framework for large-scale kernel SVM training using BSGD algorithms. We showed that budgeted versions of three popular online algorithms, Pegasos, Norma, and Margin Perceptron, can be studied under this framework. We obtained theoretical bounds on their performance that indicate that decrease in BSGD accuracy is closely related to the model degradation due to budget maintenance. Based on the analysis, we studied budget maintenance strategies based on removal, projection, and merging. We experimentally evaluated the proposed BSGD algorithms in terms of accuracy, and training time and compared them with a

number of offline and online, as well as non-budgeted, and budgeted alternatives. The results indicate that highly accurate and compact kernel SVM classifiers can be trained on high-throughput data streams. Particularly, the results show that merging is a highly attractive budget maintenance strategy for BSGD algorithms as it results in relative accurate classifiers while achieving linear training time scaling with support vector budget and data size.

Acknowledgments

This work was supported by the U.S. National Science Foundation Grant IIS-0546155. Koby Crammer is a Horev Fellow, supported by the Taub Foundations.

Appendix A. Proof of Theorem 1

We start by showing the following technical lemma.
Lemma 1

- *Let P_t be as defined in (3).*
- *Let C be a closed convex set with radius U.*
- *Let $\mathbf{w}_1, ..., \mathbf{w}_N$ be a sequence of vectors such that $\mathbf{w}_1 \in C$ and for any $t > 1$ $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \eta_t \nabla_t - \Delta_t)$, ∇_t is the (sub)gradient of P_t at $\mathbf{w}_t$, η_t is a learning rate function, Δ_t is a vector, and $\prod_C(\mathbf{w}) = \arg\min_{\mathbf{w}' \in C} ||\mathbf{w}' - \mathbf{w}||$, is a projection operation that projects $\mathbf{w}$ to C.*
- *Assume $||E_t|| \leq 1$.*
- *Define $D_t = ||\mathbf{w}_t - \mathbf{u}||^2 - ||\mathbf{w}_{t+1} - \mathbf{u}||^2$ as the relative progress toward $\mathbf{u}$ at t-th round.*

Then, the following inequality holds for any $\mathbf{u} \in C$

$$\frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{w}_t) - \frac{1}{N}\sum_{t=1}^{N} P_t(\mathbf{u}) \leq \frac{1}{N}\left(\sum_{t=1}^{N}\frac{D_t}{2\eta_t} - \sum_{t=1}^{N}\frac{\lambda}{2}||\mathbf{w}_t - \mathbf{u}||^2 + \frac{(\lambda U + 2)^2}{2}\sum_{t=1}^{N}\eta_t\right) + 2UE. \tag{20}$$

■

Proof of Lemma 1. First, we rewrite $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \eta_t \nabla_t - \Delta_t)$ by treating Δ_t as the source of error in the gradient $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \eta_t \partial_t)$, where we defined $\partial_t = \nabla_t + E_t$. Then, we lower bound D_t as

$$\begin{aligned} D_t &= ||\mathbf{w}_t - \mathbf{u}||^2 - ||\textstyle\prod_C(\mathbf{w}_t - \eta_t \partial_t) - \mathbf{u}||^2 \\ &\geq_1 ||\mathbf{w}_t - \mathbf{u}||^2 - ||\mathbf{w}_t - \eta_t \partial_t - \mathbf{u}||^2 \\ &= -\eta_t^2 ||\partial_t||^2 + 2\eta_t \nabla_t^T(\mathbf{w}_t - \mathbf{u}) + 2\eta_t E_t^T(\mathbf{w}_t - \mathbf{u}) \\ &\geq_2 -\eta_t^2(\lambda U + 1 + 1)^2 + 2\eta_t\left(P_t(\mathbf{w}_t) - P_t(\mathbf{u}) + \tfrac{\lambda}{2}||\mathbf{w}_t - \mathbf{u}||^2\right) - 4\eta_t ||E_t|| U. \end{aligned} \tag{21}$$

In $\geq_1$, we use the fact that since C is convex, $||\prod_C(a) - b|| \leq ||a - b||$ for all $b \in C$ and a. In $\geq_2$, $||\partial_t||$ is bounded as

$$||\partial_t|| \leq ||\lambda \mathbf{w}_t + y_t \Phi(\mathbf{x}_t)|| + ||E_t|| \leq \lambda U + 1 + 1,$$

and, by applying the property of strong convexity, it follows

$$\nabla_t^T(\mathbf{w}_t - \mathbf{u}) \geq P_t(\mathbf{w}_t) - P_t(\mathbf{u}) + \lambda ||\mathbf{w}_t - \mathbf{u}||^2/2,$$

since P_t is λ-strongly convex function w.r.t. $||\mathbf{w}||^2/2$ and ∇_t is the subgradient of $P_t(\mathbf{w})$ at $\mathbf{w}_t$ (according to Lemma 1 by Shalev-Shwartz and Singer (2007). Bound $||\mathbf{w}_t - \mathbf{u}|| \leq 2U$ holds since both $||\mathbf{w}_t||$ and $\mathbf{u}$ are upper bounded by U.

Dividing both sides of inequality (21) by $2\eta_t$ and rearranging, we obtain

$$P_t(\mathbf{w}_t) - P_t(\mathbf{u}) \leq \frac{D_t}{2\eta_t} - \frac{\lambda}{2}||\mathbf{w}_t - \mathbf{u}||^2 + \frac{\eta_t(\lambda U + 1 + 1)^2}{2} + 2||E_t||U. \tag{22}$$

Summing over all t in (22) and dividing two sides of inequality by N leads to the stated bound.

Proof of Theorem 1. $\mathbf{w}_{t+1}$ is bounded as

$$\begin{aligned} ||\mathbf{w}_{t+1}|| &= ||(1-\eta_t\lambda)\mathbf{w}_t + \eta_t y_t \Phi(\mathbf{x}_t) - \Delta_t|| \\ &\leq ||(1-\eta_t\lambda)\mathbf{w}_t|| + \eta_t(1 + ||E_t||) \leq_3 2/\lambda. \end{aligned}$$

In $\leq_3$ we used the definition of η_t and recursively bounded $||\mathbf{w}_t||$.

Using the fact that $||\mathbf{w}^*|| \leq 1/\sqrt{\lambda}$ (Shalev-Shwartz et al., 2011), both $||\mathbf{w}_t||$ and $||\mathbf{w}^*||$ can be bounded by constant U defined in (6). Thus the update rule $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \eta_t\nabla_t - \Delta_t)$ in Lemma 1 is reduced to $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t\nabla_t - \Delta_t$.

Plugging $\eta_t \equiv 1/(\lambda t)$ into RHS of inequality (20) in Lemma 1 and replacing $\mathbf{u}$ with $||\mathbf{w}^*||$, the first and second term in the parenthesis on the RHS are bounded as

$$\begin{aligned} &\sum_{t=1}^{N} \tfrac{D_t}{2\eta_t} - \sum_{t=1}^{N} \tfrac{\lambda}{2}||\mathbf{w}_t - \mathbf{w}^*||^2 \\ &\leq \tfrac{1}{2}\left((\tfrac{1}{\eta_1} - \lambda)||\mathbf{w}_1 - \mathbf{w}^*||^2 + \sum_{t=2}^{N} (\tfrac{1}{\eta_t} - \tfrac{1}{\eta_{t-1}} - \lambda)||\mathbf{w}_t - \mathbf{w}^*||^2 - \tfrac{1}{\eta_N}||\mathbf{w}_{N+1} - \mathbf{w}^*||^2 \right) \\ &= -\tfrac{1}{2\eta_N}||\mathbf{w}_{N+1} - \mathbf{w}^*||^2 \leq 0. \end{aligned} \tag{23}$$

According to the divergence rate of harmonic series, the third term on the RHS in (20) is bounded by,

$$\frac{(\lambda U + 2)^2}{2}\sum_{t=1}^{N}\eta_t = \frac{(\lambda U + 2)^2}{2\lambda}\sum_{t=1}^{N}\frac{1}{t} \leq \frac{(\lambda U + 2)^2}{2\lambda}(\ln(N) + 1). \tag{24}$$

With bounded U value, combining (23), (24) with (20) leads to (7).

Appendix B. Proof of Theorem 2

$\mathbf{w}_{t+1}$ is bounded as

$$\begin{aligned} ||\mathbf{w}_{t+1}|| &= ||(1-\eta_t\lambda)\mathbf{w}_t + \eta_t y_t \Phi(\mathbf{x}_t) - \Delta_t|| \\ &\leq ||(1-\eta_t\lambda)\mathbf{w}_t|| + \eta_t(1 + ||E_t||) \\ &\leq ||(1-\eta_t\lambda)\mathbf{w}_t|| + 2\eta_t. \end{aligned}$$

Since $||\mathbf{w}_1|| = 0$, the bound $||\mathbf{w}_{t+1}|| \leq 2/\lambda$ holds for all t (Kivinen et al., 2002). Using the fact that $||\mathbf{w}^*|| \leq 1/\sqrt{\lambda}$ (Shalev-Shwartz, Singer & Srebro, 2011), both $||\mathbf{w}_t||$ and $||\mathbf{w}^*||$ are bounded by constant U defined in (6). Thus the update rule $\mathbf{w}_{t+1} \leftarrow \prod_C(\mathbf{w}_t - \eta_t\nabla_t - \Delta_t)$ in Lemma 1 is reduced to $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t\nabla_t - \Delta_t$.

Replacing $\mathbf{u}$ by $\mathbf{w}^*$ in (20), the first term at RHS in (20) can be bounded as

$$\begin{aligned} &\sum_{t=1}^{N} \tfrac{D_t}{2\eta_t} = \sum_{t=1}^{N} \tfrac{1}{2\eta_t}(||\mathbf{w}_t - \mathbf{w}^*||^2 - ||\mathbf{w}_{t+1} - \mathbf{w}^*||^2) \\ &= \tfrac{1}{2\eta_1}||\mathbf{w}_1 - \mathbf{w}^*||^2 + \tfrac{1}{2}\sum_{t=2}^{N}(\tfrac{1}{\eta_t} - \tfrac{1}{\eta_{t-1}})||\mathbf{w}_t - \mathbf{w}^*||^2 - \tfrac{1}{2\eta_N}||\mathbf{w}_{N+1} - \mathbf{w}^*||^2 \\ &\leq \tfrac{4U^2}{2}\left(\tfrac{1}{\eta_1} + \sum_{t=2}^{N}(\tfrac{1}{\eta_t} - \tfrac{1}{\eta_{t-1}}) \right) = \tfrac{2U^2}{\eta_N} = \tfrac{2U^2\sqrt{N}}{\eta}. \end{aligned} \tag{25}$$

The third term at RHS in (20) can be bounded according to the series divergence rate as

$$\frac{(\lambda U + 2)^2}{2}\sum_{t=1}^{N}\eta_t \leq \frac{\eta(\lambda U + 2)^2}{2}(2\sqrt{N} - 1). \tag{26}$$

With bounded U value, combing (25), (26) with (20) and bounding the negative terms by zero lead to (8).

Appendix C. Proof of Theorem 3

Replacing $\mathbf{u}$ by $\mathbf{w}^*$ in (20), the first term at RHS in (20) can be bounded as

$$\begin{aligned}
&\sum_{t=1}^{N} \frac{D_t}{2\eta_N} = \sum_{t=1}^{N} \frac{1}{2\eta_t} \left(||\mathbf{w}_t - \mathbf{w}^*||^2 - ||\mathbf{w}_{t+1} - \mathbf{w}^*||^2 \right) \\
&= \frac{1}{2\eta_1} ||\mathbf{w}_1 - \mathbf{w}^*||^2 + \frac{1}{2} \sum_{t=2}^{N} (\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}) ||\mathbf{w}_t - \mathbf{w}^*||^2 - \frac{1}{2\eta_N} ||\mathbf{w}_{N+1} - \mathbf{w}^*||^2 \\
&= \frac{1}{2\eta} ||\mathbf{w}_1 - \mathbf{w}^*||^2 - \frac{1}{2\eta} ||\mathbf{w}_{N+1} - \mathbf{w}^*||^2 \leq \frac{U^2}{2\eta},
\end{aligned}$$

using the fact $\eta_t = \eta$ and $\lambda = 0$. Bounding the second term in RHS of (20) by zero (since it is always negative), Lemma 1 directly leads to (9).

References

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers for online and active learning. *Journal of Machine Learning Research*, 2005.

A. Bordes, L. Bottou, and P. Gallinari. Sgd-qn: careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 2009.

C. J. C. Burges. Simplified support vector decision rules. In *Advances in Neural Information Processing Systems*, 1996.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *Advances in Neural Information Processing Systems*, 2000.

N. Cesa-Bianchi and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. In *Annual Conference on Learning Theory*, 2006.

C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines, `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 2001.

F. Chang, C.-Y. Guo, X.-R. Lin, and C.-J. Lu. Tree decomposition for large-scale svm problems. *Journal of Machine Learning Research*, 2010a.

Y.-W. Chang, C.-J. Hsie, K.-W. Chang, M. Ringgaard, and C.-J. Lin. Training and testing low-degree polynomial data mappings via linear svm. *Journal of Machine Learning Research*, 2010b.

L. Cheng, S. V. N. Vishwanathan, D. Schuurmans, S. Wang, and T. Caelli. Implicit online earning with kernels. In *Advances in Neural Information Processing Systems*, 2007.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *International Conference on Machine Learning*, 2006.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 1995.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2001.

K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 2003.

K. Crammer, J. Kandola, and Y. Singer. Online classification on a budget. In *Advances in Neural Information Processing Systems*, 2004.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 2006.

L. Csató and M. Opper. Sparse representation for gaussian process models. In *Advances in Neural Information Processing Systems*, 2001.

O. Dekel and Y. Singer. Support vector machines on a budget. In *Advances in Neural Information Processing Systems*, 2006.

O. Dekel, S. Shalev-Shwartz, and Y. Singer. The forgetron: a kernel-based perceptron on a budget. *SIAM Journal on Computing*, 2008.

R. Duda and P. Hart. *Pattern Classification and Scene Analysis. New York: Wiley*, 1973.

Y. Engel, S. Mannor, and R. Meir. Sparse online greedy support vector regression. In *European Conference on Machine Learning*, 2002.

C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2001.

H.-P. Graf, E. Cosatto, L. Bottou, I. Dourdanovic, and V. Vapnik. Parallel support vector machines: the cascade svm. In *Advances in Neural Information Processing Systems*, 2005.

C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *International Conference on Machine Learning*, 2008.

T. Joachims. Training linear svms in linear time. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2006.

S. S. Keerthi, O. Chapelle, , and D. DeCoste. Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, 2006.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 2002.

Y.-J. Lee and O. L. Mangasarian. Rsvm: reduced support vector machines. In *SIAM Conference on Data Mining*, 2001.

B. Li, M. Chi, J. Fan, , and X. Xue. Support cluster machine. In *International Conference on Machine Learning*, 2007.

Y. Li and P. Long. The relaxed online maximum margin algorithm. *Machine Learning*, 2002.

D. Nguyen and T. Ho. An efficient method for simplifying support vector machines. In *International Conference on Machine Learning*, 2005.

F. Orabona, J. Keshet, and B. Caputo. Bounded kernel-based online learning. *Journal of Machine Learning Research*, 2009.

J. Platt. Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods - Support Vector Learning, MIT Press*, 1998.

A. Rahimi and B. Rahimi. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2007.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. Müller, G. Rätsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 1999.

S. Shalev-Shwartz and Y. Singer. Logarithmic regret algorithms for strongly convex repeated games (technical report). The Hebrew University, 2007.

S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: primal estimated sub-gradient solver for svm. *Mathematical Programming*, 2011.

S. Sonnenburg and V. Franc. Coffin: a computational framework for linear svms. In *International Conference on Machine Learning*, 2010.

I. Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 2003.

I. Sutskever. A simpler unified analysis of budget perceptrons. In *International Conference on Machine Learning*, 2009.

C.H. Teo, S. V. N. Vishwanathan, A. J. Smola, and Q. V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 2010.

I. W. Tsang, J. T. Kwok, and P.-M. Cheung. Core vector machines: fast svm training on very large data sets. *Journal of Machine Learning Research*, 2005.

I. W. Tsang, A. Kocsor, and J. T. Kwok. Simpler core vector machines with enclosing balls. In *International Conference on Machine Learning*, 2007.

S. V. N. Vishwanathan, A. J. Smola, and M. N.Murty. Simplesvm. In *International Conference on Machine Learning*, 2003.

Z. Wang and S. Vucetic. Tighter perceptron with improved dual use of cached data for model representation and validation. In *International Joint Conference on Neutral Netweok*, 2009.

Z. Wang and S. Vucetic. Online passive-aggressive algorithms on a budget. In *International Conference on Artificial Intelligence and Statistics*, 2010a.

Z. Wang and S. Vucetic. Online training on a budget of support vector machines using twin prototypes. *Statisitcal Analysis and Data Mining Journal*, 2010b.

Z. Wang, N. Djuric, K. Crammer, and S. Vucetic. Trading representability for scalability: adaptive multi-hyperplane machine for nonlinear classification. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2011.

J. Weston, A. Bordes, and L. Bottou. Online (and offline) on an even tighter budget. In *International Workshop on Artificial Intelligence and Statistics*, 2005.

M.-R. Wu, B. Schölkopf, , and G. Barik. Building sparse large margin classifiers. In *International Conference on Machine Learning*, 2005.

H.-F. Yu, C.-J. Hsieh, K.-W. Chang, and C.-J. Lin. Large linear classification when data cannot fit in memory. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2010.

K. Zhang, L. Lan, Z. Wang, and F. Moerchen. Scaling up kernel svm on limited resources: a low-rank linearization approach. In *International Conference on Artificial Intelligence and Statistics*, 2012.

T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent. In *International Conference on Machine Learning*, 2004.

Z. A. Zhu, W. Chen, G. Wang, C. Zhu, and Z. Chen. P-packsvm: parallel primal gradient descent kernel svm. In *IEEE International Conference on Data Mining*, 2009.

M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning*, 2003.

Journal of Machine Learning Research 13 (2012) 3133-3176 Submitted 1/12; Revised 5/12; Published 10/12

Bayesian Mixed-Effects Inference on Classification Performance in Hierarchical Data Sets

Kay H. Brodersen[*] BRODERSEN@BIOMED.EE.ETHZ.CH
Christoph Mathys[†] CHMATHYS@ETHZ.CH
Translational Neuromodeling Unit (TNU)
Institute for Biomedical Engineering
ETH Zurich & University of Zurich
Wilfriedstrasse 6, 8032 Zurich, Switzerland

Justin R. Chumbley JUSTIN.CHUMBLEY@ECON.UZH.CH
Laboratory for Social and Neural Systems Research (SNS)
Department of Economics
University of Zurich
Bluemlisalpstrasse 10, 8006 Zurich, Switzerland

Jean Daunizeau[‡] J.DAUNIZEAU@FIL.ION.UCL.AC.UK
Translational Neuromodeling Unit (TNU)
Institute for Biomedical Engineering
ETH Zurich & University of Zurich
Wilfriedstrasse 6, 8032 Zurich, Switzerland

Cheng Soon Ong CHENGSOON.ONG@UNIMELB.EDU.AU
Joachim M. Buhmann JBUHMANN@INF.ETHZ.CH
Machine Learning Laboratory
Department of Computer Science
ETH Zurich
Universitaetstrasse 6, 8092 Zurich, Switzerland

Klaas E. Stephan[§] STEPHAN@BIOMED.EE.ETHZ.CH
Translational Neuromodeling Unit (TNU)
Institute for Biomedical Engineering
ETH Zurich & University of Zurich
Wilfriedstrasse 6, 8032 Zurich, Switzerland

∗. Additional affiliations: Machine Learning Laboratory, Department of Computer Science, ETH Zurich, Universitaetstrasse 6, 8092 Zurich, Switzerland; and Laboratory for Social and Neural Systems Research (SNS), Department of Economics, University of Zurich, Bluemlisalpstrasse 10, 8006 Zurich, Switzerland.

†. Additional affiliation: Laboratory for Social and Neural Systems Research (SNS), Department of Economics, University of Zurich, Bluemlisalpstrasse 10, 8006 Zurich, Switzerland.

‡. Additional affiliation: Laboratory for Social and Neural Systems Research (SNS), Department of Economics, University of Zurich, Bluemlisalpstrasse 10, 8006 Zurich, Switzerland.

§. Additional affiliations: Laboratory for Social and Neural Systems Research (SNS), Department of Economics, University of Zurich, Bluemlisalpstrasse 10, 8006 Zurich, Switzerland; Wellcome Trust Centre for Neuroimaging, University College London, 12 Queen Square, London, WC1N 3BG, United Kingdom.

Abstract

Classification algorithms are frequently used on data with a natural hierarchical structure. For instance, classifiers are often trained and tested on trial-wise measurements, separately for each subject within a group. One important question is how classification outcomes observed in individual subjects can be generalized to the population from which the group was sampled. To address this question, this paper introduces novel statistical models that are guided by three desiderata. First, all models explicitly respect the hierarchical nature of the data, that is, they are mixed-effects models that simultaneously account for within-subjects (fixed-effects) and across-subjects (random-effects) variance components. Second, maximum-likelihood estimation is replaced by full Bayesian inference in order to enable natural regularization of the estimation problem and to afford conclusions in terms of posterior probability statements. Third, inference on classification accuracy is complemented by inference on the balanced accuracy, which avoids inflated accuracy estimates for imbalanced data sets. We introduce hierarchical models that satisfy these criteria and demonstrate their advantages over conventional methods using MCMC implementations for model inversion and model selection on both synthetic and empirical data. We envisage that our approach will improve the sensitivity and validity of statistical inference in future hierarchical classification studies.

Keywords: beta-binomial, normal-binomial, balanced accuracy, Bayesian inference, group studies

1. Introduction

Classification algorithms are frequently applied to data whose underlying structure is hierarchical. One example is the domain of brain-machine interfaces, where classifiers are used to decode intended actions from trial-wise measurements of neuronal activity in individual subjects (Sitaram et al., 2008). Another example is spam detection, where a classifier is trained separately for each user to predict content classes from high-dimensional document signatures (Cormack, 2008). A third example is the field of neuroimaging, where classifiers are used to relate subject-specific multivariate measures of brain activity to a particular cognitive or perceptual state (Cox and Savoy, 2003). In all of these scenarios, the data have a two-level structure: they comprise n experimental trials (or e-mails, or brain scans) collected from each member of a group of m subjects (or users, or patients). For each subject, the classifier is trained and tested on separate partitions of the trial-wise data. This gives rise to a set of true labels and a set of predicted labels, separately for each subject within the group. The typical question of interest for studies as those described above is: What is the accuracy of the classifier in the general population from which the subjects were sampled? This paper is concerned with such group-level inference on classification accuracy for hierarchically structured data.

In contrast to a large literature on evaluating classification performance in non-hierarchical contexts (see Langford, 2005, for a review), relatively little attention has been devoted to evaluating classification algorithms in hierarchical (i.e., group) settings (Goldstein, 2010; Olivetti et al., 2012). Rather than treating classification outcomes obtained in different subjects as samples from the same distribution, a hierarchical setting requires us to account for the fact that each subject itself has been sampled from a heterogeneous population (Beckmann et al., 2003; Friston et al., 2005). Thus, any approach to evaluating classification performance should account for two independent sources of uncertainty: *fixed-effects* variance (i.e., within-subjects variability) that results from uncertainty about the true classification accuracy in any given subject; and *random-effects variance* (i.e., between-subjects variability) that results from the distribution of true accuracies in the population from which

subjects were drawn. Taking into account both types of uncertainty requires *mixed-effects* inference. This is a central theme of the models discussed in this paper.

There are several commonly used approaches to performance evaluation in hierarchical classification studies.[1] One approach rests on the *pooled sample accuracy*, that is, the number of correctly predicted trials divided by the number of trials in total, across all subjects. Statistical significance can then be assessed using a simple binomial test that is based on the likelihood of obtaining the observed number of correct trials by chance (Langford, 2005). The second commonly used method considers the sample accuracy obtained in each individual subject. The method then (explicitly or implicitly) performs a one-tailed t-test across subjects to assess whether the true accuracy is greater than expected by chance (e.g., Harrison and Tong, 2009; Krajbich et al., 2009; Knops et al., 2009; Schurger et al., 2010).

Both of these commonly used methods suffer from limitations. First of all, they neglect the hierarchical nature of the experiment. The first method represents a fixed-effects approach and disregards variability across subjects. The second method considers random effects, but does not explicitly model the uncertainty associated with subject-specific accuracies. Moreover, both methods use maximum-likelihood estimation which has a tendency to underestimate the variance of the distribution and thus may show suboptimal predictive performance in relation to unseen data (i.e., overfitting; cf. Bishop, 2007, pp. 27–28, 147). Finally, both above methods assess performance in terms of *accuracy*, which may lead to inflated estimates for imbalanced data sets and thus to false conclusions about the significance with which the algorithm has performed better than chance (Chawla et al., 2002; Japkowicz and Stephen, 2002; Akbani et al., 2004; Wood et al., 2007; Zhang and Lee, 2008; Demirci et al., 2008; Brodersen et al., 2010a).

This paper introduces hierarchical models which implement full Bayesian mixed-effects analyses of classification performance that can flexibly deal with different performance measures.[2] These models overcome the limitations of the ritualized approaches described above: First, the models introduced here explicitly represent the hierarchical structure of the data, simultaneously accounting for fixed-effects and random-effects variance components. Second, maximum-likelihood estimation is replaced by a Bayesian framework which enables regularized estimation and model selection with conclusions in terms of posterior probability statements (Gelman et al., 2003). Third, our approach permits inference on both the accuracy and the balanced accuracy, a performance measure that avoids bias when working with imbalanced data sets (Brodersen et al., 2010a).

The paper is organized as follows. Section 2 describes both existing and novel models for inferring the accuracy and balanced accuracy of classification algorithms in the context of hierarchical data sets. Section 3 provides a set of illustrative applications of these models on both synthetic and empirical data. Section 4 reviews the key characteristics of these models and discusses their role in future classification studies.

2. Theory

In a hierarchical setting, a classifier predicts the class label of each of n trials, separately for each subject from a group. Here, we deal with the most common situation, that is, binary classification,

1. This paper focuses on parametric models for performance evaluation. Nonparametric methods are not considered in detail here.
2. All models discussed in this paper have been implemented in MATLAB and can be downloaded from: `http://mloss.org/software/view/407/`.

where class labels are taken from $\{-1,+1\}$, denoted as 'positive' and 'negative' trials. (The extension to a multiclass setting is described in the Discussion.) Typically, the algorithm is trained and tested on separate partitions of the data, resulting in $k \in \{0 \ldots n\}$ correct and $n-k$ incorrect predictions. This procedure is repeated for each subject j within a group of size m.

This setting raises three principal questions. First, what is the classification accuracy at the group level? This is addressed by inference on the mean classification accuracy in the population from which subjects were drawn. Second, what is the classification accuracy in each individual subject? Addressing this question by considering each subject in turn is possible but potentially wasteful, since within-subject inference may benefit from across-subject inference (Efron and Morris, 1971). Third, which of several classification algorithms is best? This question can be answered by estimating how well an algorithm's classification performance generalizes to new data. In particular, we wish to predict how well a trial-wise classifier will perform 'out of sample', that is, on trials from an unseen subject drawn from the same population as the one underlying the presently studied group.

This section considers different models for answering these questions. To keep the paper self-contained, we begin by briefly reviewing the well-known beta-binomial model (Pearson, 1925; Skellam, 1948; Lee and Sabavala, 1987). This introduces most of the concepts we require for subsequently introducing two new models designed to support hierarchical Bayesian inference: the twofold beta-binomial model and the bivariate normal-binomial model.

2.1 Inference on the Accuracy Using the Beta-Binomial Model

A classification algorithm, applied to n trials from a single subject, produces a sequence of classification outcomes $y_1, \ldots, y_n$ which are either correct (1) or incorrect (0). Analyses of these outcomes are typically based on the assumption that, on any given trial independently, the classifier makes a correct prediction with probability $0 \leq \pi \leq 1$, and an incorrect one with probability $1-\pi$. Thus, conditional on π, outcomes are given as a series of independent and identically distributed (i.i.d.) Bernoulli trials,

$$p(y_i \mid \pi) = \mathrm{Bern}(y_i \mid \pi) = \pi^{y_i}(1-\pi)^{1-y_i} \quad \forall i = 1 \ldots n.$$

The i.i.d. assumption derives from the assumption that the observations in the test set are i.i.d. themselves. This assumption is not always made in the context of cross-validation, but is easily justified when the data are only split once, without any cross-validation (cf. Discussion).

2.1.1 THE BETA-BINOMIAL MODEL

The i.i.d. assumption about individual classification outcomes allows us to summarize a sequence of outcomes in terms of the number of correctly predicted trials, k, and the total number of test trials, n. Thus, classification outcomes are converted into a random variable $k = \sum_{i=1}^{n} y_i$ which represents the number of successes over n trials. Since the sum of several Bernoulli variables follows a binomial distribution, the number of successes is given by:

$$p(k \mid \pi, n) = \mathrm{Bin}(k \mid \pi, n) = \binom{n}{k} \pi^k (1-\pi)^{n-k} \tag{1}$$

In this setting, Bayesian inference differs from classical maximum-likelihood estimation in that it assesses the plausibility of all possible values of π before and after observing actual data, rather than

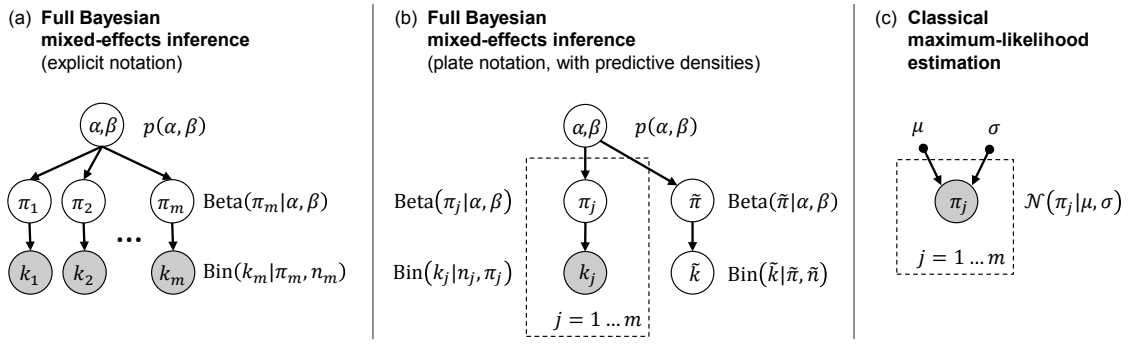


Figure 1: Models for inference on classification accuracies. This illustration shows graphical representations of different models for classical and Bayesian inference on classification accuracies, as discussed in Sections 2.1 and 2.2. Blank circles correspond to latent variables, filled circles represent observed data.

viewing π as a fixed parameter which is to be estimated. (Note that n depends on the experimental design and is not subject to inference.) It is precisely this problem that formed the basis of the first Bayesian analyses published by Bayes and Price (1763) and Laplace (1774). A natural choice for the prior distribution $p(\pi)$ is the Beta distribution,

$$p(\pi \mid \alpha_0, \beta_0) = \text{Beta}(\pi \mid \alpha_0, \beta_0) = \frac{\Gamma(\alpha_0 + \beta_0)}{\Gamma(\alpha_0)\,\Gamma(\beta_0)} \pi^{\alpha_0 - 1} (1 - \pi)^{\beta_0 - 1}, \tag{2}$$

where $\alpha_0, \beta_0 > 0$ are hyperparameters, and the Gamma function $\Gamma(\cdot)$ is required for normalization. Multiplying (1) with (2) gives rise to an overdispersed form of the binomial distribution known as the beta-binomial model (Figure 1; Pearson, 1925; Skellam, 1948; Lee and Sabavala, 1987).

In the absence of prior knowledge about π, we use a *noninformative* prior by setting $\alpha_0 = \beta_0 = 1$, which turns the Beta distribution into a uniform distribution over the $[0, 1]$ interval. The hyperparameters α_0 and β_0 can be interpreted as virtual prior counts of $\alpha_0 - 1$ correct and $\beta_0 - 1$ incorrect trials. Thus, a uniform prior corresponds to zero virtual prior observations of either kind.[3]

Because the Beta prior in (2) is a *conjugate* prior for the binomial likelihood in (1), the posterior distribution $p(\pi \mid k)$ has the same functional form as the prior,

$$p(\pi \mid k) = \text{Beta}(\pi \mid \alpha_n, \beta_n), \tag{3}$$

with updated observation counts $\alpha_n = \alpha_0 + k$ and $\beta_n = \beta_0 + n - k$.

In our context, classification is carried out separately for each subject within a group, hence the available data are k_j out of n_j correct predictions for each subject $j = 1 \ldots m$. One might be tempted to concatenate these data, form group summaries $k = \sum_{j=1}^{m} k_j$ and $n = \sum_{j=1}^{m} n_j$, and proceed to inference on π. However, this would treat π as a fixed effect in the population and disregard how the data were generated. For example, when there are many heterogeneous subjects with few trials each and a single subject with many trials, the data from this single subject would unduly dominate the

3. For a discussion of alternative priors, see Gustafsson et al. (2010).

inference at the group level. Put differently, concatenation falsely assumes zero between-subjects variability.

This limitation is resolved by explicitly modelling both within-subjects (fixed-effects) and between-subjects (random-effects) variance components in a hierarchical model comprising two levels. At the level of individual subjects, for each subject j, the number of correctly classified trials k_j can be modelled as

$$p(k_j \mid \pi_j, n_j) = \mathrm{Bin}(k_j \mid \pi_j, n_j) = \binom{n_j}{k_j} \pi_j^{k_j} (1-\pi_j)^{n_j - k_j}, \tag{4}$$

where n_j is the total number of trials in subject j, and π_j represents the fixed but unknown accuracy that the classification algorithm achieves on that subject. (Note that our notation will suppress n_j unless this introduces ambiguity.) At the group level, the model must account for variability across subjects. This is achieved by modelling subject-wise accuracies as drawn from a population distribution described by a Beta density,

$$p(\pi_j \mid \alpha, \beta) = \mathrm{Beta}(\pi_j \mid \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \pi_j^{\alpha-1} (1-\pi_j)^{\beta-1}, \tag{5}$$

such that α and β characterize the population as a whole. This step is formally identical with the Beta prior placed on the accuracy in (2) which represents uncertainty about π before observing the outcome k. Equation (5) states that uncertainty about any particular subject is best quantified by our knowledge about variability in the population, that is, the distribution of π_j over subjects (which, as described below, can be learnt from the data). Formally, a particular subject's π_j is drawn from a population characterized by α and β: subject-specific accuracies are assumed to be i.i.d., conditional on the population parameters α and β.

To describe our uncertainty about the population parameters, we use a diffuse prior on α and β which ensures that the posterior will be dominated by the data. One option would be to assign uniform densities to both the prior expected accuracy $\alpha/(\alpha+\beta)$ and the prior virtual sample size $\alpha+\beta$, using logistic and logarithmic transformations to put each on a $(-\infty, \infty)$ scale; but this prior would lead to an improper posterior density (Gelman et al., 2003). An alternative is to put a uniform density on the prior expected accuracy $\alpha/(\alpha+\beta)$ and the inverse root of the virtual sample size $(\alpha+\beta)^{-1/2}$ (Gelman et al., 2003). This combination corresponds to the prior

$$\tilde{p}(\alpha, \beta) \propto (\alpha+\beta)^{-5/2} \tag{6}$$

on the natural scale. However, although this prior leads to a proper posterior density, it is improper itself (as indicated by the tilde) and thus prevents computation of the model evidence, that is, the marginal likelihood of the data given the model, which will later become important for model comparison. We resolve this limitation by using a proper (i.e., integrable and normalized) variant,

$$p(\alpha, \beta) = \frac{3}{4} (\alpha+\beta+1)^{-5/2} \tag{7}$$

which represents a special case of the generalization of (6) proposed by Everson and Bradlow (2002). This prior can be rewritten in an unnormalized, reparameterized form as

$$\tilde{p}\left(\ln\left(\frac{\alpha}{\beta}\right), \ln(\alpha+\beta)\right) = \alpha\beta(\alpha+\beta+1)^{-5/2},$$

which will be useful in the context of model inversion (Gelman et al., 2003). Two equivalent graphical representations of this model (using the formalism of Bayesian networks; Jensen and Nielsen, 2007) are shown in Figures 1a and 1b.

2.1.2 MODEL INVERSION

Inverting the beta-binomial model allows us to infer on (i) the posterior population mean accuracy, (ii) the subject-specific posterior accuracies, and (iii) the posterior predictive accuracy. We propose a numerical procedure for model inversion which is described in detail in Appendix A. Below, we restrict ourselves to a brief conceptual summary.

First, to obtain the posterior density over the population parameters α and β we need to evaluate

$$p(\alpha,\beta \mid k_{1:m}) = \frac{p(k_{1:m} \mid \alpha,\beta)\, p(\alpha,\beta)}{\iint p(k_{1:m} \mid \alpha,\beta)\, p(\alpha,\beta)\, \mathrm{d}\alpha\, \mathrm{d}\beta} \tag{8}$$

with $k_{1:m} := (k_1, k_2, \ldots, k_m)$. Under i.i.d. assumptions about subject-specific accuracies π_j we obtain the likelihood function

$$\begin{aligned} p(k_{1:m} \mid \alpha,\beta) &= \prod_{j=1}^{m} \int p(k_j \mid \pi_j)\, p(\pi_j \mid \alpha,\beta)\, \mathrm{d}\pi_j \\ &= \prod_{j=1}^{m} \mathrm{Bb}(k_j \mid \alpha,\beta), \end{aligned} \tag{9}$$

where $\mathrm{Bb}(\cdot)$ denotes the beta-binomial distribution. Since the integral on the right-hand side of (8) cannot be evaluated in closed form, we resort to a Markov chain Monte Carlo (MCMC) procedure. Specifically, we use a Metropolis algorithm (Metropolis and Ulam, 1949; Metropolis et al., 1953) to sample from the variables at the top level of the model and obtain a set $\{(\hat{\alpha}^{(\tau)}, \hat{\beta}^{(\tau)})\}$ for $\tau = 1 \ldots c$. This set allows us to obtain samples from the posterior population mean accuracy,

$$p\left(\frac{\alpha}{\alpha+\beta} \,\middle|\, k_{1:m}\right).$$

We can use these samples in various ways, for example, to obtain a point estimate of the population mean accuracy using the posterior mean,

$$\frac{1}{c}\sum_{\tau=1}^{c} \frac{\hat{\alpha}^{(\tau)}}{\hat{\alpha}^{(\tau)} + \hat{\beta}^{(\tau)}}.$$

We could also numerically evaluate the posterior probability that the mean classification accuracy in the population does not exceed chance,

$$p = \Pr\left(\frac{\alpha}{\alpha+\beta} \leq 0.5 \,\middle|\, k_{1:m}\right)$$

which can be viewed as a Bayesian analogue of a classical p-value. We shall refer to this quantity as the (posterior) *infraliminal probability* of the classifier. It lives on the same $[0,1]$ scale as a classical p-value, but has a much more intuitive (and less error-prone) interpretation: rather than denoting the probability of observing the data (or more extreme data) under the 'null hypothesis' of a chance

classifier (classical p-value), the infraliminal probability represents the (posterior) probability that the classifier operates at or below chance. We will revisit this aspect in the Discussion.

Finally, we could compute the posterior probability that the mean accuracy in one population is greater than in another,

$$p = \Pr\left(\frac{\alpha^{(1)}}{\alpha^{(1)} + \beta^{(1)}} > \frac{\alpha^{(2)}}{(\alpha^{(2)} + \beta^{(2)})} \,\middle|\, k_{1:m^{(1)}}, k_{1:m^{(2)}} \right).$$

The second question of interest concerns the classification accuracies in individual subjects. Specifically, we wish to infer on $p(\pi_j \mid k_{1:m})$ to characterize our posterior uncertainty about the true classification accuracy in subject j. Given a pair of samples $(\alpha^{(\tau)}, \beta^{(\tau)})$, we can obtain samples from subject-specific posteriors simply by drawing from

$$\mathrm{Beta}\left(\pi_j^{(\tau)} \,\middle|\, \alpha^{(\tau)} + k_j,\ \beta^{(\tau)} + n_j - k_j\right).$$

Because samples for α and β are influenced by data $k_1 \ldots k_m$ from the entire group, so are the samples for π_j. In other words, each subject's individual posterior accuracy is informed by what we have learned about the group as a whole, an effect known as *shrinking to the population*. It ensures that each subject's posterior mean lies between its sample accuracy and the group mean. Subjects with fewer trials will exert a smaller effect on the group and shrink more, while subjects with more trials will have a larger influence on the group and shrink less.

The third question of interest is how one classifier compares to another. To address this, we must assess how well the observed performance generalizes across subjects. In this case, we are typically less interested in the average effect in the group but more in the effect that a new subject from the same population would display, as this estimate takes into account both the population mean and the population variance. The expected performance is expressed by the posterior predictive density,

$$p(\tilde{\pi} \mid k_{1:m}),$$

in which $\tilde{\pi}$ denotes the classification accuracy in a new subject drawn from the same population as the existing group of subjects with latent accuracies $\pi_1, \ldots, \pi_m$ (cf. Figure 1b).[4] Samples for this density can easily be obtained using the samples $\alpha^{(\tau)}$ and $\beta^{(\tau)}$ from the posterior population mean.[5]

The computational complexity of a full Bayesian approach can be diminished by resorting to an empirical Bayes approximation (Deely and Lindley, 1981). This approach, however, is not without criticism (Robert, 2007). Here, we will keep our treatment fully Bayesian.

2.2 Inference on the Balanced Accuracy Using the Twofold Beta-Binomial Model

A well-known phenomenon in binary classification is that a training set consisting of different numbers of representatives from either class may result in a classifier that is biased towards the majority class. When applied to a test set that is similarly imbalanced, this classifier yields an optimistic accuracy estimate. In an extreme case, the classifier might assign every single test case to the majority

4. The term 'posterior predictive density' is sometimes exclusively used for densities over variables that are unobserved but are observable in principle. Here, we use the term to refer to the posterior density of any unobserved variable, whether observable in principle (such as $\tilde{k}$) or not (such as $\tilde{\pi}$).

5. If data were indeed obtained from a new subject (represented in terms of $\tilde{k}$ correct predictions in $\tilde{n}$ trials), then $p(\tilde{\pi} \mid k_{1:m}, n_{1:m})$ would be used as a prior to compute the posterior $p(\tilde{\pi} \mid \tilde{k}, \tilde{n}, k_{1:m}, n_{1:m})$.

class, thereby achieving an accuracy equal to the proportion of test cases belonging to the majority class.

In previous literature (Chawla et al., 2002; Japkowicz and Stephen, 2002; Akbani et al., 2004; Wood et al., 2007; Zhang and Lee, 2008; Demirci et al., 2008; Brodersen et al., 2010a), this has motivated, amongst other strategies, the use of a different performance measure: the *balanced accuracy*, defined as the arithmetic mean of sensitivity and specificity, or the average accuracy obtained on either class,

$$\phi = \frac{1}{2}\left(\pi^+ + \pi^-\right). \tag{10}$$

where π^+ and π^- denote classification accuracies on positive and negative trials, respectively. If the classifier performs equally well on either class, this term reduces to the conventional accuracy (i.e., the number of correct predictions divided by the total number of predictions). In contrast, if the conventional accuracy is above chance *only* because the classifier takes advantage of an imbalanced test set, then the balanced accuracy, as appropriate, will drop to chance. We can evaluate the balanced accuracy in a hierarchical setting by extending the beta-binomial model, as described next.

2.2.1 THE TWOFOLD BETA-BINOMIAL MODEL

One way of inferring on the balanced accuracy is to duplicate the beta-binomial model and apply it separately to the two classes (Figure 2a). In other words, we consider the number of correctly predicted positive trials k^+ and the number of correctly predicted negative trials k^-, and express our uncertainty about ϕ (10) before and after observing k^+ and k^-. In a single-subject setting, as in (2), we can place separate noninformative Beta priors on π^+ and π^-,

$$\begin{aligned} p(\pi^+ \mid \alpha_0^+, \beta_0^+) &= \text{Beta}(\pi^+ \mid \alpha_0^+, \beta_0^+), \\ p(\pi^- \mid \alpha_0^-, \beta_0^-) &= \text{Beta}(\pi^- \mid \alpha_0^-, \beta_0^-), \end{aligned} \tag{11}$$

where $\alpha_0^+ = \beta_0^+ = \alpha_0^- = \beta_0^- = 1$. Inference on class-specific accuracies π^+ and π^- could be done in exactly the same way as discussed in the previous section. Here, however, we are primarily interested in the posterior density of the balanced accuracy,

$$p(\phi \mid k^+, k^-) = p\left(\frac{1}{2}(\pi^+ + \pi^-) \;\middle|\; k^+, k^-\right).$$

The balanced accuracy is thus a new random variable defined via two existing random variables from our model, π^+ and π^-. Even in a single-subject setting, a closed form for its posterior distribution is not available, and so we must resort to a numerical approximation (Brodersen et al., 2010a). For this, we first note that the distribution of the sum of the two class-specific accuracies, $s := \pi^+ + \pi^-$, is the convolution of the distributions for π^+ and π^-,

$$p\left(s \mid \alpha_n^+, \beta_n^+, \alpha_n^-, \beta_n^-\right) = \int_0^s p_{\pi^+}(s - z \mid \alpha_n^+, \beta_n^+)\; p_{\pi^-}(z \mid \alpha_n^-, \beta_n^-)\; \mathrm{d}z,$$

where the subscripts of the posterior distributions $p_{\pi^+}(\cdot)$ and $p_{\pi^-}(\cdot)$ serve to remove ambiguity. We can now obtain the posterior distribution of the balanced accuracy by replacing the sum of class-

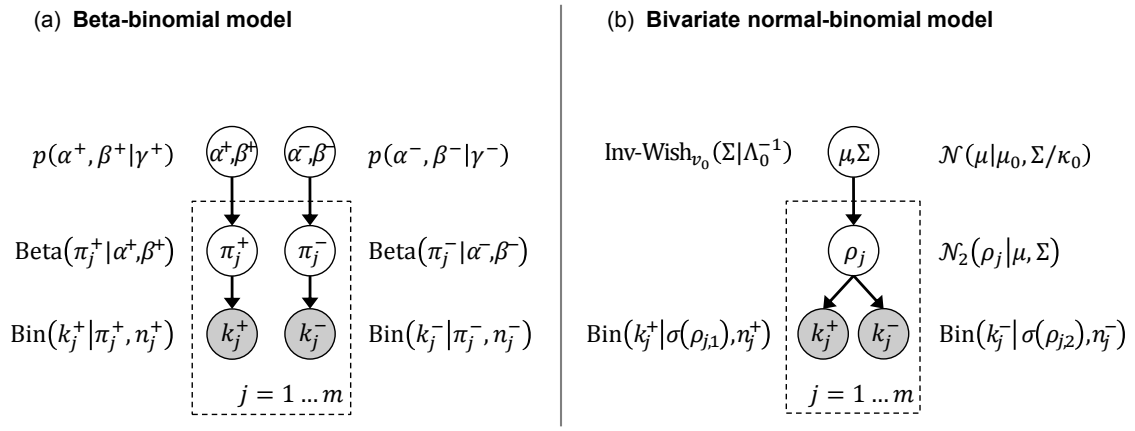


Figure 2: Models for inference on balanced classification accuracies. This figure shows two models for Bayesian mixed-effects inference on the balanced accuracy, as discussed in Sections 2.2 and 2.3. The models are based upon different assumptions and parameterizations and can be compared by Bayesian model comparison.

specific accuracies by their arithmetic mean,

$$\begin{aligned} p(\phi \mid \alpha_n^+, \beta_n^+, \alpha_n^-, \beta_n^-) &= \int_0^{2\phi} p_{\pi^+}(2\phi - z \mid \alpha_n^+, \beta_n^+)\; p_{\pi^-}(z \mid \alpha_n^-, \beta_n^-)\; \mathrm{d}z \\ &= \int_0^{2\phi} \mathrm{Beta}(2\phi - z \mid \alpha_n^+, \beta_n^+)\; \mathrm{Beta}(z \mid \alpha_n^-, \beta_n^-)\; \mathrm{d}z. \end{aligned}$$

This expression can be approximated by a simple one-dimensional grid integration over the $[0, 1]$ interval. In the same way, we can obtain approximations to the posterior mean, the posterior mode, or a posterior probability interval.

In a group setting, one can expand the above model in precisely the same way as for the simpler case of the classification accuracy in Section 2.1. Specifically, we define diffuse priors on the class-specific population parameters α^+ and β^+ as well as α^- and β^-, in analogy to (7). A graphical representation of this model is shown in Figure 2a.

2.2.2 MODEL INVERSION

Given that the twofold beta-binomial model consists of two independent instances of the simple beta-binomial model considered in Section 2.1 (Figure 1b), statistical inference follows the same approach as described previously (see Section 3.3 for an application). For instance, we can obtain the posterior population parameters, $p(\alpha^+, \beta^+ \mid k_{1:m}^+)$ and $p(\alpha^-, \beta^- \mid k_{1:m}^-)$ using the same sampling procedure as summarized in Section 2.1, except that we are now applying the procedure twice. The two sets of samples can then be averaged in a pairwise fashion to obtain samples from the posterior mean balanced accuracy in the population,

$$p\left(\phi \mid k_{1:m}^+, k_{1:m}^-\right),$$

where we have defined

$$\phi := \frac{1}{2}\left(\frac{\alpha^+}{\alpha^+ + \beta^+} + \frac{\alpha^-}{\alpha^- + \beta^-}\right).$$

Similarly, we can average pairs of posterior samples from π_j^+ and π_j^- to obtain samples from the posterior densities of subject-specific balanced accuracies,

$$p\left(\phi_j \mid k_{1:m}^+, k_{1:m}^-\right).$$

Using the same idea, we can obtain samples from the posterior predictive density of the balanced accuracy that can be expected in a new subject from the same population,

$$p\left(\tilde{\phi} \mid k_{1:m}^+, k_{1:m}^-\right).$$

2.3 Inference on the Balanced Accuracy Using the Bivariate Normal-Binomial Model

In the previous section, we saw that the twofold beta-binomial model enables mixed-effects inference on the balanced accuracy. However, it may not always be optimal to treat accuracies on positive and negative trials separately (cf. Leonard, 1972). That is, if π^+ and π^- were related in some way, the model should reflect this. For example, one could imagine a group study in which some subjects exhibit a more favourable signal-to-noise ratio than others, leading to well-separated classes. In this case, an unbiased classifier yields high accuracies on either class in some subjects and lower accuracies in others, inducing a positive correlation between class-specific accuracies. On the other hand, within each subject, any classification algorithm faces a trade-off between performing better on one class at the expense of the other class. Thus, any variability in setting this threshold leads to negatively correlated class-specific accuracies, an argument that is formally related to receiver-operating characteristics. Moreover, if the degree of class imbalance in the data varies between subjects, classifiers might be biased in different ways, again leading to negatively correlated accuracies.

In summary, π^+ and π^- may not always be independent. We therefore turn to an alternative model for mixed-effects inference on the balanced accuracy that embraces potential dependencies between class-specific accuracies (Figure 2b).

2.3.1 THE BIVARIATE NORMAL-BINOMIAL MODEL

The bivariate normal-binomial model no longer assumes that π^+ and π^- are drawn from separate populations. Instead, we use a bivariate population density whose covariance structure defines the form and extent of the dependency between π^+ and π^-.

For this combined prior, we use a bivariate normal density. Because this density has infinite support, we do not define it on the accuracies themselves but on their log odds. In this way, each subject j is associated with a two-dimensional vector of class-specific accuracies,

$$\rho_j = \begin{pmatrix}\rho_j^+ \\ \rho_j^-\end{pmatrix} = \begin{pmatrix}\sigma^{-1}(\pi_j^+) \\ \sigma^{-1}(\pi_j^-)\end{pmatrix} \in \mathbb{R}^2,$$

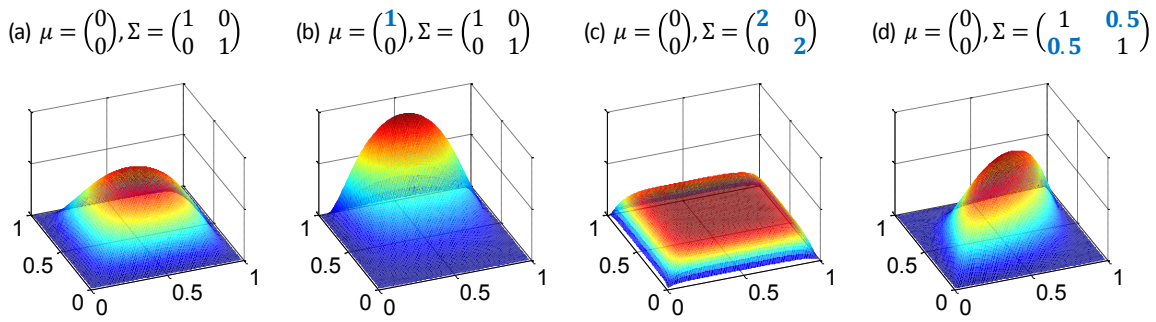


Figure 3: Distributions of class-specific accuracies in the bivariate normal-binomial model. In the bivariate normal-binomial model (Section 2.3), class-specific accuracies are assumed to follow a bivariate logit-normal distribution. This figure illustrates the flexibility of this distribution. Specifically, (a) the standard parameterization is compared to a distribution with (b) an increased accuracy on one class but not the other, (c) an increased population heterogeneity, and (d) a correlation between class-specific accuracies. The x- and y-axis represent the accuracies on positive and negative trials, respectively.

where $\sigma^{-1}(\pi) := \ln\pi - \ln(1-\pi)$ represents the logit (or inverse-logistic) transform. Conversely, class-specific accuracies can be recovered using

$$\pi_j = \begin{pmatrix} \pi_j^+ \\ \pi_j^- \end{pmatrix} = \begin{pmatrix} \sigma(\rho_j^+) \\ \sigma(\rho_j^-) \end{pmatrix} \in [0,1]^2,$$

where $\sigma(\rho) := 1/(1+\exp(-\rho))$ denotes the sigmoid (or logistic) transform. Thus, we can replace the two independent Beta distributions for π^+ and π^- in (11) by a single bivariate Gaussian prior,

$$p(\rho_j \mid \mu, \Sigma) = \mathcal{N}_2(\rho_j \mid \mu, \Sigma), \tag{12}$$

in which $\mu \in \mathbb{R}^2$ represents the population mean and $\Sigma \in \mathbb{R}^{2\times 2}$ encodes the covariance structure between accuracies on positive and negative trials. The resulting density on $\pi \in \mathbb{R}^2$ is a bivariate logit-normal distribution (Figure 3).

In analogy with the prior placed on α and β in Section 2.1, we now specify a prior for the population parameters μ and Σ. Specifically, we seek a diffuse prior that induces a weakly informative bivariate distribution over $[0,1]\times[0,1]$. We begin by considering the family of conjugate priors for (μ, Σ), that is, the bivariate normal-inverse-Wishart distribution,

$$\begin{aligned} &p(\mu, \Sigma \mid \mu_0, \kappa_0, \Lambda_0, \nu_0) \\ &\propto |\Sigma|^{-\left(\frac{\nu_0}{2}+2\right)} \exp\left(-\frac{1}{2}\mathrm{tr}(\Lambda_0\Sigma^{-1}) - \frac{\kappa_0}{2}(\mu-\mu_0)^{\mathrm{T}}\Sigma^{-1}(\mu-\mu_0)\right). \end{aligned}$$

In this distribution, the population hyperparameters Λ_0 and ν_0 specify the scale matrix and the degrees of freedom, while the parameters μ_0 and κ_0 represent the prior mean and the number of prior measurements on the Σ scale, respectively (Gelman et al., 2003). A more convenient representation

can be obtained by factorizing the density into

$$p(\Sigma \mid \Lambda_0, \nu_0) = \text{Inv-Wishart}_{\nu_0}(\Sigma \mid \Lambda_0^{-1}) \quad \text{and}$$
$$p(\mu \mid \Sigma, \mu_0, \kappa_0) = \mathcal{N}_2(\mu \mid \mu_0, \Sigma/\kappa_0).$$

In order to illustrate the flexibility offered by the bivariate normal density on ρ, we derive $p(\pi \mid \mu, \Sigma)$ in closed form (Appendix B) and then compute the bivariate density on a two-dimensional grid (Figure 3).

For the purpose of specifying a prior, we seek hyperparameters μ_0, κ_0, Λ_0, and ν_0 that induce a diffuse bivariate distribution over π. This can be achieved using

$$\mu_0 = (0,0)^{\mathrm{T}},\ \kappa_0 = 1,\ \Lambda_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}^{-1},\ \nu_0 = 5.$$

2.3.2 MODEL INVERSION

In contrast to the twofold beta-binomial model discussed in the previous section, the bivariate normal-binomial model makes it difficult to sample from the posterior densities over model parameters using a Metropolis implementation. In order to sample from $p(\mu, \Sigma \mid k^+_{1:m}, k^-_{1:m})$, we would have to evaluate the likelihood $p(k^+_{1:m}, k^-_{1:m} \mid \mu, \Sigma)$; this would require us to integrate out π^+ and π^-, which is difficult.

A simpler strategy is to design a Gibbs sampler (Geman and Geman, 1984) to draw from the joint posterior $p(\rho_{1:m}, \mu, \Sigma \mid k^+_{1:m}, k^-_{1:m})$, from which we can derive samples for the conditional posteriors $p(\rho_{1:m} \mid k^+_{1:m}, k^-_{1:m})$ and $p(\mu, \Sigma \mid k^+_{1:m}, k^-_{1:m})$. In contrast to a Metropolis scheme, Gibbs sampling requires only full conditionals, that is, distributions of one latent variable conditioned on all other variables in the model (Gelfand and Smith, 1990). Whenever a full conditional is not available, we can sample from it using a Metropolis step. Thus, we combine a Gibbs skeleton with interleaved Metropolis steps to sample from the posterior $p(\rho_{1:m}, \mu, \Sigma \mid k^+_{1:m}, k^-_{1:m})$. See Section 3.3 for an application.

First, population parameter estimates can be obtained by sampling from the posterior density $p(\mu, \Sigma \mid k^+_{1:m}, k^-_{1:m})$ using a Metropolis-Hastings approach. Second, subject-specific accuracies are estimated by first sampling from $p(\rho_j \mid k^+_{1:m}, k^-_{1:m})$ and then applying a sigmoid transform to obtain samples from the posterior density over subject-specific balanced accuracies, $p(\phi_j \mid k^+_{1:m}, k^-_{1:m})$. Finally, the predictive density $p(\tilde{\phi} \mid k^+_{1:m}, k^-_{1:m})$ can be obtained using an ancestral-sampling step on the basis of $\mu^{(\tau)}$ and $\Sigma^{(\tau)}$ followed by a sigmoid transform. As before, we use the obtained samples in all three cases to compute approximate posterior probability intervals or Bayesian p-values. A detailed description of this algorithm can be found in Appendix C.

2.4 Bayesian Model Selection

While the twofold beta-binomial model assumes independent class-specific accuracies, the bivariate normal-binomial model relaxes this assumption and allows for correlations between accuracies. This raises two questions. First, given a particular data set, which model is best at explaining observed classification outcomes? And second, can we combine the two models to obtain posterior inferences that integrate out uncertainty about which model is best? Both questions can be answered using the marginal likelihood, or model evidence, that is, the probability of the data given the model,

after integrating out the parameters:

$$p(k^+_{1:m}, k^-_{1:m} \mid M) = \int p(k^+_{1:m}, k^-_{1:m} \mid \theta)\; p(\theta \mid M)\; \mathrm{d}\theta$$

Here, θ serves as a placeholder for all model parameters and $p(\theta \mid M)$ represents its prior distribution under a given model M. Under a flat prior over models, Bayes' theorem indicates that the model with the highest evidence has the highest posterior probability given the data:

$$p(M \mid k^+_{1:m}, k^-_{1:m}) \propto p(k^+_{1:m}, k^-_{1:m} \mid M)$$

In practice, the model evidence is usually replaced by the log model evidence, which is monotonically related but numerically advantageous.

Concerning the first model described in this section, the twofold beta-binomial model M_{bb}, the log model evidence is given by

$$\begin{aligned}
&\ln p(k^+_{1:m}, k^-_{1:m} \mid M_{bb}) \\
&= \ln \int p(k^+_{1:m} \mid \pi^+_{1:m})\, p(\pi^+_{1:m})\; \mathrm{d}\pi^+_{1:m} + \ln \int p(k^-_{1:m} \mid \pi^-_{1:m})\, p(\pi^-_{1:m})\; \mathrm{d}\pi^-_{1:m} && (13) \\
&= \ln \left\langle \prod_{j=1}^{m} p(k^+_j \mid \pi^+_j) \right\rangle_{\pi^+_{1:m}} + \ln \left\langle \prod_{j=1}^{m} p(k^-_j \mid \pi^-_j) \right\rangle_{\pi^-_{1:m}} && (14)
\end{aligned}$$

where we have omitted the conditional dependence on M_{bb} in (13) and (14).[6] The expression can be approximated by

$$\approx \ln \frac{1}{c} \sum_{\tau=1}^{c} \prod_{j=1}^{m} \mathrm{Bin}\left(k^+_j \;\middle|\; \pi^{+(\tau)}_j\right) + \ln \frac{1}{c} \sum_{\tau=1}^{c} \prod_{j=1}^{m} \mathrm{Bin}\left(k^-_j \;\middle|\; \pi^{-(\tau)}_j\right),$$

where $\pi^{+(\tau)}_j$ and $\pi^{-(\tau)}_j$ represent independent samples from the prior distribution over subject-specific accuracies. They can be obtained using ancestral sampling, starting from the prior over α and β, as given in (7).

In the case of the bivariate normal-binomial model M_{nb}, the model evidence no longer sums over model partitions as in (13), and so the approximation is derived differently,

$$\begin{aligned}
&\ln p(k^+_{1:m}, k^-_{1:m} \mid M_{nb}) \\
&= \ln \int p\left(k^+_{1:m}, k^-_{1:m} \mid \rho_{1:m}\right)\, p\left(\rho_{1:m} \mid M_{nb}\right)\, \mathrm{d}\rho_{1:m} && (15) \\
&\approx \ln \frac{1}{c} \sum_{\tau=1}^{c} \prod_{j=1}^{m} \mathrm{Bin}\left(k^+_j \;\middle|\; \sigma\left(\rho^{(\tau,1)}_j\right)\right) \mathrm{Bin}\left(k^-_j \;\middle|\; \sigma\left(\rho^{(\tau,2)}_j\right)\right), && (16)
\end{aligned}$$

for which we provide additional details in Appendix C (24). Having computed the model evidences, one can proceed to Bayesian model selection (BMS) by evaluating the log Bayes factor,

$$\ln BF_{bb,nb} = \ln p(k^+_{1:m}, k^-_{1:m} \mid M_{bb}) - \ln p(k^+_{1:m}, k^-_{1:m} \mid M_{nb}), \qquad (17)$$

6. One could also express the model evidence in terms of an expectation with respect to $p(\alpha, \beta \mid M_{bb})$.

representing the evidence in favour of the beta-binomial over the normal-binomial model. By convention, a log Bayes factor greater than 3 is considered strong evidence in favour of one model over another, whereas a log Bayes factor greater than 5 is referred to as very strong evidence (Kass and Raftery, 1995). The best model can then be used for posterior inferences on the mean accuracy in the population or the predictive accuracy in a new subject from the new population.

The second option to make use of the model evidences of competing models is Bayesian model averaging (Cooper and Herskovits, 1992; Madigan and Raftery, 1994; Madigan et al., 1996). Under this view, we do not commit to a particular model but average the predictions made by all of them, weighted by their respective posteriors. In this way, we obtain a mixture expression for the posterior of the mean accuracy in the population,

$$p\left(\phi \mid k^+_{1:m}, k^-_{1:m}\right) = \sum_M p\left(\phi \mid k^+_{1:m}, k^-_{1:m}, M\right)\ p\left(M \mid k^+_{1:m}, k^-_{1:m}\right).$$

Similarly, we can obtain the posterior predictive distribution of the balanced accuracy in a new subject from the same population,

$$p\left(\tilde{\phi} \mid k^+_{1:m}, k^-_{1:m}\right) = \sum_M p\left(\tilde{\phi} \mid k^+_{1:m}, k^-_{1:m}, M\right)\ p\left(M \mid k^+_{1:m}, k^-_{1:m}\right).$$

The computational complexity of the above stochastic approximations is considerable, and so it can sometimes be useful to resort to a deterministic approximation instead, such as variational Bayes (see Discussion). While we do not consider this approach in detail here, it does provide a helpful perspective on interpreting the model evidence. Specifically, the model evidence can be approximated by a variational lower bound, the negative free-energy $\mathcal{F}$. In the case of the beta-binomial model for instance, this quantity can be written as

$$\mathcal{F} = \left\langle \ln p\left(k_{1:m} \mid \alpha, \beta, \pi_{1:m}\right)\right\rangle_q - \mathrm{KL}\left[q(\alpha, \beta, \pi_{1:m}) \parallel p(\alpha, \beta, \pi_{1:m})\right].$$

The first term is the log-likelihood of the data expected under the approximate posterior $q(\alpha, \beta, \pi_{1:m})$; it represents the goodness of fit (or accuracy) of the model. The second term is the Kullback-Leibler divergence between the approximate posterior and the prior; it represents the complexity of the model. This complexity term increases with the number of parameters, their inverse prior covariances, and with the deviation of the posterior from the prior that is necessary to fit the data. Thus, the free-energy approximation shows that the model evidence incorporates a trade-off between explaining the observed data (i.e., goodness of fit) and remaining consistent with our prior (i.e., simplicity or negative complexity). In other words, the model evidence encodes how well a model strikes the balance between explaining the data and remaining simple (Pitt and Myung, 2002; Beal, 2003; Stephan et al., 2009).

Classical approaches differ from the Bayesian framework presented above in several ways. For a comparison between classical and Bayesian inference, see Appendix D.

3. Applications

This section illustrates the practical utility of the Bayesian models discussed in the previous section and compares them to inference obtained through classical (frequentist) statistics. We begin by simulating classification outcomes to highlight the key features of Bayesian mixed-effects inference (Sections 3.1 and 3.2). We then contrast inference on accuracies with inference on balanced accuracies (Section 3.3). Finally, we illustrate the application of our approach to synthetic data (Section 3.4) as well as empirical data obtained from an imaging experiment (Section 3.5).

3.1 Inference on the Population Mean and the Predictive Accuracy

In a first experiment, we simulated classification outcomes for a group of 20 subjects with 100 trials each. Outcomes were generated using the beta-binomial model with a population mean of 0.75 and a population variance of 0.02 (i.e., $\alpha \approx 6.28$ and $\beta \approx 2.09$, corresponding to a population standard deviation of 0.141; Figure 4).

Raw data, that is, the number of correct predictions within each subject, are shown in Figure 4a. Their empirical sample accuracies are shown in Figure 4b, along with the ground-truth density of the population accuracy. Inverting the beta-binomial model, using the MCMC procedure of Section 2.1 (Figure 4c), and examining the posterior distribution over the population mean accuracy showed that more than 99.9% of its mass was above 50%, in agreement with the fact that the true population mean was above chance (Figure 4d).

We also used this simulation to illustrate the differences between a Bayesian mixed-effects central 95% posterior probability interval, a fixed-effects probability interval, and a random-effects confidence interval (Figure 4e). All three schemes arrive at the same conclusion with respect to the population mean being above chance. However, while the random-effects interval (red) is very similar to the proposed mixed-effects interval (black), the fixed-effects interval (yellow) displays too small a variance as it disregards the important between-subjects variability.

We finally considered the predictive posterior distribution over the accuracy that would be observed if we were to acquire data from a new subject (Figure 4f). This posterior did not allow for the conclusion that, with a probability larger than 0.95, the accuracy in a new subject would be above chance. This result is driven by the large heterogeneity in the population, inducing a dispersed predictive density. Importantly, the dispersion of the predictive density would not vanish even in the limit of an infinite number of subjects. This is in contrast to the dispersion of the posterior over the population mean, which becomes more and more precise with an increasing amount of data.

Inference was based on 100 000 samples, generated using 8 parallel chains. We used several standard approaches to convergence evaluation. In particular, we considered trace plots for visual inspection of mixing behaviour and convergence to the target distributions. In addition, we monitored the average ratio of within-chain variance to between-chain variance, which was bigger than 0.995. In other words, the variances of samples within and between chains were practically indistinguishable. The Metropolis rejection rate was 0.475, thus ensuring an appropriate balance between exploration (of regions with a lower density) and exploitation (of regions with a higher density). Finally, we assessed the uncertainty inherent in MCMC-based quantities such as log Bayes factors by computing standard deviations across repetitions, which led us to use 10^5 or 10^6 samples for each computation (see Section 3.3). All subsequent applications were based on the same algorithmic settings.

In frequentist inference, a common way of representing the statistical properties of a test is to estimate the probability of rejecting the null hypothesis at a fixed threshold (e.g., 0.05) under different regimes of ground truth, which leads to the concept of *power curves*. Here, we adopted this frequentist perspective to illustrate the properties of Bayesian mixed-effects inference on classification performance (Figure 5).

Specifying a true population mean of 0.5 and variance of 0.001 (standard deviation 0.0316), we generated classification outcomes, in the same way as above, for a synthetic group of 20 subjects with 100 trials each. Inverting the beta-binomial model, we inferred whether the population mean was above chance by requiring more than 95% of the posterior probability mass of the population

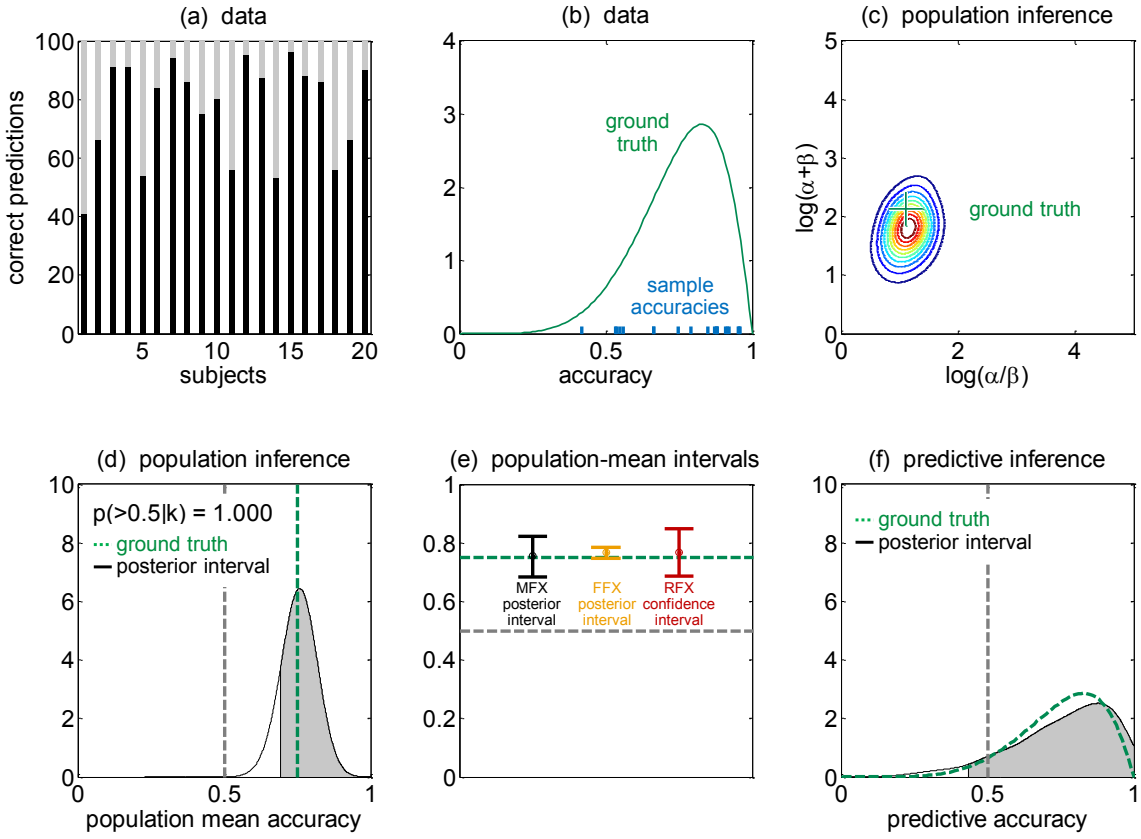


Figure 4: Inference on the population mean and the predictive accuracy. (a) Classification outcomes were generated for 20 subjects using the beta-binomial model. Each subject is fully characterized by the number of correctly classified trials (black) out of a given set of 100 trials (grey). (b) Empirical sample accuracies (blue) and their underlying population distribution (green). (c) Inverting the beta-binomial model yields samples from the posterior distribution over the population parameters, visualized using a nonparametric (bivariate Gaussian kernel) density estimate (contour lines). (d) The posterior about the population mean accuracy, plotted using a kernel density estimator (black), is sharply peaked around the true population mean (green). The upper 95% of the probability mass are shaded (grey). Because the lower bound of the shaded area is greater than 0.5, the population mean can be concluded to be above chance. (e) While the central 95% posterior interval (black) and the classical 95% confidence interval (red) look similar, the two intervals are conceptually very different. The fixed-effects interval (orange) is overly optimistic as it disregards between-subjects variability. (f) The posterior predictive distribution over $\tilde{\pi}$ represents the posterior belief of the accuracy expected in a new subject (black). Its dispersion reflects irreducible population heterogeneity.

mean to be greater than 0.5, that is, by requiring an infraliminal probability of less than 5%. We repeated this process 1 000 times and counted how many times the population mean was deemed

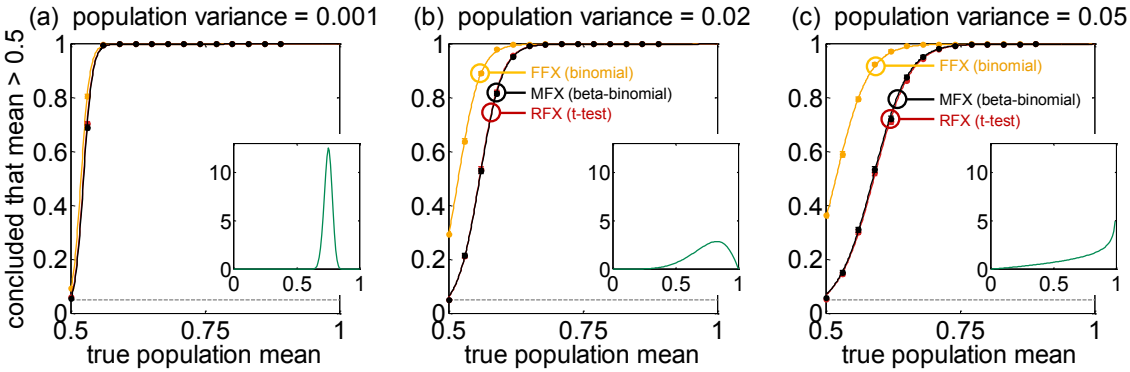


Figure 5: Inference on the population mean under varying population heterogeneity. The figure shows Bayesian estimates of the frequentist probability of above-chance classification performance, as a function of the true population mean, separately for three different level of population heterogeneity (a,b,c). Each data point is based on 1 000 simulations, each of which used 10 000 samples from every subject-specific posterior to make a decision. The figure shows that, in this setting, frequentist inference based on *t*-tests (red) agrees with Bayesian inference based on the beta-binomial model (black). By contrast, a fixed-effects approach (orange) offers invalid population inference as it disregards between-subjects variability; at a true population mean of 0.5, the hypothesis of chance-level performance is rejected more frequently than prescribed by the test size. Each data point is plotted in terms of the fraction of above-chance conclusions and a 95% central posterior interval, based on a Beta model with a flat prior. Points are joined by a sigmoidal function that was constrained to start at 0 and end at 1, with two remaining degrees of freedom. Where the true population mean exceeds 0.5, the graphs reflect the empirical sensitivity of the inference scheme. Its empirical specificity corresponds to the vertical distance between the graphs and 1 at the point where the population mean is 0.5. Insets show the distribution of the true underlying population accuracy (green) for a population mean accuracy of 0.75.

greater than chance. We then varied the true population mean and plotted the fraction of decisions for an above-chance classifier as a function of population mean (Figure 5a). At a population mean of 0.5, the vertical distance between the data points and 1 represents the empirical specificity of the test (which was designed to be $1-\alpha = 0.95$). At population means above 0.5, the data points show the empirical sensitivity of the test, which grows rapidly with increasing population mean. In this setting, the inferences that one would obtain by a frequentist *t*-test (red) are in excellent agreement with those afforded by the proposed beta-binomial model (black). Since the population variance was chosen to be very low in this initial simulation, the inferences afforded by a fixed-effects analysis (yellow) prove very similar as well; but this changes drastically when increasing the population variance to more realistic levels, as described below.

One important issue in empirical studies is the heterogeneity of the population. We studied the effects of population variance by repeating the above simulations with different variances (Figures 5b,c). As expected, an increase in population variance reduced statistical sensitivity. For

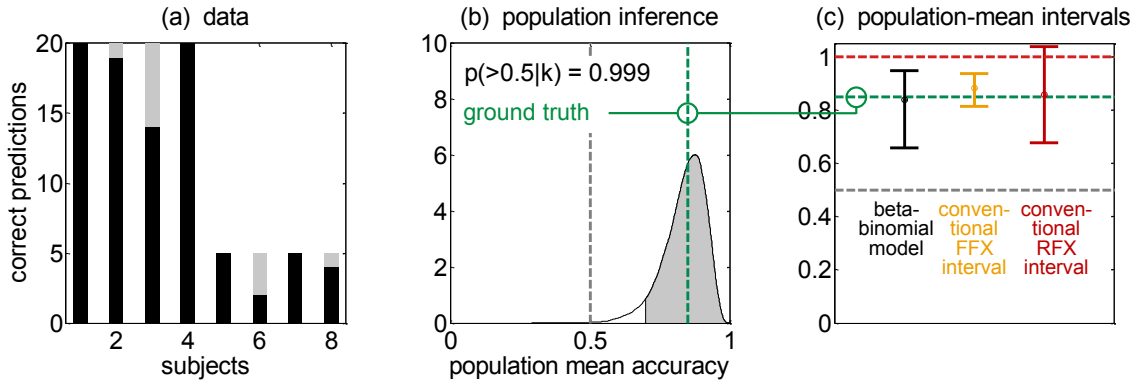


Figure 6: Inadequate inferences provided by fixed-effects and random-effects models. (a) The simulation underlying this figure represents the case of a small heteroscedastic group with varying numbers of trials across subjects. Classification outcomes were generated in the same way as in the simulation underlying Figure 5a. (b) The (mixed-effects) posterior density of the population mean (black) provides a good estimate of ground truth (green). (c) A central 95% posterior probability interval, based on the density shown in (b), comfortably includes ground truth. By contrast, a fixed-effects interval (orange) is overoptimistic as it disregards between-subjects variability. The corresponding random-effects confidence interval (red) is similar to the mixed-effects interval but lacks asymmetry, thus inappropriately including accuracies above 100% (red dashed line).

example, given a fairly homogeneous population with a true population mean accuracy of 60% and a variance of 0.001 (standard deviation 0.0316; Figure 5a), we can expect to correctly infer above-chance performance in more than 99.99% of all cases. By contrast, given a more heterogeneous population with a variance of 0.05 (standard deviation ≈ 0.22), the fraction of correct conclusions drops to 61%; in all other cases we would fail to recognize that the classifier was performing better than chance.

The above simulations show that a fixed-effects analysis (yellow) becomes an invalid procedure to infer on the population mean when the population variance is non-negligible. In more than the prescribed 5% of simulations with a true population mean of 0.5, the procedure concluded that the population mean was above chance. This is because a fixed-effects analysis yields too small variances on the population mean and therefore too easily makes above-chance conclusions.

All above simulations were based on a group of 20 subjects with 100 trials each. This emulated a setting as it frequently occurs in practice, for example, in neuroimaging data analyses. We repeated the same analysis as above on a sample data set from a second simulation setting (Figure 6). This setting was designed to represent the example of a small heterogeneous group with varying numbers of trials across subjects. Specifically, we generated data for 8 subjects, half of which had 20 trials, and half of which had only 5 trials. Classification outcomes were generated using the beta-binomial model with a population mean of 0.85 and a population variance of 0.02 (corresponding to a population standard deviation of 0.14; Figure 6a).

The example shows that the proposed beta-binomial model yields a posterior density with the necessary asymmetry; it comfortably includes the true population mean (Figure 6b). By contrast,

the fixed-effects probability interval (based on a Beta density) is overly optimistic. Finally, the random-effects confidence interval is similar to the mixed-effects interval but lacks the necessary asymmetry, including accuracies above 100% (Figure 6c).

3.2 Inference on Subject-Specific Accuracies

In the Bayesian models of this paper, classification accuracies of individual subjects are represented by a set of latent variables $\pi_1, \ldots, \pi_m$. A consequence of hierarchical Bayesian inference is that such subject-specific variables are informed by data from the entire group. Effectively, they are *shrunk* to the group mean, where the amount of shrinkage depends on the subject-specific posterior uncertainty.

To illustrate this, we generated synthetic classification outcomes and computed subject-specific posterior inferences (Figure 7). This simulation was based on 45 subjects overall; 40 subjects were characterized by a relatively moderate number of trials ($n = 20$) while 5 subjects had even fewer trials ($n = 5$). The population accuracy had a mean of 0.8 and a variance of 0.01 (standard deviation 0.1). Using this data set, we computed subject-specific central 95% posterior probability intervals and sorted them in ascending order by subject-specific sample accuracy (Figure 7a). The plot shows that, in each subject, the posterior mode (black) represents a compromise between the observed sample accuracy (blue) and the population mean (0.8). This compromise in turn provides a better estimate of ground truth (green) than sample accuracies by themselves. This effect demonstrates a key difference between the two quantities: subject-specific posteriors are informed by data from the entire group, whereas sample accuracies are based on the data from an individual subject.

Another way of demonstrating the shrinkage effect is by illustrating the transition from ground truth to sample accuracies (with its increase in dispersion) and from sample accuracies to posterior means (with its decrease in dispersion). This shows how the high variability in sample accuracies is reduced, informed by what has been learned about the population (Figure 7b). Notably, because the amount of shrinking depends on each subject's posterior uncertainty, the shrinking effect may modify the order of subjects, as indicated by crossing lines. Here, subjects with only 5 trials were shrunk more than subjects with 20 trials.

In a next step, we examined power curves, systematically changing the true population accuracy and repeating the above simulation 1 000 times (Figure 7c). Within a given simulation, we concluded that a subject-specific accuracy was above chance if more than 95% of its posterior probability mass was above 0.5. We binned subjects across all simulations into groups of similar accuracies and plotted the fraction of above-chance decisions against these true accuracies, contrasting the Bayesian model with a conventional t-test. As shown in Figure 7c, t-tests falsely detected above-chance subject-specific accuracies in about 5% of the cases, in agreement with the intended test size. By contrast, our Bayesian scheme was considerably more sensitive and detected above-chance accuracy in subjects whose true accuracy was within a small bin around 0.5. This reflected the fact that the Bayesian procedure incorporated what had been learned about the population when deciding on individual subjects. That is, a population mean well above chance (here: 0.8) made it more likely that individual subjects performed above chance as well, even in the presence of a low sample accuracy.

In addition to enabling decisions that take into account information about the group, the posterior distributions of subject-specific accuracies also yield more precise point estimates. To illustrate this, we simulated 1 000 data sets in the same way as above. Within each simulation, we compared

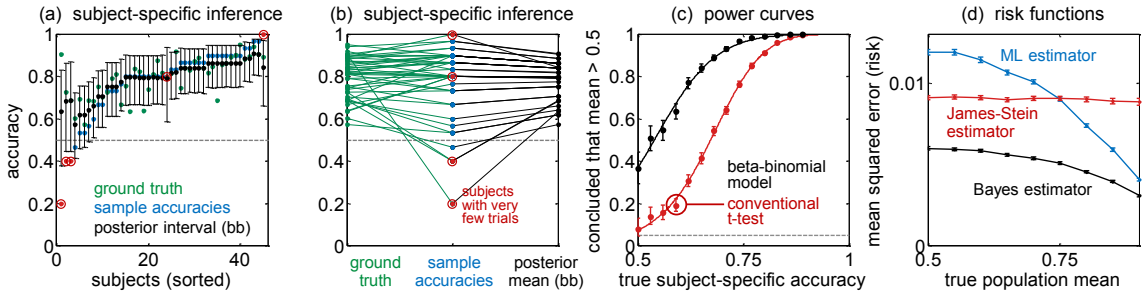


Figure 7: Inference on subject-specific accuracies. (a) Classification outcomes were generated for a synthetic heterogeneous group of 45 subjects (40 subjects with 20 trials each, 5 subjects with 5 trials each). All data were generated using the beta-binomial model (population mean 0.8, standard deviation 0.1). The figure shows subject-specific posterior means and central 95% posterior probability intervals (black), sample accuracies (blue if based on 20 trials, red if based on 5 trials), and true subject-specific accuracies (green) as a function of subject index, sorted in ascending order by sample accuracy. Whenever a subject's sample accuracy is very low or very high in relation to the remaining group, the Bayesian posterior interval is shrunk to the population. (b) Another way of visualizing the shrinking effect is to contrast the increase in dispersion (as we move from ground truth to sample accuracies) with the decrease in dispersion (as we move from sample accuracies to posterior means) enforced by the hierarchical model. Shrinking changes the order of subjects (when sorted by posterior mean as opposed to by sample accuracy) as the amount of shrinking depends on the subject-specific (first-level) posterior uncertainty. Subjects with just 5 trials (red) are shrunk more than subjects with 20 trials (blue). (c) Based on 1 000 simulations, the plot shows the fraction of simulations in which a subject's accuracy was concluded to be above chance, based on a Bayesian posterior interval (black) or a frequentist t-test (red). In contrast to classical inference, the Bayesian procedure incorporates a desirable shift towards the population in making decisions about individual group members. (d) Across the same 1 000 simulations, a Bayes estimator, based on the posterior means of subject-specific accuracies (black), was superior to both a classical ML estimator (blue) and a James-Stein estimator (red).

three different ways of obtaining an estimator for each subject's accuracy: (i) a Bayes estimator (posterior mean of the subject-specific accuracy); (ii) a maximum-likelihood estimator (sample accuracy); and (iii) a James-Stein estimator, with a similar shrinkage effect as the Bayes estimator but less explicit distributional assumptions (Figure 7d). For each estimator, we computed the mean squared error (or risk) across all subjects, averaged across all simulations. We then repeated this process for different population means. We found that the James-Stein estimator outperformed the ML estimator for low accuracies. However, both estimators were dominated by (i.e., inferior to) the Bayes estimator which provided the lowest risk throughout.

It is important to keep in mind that the above simulations are based on synthetic classification outcomes which fulfil the assumptions of the normal-binomial model by design, in particular the assumption of logit-normally distributed subject-specific accuracies and the assumption of condi-

tional independence given the population parameters. For empirical data, these assumptions may not always hold and so posterior inferences, including the shrinkage effect, may be suboptimal. This highlights the importance of model checking when using the models presented in this paper in practical applications.

3.3 Inference on the Balanced Accuracy

The balanced accuracy is a more useful performance measure than the accuracy, especially when a classifier was trained on an imbalanced test set and may thus exhibit bias. In order to illustrate the relative utility of these two measures in our Bayesian models, we simulated an imbalanced data set, composed of 20 subjects with 100 trials each, where each subject had between 70 and 90 positive trials (drawn from a uniform distribution) and between 10 and 30 negative trials.

An initial simulation specified a high population accuracy on the positive class and a low accuracy on the negative class, with equal variance in both (Figure 8a,b). These accuracies were chosen such that the classifier would perform at chance on a hypothetical balanced sample. This allowed us to mimic the commonly observed situation in which a classifier takes advantage of the imbalance in the data and preferably predicts the majority class. We independently inverted three competing models: (i) the beta-binomial model to infer on the classification accuracy; and the (ii) twofold beta-binomial and (iii) bivariate normal-binomial models to infer on the balanced accuracy. As expected, the beta-binomial model falsely suggested high evidence for above-chance classification. In contrast, the twofold beta-binomial and normal-binomial models correctly indicated the absence of a statistical relation between data and class labels (Figure 8c).

These characteristics were confirmed across a large set of simulations. As expected, inference on the accuracy falsely concluded above-chance performance, especially in the presence of a significant degree of class imbalance. By contrast, inference on the balanced accuracy did not incorrectly reject the hypothesis of the classifier operating at the level of chance more often than prescribed by the test size (Figure 8d).

We compared the two models for inference on the balanced accuracy by means of Bayesian model comparison. Using 10^6 samples with Equation (17), we obtained a log Bayes factor of 33.2 in favour of the twofold beta-binomial model (i.e., under a flat prior over models, the posterior belief in the beta-binomial model is greater than 99.99%). This result represents very strong evidence (Kass and Raftery, 1995) that the beta-binomial model provided a better explanation of the synthetic classification outcomes than the normal-binomial model. This finding is plausible since no correlation structure among class-specific accuracies was imposed in the simulation; thus, in this case, the beta-binomial model is a better model than the more complex normal-binomial model.

To assess the sampling-induced uncertainty about this result, we repeated the computation of the log Bayes factor 100 times. We obtained a sample standard deviation of 8.0, that is, the uncertainty was small in relation to the overall strength of evidence. By comparison, when using only 10^3 samples instead of 10^6, the standard deviation increased to 25.5. We used 10^6 samples for all subsequent analyses.

We repeated the main analysis above 1 000 times and plotted the fraction of above-chance conclusions against the degree of class imbalance. Note that the resulting curve is not a power curve in the traditional sense, as its independent variable is not the true (balanced) accuracy but the accuracy on positive trials, that is, an indicator of the degree of class imbalance. Figure 8d shows that the simple beta-binomial model provides progressively misleading conclusions with class imbalance at

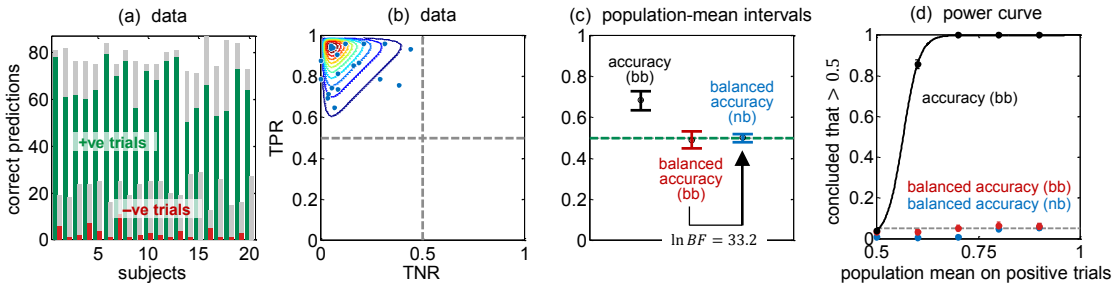


Figure 8: Inference on the balanced accuracy. (a) The simulation underlying this figure mimics an imbalanced data set which has led the classification algorithm to acquire a bias in favour of the majority class. The plot shows, for each subject, the number of correctly classified positive (green) and negative (red) trials, as well as the respective total number of trials (grey). (b) Visualizing sample accuracies separately for the two classes gives rise to a two-dimensional plot, in which the true positive rate on the y-axis and the true negative rate on the x-axis represent the accuracies on positive and negative trials, respectively. The underlying true population distribution is represented by a bivariate Gaussian kernel density estimate (contour lines). The plot shows that the population accuracy is high on positive trials and low on negative trials. (c) Central 95% posterior probability intervals based on three models: the simple beta-binomial model for inference on the population accuracy; and the twofold beta-binomial model as well as the bivariate normal-binomial model for inference on the balanced accuracy. The true mean balanced accuracy in the population is at chance (green). It is accurately estimated by models inferring on the balanced accuracy (red, blue). Bayesian model selection yielded very strong evidence (Kass and Raftery, 1995) in favour of the normal-binomial model (posterior model probability = 97.7%). (d) Probability of falsely detecting above-chance performance, using different inference schemes. The true balanced accuracy is 0.5. The x-axis represents the degree of class imbalance.

the group level (cf. Figure 5). In contrast, both schemes for inference on the balanced accuracy made above-chance conclusions in less than 5% of the simulations, as intended by their test size.

All models considered in this paper are based on diffuse priors designed in such a way that posterior inferences are clearly dominated by the data. However, one might ask to what extent such inferences depend on the exact form of the prior. To examine this dependence, we carried out a sensitivity analysis in which we considered the infraliminal probability of the posterior population mean as a function of prior moments (Figure 9). We found that inferences were extremely robust, in the sense that the influence of the prior moments on the resulting posterior densities was negligible in relation to the variance resulting from the fact that we are using a (stochastic) approximate inference method for model inversion. In particular, varying the constant (originally: 1) in Equation (7) for the beta-binomial prior left the infraliminal probability of the posterior accuracy unaffected (Figure 9a,b). Similarly, varying μ_0, κ_0, or ν_0 in the normal-binomial model had practically no influence on the infraliminal probability of the posterior balanced accuracy (Figure 9c,d,e).

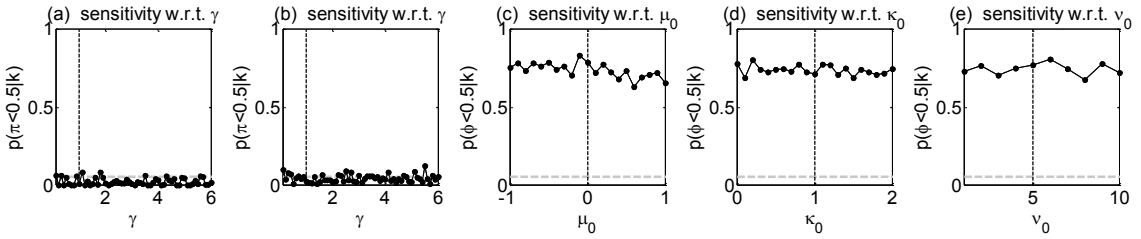


Figure 9: Sensitivity analysis. This figure illustrates the dependence of posterior inferences on the exact form of the priors proposed in this paper. Each graph shows the infraliminal probability of the population mean accuracy (i.e., the posterior probability mass below 0.5) as a function of a particular parameter of the prior distribution used for model inversion. (a,b) Same data sets as shown those shown in Figures 4a and 6a, but with a slightly lower population mean of 0.7. Inferences on the population accuracy are based on the beta-binomial model. (c,d,e) Same data set as shown in Figure 8a. Inferences on the population balanced accuracy are based on the bivariate normal-binomial model.

3.4 Application to Synthetic Data

All experiments described so far were based on classification outcomes sampled from the beta-binomial or normal-binomial model. This ensured, by construction, that the distributional assumptions underlying the models were fulfilled. To illustrate the generic applicability of our approach when its assumptions are not satisfied by construction, we applied models for mixed-effects inference to classification outcomes obtained on synthetic data features for a group of 20 subjects with 100 trials each (Figure 10). In addition to probing the models' robustness with regard to distributional assumptions, this allows one to examine what correlations between class-specific accuracies may be observed in practice.

Synthetic data were generated using a two-level sampling approach that reflected the hierarchical nature of group studies. We specified a population distribution, sampled subject-specific means and variances from it, and then used these to generate trial-specific feature vectors. In a first simulation (Figure 10, top row), we generated 50 positive trials and 50 negative trials for each individual subject j by drawing one-dimensional feature vectors from two normal distributions, $\mathcal{N}(x_{ij} \mid \mu_j^+, \sigma_j)$ and $\mathcal{N}(x_{ij} \mid \mu_j^-, \sigma_j)$, respectively. The moments of these subject-specific distributions, in turn, were drawn from a population distribution, using $\mathcal{N}(\mu_j^+ \mid \frac{1}{2}, \frac{1}{2})$ and $\mu_j^- = -\mu_j^+$ for the means, and $\mathrm{Ga}^{-1}(\sigma_j \mid 10, \frac{1}{10})$ for the variance. The normal distribution and the inverse Gamma distribution are conjugate priors for the mean and variance of a univariate normal distribution and, thus, represent natural choices for the population distribution.

To obtain classification outcomes, separately for each subject, we trained and tested a linear support vector machine (SVM), as implemented by Chang and Lin (2011), using 5-fold cross-validation. Classification outcomes are shown in Figure 10a, in which the numbers of correctly classified trials are illustrated separately for the two classes and for each subject. The same data are represented in terms of sample accuracies in Figure 10b (blue dots). To illustrate ground truth, we repeated the above procedure (of generating synthetic data and applying an SVM) 1 000 times and added a contour plot of the resulting distribution of sample accuracies in the same figure. This dis-

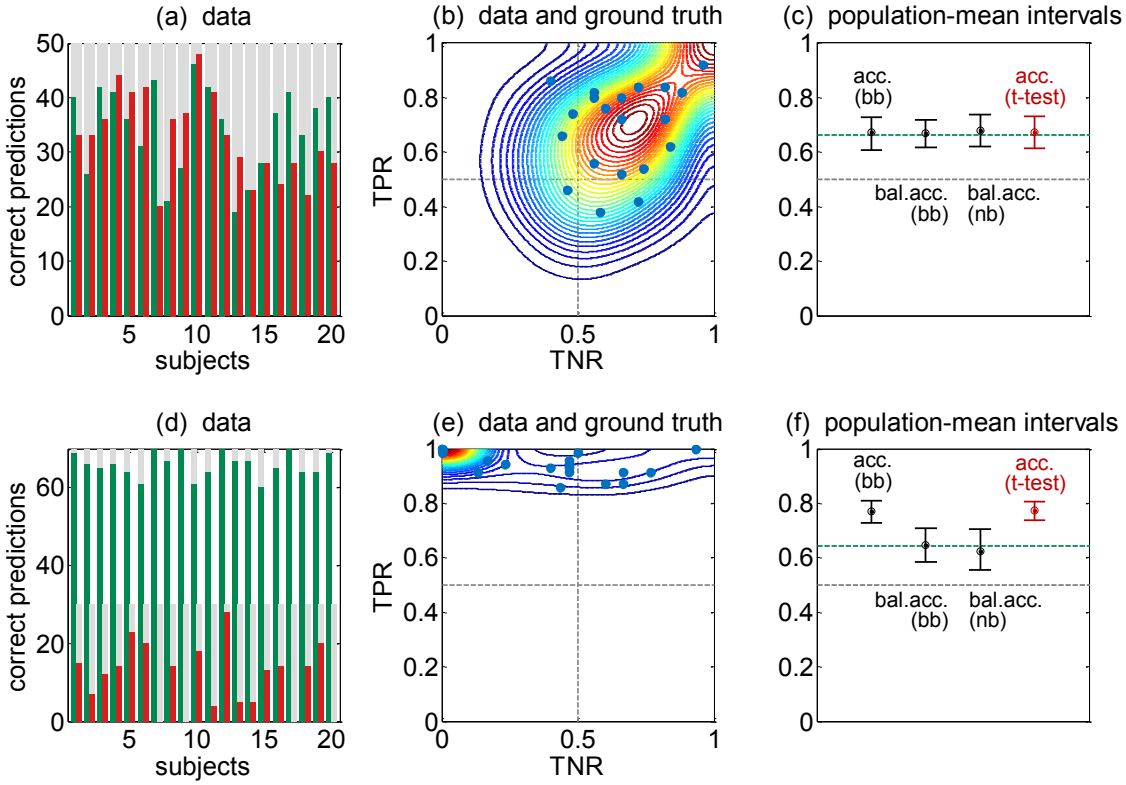


Figure 10: Application to synthetic data. (a) Classification outcomes obtained by applying a linear support vector machine to synthetic data, using 5-fold cross-validation. (b) Sample accuracies on positive (TPR) and negative classes (TNR) show the positive correlation between class-specific accuracies (blue). The underlying population distribution is represented by a bivariate Gaussian kernel density estimate (contour lines). (c) Central 95% posterior probability intervals, resulting from inversion of the beta-binomial model for inference on the population mean accuracy as well as the twofold beta-binomial model (*bb*) and the bivariate normal-binomial model (*nb*) for inference on the population mean balanced accuracy (all black). A frequentist 95% confidence interval (red) is shown for comparison. Bayesian model selection yielded very strong evidence (Kass and Raftery, 1995) in favour of the normal-binomial model (posterior model probability = 99.99%). (d) A second simulation was based on a synthetic heterogeneous group with varying numbers of trials. (e) In this case, the classifier acquires a strong bias in favour of the majority class. (f) As a result, inference on the accuracy is misleading; the balanced accuracy is a much better performance indicator, whether based on the beta-binomial (*bb*) or normal-binomial model (*nb*).

tribution was symmetric with regard to class-specific accuracies while these accuracies themselves were strongly positively correlated, as one would expect from a linear classifier tested on perfectly balanced data sets.

We applied all three models discussed in this paper for inference: the beta-binomial model for inference on the accuracy (Section 2.1), and the twofold beta-binomial and normal-binomial model for inference on the balanced accuracy (Sections 2.2 and 2.3). Central 95% posterior probability intervals about the population mean are shown in Figure 10c, along with a frequentist confidence interval of the population mean accuracy. All four approaches provided precise intervals around the true population mean. Comparing the two competing models for inference on the balanced accuracy, we obtained a log Bayes factor of 22.1 in favour of the twofold beta-binomial model (posterior model probability $> 99.99\%$; standard deviation of log Bayes factor across computations ≈ 5.1), representing very strong evidence (Kass and Raftery, 1995) that this model provided a better explanation of the data (i.e., a better balance between fit and complexity) than the bivariate normal-binomial model. This finding makes sense in light of the posterior covariance matrix of the normal-binomial model (cf. Figure 2b). Its off-diagonal elements (accounting for the potential dependency between class-specific accuracies) did not only have a very small mean ($\Sigma_{12} = \Sigma_{21} = 0.19$); they were also associated with considerable posterior uncertainty (95% credible interval $[-0.01, 0.44]$). In other words, the small additional fit provided by the off-diagonal elements was outweighed by the additional model complexity incurred.

We repeated the above analysis with a subtle but important modification: instead of using perfectly balanced data (50 positive and 50 negative trials), we created imbalanced synthetic data using 70 positive and 30 negative trials per subject. All other details of the analysis remained unchanged (Figure 10, bottom row). We observed that, as expected, the class imbalance caused the classifier to acquire a bias in favour of the majority class. This can be seen from the raw classification outcomes in which many more positive trials (green) than negative trials (red) were classified correctly, relative to their respective prevalence in the data (grey; Figure 10d). The bias is reflected accordingly by the estimated bivariate density of class-specific classification accuracies, in which the majority class consistently performs well whereas the accuracy on the minority class varies strongly (Figure 10e). In this setting, we found that both the twofold beta-binomial model and the normal-binomial model provided excellent estimates of the true balanced accuracy (Figure 10f; log Bayes factor in favour of the beta-binomial model: 47.3; standard deviation 11.3). In stark contrast, using the single beta-binomial model or a conventional mean of sample accuracies to infer on the population accuracy (as opposed to balanced accuracy) resulted in estimates that were overly optimistic and therefore misleading.

3.5 Application to Empirical Data

In order to illustrate the practical application of the approaches discussed in this paper, we analysed a neuroimaging data set, obtained by functional magnetic resonance imaging (fMRI). In neuroimaging, classifiers are often used as part of decoding models designed to infer a perceptual or cognitive state from brain activity, typically on a trial-by-trial basis, but across a group of subjects. The interpretation of the ensuing results critically relies on the validity of the models used for inference on classification performance.

Here, we analysed data from an fMRI experiment involving 16 volunteers designed to study the cognitive processes underlying decision making. During the experiment, subjects had to choose, on each trial, between two alternative options. Choices were indicated by button press (left/right index finger). Over the course of the experiment, subjects learned, by trial and error, the reward probabilities of these two options. Details on experimental design, data acquisition, and preprocessing can

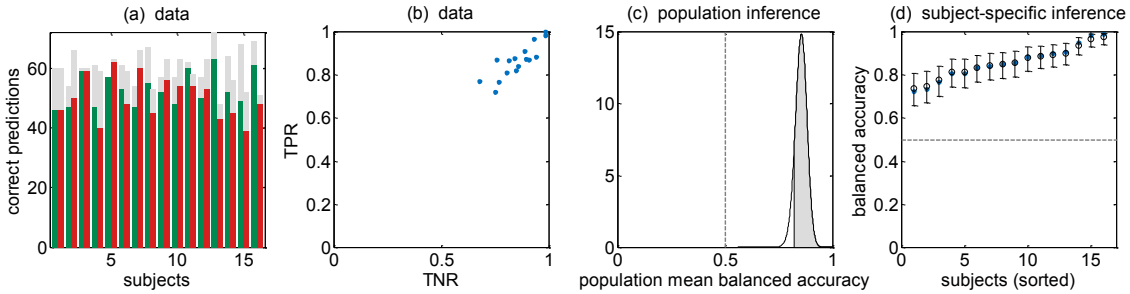


Figure 11: Application to empirical data. (a) Classification outcomes obtained by applying a linear SVM to trial-wise fMRI data from a decision-making task. (b) Replotting classification outcomes in terms of sample accuracies on positive (TPR) and negative trials (TNR) reveals the positive correlation between class-specific accuracies. (c) In this data set, when inferring on the balanced accuracy, the bivariate normal-binomial model has a higher evidence (marginal likelihood) than the twofold beta-binomial model. Inverting the former model, which captures potential dependencies between class-specific accuracies, yields a posterior distribution over the population mean balanced accuracy (black) which shows that the classifier is performing above chance. (d) The same model can be used to obtain subject-specific posterior inferences. The plot contrasts sample accuracies (blue) with central 95% posterior probability intervals (black), which avoid overfitting by shrinking to the population mean.

be found elsewhere (Behrens et al., 2007). Here, we predicted from the fMRI data, on a trial-by-trial basis, which option had been chosen. Because choices were indicated by button presses, we expected highly discriminative activity in the primary motor cortex.

Separately for each subject, a general linear model (Friston et al., 1995) was used to create a set of parameter images representing trial-specific estimates of evoked brain activity in each volume element. These images were used for subsequent classification. We trained a linear support vector machine (SVM) using 5-fold cross-validation. Comparing predicted to actual choices resulted in 120 classification outcomes for each of the 16 subjects. These data were used for inference on the classification accuracy using the beta-binomial model (Figure 11).

As can be seen from raw classification outcomes, class-specific accuracies seemed to be positively correlated (Figures 11a,b), in a similar way as for the synthetic data considered above. Thus, we used both the twofold beta-binomial model and the bivariate normal-binomial model for inference. Bayesian model comparison yielded a log Bayes factor of 12.5 in favour of the beta-binomial model (standard deviation across computations ≈ 4.69), suggesting that the additional complexity of the normal-binomial model may not have balanced its higher flexibility in explaining the correlations between class-specific accuracies. Using the beta-binomial model for inference on the population mean balanced accuracy, we obtained very strong evidence (infraliminal probability $p < 0.001$) that the classifier was operating above chance (Figure 11c).

Inference on subject-specific accuracies yielded fairly precise posterior intervals (Figure 11d). The shrinkage effect in these subject-specific accuracies was rather small: the average absolute difference between sample accuracies and posterior means amounted to 1.39 percentage points.

Even the biggest observed shift among all subjects was no more than 3.05 percentage points (from a sample accuracy of 99.2% down to a posterior mean of 96.2%). This minor impact of shrinkage is expected given the relatively small number of subjects (16) and the relatively large number of trials per subject (120).

4. Discussion

Canonical classification algorithms are frequently used on multilevel or hierarchically structured data sets, where a classifier is trained and tested for each subject within a group. This paper showed how the evaluation of classification performance in this setting may benefit from mixed-effects models that explicitly capture the hierarchical structure of the data. We organize the following discussion around the three principal features of this approach.

4.1 Replacing Fixed-Effects by Mixed-Effects Models

The primary contribution of this paper is the introduction and analysis of several models for Bayesian mixed-effects inference for group-level classification studies. To capture the two key sources of variation in hierarchical data sets, we simultaneously account for fixed-effects (within-subjects) and random-effects (across-subjects) variance components. This idea departs from previous models which are widely used for classification studies but ignore within- or between-subjects variability. Fixed-effects models make inappropriate assumptions and yield overconfident inference. Conversely, random-effects models treat subject-specific sample accuracies as observed, rather than inferred, and thus omit uncertainty associated with such sample accuracies.

The mixed-effects models considered in this paper ensure that known dependencies between inferences on subject-specific accuracies are accommodated within an internally consistent representation of the data. Specifically, the posterior distribution of the accuracy of one subject is partially influenced by the data from all other subjects, correctly weighted by their respective posterior precisions (see Section 3.2). Thus, the available group data are exploited to constrain individual inference appropriately. Non-hierarchical models, by contrast, risk being under-parameterized or over-parameterized. For example, pooling classification outcomes across subjects and modelling them as being drawn from a single distribution corresponds to an under-parameterized model whose single parameter (i.e., the latent accuracy) is insufficient to explain any population variability. Conversely, replicating the single-subject model in Equations (1)–(3) for each subject leads to an over-parameterized model with $2n$ parameters that is likely to overfit the data and generalize poorly. Hierarchical models overcome this problem in a natural way: they regularize the inversion problem by incorporating the structural dependencies that are assumed to govern the observed data.

An important aspect to keep in mind is that shrinkage is a posterior inference, and as such is conditional on the model. A corollary of this is that shrinkage is suboptimal when the hierarchical model structure represents an unreasonable assumption. This highlights the importance of model checking as an integral part of statistical inference. In particular, researchers applying the models proposed in this paper are advised to check whether the hierarchical structure of the models can be defended on substantive grounds. For example, in an experiment where each subject was either assigned to a treatment group or a control group, it may no longer be justified to treat their accuracies as conditionally independent and identically distributed given a single vector of population parameters; instead, it might be more appropriate to analyse the two subgroups separately (or augment the present models by a third level).

In those situations where a hierarchical structure is justified, we are not aware of alternatives that are superior to shrinkage. One possibility is to use *no pooling* of information across subjects, leading to a set of isolated subject-specific sample accuracies. Another possibility is *complete pooling*, leading to a single group mean accuracy. Between these two extremes lie the weighted estimates provided by a hierarchical model. Its shrinkage effect ensures that information from different sources is weighted correctly and incorporated into the posterior density of each model parameter.

Shrinkage is not a consequence of the Bayesian perspective adopted in this paper. It is a fundamental aspect of statistical dependencies in hierarchical structures which has been known for more than a century, dating back to work as early as Galton's law of 'regression towards mediocrity' (Galton, 1886). It is perfectly possible to obtain shrinkage through classical inference where it has undergone considerable scrutiny; one of the best-known examples is the James-Stein estimator (Appendix E) whose beneficial effect on estimation precision has long been recognized in frequentist statistics. For early contributions to the extensive literature on shrinkage effects, see Stein (1956), James and Stein (1961), and Efron and Morris (1971, 1972). For typical applications in other fields of science, see the many examples described by Gelman et al. (2003).

The hierarchical models presented in this paper are motivated by two-level designs that distinguish between inference at the subject level and inference at the group level. However, it should be noted that these models can easily be extended to accommodate multi-level studies. For example, in order to model classification performance in different task conditions or in different sessions, one could introduce separate latent accuracies $\pi_j^a, \pi_j^b, \ldots$, all of which are drawn from a common subject-specific accuracy π_j. In this way, one would explicitly model task- or session-specific accuracies to be conditionally independent from one another given an overall subject-specific effect π_j, and conditionally independent from other subjects given the population parameters. This example shows that additional relationships between portions of the acquired data can be naturally expressed in a hierarchical model to appropriately constrain inferences.

Mixed-effects models are not only useful when *evaluating* a classifier but also when *designing* it. For instance, Schelldorfer et al. (2011) proposed a linear mixed-effects model for classification that accounts for different sources of variation in the data. The model has been shown to improve classification performance in the domain of brain-computing interfaces (Fazli et al., 2011).

4.2 Replacing Frequentist by Bayesian Inference

The second feature of our approach is to provide Bayesian alternatives to the frequentist procedures that have been dominating classification group studies so far. Although these two schools share commonalities, there are primarily deep conceptual differences. Frequentist approaches consider the distribution of an estimator as a function of the unknown true parameter value and view probabilities as long-term frequencies; estimation yields point estimates and confidence intervals, while inference takes the form of statements on the probability of estimator values under a 'null hypothesis.' Bayesian methods, by contrast, consider the subjective belief about the parameter, before and after having observed actual data, drawing on probability theory to optimally quantify inferential uncertainty.

An additional aspect of Bayesian approaches is that one can evaluate different models by comparing their respective model evidences. This corresponds to inference about model structure as defined by the model's priors. For example, in Section 2.4 we showed how alternative *a priori* as-

sumptions about the population covariance of class-specific accuracies can be evaluated, relative to the priors of the models, using Bayesian model selection.

Bayesian inference in hierarchical models is typically analytically intractable, which is why we resort to approximate inference, for example by using stochastic approximation schemes based on MCMC methods. While computationally less efficient than deterministic approximations (e.g., variational Bayes, VB), these are easy to implement, avoid additional distributional assumptions, and are asymptotically exact. This paper exclusively relied on MCMC for model inversion. In future work, we will also provide VB algorithms for inverting models of the sort presented in this paper (see below).

It is worth noting that classical inference does not necessarily have to assume the form currently prevalent in the evaluation of hierarchical classification studies. For example, as noted by one of our reviewers, the t-test that is presently used by the large majority of classification analyses could be replaced by a classical mixed-effects model. This would require two things. Firstly, the definition of a decision statistic, for example, the fraction of correctly classified trials, pooled across subjects, or more simply, a hierarchical model such as the beta-binomial model, but estimated using maximum-likelihood estimation (for an example using logistic regression, see Dixon, 2008). Secondly, an inference scheme: under the null hypothesis that the classifiers perform at chance, the number of correctly/incorrectly classified trials can be swapped across subjects; this would provide a permutation mechanism to test the significance of the decision statistic.

An advantage of the above frequentist scheme would be that it no longer requires an assumption common to all other approaches considered in this paper: the assumption that trial-wise classification outcomes y_i are conditionally independent and identically distributed (i.i.d.) given a subject-specific accuracy π. This is typically justified by assuming that, in a classification analysis, test observations are i.i.d. themselves, conditional on the parameters of the latent process that generated the data. The situation is less clear in a cross-validation setting, where, strictly speaking, classification outcomes are no longer independent of one another (Kohavi, 1995; Wickenberg-Bolin et al., 2006; Gustafsson et al., 2010). Because violations of i.i.d. assumptions lead to conservative inference when controlling false positive rates, the i.i.d. assumption has generally not been a major concern in the literature; however, it remains a relevant topic, and further research into the ensuing statistical bias and its adequate correction is required. In the present paper, we used 5-fold cross-validation. If trial-by-trial dependence is an issue, then one possibility is to resort to a single-split (or hold-out) scheme, by training on one half of the data, and testing on the other (see Breiman and Spector, 1992, for details).

4.3 Replacing the Accuracy by the Balanced Accuracy

The third feature of our approach is its flexibility with regard to performance measures. While it is common to compare algorithms with regard to their accuracy, the limitations of this metric are well-known. For example, when a classifier is tested on an imbalanced data set, the accuracy may be inflated and lead to false conclusions about the classifier's performance. There are different potential solutions to this problem (Akbani et al., 2004; Chawla et al., 2002; Japkowicz and Stephen, 2002). One can, for example, restore balance by undersampling the large class or by oversampling the small class, or modify the costs of misclassification (Zhang and Lee, 2008). A more generic safeguard is to replace the accuracy with the balanced accuracy, defined as the arithmetic mean of

the class-specific accuracies. Unlike the measure described by Velez et al. (2007), the balanced accuracy is symmetric with respect to the type of class.[7]

Notably, the balanced accuracy is not confined to binary classification but can easily be generalized to K classes, by defining the balanced accuracy as the arithmetic mean of all K class-specific accuracies. For the twofold beta-binomial model, one could then replace π^+ and π^- by $\pi^{(1)}, \pi^{(2)}, \ldots, \pi^{(K)}$, whereas for the normal-binomial model, the bivariate normal distribution would be replaced by a K-dimensional normal distribution.

Using the example of the balanced accuracy, we have described how hierarchical models enable Bayesian inference on performance measures other than the accuracy. Future examples might include functional measures such as the receiver-operating characteristic (ROC) or the precision-recall curve (cf. Brodersen et al., 2010b). We also demonstrated that there may be multiple plausible models *a priori*. In this case, Bayesian model selection can be used to decide between competing models. Alternatively, Bayesian model averaging produces predictions which account for posterior model uncertainty. This approach can be adopted with any other performance measure of interest.[8]

The choice of a versatile yet convenient parameterization of the distributions for class-specific accuracies π^+ and π^- has been a recurring theme in the literature. Whereas early treatments adopted an empirical Bayes approach (e.g., Albert, 1984; Good, 1956; Griffin and Krutchkoff, 1971), the more recent literature has discussed various fully hierarchical approaches (see Agresti and Hitchcock, 2005, for an overview). For instance, Leonard (1972) proposed to replace independent Beta priors on each element of π, such as those in (2.2), by independent normal priors on each element of $\text{logit}(\pi)$. While this is analytically convenient, it requires independence assumptions in relation to the elements of π. This limitation was addressed by Berry and Christensen (1979), who placed a Dirichlet process prior on the elements of π. A related approach was proposed by Albert and Gupta (1983), who placed Beta priors on the components of π such that their degree of correlation could be controlled by a common hyperparameter. As mentioned above, a principled way of evaluating such different propositions rests upon Bayesian model comparison (MacKay, 1992; Madigan and York, 1997; Penny et al., 2004), which we illustrate by deciding between alternative parameterizations for inference on the balanced accuracy.

A similar approach to the one discussed in this article has been suggested by Olivetti et al. (2012), who carry out inference on the population mean accuracy by comparing two beta-binomial models: one with a population mean prior at 0.5 (i.e., chance), and one with a uniform prior on the interval $[0.5, 1]$. Inference then takes the form of model selection, resulting in a Bayes factor and its conventional interpretation (Kass and Raftery, 1995). Our approach differs from the above work in four ways: (i) in addition to classification accuracy, we consider the balanced accuracy, which is a more useful performance measure whenever the data are not perfectly balanced, and for which we offer different parameterizations that can be optimized using Bayesian model selection; (ii) we explicitly frame our approach in terms of fixed-effects (FFX), random-effects (RFX), and mixed-effects (MFX) inference, and we provide the respective graphical models; (iii) we emphasize the use of uninformative priors on the interval $[0, 1]$ to obtain unbiased posterior estimates, which allows us to use infraliminal probabilities for inference; (iv) finally, we provide extensive simulation results

7. If desired, this symmetry assumption can be dropped by introducing class-specific misclassification costs.

8. It should be noted that, in this context, model selection is carried out to ask which model best explains observed classification outcomes. This is different from asking what sort of model (i.e., classification algorithm) might be best at classifying the data in the first place.

that demonstrate the differences between FFX, RFX, and MFX approaches, shrinkage effects, and reduced estimation risks.

4.4 Summary of Present Results and Conclusions

To examine the properties of our approach and demonstrate its practical applicability, we reported several applications of the different models to synthetic and empirical data. Our results illustrated the characteristic features of our approach: (i) posterior densities as opposed to point estimates of parameters; (ii) the ability to compare alternative (non-nested) models; (iii) the 'shrinking-to-the-population' effect that regularizes estimates of classification performance in individual subjects (Figure 7b); (iv) increased sensitivity (Figure 7c); (v) more precise parameter estimates (Figure 7d); (vi) avoidance of classifier bias for imbalanced data sets using the balanced accuracy (Figure 8).

One practical limitation of our approach lies in the high computational complexity of our current inversion methods. In particular, our MCMC algorithms lack guarantees about convergence rates. Our algorithms also include heuristics regarding the number of burn-in samples, the precision of the overdispersed initial distributions and the proposal densities, and regarding the number of chains run in parallel. To address these issues, we are currently preparing a variational Bayesian approach that may offer computationally highly efficient model inversion.

We hope that the models for Bayesian mixed-effects analyses introduced in this paper will find widespread use, improving the sensitivity and validity of future classification studies at the group level. To facilitate the use of our approach, an open-source MATLAB implementation of all models discussed in this paper is available for download (`http://mloss.org/software/view/407/`).

Acknowledgments

This research was supported by the University Research Priority Program 'Foundations of Human Social Behaviour' at the University of Zurich (KHB, KES), by the SystemsX.ch project 'Neurochoice' (KHB, KES), and by the NCCR 'Neural Plasticity' (KES). The authors wish to thank Thomas Nichols, Timothy E.J. Behrens, Mark W. Woolrich, Adrian Groves, Ged Ridgway, and Anne Broger for insightful discussions during the course of this research.

Appendix A. Inversion of the Beta-Binomial Model

The algorithm is initialized by drawing initial values for $\alpha^{(0)}$ and $\beta^{(0)}$ from an overdispersed starting distribution. We represent these as

$$\omega^{(0)} = \left(\ln\left(\frac{\alpha^{(0)}}{\beta^{(0)}}\right), \ln\left(\alpha^{(0)} + \beta^{(0)}\right) \right)^{\mathrm{T}}.$$

This coordinate transformation makes sampling more efficient (Gelman et al., 2003). Subsequently, on each iteration τ, a new candidate ω^* is drawn from a symmetric proposal distribution

$$q_\tau\left(\omega^* \,\middle|\, \omega^{(\tau-1)}\right) = \mathcal{N}_2\left(\omega^* \,\middle|\, \omega^{(\tau-1)}, \begin{pmatrix} 1/8 & 0 \\ 0 & 1/8 \end{pmatrix}\right).$$

This candidate sample ω^* is accepted with probability

$$\min\left\{1,\ \frac{p(k_{1:m} \mid \alpha^*, \beta^*)\, p(\alpha^*, \beta^*)}{p(k_{1:m} \mid \alpha^{(\tau-1)}, \beta^{(\tau-1)})\, p(\alpha^{(\tau-1)}, \beta^{(\tau-1)})}\right\}$$
$$= \min\left\{1,\ \exp\left(\sum_{j=1}^{m} f(\alpha^*, \beta^*, k_j) - f(\alpha^{(\tau-1)}, \beta^{(\tau-1)}, k_j)\right)\right\}$$

where (7) and (9) (main text) were used in defining

$$f(\alpha, \beta, k) := \ln \mathrm{Bb}(k \mid \alpha, \beta) + \ln p(\alpha, \beta).$$

In order to assess whether the mean classification performance achieved in the population is above chance, we must evaluate our posterior knowledge about the population parameters α and β. Specifically, inference on $\alpha/(\alpha+\beta)$ serves to assess the mean accuracy achieved in the population. For example, its posterior expectation represents a point estimate that minimizes a squared-error loss function,

$$\mathbb{E}\left[\frac{\alpha}{\alpha+\beta} \;\middle|\; k_{1:m}\right] \approx \frac{1}{c}\sum_{\tau=1}^{c} \frac{\alpha^{(\tau)}}{\alpha^{(\tau)}+\beta^{(\tau)}}.$$

Another informative measure is the posterior probability that the mean classification accuracy in the population does not exceed chance,

$$p = \Pr\left(\frac{\alpha}{\alpha+\beta} \le 0.5 \;\middle|\; k_{1:m}\right) \approx \#\left\{\frac{\alpha^{(\tau)}}{(\alpha^{(\tau)}+\beta^{(\tau)}} \le 0.5\right\},$$

which we refer to as the (posterior) infraliminal probability of the classifier. The symbol $\#\{\cdot\}$ denotes a count of samples.

When we are interested in the classification accuracies of individual subjects, we wish to infer on $p(\pi_j \mid k_{1:m})$. This expression fully characterizes our posterior uncertainty about the true classification accuracy in subject j. Given a pair of samples $\alpha^{(\tau)}, \beta^{(\tau)}$, we can obtain samples from these posterior distributions simply by drawing from

$$\mathrm{Beta}\left(\pi_j^{(\tau)} \;\middle|\; \alpha^{(\tau)} + k_j,\ \beta^{(\tau)} + n_j - k_j\right).$$

This can be derived by relating the full conditional $p(\pi_j \mid \alpha, \beta, \pi_{1:j-1, j+1:m}, k_{1:m})$ to the closed-form posterior in (3) (see main text; cf. Gelman et al., 2003).

In order to infer on the performance that may be expected in a new subject from the same population, we are interested in the posterior predictive density,

$$p(\tilde{\pi} \mid k_{1:m}),$$

in which $\tilde{\pi}$ denotes the classification accuracy in a new subject drawn from the same population as the existing group of subjects with latent accuracies $\pi_1, \ldots, \pi_m$.[9] Unlike the posterior on $\alpha/(\alpha+\beta)$,

9. As noted before, the term 'posterior predictive density' is sometimes exclusively used for densities over variables that are unobserved but observable in principle. Here, we use the term to refer to the posterior density of any unobserved variable, whether observable in principle (such as $\tilde{k}$) or not (such as $\tilde{\pi}$).

the posterior predictive density on $\tilde{\pi}$ reflects both the mean and the variance of the performance achieved in the population.[10]

In order to derive an expression for the posterior predictive distribution in closed form, one would need to integrate out the population parameters α and β,

$$p(\tilde{\pi} \mid k_{1:m}) = \iint p(\tilde{\pi} \mid \alpha, \beta)\, p(\alpha, \beta \mid k_{1:m})\, \mathrm{d}\alpha\, \mathrm{d}\beta,$$

which is analytically intractable. However, the integral shows that values can be drawn from the posterior predictive density on $\tilde{\pi}$ using a single ancestral-sampling step. Specifically, within each iteration τ, the current samples $\alpha^{(\tau)}$ and $\beta^{(\tau)}$ can be used to obtain a new sample $\tilde{\pi}^{(\tau)}$ by drawing from

$$\mathrm{Beta}\left(\tilde{\pi}^{(\tau)} \;\middle|\; \alpha^{(\tau)}, \beta^{(\tau)}\right).$$

Once a number of samples from $p(\tilde{\pi} \mid k_{1:m})$ have been obtained, summarizing posterior inferences becomes straightforward, for example, by reporting

$$p(\tilde{\pi} \leq 0.5) \approx \#\{\pi^{(\tau)} \leq 0.5\},$$

which represents the probability that the classifier, when applied to a new subject from the same population, will not perform better than chance.

Appendix B. Bivariate Normal Prior

In order to illustrate the flexibility offered by the bivariate Normal density on ρ, we derive $p(\pi \mid \mu, \Sigma)$ in closed form and then compute the bivariate density on a two-dimensional grid. We begin by noting that

$$p_\pi(\pi \mid \mu, \Sigma) = p_\rho(\sigma^{-1}(\pi) \mid \mu, \Sigma) \left| \frac{d\sigma}{d\rho} \right|^{-1},$$

where we have added indices to p_π and p_ρ to disambiguate between the two densities, and where σ^{-1} denotes the logit transform. The Jacobian is

$$\frac{d\sigma}{d\rho} = \begin{pmatrix} \sigma'(\rho_1) & 0 \\ 0 & \sigma'(\rho_2) \end{pmatrix},$$

in which σ' represents the first derivative of the sigmoid transform. From this, we obtain the inverse determinant of the Jacobian as

$$\left| \frac{d\sigma}{d\rho} \right|^{-1} = \frac{1}{\sigma'(\rho_1)\,\sigma'(\rho_2)}.$$

Thus, the conditional bivariate density $p_\pi(\pi \mid \mu, \Sigma)$ is given by

$$p_\pi(\pi \mid \mu, \Sigma) = \mathcal{N}_2\left(\sigma^{-1}(\pi) \;\middle|\; \mu, \Sigma\right) \frac{1}{\sigma'(\sigma^{-1}(\pi_1))\,\sigma'(\sigma^{-1}(\pi_2)))}$$

10. If data were indeed obtained from a new subject (represented in terms of $\tilde{k}$ correct predictions in $\tilde{n}$ trials), then $p(\tilde{\pi} \mid k_{1:m}, n_{1:m})$ would be used as a prior to compute the posterior $p(\tilde{\pi} \mid \tilde{k}, \tilde{n}, k_{1:m}, n_{1:m})$.

where $\sigma^{-1}(\pi) := (\sigma^{-1}(\pi_1), \sigma^{-1}(\pi_2))^{\mathrm{T}}$. When evaluating this density on a $[0,1] \times [0,1]$ grid, the normalization constant is no longer needed, so that we can use the simpler expression

$$p_\pi(\pi \mid \mu, \Sigma) \propto \frac{1}{\pi_1 \pi_2 (1-\pi_1)(1-\pi_2)} \exp\left\{ -\frac{1}{2} \left(\sigma^{-1}(\pi) - \mu\right)^{\mathrm{T}} \Sigma^{-1} \left(\sigma^{-1}(\pi) - \mu\right) \right\},$$

where we have used the fact that $\sigma'(x) = \sigma(x)(1-\sigma(x))$. This derivation allows us to illustrate the degrees of freedom of our family of prior distributions over μ and Σ.

Appendix C. Inversion of the Bivariate Normal-Binomial Model

The algorithm is initialized by drawing initial values for $\mu^{(0)}$, $\Sigma^{(0)}$, and $\rho_1^{(0)}, \ldots, \rho_m^{(0)}$ from overdispersed starting distributions. On each iteration $\tau = 1 \ldots c$, we then update one variable after another, by sampling from the full conditional distribution of one variable given the current values of all others.[11] We begin by finding a new sample $(\mu, \Sigma)^{(\tau)}$, which can be implemented in a two-step procedure (Gelman et al., 2003). We first set

$$\begin{aligned}
\kappa_m &= \kappa_0 + m \\
\nu_m &= \nu_0 + m \\
\mu_m &= \frac{\kappa_0}{\kappa_m}\mu_0 + \frac{m}{\kappa_m}\rho^{(\tau-1)} \\
S &= \Sigma_{j=1}^{m} \left(\rho_j^{(\tau-1)} - \rho^{(\tau-1)}\right)\left(\rho_j^{(\tau-1)} - \rho^{(\tau-1)}\right)^{\mathrm{T}} \\
\Lambda_m &= \Lambda_0 + S + \frac{\kappa_0 m}{\kappa_m}\left(\rho^{(\tau-1)} - \mu_0\right)\left(\rho^{(\tau-1)} - \mu_0\right)^{\mathrm{T}},
\end{aligned}$$

where $\rho^{(\tau-1)} = \frac{1}{m}\sum_{j=1}^{m} \rho^{(\tau-1)}$, to draw

$$\Sigma^{(\tau)} \sim \text{Inv-Wishart}_{\nu_m}\left(\Sigma^{(\tau)} \,\middle|\, \Lambda_m^{-1}\right).$$

We then complete the first step by drawing

$$\mu^{(\tau)} \sim \mathcal{N}_2\left(\mu^{(\tau)} \,\middle|\, \mu_m, \Sigma^{(\tau)}/\kappa_m\right),$$

which we can use to obtain samples from the posterior mean balanced accuracy using

$$\phi^{(\tau)} := \frac{1}{2}\left(\mu_1^{(\tau)} + \mu_2^{(\tau)}\right).$$

Next, we update the bivariate variables $\rho_1, \ldots, \rho_m$. For each subject j, we wish to draw from the full conditional distribution

$$\begin{aligned}
&p\left(\rho_j^{(\tau)} \,\middle|\, k_{1:m}^{+}, k_{1:m}^{-}, \rho_{1:j-1}^{(\tau)}, \rho_{j+1:m}^{(\tau-1)}, \mu^{(\tau)}, \Sigma^{(\tau)}\right) \\
&= p\left(\rho_j^{(\tau)} \,\middle|\, k_j^{+}, k_j^{-}, \mu^{(\tau)}, \Sigma^{(\tau)}\right),
\end{aligned} \tag{18}$$

11. Here, we define one iteration as an update of all latent variables. Alternatively, one might update only one variable (or a subset of variables) per iteration, chosen randomly or systematically, as long as each variable is updated periodically.

which we have simplified by omitting all variables that are not part of the Markov blanket of ρ_j (cf. Figure 2b). Because we cannot sample from this distribution directly, we generate a candidate from a symmetric proxy distribution

$$q(\rho_j^*) = \mathcal{N}_2\left(\rho_j^* \,\middle|\, \rho_j^{(\tau-1)}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}^{\mathrm{T}}\right),$$

and then construct a Metropolis acceptance test. For this, we note that

$$\begin{aligned}
&p\left(\rho_j^* \,\middle|\, k_j^+, k_j^-, \mu^{(\tau)}, \Sigma^{(\tau)}\right) \\
&\propto \tilde{p}\left(\rho_j^* \,\middle|\, k_j^+, k_j^-, \mu^{(\tau)}, \Sigma^{(\tau)}\right) && (19)\\
&= p\left(k_j^+, k_j^- \,\middle|\, \rho_j^*, \mu^{(\tau)}, \Sigma^{(\tau)}\right)\; p\left(\rho_j^* \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right) && (20)\\
&= p\left(k_j^+, k_j^- \,\middle|\, \rho_j^*\right)\; p\left(\rho_j^* \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right) && (21)\\
&= p\left(k_j^+ \,\middle|\, \rho_{j,1}^*\right)\; p\left(k_j^- \,\middle|\, \rho_{j,2}^*\right)\; p\left(\rho_j^* \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right) && (22)\\
&= \mathrm{Bin}\left(k_j^+ \,\middle|\, \sigma(\rho_{j,1}^*)\right) \mathrm{Bin}\left(k_j^- \,\middle|\, \sigma(\rho_{j,2}^*)\right) \mathcal{N}_2\left(\rho_j^* \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right), && (23)
\end{aligned}$$

where (19) places our focus on the unnormalized density, (20) uses Bayes' theorem, (21) is based on the Markov blanket, (22) exploits the conditional independence of class-specific outcomes k_j^+ and k_j^-, and (23) relies on the model assumptions introduced in (4) and (12) (main text). We can use this result to accept the candidate sample ρ_j^* with probability

$$\min\{1, \exp(r)\},$$

where

$$\begin{aligned}
r &= \ln \frac{\tilde{p}\left(\rho_j^* \,\middle|\, k_j^+, k_j^-, \mu^{(\tau)}, \Sigma^{(\tau)}\right)}{\tilde{p}\left(\rho_j^{(\tau-1)} \,\middle|\, k_j^+, k_j^-, \mu^{(\tau)}, \Sigma^{(\tau)}\right)} \\
&= \ln \mathrm{Bin}\left(k_j^+ \,\middle|\, \sigma\left(\rho_{j,1}^*\right)\right) + \ln \mathrm{Bin}\left(k_j^- \,\middle|\, \sigma\left(\rho_{j,2}^*\right)\right) + \ln \mathcal{N}_2\left(\rho_j^* \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right) \\
&\quad - \ln \mathrm{Bin}\left(k_j^+ \,\middle|\, \sigma\left(\rho_{j,1}^{(\tau-1)}\right)\right) - \ln \mathrm{Bin}\left(k_j^- \,\middle|\, \sigma\left(\rho_{j,2}^{(\tau-1)}\right)\right) - \ln \mathcal{N}_2\left(\rho_j^{(\tau-1)} \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right).
\end{aligned}$$

We can now obtain samples from the posterior densities $p(\pi_j \mid k_{1:m}^+, k_{1:m}^-)$ for each subject j simply by sigmoid-transforming the current sample,

$$\pi_j^{(\tau)} = \sigma\left(\rho_j^{(\tau)}\right).$$

Based on this, we can obtain samples from the subject-specific balanced accuracies by setting

$$\phi_j^{(\tau)} := \frac{1}{2}\left(\pi_{j,1}^{(\tau)} + \pi_{j,2}^{(\tau)}\right).$$

Apart from using $\mu^{(\tau)}$ and $\Sigma^{(\tau)}$ to obtain samples from the posterior distributions over ρ_j, we can further use the two vectors to draw samples from the posterior predictive distribution $p(\tilde{\pi}^+_{1:m}, k^-_{1:m})$. For this we first draw

$$\tilde{\rho}^{(\tau)} \sim \mathcal{N}_2\left(\tilde{\rho}^{(\tau)} \,\middle|\, \mu^{(\tau)}, \Sigma^{(\tau)}\right),$$

and then obtain the desired sample using

$$\tilde{\pi}^{(\tau)} = \sigma\left(\tilde{\rho}^{(\tau)}\right),$$

from which we can obtain samples from the posterior predictive balanced accuracy using

$$\tilde{\phi}^{(\tau)} := \frac{1}{2}\left(\tilde{\pi}^{(\tau)}_1 + \tilde{\pi}^{(\tau)}_2\right).$$

In all above cases, we can use the obtained samples to compute approximate posterior probability intervals or Bayesian p-values.

The approximate expression for the model evidence in (16) can be obtained as follows:

$$\begin{aligned}
&\ln p(k^+_{1:m}, k^-_{1:m} \mid M_{nb}) \qquad (24)\\
&= \ln \int p(k^+_{1:m}, k^-_{1:m} \mid \rho_{1:m})\, \mathrm{d}\rho_{1:m}\\
&= \ln \left\langle p(k^+_{1:m}, k^-_{1:m} \mid \rho_{1:m}) \right\rangle_{\rho_{1:m}}\\
&= \ln \left\langle \prod_j^m p(k^+_j, k^-_j \mid \rho_j) \right\rangle_{\rho_{1:m}}\\
&= \ln \left\langle \prod_j^m p(k^+_j \mid \rho^{(1)}_j)\; p(k^-_j \mid \rho^{(2)}_j) \right\rangle_{\rho_{1:m}}\\
&\approx \ln \frac{1}{c} \sum_{\tau=1}^{c} \prod_j^m p\left(k^+_j \mid \rho^{(\tau,1)}_j\right) p\left(k^-_j \mid \rho^{(\tau,2)}_j\right)\\
&= \ln \frac{1}{c} \sum_{\tau=1}^{c} \prod_j^m \mathrm{Bin}\left(k^+_j \,\middle|\, \sigma\left(\rho^{(\tau,1)}_j\right)\right) \mathrm{Bin}\left(k^-_j \,\middle|\, \sigma\left(\rho^{(\tau,2)}_j\right)\right)
\end{aligned}$$

Appendix D. Comparison to Classical Inference

In a maximum-likelihood (ML) setting, one typically aims to obtain a point estimate for π, the true accuracy of the classifier under the binomial model.

D.1 Classical Inference for a Single Subject

In the case of a single-subject setting, the ML estimate for π is

$$\hat{\pi}_{\mathrm{ML}} = \arg\max_\pi \mathrm{Bin}(k \mid \pi, n) = \frac{k}{n},$$

which corresponds to the *sample accuracy*, that is, the number of correctly classified trials divided by the total number of trials.

Classical inference in the binomial model proceeds by asking how probable the observed value (or greater values) of the estimator are, assuming that the true accuracy π is at chance. This tests the null hypothesis $H_0 : \pi = 0.5$, yielding a p-value,

$$p = 1 - \mathcal{F}_{\text{Bin}}(k \mid 0.5),$$

where $\mathcal{F}_{\text{Bin}}(k \mid 0.5)$ is the cumulative distribution function of the binomial distribution with $\pi = 0.5$.

The practical simplicity of maximum likelihood is offset by its conceptual limitations. Specifically, using the sample accuracy k/n to estimate the true accuracy π risks overfitting. Furthermore, a point estimate for π ignores both (prior and posterior) uncertainty about classification performance.

D.2 Classical Inference in a Group Study

In a hierarchical setting, group-level inference frequently proceeds by applying a one-sample, one-tailed t-test to subject-specific sample accuracies.[12] This tests the null hypothesis that subject-specific accuracies are drawn from a distribution with a mean at chance level, using the t-statistic

$$\sqrt{m}\frac{\pi - \pi_0}{\hat{\sigma}_{m-1}} \sim t_{m-1}, \tag{25}$$

where π and $\hat{\sigma}_{m-1}$ are the sample mean and sample standard deviation of subject-specific sample accuracies, π_0 is the accuracy at chance (e.g., 0.5 for binary classification), and t_{m-1} is Student's t-distribution on $m-1$ degrees of freedom.

Additionally, it is common practice to indicate the uncertainty about the population mean of the classification accuracy by reporting the 95% confidence interval

$$\left[\pi \pm t_{0.025,m-1} \times \frac{\hat{\sigma}_{m-1}}{\sqrt{m}}\right], \tag{26}$$

where $t_{0.025,m-1}$ is a quantile from the t-distribution. It is worth emphasizing that this confidence interval has a merely illustrative purpose. This is because a central interval corresponds to a two-tailed test, whereas the t-test above is one-tailed. Since it is based on Gaussian assumptions, a one-tailed confidence interval would include the entire real line up to $+\infty$. Thus, a (two-sided) confidence-interval test actually has a false positive rate of $\alpha/2 = 0.025$. Similarly, under the null distribution, the 95% confidence interval will lie entirely below 0.5 in 2.5% of the cases. In a classical framework, one would have to call this 'significant,' in the sense of the classifier operating below chance. However, this is not the hypothesis one would typically want to test. Rather, it is more desirable to formulate a one-tailed test. In a Bayesian setting, this can be achieved by quantifying the (posterior) probability that the true accuracy is above chance.

Fundamentally, the differences between the classical procedure and the full Bayesian approach discussed earlier can best be understood by considering their respective assumptions. The distributional assumption underlying both the t-statistic in (25) and the confidence interval in (26) is that the sample mean of the subject-wise accuracies, under the null hypothesis, is normally distributed,

$$\pi \sim \mathcal{N}\left(\mu, \frac{\sigma}{\sqrt{m}}\right), \tag{27}$$

12. It should be noted that the present manuscript focuses on those classical procedures that are widely used in application domains such as neuroimaging and brain-machine interfaces. However, it is worth noting that alternative maximum-likelihood procedures exist that eschew the normality assumption implicit in a classical t-test (e.g., Dixon, 2008, see also Discussion).

where the population standard deviation σ has been estimated by the sample standard deviation $\hat{\sigma}_{m-1}$. The corresponding graphical model is shown in Figure 1d.

This analysis is popular but suffers from two faults that are remedied by our Bayesian treatment. (For a classical mixed-effects approach, see Discussion.) First, accuracies are confined to the $[0,1]$ interval, but are modelled by a normal distribution with infinite support. Consequently, error bars based on confidence intervals (26) may well include values above 1 (see Figure 6c for an example). By contrast, the Beta distribution used in the Bayesian approach has the desired $[0,1]$ support and thus represents a more natural candidate distribution.[13]

Second, the random-effects group analysis under (27) does not acknowledge within-subject estimation uncertainty and only provides a summary-statistics approximation to full mixed-effects inference. More specifically, the model is based on subject-wise sample accuracies π_j as the units of observation, rather than using the number of correctly classified trials k to infer on the accuracy in each subject. Put differently, the model assumes that subject-wise accuracies have all been estimated with infinite precision. But the precision is finite, and it varies both with the number of observed trials n_j and with the sample accuracy k_j/n_j. (This can be seen from the expression for the variance of a Bernoulli variable, which is largest at the centre of its support.) In summary, classifier performance cannot be observed directly; it must be inferred. While the classical model above does allow for inference on random-effects (between-subjects) variability, it does not explicitly account for fixed-effects (within-subject) uncertainty. This uncertainty is only taken into account indirectly by its influence on the variance of the observed sample accuracies.

With regard to subject-specific accuracies, one might be tempted to use $\hat{\pi}_j = k_j/n_j$ as individual estimates. However, in contrast to Bayesian inference on subject-specific accuracies (see Section 2.1), individual sample accuracies do not take into account the moderating influence provided by knowledge about the group (i.e., 'shrinkage'). An effectively similar outcome is found in classical inference using the James-Stein estimator (James and Stein, 1961, see Appendix E). All of these conceptual differences can be illustrated best using synthetic and empirical data, as described in Section 3.

Appendix E. Classical Shrinkage Using the James-Stein Estimator

When inferring on subject-specific accuracies π_j, the beta-binomial model uses data from the entire group to inform inferences in individual subjects. Effectively, subject-specific posteriors are 'shrunk' to the population mean. This is in contrast to using sample accuracies $\hat{\pi} = k_j/n_j$ as individual estimates. In classical inference, a similar shrinkage effect can be achieved using the positive-part James-Stein estimator (James and Stein, 1961). It is given by

$$\hat{\pi}^{\mathrm{JS}}_{1:m} = (1-\xi)\hat{\pi}_{1:m} + \xi\hat{\pi}_{1:m}$$
$$\xi = \left(1 - \frac{(m-2)\,\hat{\sigma}^2_m(\hat{\pi}_{1:m})}{\|\hat{\pi}_{1:m} - \hat{\pi}_{1:m}\|^2_2}\right)^+$$

where $\hat{\pi}_{1:m} = (k_j/n_j)_{1:m}$ is a vector of sample accuracies, $\hat{\pi}_{1:m}$ is its sample average, and $\hat{\sigma}^2_m$ denotes the population standard deviation. The weighing factor ξ balances the influence of the data ($\hat{\pi}_j$ for a given subject j) and the population ($\hat{\pi}_{1:m}$) on the estimate.

13. A classical approach to obtaining more reasonable confidence intervals would be to apply a logit transform or a z-transform to sample accuracies and then compute confidence intervals in the space of log odds or z-scores.

References

A. Agresti and D. B. Hitchcock. Bayesian inference for categorical data analysis. *Statistical Methods and Applications*, 14(3):297–330, Dec. 2005. ISSN 1618-2510. doi: 10.1007/s10260-005-0121-y.

R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004*, pages 39–50. 2004.

J. H. Albert. Empirical Bayes estimation of a set of binomial probabilities. *Journal of Statistical Computation and Simulation*, 20(2):129–144, 1984. ISSN 0094-9655.

J. H. Albert and A. K. Gupta. Estimation in contingency tables using prior information. *Journal of the Royal Statistical Society. Series B (Methodological)*, 45(1):60–69, Jan. 1983. ISSN 00359246.

T. Bayes and R. Price. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society*, 53:370–418, 1763. ISSN 0261-0523. doi: doi:10.1098/rstl.1763.0053.

M. Beal. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis, University College London, United Kingdom, 2003.

C. F. Beckmann, M. Jenkinson, and S. M. Smith. General multilevel linear modeling for group analysis in fMRI. *NeuroImage*, 20(2):1052–1063, Oct. 2003. ISSN 1053-8119. doi: 10.1016/S1053-8119(03)00435-X.

T. E. J. Behrens, M. W. Woolrich, M. E. Walton, and M. F. S. Rushworth. Learning the value of information in an uncertain world. *Nature Neuroscience*, 10:1214–1221, 2007.

D. A. Berry and R. Christensen. Empirical Bayes estimation of a binomial parameter via mixtures of Dirichlet processes. *The Annals of Statistics*, 7(3):558–568, May 1979. ISSN 0090-5364. doi: 10.1214/aos/1176344677.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer New York., 2007.

L. Breiman and P. Spector. Submodel selection and evaluation in regression. The x-random case. *International Statistical Review/Revue Internationale de Statistique*, page 291–319, 1992.

K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann. The balanced accuracy and its posterior distribution. In *Proceedings of the 20th International Conference on Pattern Recognition*, pages 3121–3124. IEEE Computer Society, 2010a. ISBN 1051-4651. doi: 10.1109/ICPR.2010.764.

K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann. The binormal assumption on precision-recall curves. In *Proceedings of the 20th International Conference on Pattern Recognition*, pages 4263–4266. IEEE Computer Society, 2010b. ISBN 1051-4651. doi: 10.1109/ICPR.2010.1036.

C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1—27:27, 2011.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(3):321–357, 2002.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, Oct. 1992. ISSN 0885-6125. doi: 10.1007/BF00994110.

G. V. Cormack. Email spam filtering: a systematic review. *Foundations and Trends in Information Retrieval*, 1(4):335–455, Apr. 2008. ISSN 1554-0669. doi: 10.1561/1500000006.

D. D. Cox and R. L. Savoy. Functional magnetic resonance imaging (fMRI) "brain reading": Detecting and classifying distributed patterns of fMRI activity in human visual cortex. *NeuroImage*, 19(2):261–270, 2003. ISSN 1053-8119. doi: 10.1016/S1053-8119(03)00049-1.

J. J. Deely and D. V. Lindley. Bayes empirical Bayes. *Journal of the American Statistical Association*, 76(376):833–841, Dec. 1981. ISSN 01621459. doi: 10.2307/2287578.

O. Demirci, V. P. Clark, V. A. Magnotta, N. C. Andreasen, J. Lauriello, K. A. Kiehl, G. D. Pearlson, and V. D. Calhoun. A review of challenges in the use of fMRI for disease classification / characterization and a projection pursuit application from a multi-site fMRI schizophrenia study. *Brain Imaging and Behavior*, 2(3):207–226, Aug. 2008. ISSN 1931-7557. doi: 10.1007/s11682-008-9028-1.

P. Dixon. Models of accuracy in repeated-measures designs. *Journal of Memory and Language*, 59 (4):447–456, Nov. 2008. ISSN 0749-596X. doi: 10.1016/j.jml.2007.11.004.

B. Efron and C. Morris. Limiting the risk of Bayes and empirical Bayes estimators - part I: the Bayes case. *Journal of the American Statistical Association*, pages 807–815, 1971.

B. Efron and C. Morris. Limiting the risk of Bayes and empirical Bayes estimators – part II: the empirical Bayes case. *Journal of the American Statistical Association*, page 130–139, 1972.

P. J. Everson and E. T. Bradlow. Bayesian inference for the beta-binomial distribution via polynomial expansions. *Journal of Computational and Graphical Statistics*, 11(1):202–207, Mar. 2002. ISSN 10618600.

S. Fazli, M. Danoczy, J. Schelldorfer, and K.-R. Müller. L1-penalized linear mixed-effects models for high dimensional data with application to BCI. *NeuroImage*, 56(4):2100–2108, June 2011. ISSN 1053-8119. doi: 16/j.neuroimage.2011.03.061.

K. J. Friston, A. P. Holmes, K. J. Worsley, J. P. Poline, C. D. Frith, and R. S. J. Frackowiak. Statistical parametric maps in functional imaging: A general linear approach. *Human Brain Mapping*, 2(4):189–210, 1995.

K. J. Friston, K. E. Stephan, T. E. Lund, A. Morcom, and S. Kiebel. Mixed-effects and fMRI studies. *NeuroImage*, 24(1):244–252, Jan. 2005. ISSN 1053-8119. doi: 10.1016/j.neuroimage.2004.08.055.

F. Galton. Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15:246–263, 1886.

A. Gelfand and A. Smith. Sampling-based approaches to computing marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2 edition, July 2003. ISBN 9781584883883.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, vol. 30, p. 712-727*, 6(1):721–741, 1984.

H. Goldstein. *Multilevel Statistical Models*, volume 847. Wiley, 2010.

I. J. Good. On the estimation of small frequencies in contingency tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 18(1):113–124, 1956. ISSN 0035-9246.

B. S. Griffin and R. G. Krutchkoff. An empirical Bayes estimator for P[success] in the binomial distribution. *The Indian Journal of Statistics, Series B*, 33(3/4):217–224, Dec. 1971. ISSN 05815738.

M. G. Gustafsson, M. Wallman, U. Wickenberg Bolin, H. Göransson, M. Fryknäs, C. R. Andersson, and A. Isaksson. Improving Bayesian credibility intervals for classifier error rates using maximum entropy empirical priors. *Artificial Intelligence in Medicine*, 49(2):93–104, June 2010. ISSN 0933-3657. doi: 10.1016/j.artmed.2010.02.004.

S. A. Harrison and F. Tong. Decoding reveals the contents of visual working memory in early visual areas. *Nature*, 458(7238):632–635, 2009. ISSN 0028-0836. doi: 10.1038/nature07832.

W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, page 361, 1961.

N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5):429–449, 2002.

F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer, 2007. ISBN 9780387682815.

R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90 (430):773–795, 1995. ISSN 01621459.

A. Knops, B. Thirion, E. M. Hubbard, V. Michel, and S. Dehaene. Recruitment of an area involved in eye movements during mental arithmetic. *Science (New York, N.Y.)*, 324(5934):1583–1585, May 2009. ISSN 1095-9203. doi: 10.1126/science.1171599.

R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*, volume 14, pages 1137–1145. Lawrence Erlbaum Associates Ltd., 1995.

I. Krajbich, C. Camerer, J. Ledyard, and A. Rangel. Using neural measures of economic value to solve the public goods free-rider problem. *Science (New York, N.Y.)*, 326(5952):596–599, Oct. 2009. ISSN 1095-9203. doi: 10.1126/science.1177302.

J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6:273–306, 2005.

P. S. Laplace. *Memoire sur la probabilité des causes par les évènemens*. De l'Imprimerie Royale, 1774.

J. C. Lee and D. J. Sabavala. Bayesian estimation and prediction for the beta-binomial model. *Journal of Business & Economic Statistics*, 5(3):357–367, July 1987. ISSN 07350015. doi: 10.2307/1391611.

T. Leonard. Bayesian methods for binomial data. *Biometrika*, 59(3):581–589, Dec. 1972. doi: 10.1093/biomet/59.3.581.

D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4:415–447, 1992.

D. Madigan and A. E. Raftery. Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546, Dec. 1994. ISSN 0162-1459. doi: 10.2307/2291017.

D. Madigan and J. C. York. Bayesian methods for estimation of the size of a closed population. *Biometrika*, 84(1):19 –31, 1997. doi: 10.1093/biomet/84.1.19.

D. Madigan, A. E. Raftery, C. Volinsky, and J. Hoeting. Bayesian model averaging. In *Proceedings of the AAAI Workshop on Integrating Multiple Learned Models, Portland, OR*, pages 77–83, 1996.

N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of American Statistical Association*, 44:335–341, 1949.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087, 1953. ISSN 00219606. doi: 10.1063/1.1699114.

E. Olivetti, S. Veeramachaneni, and E. Nowakowska. Bayesian hypothesis testing for pattern discrimination in brain decoding. *Pattern Recognition*, 45(6):2075–2084, June 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2011.04.025.

E. S. Pearson. Bayes' theorem, examined in the light of experimental sampling. *Biometrika*, 17 (3/4):388–442, 1925. ISSN 0006-3444.

W. D. Penny, K. E. Stephan, A. Mechelli, and K. J. Friston. Comparing dynamic causal models. *NeuroImage*, 22(3):1157–1172, 2004.

M. A. Pitt and I. J. Myung. When a good fit can be bad. *Trends in Cognitive Sciences*, 6(10), 2002.

C. P. Robert. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*. Springer, 2007. ISBN 9780387715988.

J. Schelldorfer, P. Bühlmann, and S. V. De Geer. Estimation for high-dimensional linear mixed-effects models using l1-penalization. *Scandinavian Journal of Statistics*, 38(2):197–214, 2011. ISSN 1467-9469. doi: 10.1111/j.1467-9469.2011.00740.x. URL `http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9469.2011.00740.x/abstract`.

A. Schurger, F. Pereira, A. Treisman, and J. D. Cohen. Reproducibility distinguishes conscious from nonconscious neural representations. *Science*, 327(5961):97–99, Jan. 2010. doi: 10.1126/science.1180029.

R. Sitaram, N. Weiskopf, A. Caria, R. Veit, M. Erb, and N. Birbaumer. fMRI brain-computer interfaces: A tutorial on methods and applications. *Signal Processing Magazine, IEEE*, 25(1): 95–106, 2008. ISSN 1053-5888. doi: 10.1109/MSP.2008.4408446.

J. G. Skellam. A probability distribution derived from the binomial distribution by regarding the probability of success as variable between the sets of trials. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2):257–261, 1948. ISSN 0035-9246.

C. Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley symposium on mathematical statistics and probability*, volume 1, page 197–206, 1956.

K. E. Stephan, W. D. Penny, J. Daunizeau, R. J. Moran, and K. J. Friston. Bayesian model selection for group studies. *NeuroImage*, 46(4):1004–1017, July 2009. ISSN 1095-9572. doi: 10.1016/j.neuroimage.2009.03.025.

D. R. Velez, B. C. White, A. A. Motsinger, W. S. Bush, M. D. Ritchie, S. M. Williams, and J. H. Moore. A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction. *Genetic Epidemiology*, 31(4):306–315, May 2007. ISSN 0741-0395. doi: 10.1002/gepi.20211.

U. Wickenberg-Bolin, H. Goransson, M. Fryknas, M. Gustafsson, and A. Isaksson. Improved variance estimation of classification performance via reduction of bias caused by small sample size. *BMC Bioinformatics*, 7(1):127, 2006. ISSN 1471-2105. doi: 10.1186/1471-2105-7-127.

I. A. Wood, P. M. Visscher, and K. L. Mengersen. Classification based upon gene expression data: bias and precision of error rates. *Bioinformatics*, 23(11):1363 –1370, June 2007. doi: 10.1093/bioinformatics/btm117.

D. Zhang and W. S. Lee. Learning classifiers without negative examples: A reduction approach. In *Third International Conference on Digital Information Management, 2008. ICDIM 2008*, pages 638 –643, 2008. doi: 10.1109/ICDIM.2008.4746761.

Journal of Machine Learning Research 13 (2012) 3177-3206 Submitted 1/12; Revised 10/12; Published 11/12

Quantum Set Intersection and its Application to Associative Memory

Tamer Salman TAMER@CS.TECHNION.AC.IL
Yoram Baram BARAM@CS.TECHNION.AC.IL
Department of Computer Science
Technion - Israel Institute of Technology
Haifa, 32000, Israel

Editor: Manfred Opper

Abstract

We describe a quantum algorithm for computing the intersection of two sets and its application to associative memory. The algorithm is based on a modification of Grover's quantum search algorithm (Grover, 1996). We present algorithms for pattern retrieval, pattern completion, and pattern correction. We show that the quantum associative memory can store an exponential number of memories and retrieve them in sub-exponential time. We prove that this model has advantages over known classical associative memories as well as previously proposed quantum models.

Keywords: associative memory, pattern completion, pattern correction, quantum computation, quantum search

1. Introduction

The introduction of Shor's algorithm for factoring numbers in polynomial time (Shor, 1994) has demonstrated the ability of quantum computation to solve certain problems more efficiently than classical computers. This perception was ratified two years later, when Grover (1996) introduced a sub-exponential algorithm for quantum searching a database.

The field of quantum computation is based on the combination of computation theory and quantum mechanics. Computation theory concerns the design of computational models and the study of their time and space complexities. Quantum mechanics, on the other hand, concerns the study of systems governed by the rules of quantum physics. The combination of the two fields addresses the nature of computation in the physical world. However, there is still no efficient reduction of quantum mechanical behavior to classical computation.

Quantum mechanics is a conceptual framework that mathematically describes physical systems. It is based on four postulates, known as the postulates of quantum mechanics (Nielsen and Chuang, 2000), which provide a connection between the physical world and mathematical formalism. Through these postulates, it is possible to better understand the nature of physical computation and what can be physically computed.

The first postulate states that a physical system is completely described by a state in a Hilbert space known as the state space. The second states that the evolution of a closed quantum system is described by a unitary transformation. The third states that a quantum measurement is described by a collection of measurement operators that satisfy the completeness equality, that is, the sum of all possible measurements adds up to 1. The forth states that the state space of a composite physical system is the tensor product of the state spaces of its components.

Next, we introduce the basic building blocks of quantum computation. The reader is referred to Nielsen and Chuang (2000) for a detailed introduction.

1.1 States and Qubits

The basic entity of classical computation is the classical bit. Each classical bit can have one of two values, 0 and 1. The state of any finite physical system that can be found in a finite number of states can be described by a string of bits. A string of n bits represents one of 2^n possible states of a system enumerated 0, ..., $2^n - 1$.

In quantum computation, the basic entity is called a qubit (quantum bit). The qubit can have the analogue values $|0\rangle$ and $|1\rangle$, known as the computational basis states, where $|\cdot\rangle$ is the Dirac notation. Yet, the qubit can also have any other value that is a linear combination of $|0\rangle$ and $|1\rangle$:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

where α and β are any complex numbers (called the amplitudes of the basis states 0 and 1, respectively), such that $|\alpha|^2 + |\beta|^2 = 1$. Consequently, the qubit can be in any one of an infinite number of states described by unit vectors in a 2-dimensional complex vector space. The unary representation of a qubit can be given as a vector of two values

$$|\Psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}.$$

Analogously to the classical bit string, qubit strings (or quregisters) describe the state of a system. A two qubit system comprising two qubits $|a\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|b\rangle = \gamma|0\rangle + \delta|1\rangle$ is described by the tensor product of the two qubits $|a,b\rangle \equiv |a\rangle \otimes |b\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle$.

1.2 Measurement

The measurement of a qubit reveals only one of two possible outcomes. The value of α and β cannot be extracted from the measurement of a qubit. Instead, when measuring the qubit $\alpha|0\rangle + \beta|1\rangle$ in the computational basis, the result can be either 0 or 1 with probabilities $|\alpha|^2$ and $|\beta|^2$ respectively. For example, the state $\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$, when measured, yields any of the two results 0 or 1 with probability $1/2$. The measurement operation is not reversible and, once made, the qubit no longer exists in its state before the measurement. Measurements can be performed in different bases. For example, measuring the qubit $\alpha|0\rangle + \beta|1\rangle$ in the Hadamard basis defined by the two basis states $|+\rangle \equiv \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$ and $|-\rangle \equiv \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)$ gives $|+\rangle$ with probability $\frac{(\alpha+\beta)^2}{2}$ and $|-\rangle$ with probability $\frac{(\alpha-\beta)^2}{2}$, since $\alpha|0\rangle + \beta|1\rangle = \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle$.

An n-qubit system can be either measured completely or partially. When measured partially, the unmeasured subsystem can retain quantum superposition and further quantum manipulations can be performed upon it. However, any measurement can be delayed to the end of the computation process.

1.3 Operators

In quantum computation, a system changes its state under a unitary quantum operator U from $|\Psi\rangle$ to $U|\Psi\rangle$. An operator U can be described as a $2^n \times 2^n$ matrix operating on the unary representation

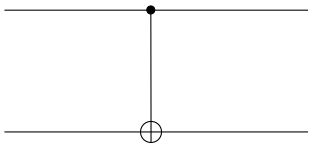


Figure 1: The 2-qubit Controlled-Not (CNOT) quantum gate.

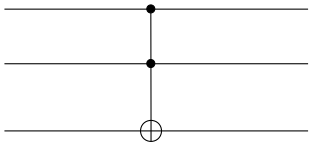


Figure 2: The 3-qubit Toffoli quantum gate.

of the system state. A unitary operator satisfies $UU^{\dagger} = I$, where $U^{\dagger}$ is the conjugate transpose of U (transpose the matrix U then substitute the conjugate complex of each element in the matrix).

Quantum operators can be implemented using quantum gates, which are the analogue of the classical gates that compose classical electrical circuits. In this analogy, the wires of a circuit carry the information on the system's state, while the quantum gates manipulate their contents to different states. For example, the Hadamard operator

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

transforms a qubit in the state $|0\rangle$ into the state $|+\rangle$ and the state $|1\rangle$ into the state $|-\rangle$.

Operators can also be quantum gates operating on multiple qubits. An n-qubit quantum operator has n inputs and n outputs. For example, the 2-qubit controlled-not (CNOT) gate depicted in Figure 1, flips the target (second) qubit if the control qubit (first) has value $|1\rangle$ and leaves it unchanged if the control qubit has the value $|0\rangle$. Specifically, the CNOT gate performs the following transformations on the four computational basis states: $|00\rangle \rightarrow |00\rangle$, $|01\rangle \rightarrow |01\rangle$, $|10\rangle \rightarrow |11\rangle$, and $|11\rangle \rightarrow |10\rangle$. It can be described as a unitary matrix operating on the unary representation of the state as follows:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Another gate is the 3-qubit controlled-controlled-not gate also known as the Toffoli gate depicted in Figure 2, which flips the target (third) qubit if the two control bits (first and second) both have the values $|1\rangle$ and leaves it unchanged otherwise.

The Hadamard operator can also be seen as operating on n-qubits by the tensor product of n single qubit Hadamard operators. Each qubit is then transformed according to the single qubit Hadamard transform, that is,

$$H^{\otimes n} |x_{n-1}, ..., x_1, x_0\rangle = H |x_{n_1}\rangle \otimes ... \otimes H |x_1\rangle \otimes H |x_0\rangle .$$

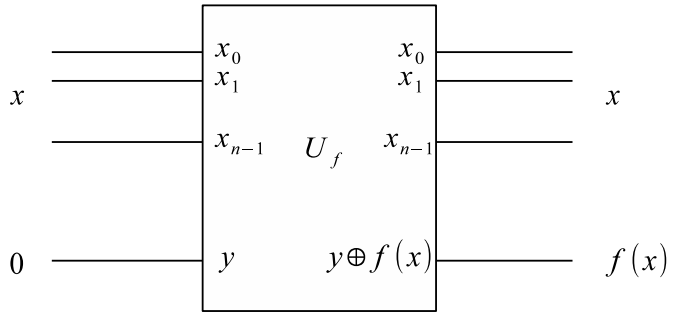


Figure 3: A classical reversible Oracle.

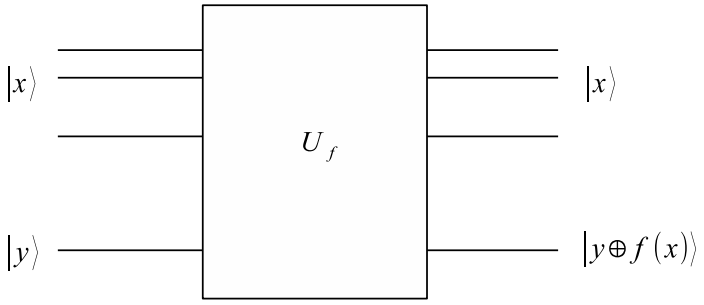


Figure 4: A quantum Oracle.

1.4 Quantum Parallelism and Interference

Consider an oracle of an n-dimensional function f implemented by a classical circuit, which, for every input x, produces $f(x)$ at the output. The oracle can be made reversible by adding an additional bit to the input, initialized by 0, and letting the output be $x, 0 \oplus f(x)$ as depicted in Figure 3, where $\oplus$ is the addition modulo 2. Initializing the additional bit with 1 produced $\overline{f(x)}$ at the output.

A quantum oracle is a reversible oracle that accepts a superposition of inputs and produces a superposition of outputs as depicted in Figure 4. When the additional qubit is initialized by $|0\rangle$, the oracle performs the following transformation: $|x\rangle|0\rangle \rightarrow |x\rangle|0 \oplus f(x)\rangle$. When the additional qubit is initialized by $|-\rangle$ the oracle is called a quantum phase oracle that gives $f(x)$ in the phase of the state $|x\rangle$ as follows: $|x\rangle|-\rangle \rightarrow (-1)^{f(x)}|x\rangle|-\rangle$.

Suppose that we have constructed a quantum circuit U_f that implements a function $f : \{0,1\}^n \rightarrow \{0,1\}$, such that when introduced with an input $|x\rangle|y\rangle$, the output of the circuit would be $|x\rangle|y \oplus f(x)\rangle$. Quantum parallelism is the ability of the quantum circuit to process many inputs simultaneously and receive all the outcomes at the output. Consider the case where $|y\rangle = |0\rangle$, and the $|x\rangle = H^{\otimes n}|0\rangle^{\otimes n}$. Applying the n-qubit Hadamard operator to the $|0\rangle$ state yields a superposition of all basis states

$\frac{1}{\sqrt{2^n}}\sum i=0^{2^n}|i\rangle$. The superposition will be maintained through the quantum circuit and the resulting output would be $\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n}|i\rangle|f(i)\rangle$. This can be computed as follows:

$$U_f^{\otimes n+1}\left(\left(H^{\otimes n}\otimes I\right)|0\rangle^{\otimes n}|0\rangle\right)=U_f^{\otimes n+1}\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n}|i\rangle|0\rangle=\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n}|i\rangle|f(i)\rangle.$$

Although this computes f for all values of i simultaneously there is no immediate way of seeing them all together, because once the output state is measured, only one value of $f(i)$ is revealed and the rest vanish. However, further quantum computation allows the different values to interfere together and reveal some information concerning the function f.

It has been shown (Deutsch, 1985; Deutsch and Jozsa, 1992; Simon, 1994) that the revealed information can be considerably more than what classical computation would achieve after one query of the circuit that implements the function f. Deutsch and Jozsa (1992) showed that if a binary n-dimensional function is guaranteed to be either constant or balanced, then determination can be done using only one query to a quantum oracle implementing it, while a classical solution would require an exponential number of queries. Another example of the advantage of quantum algorithms over classical ones was presented by Simon (1994), in which a function is known to have the property that there exists some $s\in\{0,1\}^n$ for which for all $x,y\in\{0,1\}^n$ it hold that $f(x)=f(y)$ if and only if $x=y$ or $x\oplus y=s$. Simon (1994) proved that using quantum computations, s can be found exponentially faster than with any classical algorithm, including probabilistic algorithms.

1.5 Solving Problems using Quantum Computation

According to the above definitions of a system state, measurement, and operators, a quantum computer drives the dynamics of the quantum system through operators that change its state and measures the final state to reveal classical information. In the general case, one might think of the process of quantum computation as a multi-phase procedure, which performs some classical computation on the data at hand, creates a quantum state describing the system, drives it through quantum operators, which might depend on the data, to the target state, measures the outcome, and performs some more classical computation to receive a result.

Consequently, we can describe a schematic process for solving problems using quantum computation as follows:

A general solution scheme using quantum computations

Given:

Classical input data

1. Preliminary classical computation of the input
2. Create initial quantum state
3. Apply quantum circuit to the initial quantum state
4. Measure the resulting state
5. Apply classical computation to the measured state

1.6 Grover's Quantum Search Algorithm

Given a database of $N\equiv 2^n$ unsorted elements of n bits each, any classical search would require $O(N)$ queries to find a desired element. In 1996, Grover presented a quantum computational algorithm that searches an unsorted database with $O(\sqrt{N})$ operations (Grover, 1996; Boyer et al., 1996).

The algorithm performs a series of $O(\sqrt{N})$ unitary operations on the superposition of all basis states that amplify the solution states causing the probability of measuring one of the solutions at the end of the computation to be close to 1.

Suppose that the search problem has a set X of r solutions and that we own an oracle function f_X that identifies the solution $x \in X$ according to the following:

$$f_X(x) = \begin{cases} 1, & x \in X \\ 0, & x \notin X \end{cases}.$$

Any classical algorithm that attempts to find the solution clearly needs to query the oracle N times in the worst case. Grover's algorithm shows that we can find the solution with the help of the oracle by querying it only $O(\sqrt{N})$ times.

The quantum phase oracle of the function f_X flips (rotates by π) the amplitude of the states of X, while leaving all other states unchanged. This is done by the operator $I_X = I - 2\sum_{x\in X} |x\rangle \langle x|$. In matrix formulation, I_X is similar to the identity matrix I except it has -1 on the xth elements of the diagonal.

Grover's algorithm starts with the superposition of all basis states created by applying the Hadamard operator on the zero state, $H^{\otimes n} |0\rangle^{\otimes n}$ (shortened by $H|0\rangle$), and goes about performing multiple iterations, in which each iteration consists of applying the phase oracle followed by the operator HI_0H, where I_0 flips the phase of the state $|0\rangle^{\otimes n}$. Grover's iterator is thus defined as

$$Q = -HI_0HI_X \tag{1}$$

where the sign "$-$" stands for the global phase flip that has no physical meaning and is performed only for analytical convenience.

The operator in Equation 1 can be viewed in the space defined by the two basis states

$$|l_1\rangle = \frac{1}{\sqrt{r}} \sum_{i\in X} |i\rangle$$

and

$$|l_2\rangle = \frac{1}{\sqrt{N-r}} \sum_{i\notin X} |i\rangle$$

as the rotation

$$\begin{pmatrix} 1-\frac{2r}{N} & 2\frac{\sqrt{r(N-r)}}{N} \\ -2\frac{\sqrt{r(N-r)}}{N} & 1-\frac{2r}{N} \end{pmatrix}$$

which is depicted in Figure 5, where the rotation angle is

$$w = \arccos\left(1 - \frac{2r}{N-r}\right).$$

The initial state has an angle

$$\phi = \arctan\left(\sqrt{\frac{r}{N-r}}\right)$$

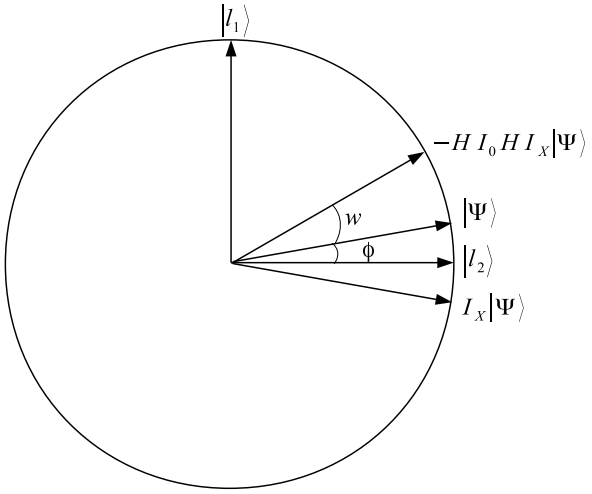


Figure 5: The effect of Grover's rotation on the state $|\Psi\rangle$.

with $|l_2\rangle$, and after some analysis one can find that applying the operator T times starting from the initial state yields a solution state with a maximal probability that is very close to 1 upon measurement when

$$T = \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{r}} \right\rfloor = O\left(\sqrt{\frac{N}{r}}\right).$$

The algorithm performs $O\left(\sqrt{\frac{N}{r}}\right)$ iterations, where $r = |X|$ is the number of marked states. Additional improvements were made by Boyer et al. (1996) and Brassard et al. (1998) coping with an unknown number of marked states in complexity $O\left(\sqrt{\frac{N}{r}}\right)$. Biham et al. (1999) introduced a third improvement that outputs a marked state when initiated with a state of an arbitrary amplitude distribution.

In the next sections, we present our algorithm for set intersection and its use in a quantum model of associative memory. We focus our analysis on the auto-associative memory model. However, presenting the algorithm with the quantum superposition of pairs x_i and y_i created by the use of the oracle B makes it a model for general associative memory with no additional cost, as a quantum search can yield y_i upon measurement of x_i.

1.7 Associative Memory

Associative memory stores and retrieves patterns with error correction or pattern completion of the input. The task can be defined as memorizing m pairs of data $\left(x^i, y^i\right)$, where x^i is an n dimensional vector and y^i is a q dimensional vector, and outputting y^i when presented with $\tilde{x}$, which is a faulty or a partial version of x^i. A specific case of associative memory is the auto-associative memory, in which $y^i \equiv x^i,\ \forall i \in \{1,\ldots,m\}$. Associative memory can be defined over a continuum, where $\left(x^i, y^i\right) \in R^n \times R^q$, or over a binary space, where $\left(x^i, y^i\right) \in \{0,1\}^n \times \{0,1\}^q$. In this paper, we concentrate on the binary model.

Algorithms for implementation of associative memory (Hopfield, 1982; Kanerva, 1993) have been extensively studied in the neural networks literature. The Hopfield model (Hopfield, 1982) consisting of n threshold neurons stores n-dimensional patterns $x \in \{\pm 1\}^n$ by the sum of outer products and retrieves a stored pattern when presented with a partial or a noisy version of the pattern.

The maximal storage capacity for which the stored patterns will be retrieved correctly with high probability (McEliece et al., 1987) is

$$M_{max} = \frac{n}{2\ln n}.$$

Sparse encoding has been shown to increase the storage capacity considerably (Baram, 1991). A capacity exponential in the input dimension has been shown to result from a network size also exponential in the input dimension (Baram and Sal'ee, 1992).

2. Quantum Intersection

Given a set of marked states K of size k, Grover's quantum search algorithm for multiple marked states yields any member of the set with probability $O\left(k^{-1/2}\right)$ when given a phase version I_K of an oracle f_K of the form

$$f_K(x) = \begin{cases} 1, & x \in K \\ 0, & x \notin K \end{cases}. \tag{2}$$

Consequently, given a phase oracle I_K, Grover's algorithm chooses a member of the subset K. Suppose that, in addition to the oracle f_K, we have an oracle f_M, such that

$$f_M(x) = \begin{cases} 1, & x \in M \\ 0, & x \notin M \end{cases} \tag{3}$$

where M is another set of size m of marked states.

We define the problem of quantum intersection as the choice of any member of the intersection set $K \cap M$ of size r with probability $O\left(r^{-1/2}\right)$.

A straightforward algorithm for finding a member of the intersection between two sets of marked states K and M, based on the oracles f_K and f_M involves the use of the intersection oracle, comprising the oracles in Equation 2, Equation 3 in sequence, and a Toffoli gate as depicted in Figure 6.

When the input to the oracle is $|x\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle$, then the output is $|x\rangle \otimes |f_K(x)\rangle \otimes |f_M(x)\rangle \otimes |f_{K\cap M}(x)\rangle$. The intersection phase oracle is realized when $|b\rangle = |-\rangle$, then, the input $|x\rangle \otimes |0\rangle \otimes |0\rangle \otimes |-\rangle$ will cause the output to be $|x\rangle \otimes |f_K(x)\rangle \otimes |f_M(x)\rangle \otimes \left[(-1)^{f_{K\cap M}(x)}|-\rangle\right]$.

Retrieving a state in the intersection between the two sets is accomplished by using Grover's quantum search algorithm with the phase version $I_{K\cap M}$ of the oracle $f_{M\cap K}$.

However, under certain conditions, one might not have the two oracles at hand, and, thus, the use of the intersection oracle might not be feasible. Another usage is when one constant oracle is available and the second oracle needs to be changed with each activation. We present an algorithm that applies a series of computations carried out by the two owners of the oracles in an alternating fashion.

Algorithm 1 : Quantum Set Intersection
Given: Phase oracles I_M and I_K
Denote: $Q_M \equiv -HI_0HI_M$, $Q_K \equiv -HI_0HI_K$
1. Let $|\Psi\rangle = H|0\rangle^{\otimes n}$

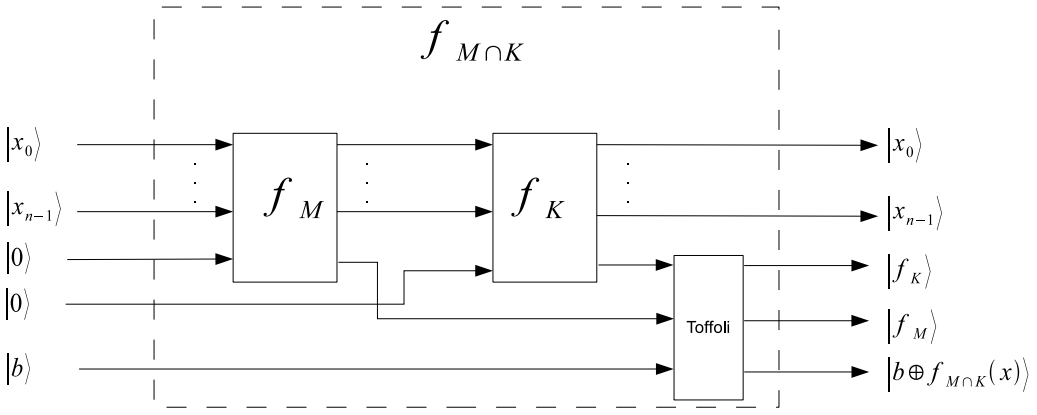


Figure 6: The intersection oracle $F_{K\cap M}$ created using f_M, f_K, and $Toffoli$.

2. Repeat

$|\Psi\rangle = Q_K|\Psi\rangle$

$|\Psi\rangle = Q_M|\Psi\rangle$

$T = O\left(\sqrt{\frac{N}{|K\cap M|}}\right)$ *times*

3. Measure $|\Psi\rangle$

Algorithm 1 assumes that the size of the intersection set $|K\cap M|$ is known in order to determine the number of iterations. In the more general case where $|K\cap M|$ is unknown we apply the modification for an unknown number of marked states (Boyer et al., 1996). Alteratively, we can use the quantum counting algorithm (Brassard et al., 1998) for the apriori estimation of the number of marked states. In both cases the time complexity of the set intersection algorithm is sub-exponential. Next, we present two theorems that prove that Algorithm 1 measures a member of the intersection set with maximal probability and this probability is asymptotically close to 1.

Theorem 1 *Let I_K and I_M be phase oracles that mark two sets of n-qubit states K and M with $|K|,|M| << N$. Let us denote $|K| = k$, $|M| = m$,$|K\cap M| = r$,*

$$Q \equiv Q_M Q_K = (HI_0HI_MHI_0HI_K) \tag{4}$$

and

$$|\Psi(t)\rangle = Q^T H|0\rangle.$$

Then, the maximal probability of measuring a state in the intersection $K\cap M$ is achieved at

$$T = \arg\max_t \sum_{x\in K\cap M} |\langle x|\Psi(t)\rangle|^2 = \left\lfloor \frac{\pi/2 - \arctan\left(\sqrt{\frac{r}{N-r}}\right)}{\arccos\left(\frac{4km}{N^2} - \frac{4r}{N} + \Gamma\right)} \right\rfloor \tag{5}$$

where

$$\Gamma = \sqrt{1 - \frac{8rN^3 + 8kmN^2 - 16rkN^2 - 16rmN^2 + 32rkmN - 16k^2m^2}{N^4}}.$$

Proof The Hilbert space spanned by all computational basis states of the n-qubit register can be divided into four different subspaces: the subspace spanned by the states in $K \cap M$, the subspace spanned by the states in $K \backslash M$, the subspace spanned by the states in $M \backslash K$, and the subspace spanned by the states in $\overline{K \cup M}$. Accordingly, we select an orthonormal basis for representing the operator, in which the first four basis states are:

$$|l_1\rangle \equiv \frac{1}{\sqrt{|K \cap M|}} \sum_{i \in K \cap M} |i\rangle, \tag{6}$$

$$|l_2\rangle \equiv \frac{1}{\sqrt{|K \backslash M|}} \sum_{i \in K \backslash M} |i\rangle, \tag{7}$$

$$|l_3\rangle \equiv \frac{1}{\sqrt{|M \backslash K|}} \sum_{i \in M \backslash K} |i\rangle, \tag{8}$$

$$|l_4\rangle \equiv \frac{1}{\sqrt{|\overline{K \cup M}|}} \sum_{i \in \overline{K \cup M}} |i\rangle \tag{9}$$

and the rest of the basis consists of orthonormal extensions of these states, then

$$H|0\rangle = \sqrt{\frac{r}{N}}|l_1\rangle + \sqrt{\frac{k-r}{N}}|l_2\rangle + \sqrt{\frac{m-r}{N}}|l_3\rangle + \sqrt{\frac{N-m-k+r}{N}}|l_4\rangle.$$

The operator Q_K affects only the four states $(|l_1\rangle, |l_2\rangle, |l_3\rangle, |l_4\rangle)$ as follows:

$$\begin{aligned} Q_K|l_1\rangle = \quad & -HI_0HI_K|l_1\rangle = -H(I-2|0\rangle\langle 0|)H\left(I-2\sum_{i\in K}|i\rangle\langle i|\right)|l_1\rangle = \\ & \left(1-\frac{2r}{N}\right)|l_1\rangle + \left(-\frac{2\sqrt{r(k-r)}}{N}\right)|l_2\rangle \\ + \quad & \left(-\frac{2\sqrt{r(m-r)}}{N}\right)|l_3\rangle + \left(-\frac{2\sqrt{r(N-m-k+r)}}{N}\right)|l_4\rangle, \end{aligned}$$

$$\begin{aligned} Q_K|l_2\rangle = \quad & -HI_0HI_K|l_2\rangle = -H(I-2|0\rangle\langle 0|)H\left(I-2\sum_{i\in K}|i\rangle\langle i|\right)|l_2\rangle = \\ & \left(-\frac{2\sqrt{r(k-r)}}{N}\right)|l_1\rangle + \left(1-\frac{2(k-r)}{N}\right)|l_2\rangle \\ + \quad & \left(-\frac{2\sqrt{(k-r)(m-r)}}{N}\right)|l_3\rangle + \left(-\frac{2\sqrt{(k-r)(N-m-k+r)}}{N}\right)|l_4\rangle, \end{aligned}$$

$$
\begin{aligned}
Q_K|l_3\rangle = \quad & -HI_0HI_K|l_3\rangle = -H(I-2|0\rangle\langle 0|)H\left(I-2\sum_{i\in K}|i\rangle\langle i|\right)|l_3\rangle = \\
& \left(\frac{2\sqrt{r(m-r)}}{N}\right)|l_1\rangle + \left(\frac{2\sqrt{(k-r)(m-r)}}{N}\right)|l_2\rangle \\
+ \quad & \left(\frac{2(m-r)}{N}-1\right)|l_3\rangle + \left(\frac{2\sqrt{(m-r)(N-m-k+r)}}{N}\right)|l_4\rangle,
\end{aligned}
$$

$$
\begin{aligned}
Q_K|l_4\rangle = \quad & -HI_0HI_K|l_4\rangle = -H(I-2|0\rangle\langle 0|)H\left(I-2\sum_{i\in K}|i\rangle\langle i|\right)|l_4\rangle = \\
& \left(\frac{2\sqrt{r(N-m-k+r)}}{N}\right)|l_1\rangle + \left(\frac{2\sqrt{(k-r)(N-m-k+r)}}{N}\right)|l_2\rangle \\
+ \quad & \left(\frac{2\sqrt{(m-r)(N-m-k+r)}}{N}\right)|l_3\rangle + \left(\frac{2(N-m-k-+r)}{N}-1\right)|l_4\rangle
\end{aligned}
$$

yielding Q_K in matrix form

$$
Q_K = \begin{pmatrix}
1-\frac{2r}{N} & \frac{-2\sqrt{r}\sqrt{k-r}}{N} & \frac{2\sqrt{r}\sqrt{m-r}}{N} & \frac{2\sqrt{r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{k-r}}{N} & 1-\frac{2(k-r)}{N} & \frac{2\sqrt{k-r}\sqrt{m-r}}{N} & \frac{2\sqrt{k-r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{m-r}}{N} & \frac{-2\sqrt{k-r}\sqrt{m-r}}{N} & \frac{2(m-r)}{N}-1 & \frac{2\sqrt{m-r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{N-k-m+r}}{N} & \frac{-2\sqrt{k-r}\sqrt{N-k-m+r}}{N} & \frac{2\sqrt{m-r}\sqrt{N-k-m+r}}{N} & \frac{2(N-k-m+r)}{N}-1
\end{pmatrix}. \tag{10}
$$

Similarly, we obtain the matrix form of Q_M

$$
Q_M = \begin{pmatrix}
1-\frac{2r}{N} & \frac{2\sqrt{r}\sqrt{k-r}}{N} & \frac{-2\sqrt{r}\sqrt{m-r}}{N} & \frac{2\sqrt{r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{k-r}}{N} & \frac{2(k-r)}{N}-1 & \frac{-2\sqrt{k-r}\sqrt{m-r}}{N} & \frac{2\sqrt{k-r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{m-r}}{N} & \frac{2\sqrt{k-r}\sqrt{m-r}}{N} & \frac{1-2(m-r)}{N} & \frac{2\sqrt{m-r}\sqrt{N-k-m+r}}{N} \\
\frac{-2\sqrt{r}\sqrt{N-k-m+r}}{N} & \frac{2\sqrt{k-r}\sqrt{N-k-m+r}}{N} & \frac{-2\sqrt{m-r}\sqrt{N-k-m+r}}{N} & \frac{2(N-k-m+r)}{N}-1
\end{pmatrix}. \tag{11}
$$

Substituting Equation 10 and Equation 11 into Equation 4 yields

$$
\begin{aligned}
Q \quad = \quad & \begin{pmatrix}
1-\frac{8r(N-m)}{N^2} & \frac{-4\sqrt{r}\sqrt{k-r}(N-2m)}{N^2} & 0 & 0 \\
\frac{-4\sqrt{r}\sqrt{k-r}(N-2m)}{N^2} & \frac{8m(k-r)}{N^2}-1 & 0 & 0 \\
\frac{-8\sqrt{r}\sqrt{m-r}(N-m)}{N^2} & \frac{-4\sqrt{k-s}\sqrt{m-r}(N-2m)}{N^2} & 0 & 0 \\
\frac{-4\sqrt{r}\sqrt{N-k-m+r}(N-2m)}{N^2} & \frac{8m\sqrt{k-r}\sqrt{N-k-m+r}}{N^2} & 0 & 0
\end{pmatrix} \\
+ \quad & \begin{pmatrix}
0 & 0 & \frac{8\sqrt{r}\sqrt{m-r}(N-m)}{N^2} & \frac{4\sqrt{r}\sqrt{N-k-m+r}(N-2m)}{N^2} \\
0 & 0 & \frac{4\sqrt{k-r}\sqrt{m-r}(N-2m)}{N^2} & \frac{-8m\sqrt{k-r}\sqrt{N-k-m+r}}{N^2} \\
0 & 0 & \frac{8(N-m)(m-r)}{N^2}-1 & \frac{4\sqrt{m-r}\sqrt{N-k-m+r}(N-2m)}{N^2} \\
0 & 0 & \frac{4\sqrt{m-r}\sqrt{N-k-m+r}(N-2m)}{N^2} & 1-\frac{8m(N-k-m+r)}{N^2}
\end{pmatrix}.
\end{aligned}
$$

The compound operator Q is a rotation in the 4-dimensional space spanned by Equations 6 - Equation 9. Finding the rotation angles requires the diagonalized matrix of Q. Let V be the matrix whose columns are the eigenvectors of Q, then $Q^D = V^{-1}QV$ is a diagonal matrix whose diagonal components are the eigenvalues of Q as follows:

$$Q^D = \begin{pmatrix} e^{-iw_1} & 0 & 0 & 0 \\ 0 & e^{iw_1} & 0 & 0 \\ 0 & 0 & e^{-iw_2} & 0 \\ 0 & 0 & 0 & e^{iw_2} \end{pmatrix}$$

where

$$\begin{aligned} e^{-iw_1} &= \frac{4km}{N^2} - \frac{4r}{N} + \Gamma - \Delta \\ e^{iw_1} &= \frac{4km}{N^2} - \frac{4r}{N} + \Gamma + \Delta \\ e^{-iw_2} &= \frac{4km}{N^2} - \frac{4r}{N} - \Gamma - \Delta \\ e^{iw_2} &= \frac{4km}{N^2} - \frac{4r}{N} - \Gamma + \Delta \end{aligned}$$

and Γ and Δ are given by

$$\begin{aligned} \Gamma &= \sqrt{1 - \frac{8rN^3 + 8kmN^2 - 16rkN^2 - 16rmN^2 + 32rkmN - 16k^2m^2}{N^4}}, \\ \Delta &= 2\sqrt{\frac{2N^2r(r+k+m) - N^3r(1+\Gamma) + N^2km\Gamma - N^2km - 8Nkmr + 4k^2m^2}{N^4}} \end{aligned}$$

which implies

$$\begin{aligned} w1 &= \arccos\left(\frac{4km}{N^2} - \frac{4r}{N} + \Gamma\right), \\ w2 &= \arccos\left(\frac{4km}{N^2} - \frac{4r}{N} - \Gamma\right). \end{aligned}$$

The amplitude of $|l_1\rangle$ is given by

$$a(t) = A\sin(w_1 t + \phi) \tag{12}$$

where A is the maximal amplitude, to be found, and ϕ is the angle between the initial state $H|0\rangle^{\otimes n}$ and $|l_1\rangle$:

$$\phi = \arctan\sqrt{\frac{r}{N-r}}.$$

The maximal probability A^2 that a measurement of the system will produce a state in $K \cap M$ will be obtained at time

$$T = \arg\max_t A^2 sin^2(w_1 t + \phi)$$

yielding

$$T = \left\lfloor \frac{\pi/2 - \arctan\left(\sqrt{\frac{r}{N-r}}\right)}{\arccos\left(\frac{4km}{N^2} - \frac{4r}{N} + \Gamma\right)} \right\rfloor$$

as asserted (Equation 5). ■

Theorem 1 suggests a way for approximating the time complexity of the algorithm when $m, k << N$. Employing the Taylor series, the second order approximation of the rotation angle is

$$w_1 = O\left(\sqrt{\frac{r}{N}}\right)$$

and the number of iterations can be approximated by

$$T \approx O\left(\frac{\pi/2 - \sqrt{\frac{r}{N-r}}}{\sqrt{\frac{r}{N}}}\right) = O\left(\sqrt{\frac{N}{r}}\right). \tag{13}$$

Theorem 2 *The maximal probability, A^2 from Equation 12, of measuring a marked state in $K \cap M$ from Theorem 1 is approximately* 1 *when $|K|, |M| << N$ and N is large.*

Proof The amplitude $a(t)$ of $|l_1\rangle$ behaves as in Equation 12 and the maximal amplitude A can be obtained from

$$\begin{aligned} A\sin(w_1 + \phi) = & \sqrt{\frac{r}{N}}\left(1 - \frac{8r(N-m)}{N^2}\right) \\ + & \sqrt{\frac{k-r}{N}}\left(\frac{-4\sqrt{r}\sqrt{k-r}(N-2m)}{N^2}\right) \\ + & \sqrt{\frac{m-r}{N}}\left(\frac{8\sqrt{r}\sqrt{m-r}(N-m)}{N^2}\right) \\ + & \sqrt{\frac{N-m-k+r}{N}}\left(\frac{4\sqrt{r}\sqrt{N-m-k+r}(N-2m)}{N^2}\right). \end{aligned} \tag{14}$$

Substituting w_1 and ϕ in the left hand side of Equation 14 yields

$$
\begin{aligned}
A \approx & \frac{\sqrt{\frac{r}{N}}\left(1-\frac{8r(N-m)}{N^2}\right)}{\sqrt{2-2\Gamma-\frac{8km}{N^2}+\frac{8r}{N}}+\sqrt{\frac{r}{N-r}}} \\
+ & \frac{\sqrt{\frac{k-r}{N}}\left(\frac{-4\sqrt{r}\sqrt{k-r}(N-2m)}{N^2}\right)}{\sqrt{2-2\Gamma-\frac{8km}{N^2}+\frac{8r}{N}}+\sqrt{\frac{r}{N-r}}} \\
+ & \frac{\sqrt{\frac{m-r}{N}}\left(\frac{8\sqrt{r}\sqrt{m-r}(N-m)}{N^2}\right)}{\sqrt{2-2\Gamma-\frac{8km}{N^2}+\frac{8r}{N}}+\sqrt{\frac{r}{N-r}}} \\
+ & \frac{\sqrt{\frac{N-m-k+r}{N}}\left(\frac{4\sqrt{r}\sqrt{N-m-k+r}(N-2m)}{N^2}\right)}{\sqrt{2-2\Gamma-\frac{8km}{N^2}+\frac{8r}{N}}+\sqrt{\frac{r}{N-r}}}
\end{aligned}
$$

which, under the assumption $r,k,m << N$, is close to 1.

■

3. Quantum Associative Memory

In this section we introduce our associative memory model and the retrieval procedure with pattern completion and correction abilities. We present the concept of memory as a quantum operator that flips the phase of the memory patterns. The operator is based on an oracle that identifies, or "marks", memory patterns. This allows the initial state of our algorithm to be independent of the memory set. The input of our algorithms is an n-qubit register that contains the superposition of all basis states. This input is created by applying the Hadamard operation on an n-qubit register set to zeros, $H|0\rangle$.

3.1 Pattern Completion

Let I_M be a phase oracle on a set M, called the memory set, of m n-qubit patterns and let x' be a version of a memory pattern $x \in M$ with d missing bits. We are required to output the pattern x based on I_M and x'. The partial pattern is given as a string of binary values 0 and 1 and some unknown bits marked '?'. Denoting the set of possible completions of the partial pattern K and its size k, the completion problem can be reduced to the problem of retrieving a member x of the intersection between two sets K and M, $x \in K \cap M$. For example, let $M = \{0101010, 0110100, 1001001, 1111000, 1101100, 1010101, 0000111, 0010010\}$ be a 7-bit memory set of size 8 and let "0110?0?" be a partial pattern with 2 missing bits, so the completion set is $K = \{0110000, 0110001, 0110100, 0110101\}$. Pattern completion is the computation of the intersection between K and M, which is the memory pattern 0110100.

Pattern completion can use either the intersection oracle presented in Figure 6 or the quantum intersection presented in Algorithm 1. In either case, we need to create the completion operator f_K or its phase version I_K that marks the states of the set K, which can be implemented by checking whether a state is a completion of the partial patterns x' represented by the set K. Such an implementation is shown in Figure 7, where for each given bit in the pattern an appropriate control is added

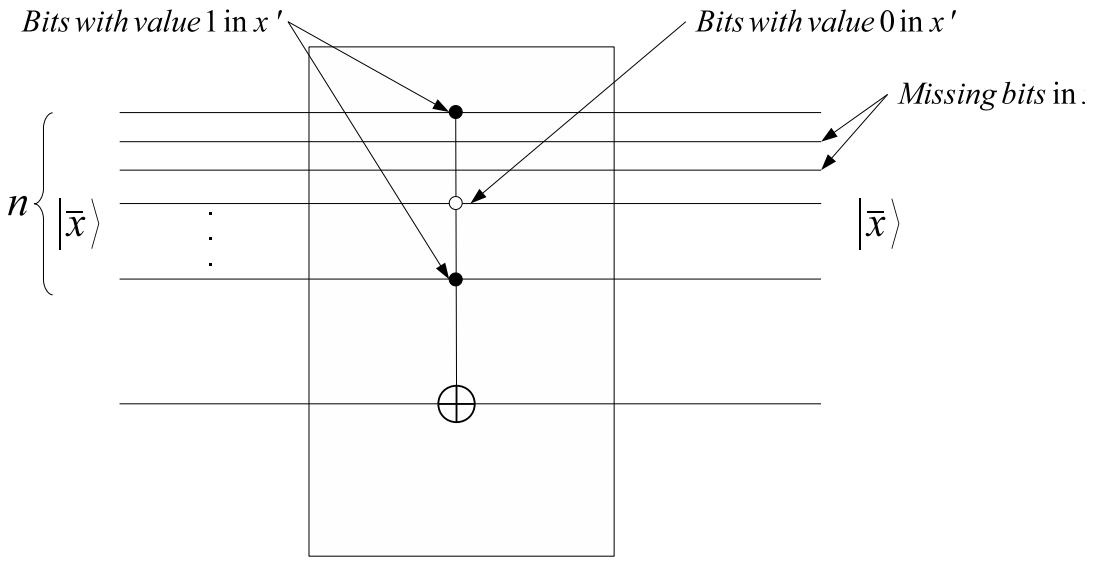


Figure 7: An implementation of a completion operator with an up to $n+1$ dimensional controlled-not operator.

to the corresponding qubit. The dark circle means that the control is activated when the bit value is 1 and the empty circle means that the control is activated when the bit value is 0. For each missing bit, no control is added for the corresponding qubit. A single n qubit operator can be implemented by $O(n)$ 2 and 3-qubit operators (Barenco et al., 1995).

The algorithm for pattern completion through the quantum intersection algorithm is

Algorithm 2 : Quantum Pattern Completion
Given: A memory operator $I_M$and a pattern $x' \in \{0,1\}^n$, which is a partial version of some memory pattern with up to d missing bits
1. Create the completion operator I_K.
2. Apply Algorithm 1 with I_M and I_K

3.2 Pattern Correction

Let I_M be a phase oracle of a memory set M of size m and let x' be a version of a memory pattern $x \in M$ with up to d faulty bits. We are required to output the pattern x based on I_M and x'. The set K of possible corrections of the faulty pattern consists of all patterns in Hamming distance up to d from x'. The correction problem can be reduced to the problem of retrieving a member x of the intersection between two sets K and M, $x \in K \cap M$. For example, let $M = \{0101010, 0110100, 1001001, 1111000, 1101100, 1010101, 0000111, 0010010\}$ be a 7-bit memory set and let "0110001" be the input pattern with 2 possible errors. The correction set K

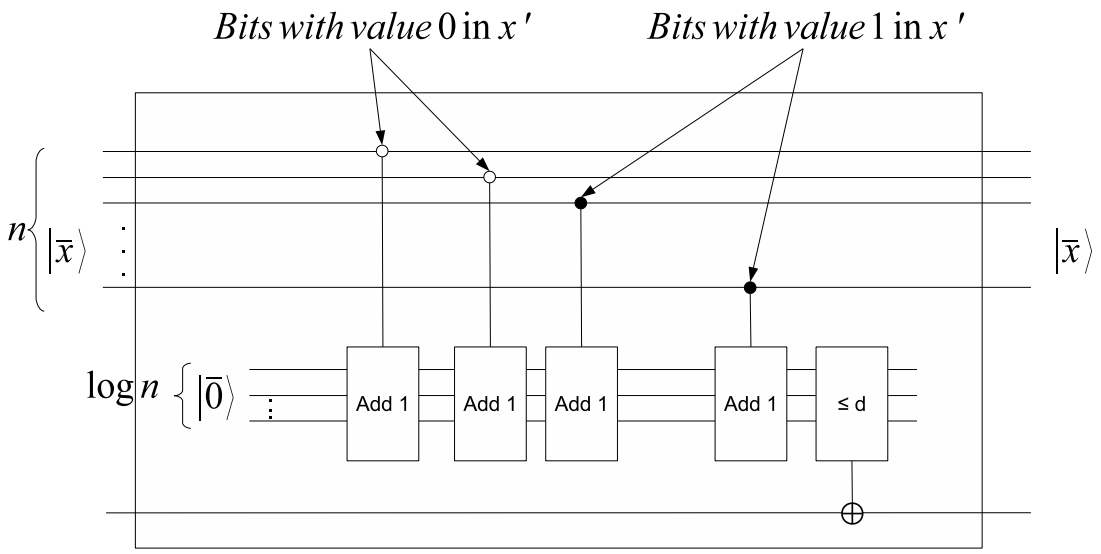


Figure 8: An implementation of a correction operator with a $\lceil \log n \rceil$ additional bits, n controlled operators that add 1, and a threshold operator.

consists of all patterns that are in Hamming distance up to 2 from x'. Pattern correction should then retrieve the memory pattern 0110100.

Pattern correction can be solved using the quantum intersection algorithm, which requires the creation of the correction operator f_K or its phase version I_K. This can be implemented by checking whether the number of different bits between any given state and the faulty pattern x' is less or equal to d. Such an implementation is shown in Figure 8. The operator applies a linear threshold on the hamming distance between x' and any input x. The operator consists of n input qubits, $\lceil \log n \rceil$ qubits for the hamming distance of x from x', and an additional qubit for the output. The operator adds 1 for each bit in x that is not equal to the corresponding bit in x', then applies a controlled not operator if the hamming distance does not exceed the threshold d. The operator can, thus, be implemented using $O(n \log n)$ 2 and 3-qubit operators.

Algorithm 3 : Quantum Pattern Correction
Given: A memory operator I_M and a pattern $x' \in \{0,1\}^n$, which is a faulty version of some memory pattern with up to d faulty bits
1. Create the correction operator I_K.
2. Apply Algorithm 1 with I_M and I_K

A generalization of both Algorithm 2 and Algorithm 3 for the case of unknown number of possible corrections is straight forward using quantum search for an unknown number of marked states (Boyer et al., 1996) or quantum counting (Brassard et al., 1998).

Algorithm 3 finds a memory pattern that is in Hamming distance up to d from x'. If the memory is within the correction capacity bounds, Algorithm 3 finds the correct pattern x with high probability, as will be proved in Section 4. However, if we are interested in ensuring that we find the closest memory pattern to x', with no dependence on the capacity bound, then we can apply Algorithm 3 for $i = 0$ bits and increase it up to $i = d$ bits. In this case we ensure that we find a pattern x, such that

$$\left\{x : (x \in M) \wedge \left(\forall x'' \in M : dist(x'',x) \geq dist(x',x)\right)\right\}$$

where $dist(\cdot,\cdot)$ is the Hamming distance.

4. Analysis of the Quantum Associative Memory

In this section we analyze the time complexity and memory capacity of the proposed quantum associative memory. We first show that the time complexity of retrieval operations is sub-exponential in the number of bits. Then we show that the number of memory patterns that can be stored while the model retains its correction and completion abilities is exponential in the number of bits.

4.1 Time Complexity Analysis

The time complexity of the retrieval procedure with either pattern completion or correction ability is determined by the complexity of the quantum intersection algorithm and the complexity of the completion and correction operators. The two operators can be implemented by a number of operations which is linear in n or in $n\log n$. According to Equation 13, the completion and correction operations are performed in

$$T \approx n\log n + O\left(\sqrt{\frac{N}{|K \cap M|}}\right) = O\left(\sqrt{\frac{N}{|K \cap M|}}\right)$$

operations, which is sub-exponential in number of bits.

4.2 Capacity Analysis

We consider three different capacity measures. The first is the equilibrium capacity M_{eq}, which is the maximal memory size that ensures that all memory patterns are equilibrium points of the model. An Equilibrium point is a pattern that, when presented to the model as an input, is also retrieved as an output with high probability. The second is the pattern completion capacity M_{com}, which is the maximal memory size that allows the completion of any partial pattern with up to d missing bits with high probability. The third is the pattern correction capacity M_{cor}, which is the maximal memory size that allows the correction of any pattern with up to d faulty bits with high probability.

Equilibrium is a special case of completion and correction with neither missing nor faulty bits. Therefore, the equilibrium capacity should be equal to the completion and correction capacities for $d = 0$.

4.2.1 EQUILIBRIUM CAPACITY

The equilibrium capacity of the quantum associative memory is

$$M_{eq} = N$$

because every memory state of an n-bit associative memory M of any size $m \leq N$ is an equilibrium state, that is, if $Q_x = -(HI_0HI_x)$, $T = \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor$, and $|\Psi(T)\rangle = Q_x^T H|0\rangle$, then

$$\forall x \in M: \ |\langle x|\Psi(T)\rangle|^2 \to 1 \ as \ n \to \infty.$$

This is a direct consequence of Grover's algorithm (Grover, 1996) and the results obtained by Boyer et al. (1996) concerning the ability to find any member of the size N database with probability close to 1.

4.2.2 COMPLETION CAPACITY

Given a pattern x', which is a partial version of some memory pattern x_c with d missing bits, we seek the maximal memory size, for which the pattern can be completed with high probability from a random uniformly distributed memory set (McEliece et al., 1987; Baram, 1991; Baram and Sal'ee, 1992).

The completion capacity is bounded from above by two different bounds. The first is a result of Grover's quantum search algorithm limitations and the second is a result of the probability of correct completion.

A Bound on Memory Size due to Grover's Quantum Search Limitations. Grover's operator flips the marked states around the zero amplitude (negating their amplitudes) then flips all amplitudes around the average of all amplitudes (Biham et al., 1999). Amplification of the desired amplitudes occurs only when the average of all amplitudes is closer to the amplitudes of the non-marked states than to the marked states. This imposes the following upper bound on the memory size:

$$m < N/2.$$

This can be observed in the first iteration of the quantum search algorithm on a number of marked states. The initial amplitude of all basis states in $H|0\rangle$ is $1/\sqrt{N}$. Flipping the phase of m marked states by I_M to $-1/\sqrt{N}$ yields an average amplitude of

$$(N-m) * \left(1/\sqrt{N}\right) - m * \left(1/\sqrt{N}\right) = \frac{N-2m}{\sqrt{N}}.$$

Then, flipping the phases of all basis states around this average by HI_0H yields two values of amplitudes. The amplitudes of marked and unmarked states become

$$2\left(\frac{N-2m}{\sqrt{N}}\right) \pm \frac{1}{\sqrt{N}} = \left(\frac{2N-4m \pm 1}{\sqrt{N}}\right) \tag{15}$$

where $\pm$ correspond to marked and unmarked states respectively.

A necessary condition for the amplification of marked states is that the absolute value of their amplitudes after an iteration of the algorithm is higher than the absolute value of the amplitude of unmarked states. The condition is satisfied if and only if the two equations given in Equation 15 satisfy

$$\left|\left(\frac{2N-4m+1}{\sqrt{N}}\right)\right| > \left|\left(\frac{2N-4m-1}{\sqrt{N}}\right)\right|$$

which holds true if and only if $m < N/2$. Therefore, if $m \geq N/2$ the amplitudes of the marked states will not increase, which gives the following upper bound on the completion capacity:

$$M_{com} < N/2. \tag{16}$$

However, this is a very loose bound and the success probability of the completion will impose a tighter bound.

A Bound on Memory Size due to Pattern Completion. The bound on memory size that ensures a high probability of correct completion depends on the definition of the pattern completion procedure. If one defines pattern completion as the process of outputting any of a number of possible memory patterns when given a partial input, then the capacity bound of our memory is the amplification bound given in Equation 16. However, this is not always the case. Pattern completion capacity is usually defined as the maximal size of a random uniformly distributed memory set that, given a partial version x' of a memory $x_c \in M$ with d missing bits, outputs x_c. The following theorem gives an upper bound on the capacity for pattern completion with high probability:

Theorem 3 *An n-bit associative memory with m random patterns can complete up to d missing bits on average when*

$$m \leq 2^{n-d}$$

with probability higher than

$$\frac{v}{e^v - 1} \sum_{i=1}^{m} \frac{v^{i-1}}{i^i} \tag{17}$$

as n grows to infinity, where

$$v = \frac{m}{2^{n-d}}.$$

Proof Let M be a random uniformly distributed memory set of size $m = v2^{n-d}$, where $0 < v < 1$. Let x' be a partial pattern of $x_c \in M$ with d missing bits. x' induces the set

$$K = \left\{ x \middle| x \text{ is a completion of } x' \right\}$$

where $|K| = 2^d$. Let Z_i be random indicator variables representing the existence of the ith member of M in K. Denoting $p = Pr(Z_i = 1) = \frac{2^d}{2^n} = \frac{1}{2^{n-d}}$, we have

$$m = v/p. \tag{18}$$

Let us denote $|\Psi(T)\rangle = Q^T H |0\rangle$ and $S = \sum_{i=1}^{m} Z_i$. If there is only one possible memory completion, then it is x_c, and if there are two, then x_c is one of the two, and so on. Therefore, the probability of successfully outputting x_c from the partial pattern x' is the sum of the conditional probabilities that there are i memory completions divided by i:

$$\begin{aligned} |\langle x_c | \Psi(T) \rangle|^2 &= \sum_{i=1}^{m} \frac{Pr(S = i \mid S \geq 1)}{i} \\ &= \sum_{i=1}^{m} \frac{Pr(S = i)}{iPr(S \geq 1)} \\ &= \sum_{i=1}^{m} \frac{\binom{m}{i} p^i (1-p)^{m-i}}{i\left(1 - (1-p)^m\right)}. \end{aligned} \tag{19}$$

Now, this probability can be lower-bounded by

$$|\langle x_c|\Psi(T)\rangle|^2 \geq \sum_{i=1}^{m} \frac{\left(\frac{m}{i}\right)^i p^i (1-p)^{m-i}}{i\left(1-(1-p)^m\right)}. \tag{20}$$

Substituting Equation 18 in Equation 20 we have

$$\begin{aligned} |\langle x_c|\Psi(T)\rangle|^2 &\geq \sum_{i=1}^{m} \frac{\left(\frac{v^i}{i^i}\right)(1-p)^{v/p-i}}{i\left(1-(1-p)^{v/p}\right)} \\ &\geq \sum_{i=1}^{m} \frac{\left(\frac{v^i}{i^i}\right) e^{-v} \frac{1}{(1-p)^i}}{i\left(1-e^{-v}\right)} \\ &\geq \frac{1}{e^v-1} \sum_{i=1}^{m} \frac{v^i}{\left(i(1-p)\right)^i} \\ &> \frac{v}{e^v-1} \sum_{i=1}^{m} \frac{v^{i-1}}{i^i}. \end{aligned} \tag{21}$$

■

Figure 9 shows that the lower bound with approximation by only three terms of the sum in Equation 17 as a function of v is higher than 75% for all possible sizes of a non-empty memory within the capacity limits.

Theorem 3 implies that $M_{com}(d) = 2^{n-d}$, which agrees with the result concerning the equilibrium capacity, since $M_{com}(0) = 2^{n-0} = 2^n = N = M_{eq}$.

4.2.3 CORRECTION CAPACITY

A bound on the correction capacity of Algorithm 3 is given be the following theorem:

Theorem 4 *An n-bit associative memory with m random patterns can correct up to d faulty bits on average when*

$$m \leq 2^{n-d} / \binom{n}{d}$$

with probability higher than

$$\frac{v}{e^v-1} \sum_{i=1}^{m} \frac{v^{i-1}}{i^i}$$

as n grows to infinity, where

$$v = \frac{m}{2^{n-d}}.$$

Proof Let M be a random uniformly distributed memory set of size

$$m = v2^{n-d} / \binom{n}{d}$$

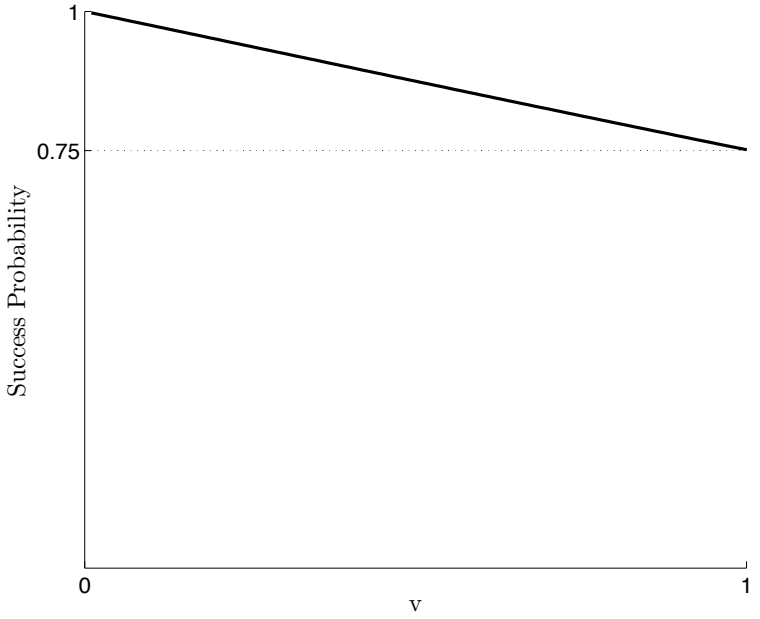


Figure 9: Success probability of pattern completion vs. memory size divided by the maximal completion capacity v for an associative memory with $n = 30$ qubits.

where $0 < v < 1$. Let x' be a pattern $x_c \in M$ with d faulty bits. x' induces a set

$$D = \left\{ x \middle| dist(x,x') \leq d \right\}$$

where $|D| = \binom{n}{d} 2^d$. Let Z_i be random indicator variables representing the existence of the ith member of M in D. Denoting $p = Pr(Z_i = 1) = \binom{n}{d} 2^{d-n}$, we have

$$m = v/p.$$

Let us denote $|\Psi(T)\rangle = Q^T H |0\rangle$ and $S = \sum_{i=1}^{m} Z_i$. The probability of successfully retrieving x_c from the pattern x' is then

$$|\langle x_c|\Psi(T)\rangle|^2 = \sum_{i=1}^{m} \frac{Pr(S = i \mid S \geq 1)}{i}$$

which, according to Equations 19 - 21, also satisfies

$$|\langle x_c|\Psi(T)\rangle|^2 \geq \frac{v}{e^v - 1} \sum_{i=1}^{m} \frac{v^{i-1}}{i^i}.$$

■

Theorem 4 yields $M_{cor}(d) = \binom{n}{d} 2^{n-d}$, which also agrees with the result concerning the equilibrium capacity, since $M_{cor}(0) = \binom{n}{0} 2^{n-0} = 2^n = N = M_{eq}$.

For example, let M be a memory set over $\{0,1\}^{100}$, then, as long as $|M| < 2^{80}$, we can complete up to $d = 100 - \log|M| = 20$ bits and correct up to $d = 13$ bits.

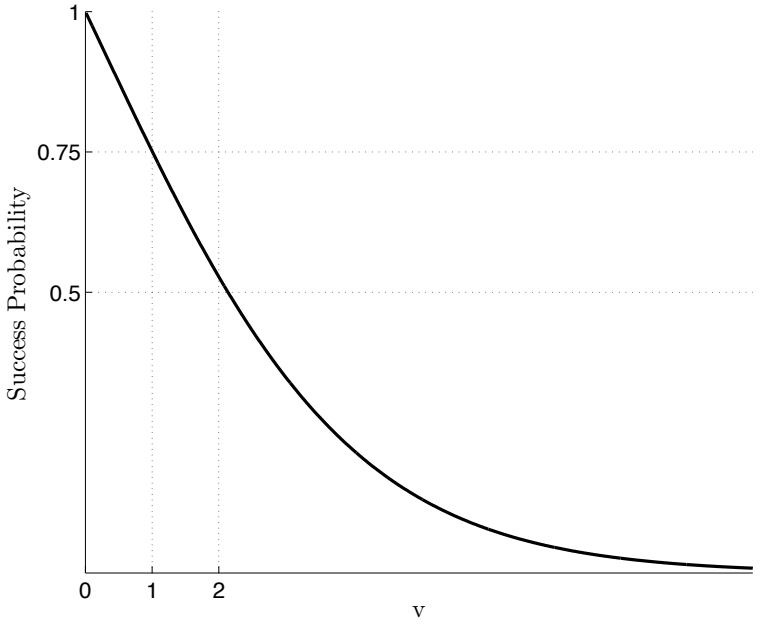


Figure 10: Pattern completion or correction probability vs. the memory size divided by the maximal completion capacity v. For $0 < v < 1$, the probability is above 75% and for $v < 2$ is above 50%.

4.2.4 INCREASING MEMORY SIZE BEYOND THE CAPACITY BOUNDS

The various capacities presented above are exponential in n under the assumption $d << n$. However, an increase of m beyond the capacity bound results in a decay of the correct completion probability as depicted in Figure 10. It can be seen that it is more likely to find the correct completion than not to find it as long as $v < 2$.

In addition, the model can also output a superposition of a number of possible outputs, by skipping the measurement operation in Algorithm 1. This is not true for most classical memory models where spurious memories arise and the output is usually not a memorized pattern, but, rather, some spurious combination of multiple memory patterns (Hopfield, 1982; Bruck, 1990; Goles and Martínez, 1990).

5. Comparison to Previous Works

Quantum computation was previously applied to associative memory by Ventura and Martinez (2000), Ezhov et al. (2000), Howell et al. (2000), and, subsequently, by others. An algorithm based on the model developed by Ventura and Martinez (2000) was proposed by Arima et al. (2008). It was further developed by Arima et al. (2009) and analyzed by Miyajima et al. (2010). We analyze the two main algorithms (Ventura and Martinez, 2000; Arima et al., 2009) and show their differences with respect to our algorithm. These algorithms are given below.

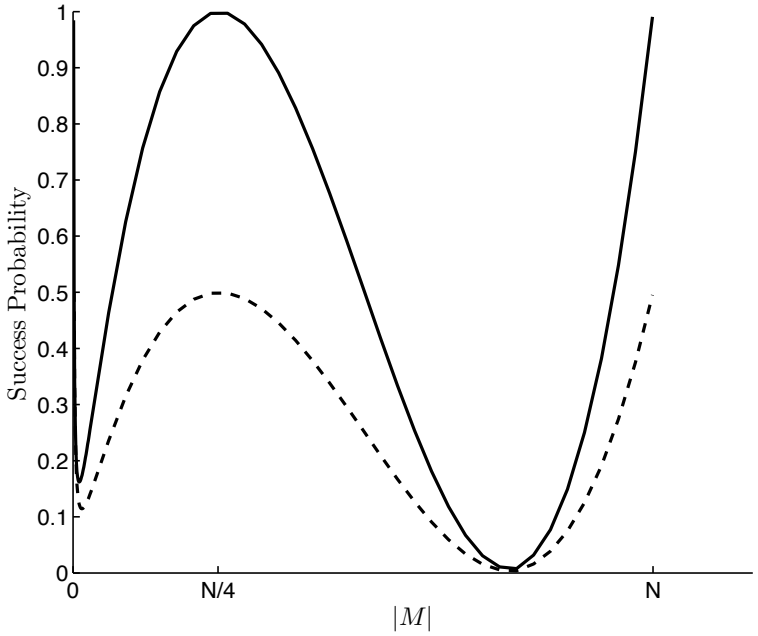


Figure 11: Memory size vs. success probability of Algorithm 4. Optimal results are achieved only when the memory size is close to $\frac{N}{4}$.

Algorithm 4 The algorithm proposed by Ventura and Martinez (2000)
Given: Phase oracles I_M and I_K
1. Denote $Q_M = -HI_0HI_M$ and $Q_K = -HI_0HI_K$
2. Let $|\Psi\rangle = \frac{1}{m}\sum_{i=1}^{m}|i\rangle$.
3. Apply Q_MQ_K on $|\Psi\rangle$
4. Apply Q_K on $|\Psi\rangle$ for $T = \left\lfloor \pi/4\sqrt{N/|K \cap M|} \right\rfloor$-2 times.
5. Measure $|\Psi\rangle$.

Algorithm 5 The algorithm proposed by Arima et al. (2009)
Given: Phase oracles I_M and I_K
1. Denote $Q_M = -HI_0HI_M$ and $Q_K = -HI_0HI_K$
2. Let $|\Psi\rangle = \frac{1}{m}\sum_{i=1}^{m}|i\rangle$.
3. Apply Q_MQ_K on $|\Psi\rangle$ for T times. (T was not found by Arima et al., 2009)
4. Measure $|\Psi\rangle$.

Algorithm 4 can find only a single marked state with high probability when the memory size m is close to $\frac{N}{4} - 2$, as shown by the solid line in Figure 11. The probability of measuring this state reduces by a half when there are two marked states and only one of them is a memory pattern, as shown by the dashed line in Figure 11, and so on.

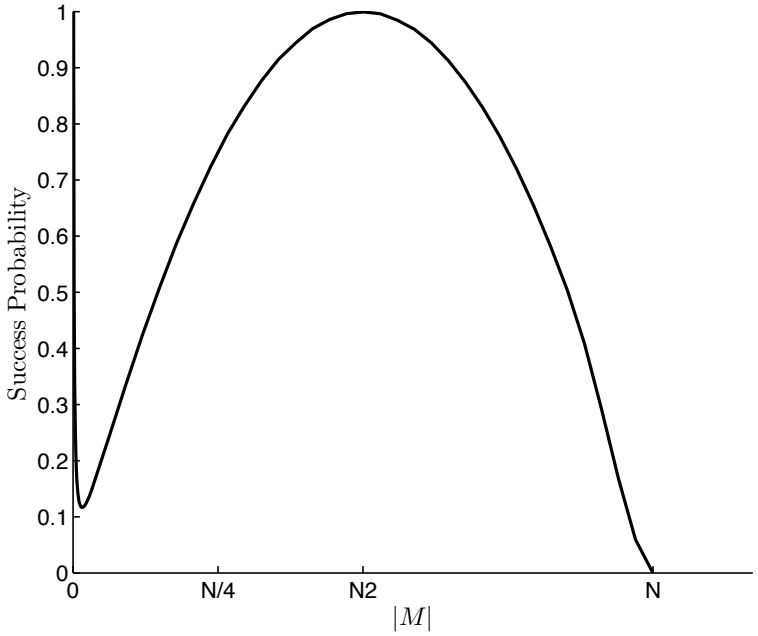


Figure 12: Memory size vs. success probability in Algorithm 5. Satisfactory results are achieved only when the memory size exceeds $\frac{N}{4}$.

Algorithm 5 gives satisfying results only when the memory size exceeds $\frac{N}{4}$, which is exponential in the number of qubits, leaving the possibility of effective pattern completion only for 2 qubits or less. It is therefore not helpful for associative memory with pattern completion and correction abilities. The success probability of Algorithm 5 vs. the memory size is depicted in Figure 12. Miyajima et al. (2010) added a control parameter to tune the algorithm, changing the memory size for which the maximal amplitude is achieved. The algorithm is presented only for one marked state with no completion and correction abilities. The time complexity and stopping criteria were not stated by Arima et al. (2008) and were later found to be $O\left(\sqrt{N}\right)$ (Arima et al., 2009; Miyajima et al., 2010).

Our algorithm, on the other hand achieves high success probability up to memory size $\frac{N}{4}$, as depicted in Figure 13.

Furthermore, both Algorithm 4 and Algorithm 5 need to initialize the system at a superposition of the memory states:

$$|\Psi\rangle = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} |i\rangle$$

which is a shortcoming for two reasons: the time complexity of initialization when the memory size is large and the need for repeated initialization upon every application of the memory. The latter is important as it adds an exponential factor to the query time, for either completion or correction, and an exponential addition to the single query time when amplitude amplification is needed. Amplitude amplification ensures that we pick the correct pattern with probability 1 by performing the algorithm

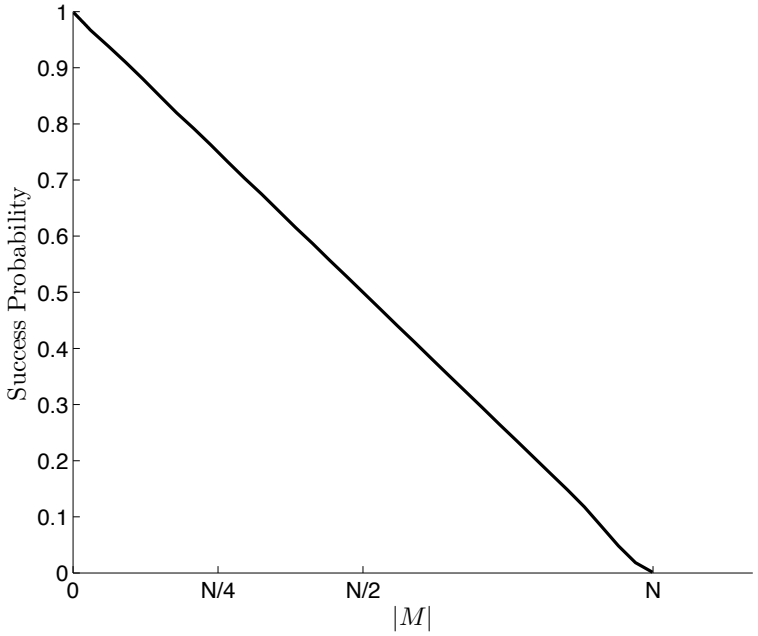


Figure 13: Memory size vs. success probability in Algorithms 2 and 3.

a multiple number of times. Our algorithm's initialization, on the other hand, does not depend on the memory patterns.

6. Numerical Examples and Simulations

Let us first consider an associative memory of 10 qubits. We have randomly chosen a set of 50 patterns M out of the possible 1024 to be stored in memory. We also chose two partial patterns, each with 4 missing qubits, yielding two completion sets K_1 and K_2 of 16 possible completions each. We chose K_1 and K_2 such that they have one and two completions in memory respectively. Figure 14(a) shows the memory set, where each vertical line represents a memory pattern, and Figure 14(b) shows the completion set K_1 in the same manner. The amplitudes of the final state of the completion algorithm are shown in Figure 14(c), where the only possible memory completion has amplitude close to 1. Figure 15 shows the high amplitudes of the two possible memory completions when the completion set is K_2.

As can be seen, applying our algorithm to both completion sets amplified the states that are possible completions in memory. The amplitudes of the desired states reached up to 96.76% in the first case and 68.44% in the second case for each one of the two high amplitudes. Therefore, the probability of measuring the correct completion in the first case is 93.62% and the probability of measuring one of the two correct completions in the second case is 93.67%.

Another simulation was carried out on a 10 qubits associative memory with 2^7 memory patterns and completion queries with 3 missing bits. The behavior of the different subgroups of the basis states is schematically described in Figure 16 for a series of iterations with the completion operator Q of Equation 4. Each amplitude value indicated represents the amplitudes of all the basis states that belong to the corresponding subgroup. It shows the amplification of states in the intersection

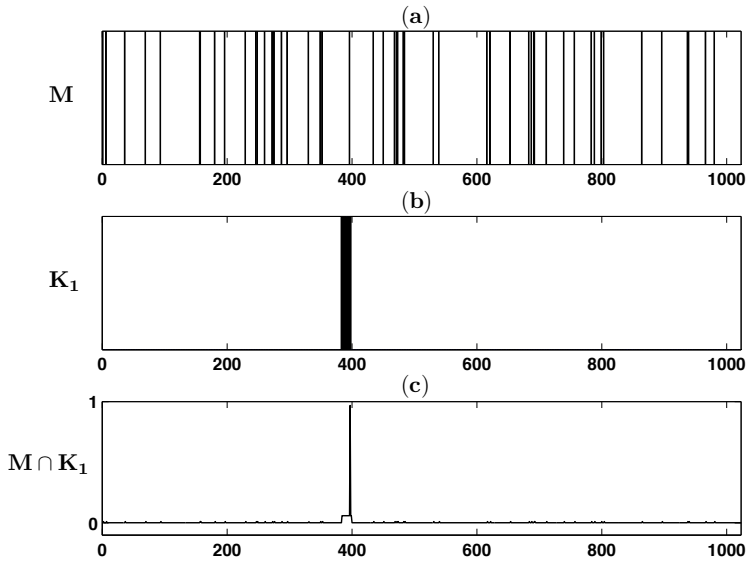


Figure 14: (a) A set of memory patterns M (b) a set of possible completions K_1 to a partial pattern, and (c) the memory completion result in amplitudes.

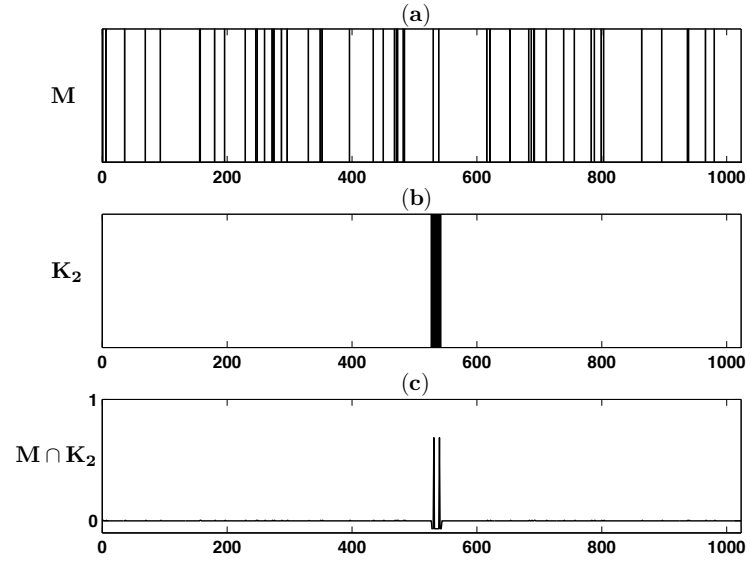


Figure 15: (a)A set of memory patterns M (b) a set of possible completions K_2 to a partial pattern, and the memory completion result in amplitudes.

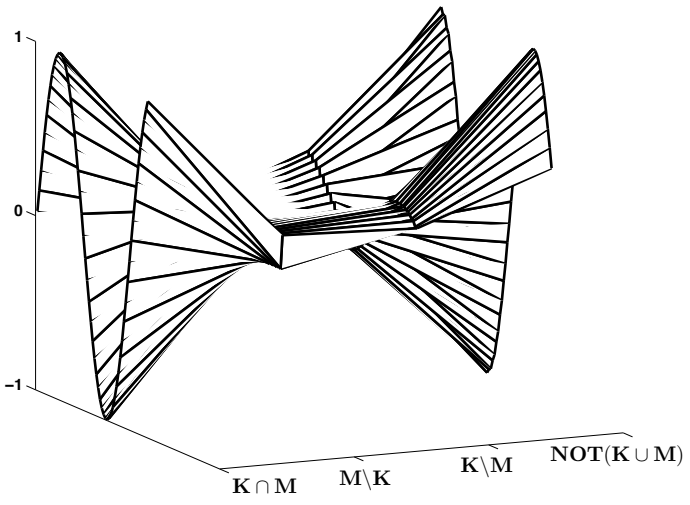


Figure 16: Simulation of a series of iterations of the completion algorithm. The graph shows the different behavior of the different subgroups of basis states. K is the completion set and M is the memory set. The memory completions and the non completions or memories are amplified alternatingly, while the amplitudes of $K\backslash M$ and $M\backslash K$ subgroups stay close to zero.

group $K \cap M$ and in $NOT(K \cup M)$ alternatingly, while the amplitudes of states in $K\backslash M$ and $M\backslash K$ stay close to zero.

We have also tested our algorithms with a larger number of qubits in order to verify that the success rate of retrieval grows asymptotically to 1 as the number of qubits grows. For instance, we tested a 30 qubit system with 2^{25} memory patterns and a completion query that has 8 missing bits. We tested different completions of 8 missing bits so that the intersection set size varied from 1 to 10 patterns. Our algorithm found a member of the memory completion set with probability 96.8%. Increasing the memory size to 2^{26} and 2^{27}, and thereby bringing the capacity close to its limit resulted in completion probabilities of 93.5% and 86.7% respectively. Figure 17 depicts the success rates of pattern completion in a 30 qubit system. An explanation of the different graphs can be found in Table 1. The solid line in Figure 17 depicts the success probability vs. the logarithm of the size of memory with completion queries set to 8 missing bits and the number of possible memory completions set to 1. The dashed line depicts the success probability vs. the logarithm of the completion query size when the memory size is set to 2^{25} patterns and the number of possible memory completions set to 1. The dotted line depicts the success probability vs. the logarithm of the number of possible memory completions when both the memory size and the completion query size are set to 2^{25}. The dash-dotted line depicts the success probability vs. the number of qubits in the system (growing from 5 to 30 qubits) when the memory size, the completion query size, and the number of possible memory completions are small constants.

Figure 17 shows that the deterioration of the success probability vs. the memory size or the completion query size is very slow. For instance, deterioration starts at memory size 2^{26}. Further-

Graph	$\lvert N\rvert$	$\lvert M\rvert$	$\lvert K\rvert$	$\lvert K\cap M\rvert\rvert$
Solid	constant	varying	constant	constant
——	2^{30}	2^3-2^{27}	2^3	1
Dashed	constant	constant	varying	constant
– – –	2^{30}	2^{25}	2^3-2^{25}	1
Dotted	constant	constant	constant	varying
...	2^{30}	2^{25}	2^{25}	$1-2^{25}$
Dash-dotted	varying	constant	constant	constant
– · – · –	2^5-2^{30}	2^3	2^3	1

Table 1: Properties of the four simulations depicted in Figure 17. The x axis in Figure 17 represents the varying set size, while the other set sizes are constant in each simulation.

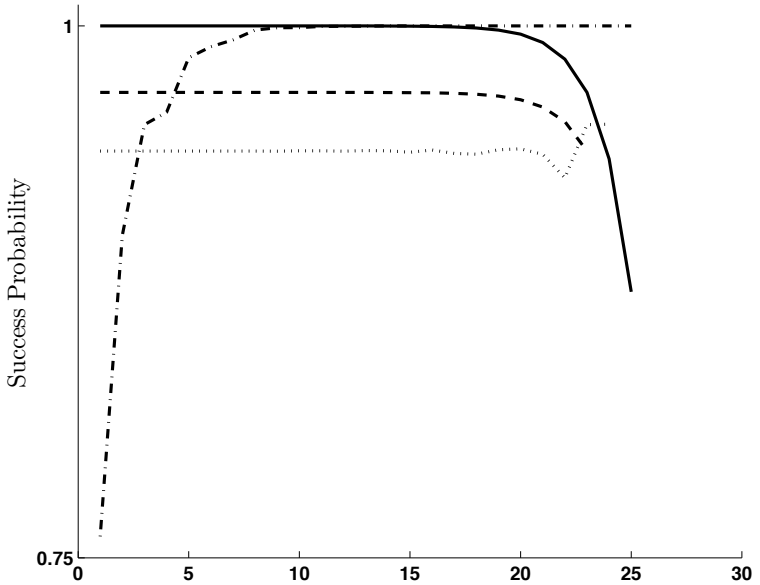


Figure 17: Success probability of measuring a desired memory completion vs. the *log* of the memory size (solid), completion query size (dashed), possible memory completions (dotted), and number of qubits (dash-dotted).

more, the success probability increases when the number of possible memory completions (the size of the intersection set) grows towards the sizes of the completion query and the memory, which indicates that choosing a member of the intersection becomes easy (by randomly choosing a possible completion). Finally, the figure also shows that, as the number of qubits in the system grows, the success probability becomes asymptotically 1, which indicates that, practically, our algorithm produces the intersection when $n >> 1$.

7. Conclusion

We have presented a quantum computational algorithm that computes the intersection between two subsets of n-bit strings. The algorithm is based on a modification of Grover's quantum search. Using the intersection algorithm, we have presented a set of algorithms that implement a model of associative memory via quantum computation. We introduced the notion of memory as a quantum operator, thus avoiding the dependence of the initial state of the system on the memory set. We have shown that our algorithms have both speed and capacity advantages with respect to classical associative memory models, consuming sub-exponential time, and are able to store a number of memory patterns which is exponential in the number of bits. Pattern retrieval algorithms with completion and correction abilities were presented. Bounds relating memory capacity to the maximal allowed signal to noise ratio were found.

References

K. Arima, H. Miyajima, N. Shigei, and M. Maeda. Some properties of quantum data search algorithms. In *Proceedings of the The 23rd International Technical Conference on Circuits/Systems, Computers and Communication (ITC-CSCC2008)*, pages 1169–1172, 2008.

K. Arima, N. Shigei, and H. Miyajima. A proposal of a quantum search algorithm. *International Conference on Convergence Information Technology*, 0:1559–1564, 2009.

Y. Baram. On the capacity of ternary hebbian networks. *Information Theory, IEEE Transactions on*, 37(3):528–534, May 1991.

Y. Baram and D. Sal'ee. Lower bounds on the capacities of binary and ternary networks storing sparse random vectors. *Information Theory, IEEE Transactions on*, 38(6):1633 –1647, nov 1992.

A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. H. Margolus, P. W. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52 (5):3457–3467, 1995.

E. Biham, O. Biham, D. Biron, M. Grassl, and D. A. Lidar. Grover's quantum search algorithm for an arbitrary initial amplitude distribution. *Physical Review A*, 60(4):2742–2745, 1999.

M. Boyer, G. Brassard, P. H*ö*yer, and A. Tapp. Tight bounds on quantum searching. Technical Report PP-1996-11, 30, 1996.

G. Brassard, P. H*ö*yer, and A. Tapp. Quantum counting. In *ICALP '98: Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, pages 820–831, 1998. ISBN 3-540-64781-3.

J. Bruck. On the convergence properties of the hopfield model. *Proceedings of the IEEE*, 78(10): 1579–1585, oct 1990.

D. Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proceedings of the Royal Society A Mathematical Physical and Engineering Sciences*, 400:97–117, 1985.

D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proceedings Mathematical and Physical Sciences*, 439(1907):553–558, 1992.

A. Ezhov, A. Nifanova, and D. Ventura. Quantum associative memory with distributed queries. *Inf. Sci. Inf. Comput. Sci.*, 128(3-4):271–293, 2000.

E. Goles and S. Martínez. *Neural and Automata Networks: Dynamical Behavior and Applications*. Kluwer Academic Publishers, 1990.

L. K. Grover. A fast quantum mechanical algorithm for database search. In *STOC '96: Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, pages 212–219, 1996.

J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79:2554–2558, 1982.

J. C. Howell, J. A. Yeazell, and D. Ventura. Optically simulating a quantum associative memory. *Physical Review A*, 62(4):042303, 2000.

P. Kanerva. Sparse distributed memory and related models. In *Associative Neural Memories*, pages 50–76. Oxford University Press, 1993.

R. McEliece, E. Posner, E. Rodemich, and S. Venkatesh. The capacity of the hopfield associative memory. *Information Theory, IEEE Transactions on*, 33(4):461–482, jul 1987.

H. Miyajima, N. Shigei, and K. Arima. Some quantum search algorithms for arbitrary initial amplitude distribution. In *Sixth International Conference on Natural Computation (ICNC)*, volume 8, pages 603–608, 2010.

M.A. Nielsen and I.L. Chuang. *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.

P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 124–134, 1994.

D.R. Simon. On the power of quantum computation. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 116 –123, nov 1994.

D. Ventura and T. Martinez. Quantum associative memory. *Information Sciences*, 124(1-4):273–296, 2000.

Journal of Machine Learning Research 13 (2012) 3207-3245 Submitted 4/10; Revised 3/12; Published 11/12

Dynamic Policy Programming

Mohammad Gheshlaghi Azar M.AZAR@SCIENCE.RU.NL
Vicenç Gómez V.GOMEZ@SCIENCE.RU.NL
Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Department of Biophysics
Radboud University Nijmegen
6525 EZ Nijmegen, The Netherlands

Editor: Ronald Parr

Abstract

In this paper, we propose a novel policy iteration method, called dynamic policy programming (DPP), to estimate the optimal policy in the infinite-horizon Markov decision processes. DPP is an incremental algorithm that forces a gradual change in policy update. This allows us to prove finite-iteration and asymptotic ℓ_∞-norm performance-loss bounds in the presence of approximation/estimation error which depend on the average accumulated error as opposed to the standard bounds which are expressed in terms of the supremum of the errors. The dependency on the average error is important in problems with limited number of samples per iteration, for which the average of the errors can be significantly smaller in size than the supremum of the errors. Based on these theoretical results, we prove that a sampling-based variant of DPP (DPP-RL) asymptotically converges to the optimal policy. Finally, we illustrate numerically the applicability of these results on some benchmark problems and compare the performance of the approximate variants of DPP with some existing reinforcement learning (RL) methods.

Keywords: approximate dynamic programming, reinforcement learning, Markov decision processes, Monte-Carlo methods, function approximation

1. Introduction

Many problems in robotics, operations research and process control can be represented as a control problem that can be solved by finding the optimal policy using *dynamic programming* (DP). DP is based on estimating some measures of the value of state-action $Q^*(x,a)$, also known as action-value function, through the Bellman equation. For high-dimensional discrete systems or for continuous systems, computing the value function by DP is intractable. The common approach to make the computation tractable is to approximate the value function using function-approximation and Monte-Carlo sampling (Szepesvári, 2010; Bertsekas and Tsitsiklis, 1996). Examples of such approximate dynamic programming (ADP) methods are approximate policy iteration (API) and approximate value iteration (AVI) (Bertsekas, 2007; Lagoudakis and Parr, 2003; Perkins and Precup, 2003; de Farias and Van Roy, 2000). In addition to these approaches, there are methods which do not rely exclusively on an approximate value function. These methods include, for instance, actor-critic methods (Barto et al., 1983), which explicitly consider two interacting processes, policy gradient methods (Baxter and Bartlett, 2001; Sutton et al., 2000), and dual dynamic programming (Wang et al., 2007, 2008).

ADP methods have been successfully applied to many real world problems, and theoretical results have been derived in the form of finite iteration and asymptotic performance guarantee on the induced policy. In particular, the formal analysis of these algorithms is usually characterized in terms of bounds on the difference between the optimal and the estimated value function induced by the algorithm (performance loss) (Farahmand et al., 2010; Thiery and Scherrer, 2010; Munos, 2005; Bertsekas and Tsitsiklis, 1996). For instance, in the case of AVI and API, the asymptotic ℓ_∞-norm performance-loss bounds in the presence of approximation error ε_k can be expressed as[1]

$$\limsup_{k\to\infty}\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\limsup_{k\to\infty}\|\varepsilon_k\|, \tag{1}$$

where γ denotes the discount factor, $\|\cdot\|$ is the ℓ_∞-norm w.r.t. the state-action pair (x,a) and π_k is the control policy at iteration k.

The bound of Equation 1 is expressed in terms of the supremum of the approximation errors. Intuitively, the dependency on the supremum error means that to have a small overall performance loss the approximation errors of all iterations should be small in size, that is, a large approximation error in only one iteration can derail the whole learning process. This can cause a major problem when the approximation error ε_k arises from sampling. In many problems of interest, the sampling error can be large and hard to control, since only a limited number of samples can be used at each iteration. Also, even in those cases where we have access to large number of samples, it may be difficult, if not impossible, to control the size of errors for all iterations. This is due to the fact that the sampling errors are random objects and, regardless of the number of samples used at each iteration, there is always a fair chance that in some few *outlier* iterations the sampling errors take large values in their interval of definition. In all those cases, a bound which depends on the average accumulated error $\bar{\varepsilon}_k = 1/(k+1)\sum_{j=0}^{k}\varepsilon_j$ instead of the supremum error is preferable. The rationale behind this idea is that the average of the sum of random variables, under some mild assumptions, can be significantly smaller in size than the supremum of the random variables. Also, the average error $\bar{\varepsilon}_k$ is less sensitive to the outliers than the supremum error. Therefore, a bound which depends on the average error can be tighter than the one with dependency on the supremum error. To the best of authors' knowledge, there exists no previous work that provides such a bound.

In this paper, we propose a new incremental policy-iteration algorithm called dynamic policy programming (DPP). DPP addresses the above problem by proving the first asymptotic and finite-iteration performance loss bounds with dependency on $\|\bar{\varepsilon}_k\|$. This implies the previously mentioned advantages in terms of performance guarantees. The intuition is that DPP, by forcing an incremental change between two consecutive policies, accumulates the approximation errors of all the previous iterations, rather than just minimizing the approximation error of the current iteration. We also introduce a new RL algorithm based on the DPP update rule, called DPP-RL, and prove that it converges to the optimal policy with the convergence rate of order $1/\sqrt{k}$. This rate of convergence leads to a PAC ("probably approximately correct") sample-complexity bound of order $O(1/((1-\gamma)^6\varepsilon^2))$ to find an ε-optimal policy with high probability, which is superior to the best existing result of standard Q-learning (Even-Dar and Mansour, 2003). See Section 6 for a detailed comparison with incremental RL algorithms such as Q-learning and SARSA.

1. For AVI the approximation error ε_k is defined as the error associated with the approximation of the Bellman optimality operator. In the case of API, ε_k is the policy evaluation error (see Farahmand et al., 2010; Bertsekas and Tsitsiklis, 1996, Chapter 6, for more details).

DPP shares some similarities with the well-known actor-critic (AC) method of Barto et al. (1983), since both methods make use of an approximation of the optimal policy by means of action preferences and soft-max policy. However, DPP uses a different update rule which is only expressed in terms of the action preferences and does not rely on the estimate of the value function to criticize the control policy.

The contribution of this work is mainly theoretical, and focused on the problem of estimating the optimal policy in an infinite-horizon MDP. Our setting differs from the standard RL setting in the following: we rely on a *generative model* from which samples can be drawn. This means that the agent has full control on the sample queries that can be made for any arbitrary state. Such an assumption is commonly made in theoretical studies of RL algorithms (Farahmand et al., 2008; Munos and Szepesvári, 2008; Kearns and Singh, 1999) because it simplifies the analysis of learning and exploration to a great extent. We compare DPP empirically with other methods that make use of this assumption. The reader should notice that this premise does not mean that the agent needs explicit knowledge of the model dynamics to perform the required updates, nor does it need to learn one.

This article is organized as follows. In Section 2, we present the notation which is used in this paper. We introduce DPP and we investigate its convergence properties in Section 3. In Section 4, we demonstrate the compatibility of our method with the approximation techniques by generalizing DPP bounds to the case of function approximation and Monte-Carlo sampling. We also introduce a new convergent RL algorithm, called DPP-RL, which relies on a sampling-based variant of DPP to estimate the optimal policy. Section 5, presents numerical experiments on several problem domains including the optimal replacement problem (Munos and Szepesvári, 2008) and a stochastic grid world. In Section 6 we briefly review some related work. Finally, in Section 7, we summarize our results and discuss some of the implications of our work.

2. Preliminaries

In this section, we introduce some concepts and definitions from the theory of Markov decision processes (MDPs) and reinforcement learning (RL) as well as some standard notation (see Szepesvári, 2010, for further reading). We begin by the definition of the ℓ_2-norm (Euclidean norm) and the ℓ_∞-norm (supremum norm). Assume that $\mathcal{Y}$ is a finite set. Given the probability measure μ over $\mathcal{Y}$, for a real-valued function $g : \mathcal{Y} \to \mathbb{R}$, we shall denote the ℓ_2-norm and the weighted $\ell_{2,\mu}$-norm of g by $\|g\|_2^2 \triangleq \sum_{y\in\mathcal{Y}} g(y)^2$ and $\|g\|_{2,\mu}^2 \triangleq \sum_{y\in\mathcal{Y}} \mu(y) g(y)^2$, respectively. Also, the ℓ_∞-norm of g is defined by $\|g\| \triangleq \max_{y\in\mathcal{Y}} |g(y)|$ and $\log(\cdot)$ denotes the natural logarithm.

2.1 Markov Decision Processes

A discounted MDP is a quintuple $(\mathcal{X}, \mathcal{A}, P, \mathcal{R}, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ are, respectively, the state space and the action space. P shall denote the state transition distribution and $\mathcal{R}$ denotes the reward kernel. $\gamma \in [0,1)$ denotes the discount factor. The transition P is a probability kernel over the next state upon taking action a from state x, which we shall denote by $P(\cdot|x,a)$. $\mathcal{R}$ is a set of real-valued numbers. A reward $r(x,a) \in \mathcal{R}$ is associated with each state x and action a.

Assumption 1 (MDP Regularity) *We assume that $\mathcal{X}$ and $\mathcal{A}$ are finite sets with the cardinalities $|\mathcal{X}|$ and $|\mathcal{A}|$, respectively. Also, the absolute value of the immediate reward $r(x,a)$ is bounded from above by $R_{\max} > 0$ for all $(x,a) \in \mathcal{Z}$.*

Remark 1 *To keep the representation succinct, we make use of the short-hand notation $\mathcal{Z}$ for the joint state-action space $\mathcal{X}\times\mathcal{A}$. We also denote $R_{\max}/(1-\gamma)$ by $V_{\max}$.*

A Markovian policy kernel determines the distribution of the control action given the current state. The policy is called stationary if the distribution of the control action is independent of time. Given the current state x, we shall denote the Markovian stationary policy, or in short only policy, by $\pi(\cdot|x)$. A policy is called deterministic if for any state $x\in\mathcal{X}$ there exists some action a such that $\pi(a|x)=1$. Given the policy π, its corresponding value function $V^\pi:\mathcal{X}\to\mathbb{R}$ denotes the expected total discounted reward in each state x, when the action is chosen by policy π, which we denote by $V^\pi(x)$. Often it is convenient to associate value functions not with states but with state-action pairs. Therefore, we introduce $Q^\pi:\mathcal{Z}\to\mathbb{R}$ as the expected total discounted reward upon choosing action a from state x and then following policy π, which we shall denote by $Q^\pi(x,a)$. We define the *Bellman operator* $\mathcal{T}^\pi$ on the action-value functions by[2]

$$\mathcal{T}^\pi Q(x,a)\triangleq r(x,a)+\gamma\sum_{(y,b)\in\mathcal{Z}}P(y|x,a)\pi(b|y)Q(y,b),\qquad\forall(x,a)\in\mathcal{Z}.$$

The goal is to find a policy π^* that attains the *optimal value function*, $V^*(x)\triangleq\sup_\pi V^\pi(x)$, at all states $x\in\mathcal{X}$. The optimal value function satisfies the Bellman equation:

$$\begin{aligned}V^*(x)&=\sup_{\pi(\cdot|x)}\sum_{\substack{y\in\mathcal{X}\\a\in\mathcal{A}}}\pi(a|x)\left[r(x,a)+P(y|x,a)V^*(y)\right]\\&=\max_{a\in\mathcal{A}}\left[r(x,a)+\sum_{y\in\mathcal{X}}P(y|x,a)V^*(y)\right],\end{aligned}\qquad\forall x\in\mathcal{X}.\tag{2}$$

Likewise, the *optimal action-value function* Q^* is defined by $Q^*(x,a)=\sup_\pi Q^\pi(x,a)$ for all $(x,a)\in\mathcal{Z}$. We shall define the *Bellman optimality operator* $\mathcal{T}$ on the action-value functions as

$$\mathcal{T}Q(x,a)\triangleq r(x,a)+\gamma\sum_{y\in\mathcal{X}}P(y|x,a)\max_{b\in\mathcal{A}}Q(y,b),\qquad\forall(x,a)\in\mathcal{Z}.$$

Q^* is the fixed point of $\mathcal{T}$. Both $\mathcal{T}$ and $\mathcal{T}^\pi$ are contraction mappings, w.r.t. the supremum norm, with the factor γ (Bertsekas, 2007, Chapter 1). In other words, for any two real-valued action-value functions Q and Q' and every policy π, we have

$$\left\|\mathcal{T}Q-\mathcal{T}Q'\right\|\le\gamma\left\|Q-Q'\right\|,\qquad\qquad\left\|\mathcal{T}^\pi Q-\mathcal{T}^\pi Q'\right\|\le\gamma\left\|Q-Q'\right\|.$$

The policy distribution π defines a right-linear operator $P^\pi\cdot$ as

$$(P^\pi Q)(x,a)\triangleq\sum_{(y,b)\in\mathcal{Z}}\pi(b|y)P(y|x,a)Q(y,b),\qquad\forall(x,a)\in\mathcal{Z}.$$

Further, we define two other right-linear operators $\pi\cdot$ and $P\cdot$ as

$$\begin{aligned}(\pi Q)(x)&\triangleq\sum_{a\in\mathcal{A}}\pi(a|x)Q(x,a),\quad\forall x\in\mathcal{X},\\(PV)(x,a)&\triangleq\sum_{y\in\mathcal{X}}P(y|x,a)V(y),\quad\forall(x,a)\in\mathcal{Z}.\end{aligned}$$

2. We note that Q^π is the fixed point of $\mathcal{T}^\pi$.

We note that for every $Q : \mathcal{Z} \to \mathbb{R}$, $V : \mathcal{X} \to \mathbb{R}$ and policy π, we have

$$\begin{aligned} (\pi[Q+V])(x) &= (\pi Q)(x) + V(x), && \forall x \in \mathcal{X}, \\ (P^\pi[Q+V])(x,a) &= (P^\pi Q)(x,a) + (PV)(x,a), && \forall (x,a) \in \mathcal{Z}. \end{aligned} \tag{3}$$

We define the max operator $\mathcal{M}$ on the action value functions as $(\mathcal{M}Q)(x) \triangleq \max_{a \in \mathcal{A}} Q(x,a)$, for all $x \in \mathcal{X}$. Based on the new definitions one can rephrase the Bellman operator and the Bellman optimality operator as

$$\mathcal{T}^\pi Q(x,a) = r(x,a) + \gamma (P^\pi Q)(x,a), \qquad \mathcal{T} Q(x,a) = r(x,a) + \gamma (P \mathcal{M} Q)(x,a).$$

In the sequel, we repress the state(-action) dependencies in our notation wherever these dependencies are clear, for example, $\Psi(x,a)$ becomes Ψ, $Q(x,a)$ becomes Q. Also, for simplicity of the notation, we remove some parenthesis, for example, writing $\mathcal{M}Q$ for $(\mathcal{M}Q)$ and $P^\pi Q$ for $(P^\pi Q)$, when there is no possible confusion.

3. Dynamic Policy Programming

In this section, we introduce and analyze the DPP algorithm. We first present the *dynamic policy programming* (DPP) algorithm in Section 3.1 (see Appendix A for some intuition on how DPP can be related to the Bellman equation). We then investigate the finite-iteration and the asymptotic behavior of DPP and prove its convergence in Section 3.2.

3.1 Algorithm

DPP is a policy iteration algorithm which represents the policy π_k in terms of some action preference numbers Ψ_k (Sutton and Barto, 1998, Chapter 2.8). Starting at Ψ_0, DPP iterates the action preferences of all state-action pairs $(x,a) \in \mathcal{Z}$ through the DPP operator $\mathcal{O}$ (the pseudo code of DPP is presented in Algorithm 1):

$$\Psi_{k+1}(x,a) = \mathcal{O}\Psi_k(x,a) \triangleq \Psi_k(x,a) - (\mathcal{M}_\eta \Psi_k)(x) + r(x,a) + \gamma (P \mathcal{M}_\eta \Psi_k)(x,a),$$

where $\mathcal{M}_\eta$ denotes the softmax operator. The softmax operator $\mathcal{M}_\eta$ is defined on every $f : \mathcal{Z} \to \mathbb{R}$ as

$$(\mathcal{M}_\eta f)(x) \triangleq \frac{\sum_{a \in \mathcal{A}} \exp(\eta f(x,a)) f(x,a)}{\sum_{b \in \mathcal{A}} \exp(\eta f(x,b))},$$

where $\eta > 0$ is the inverse temperature.

The control policy π_k is then computed as a function of Ψ_k at each iteration k:

$$\pi_k(a|x) = \frac{\exp(\eta \Psi_k(x,a))}{\sum_{b \in \mathcal{A}} \exp(\eta \Psi_k(x,b))}, \qquad \forall (x,a) \in \mathcal{Z}. \tag{4}$$

Based on Equation 4 one can re-express the DPP operator on the action preferences Ψ_k as

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \tag{5}$$

Algorithm 1: (DPP) Dynamic Policy Programming

Input: Action preferences $\Psi_0(\cdot,\cdot)$, γ and η

for $k = 0, 1, 2, \ldots, K-1$ **do** // main loop
  foreach $(x,a) \in \mathcal{Z}$ **do** // compute the control policy
    $\pi_k(a|x) := \dfrac{\exp(\eta\Psi_k(x,a))}{\sum_{b\in\mathcal{A}} \exp(\eta\Psi_k(x,b))}$;
  end
  foreach $(x,a) \in \mathcal{Z}$ **do** // compute the new action-preferences
    $\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x)$; // DPP update rule
  end
end
foreach $(x,a) \in \mathcal{Z}$ **do** // compute the last policy
  $\pi_K(a|x) := \dfrac{\exp(\eta\Psi_K(x,a))}{\sum_{b\in\mathcal{A}} \exp(\eta\Psi_K(x,b))}$;
end
return π_K;

3.2 Performance Guarantee

In this subsection, we investigate the finite-iteration and asymptotic behavior of Algorithm 1. We begin by proving a finite-iteration performance guarantee for DPP:

Theorem 2 (The ℓ_∞-norm performance loss bound of DPP) *Let Assumption 1 hold. Also, assume that Ψ_0 is uniformly bounded by $V_{\max}$ for all $(x,a) \in \mathcal{Z}$, then the following inequality holds for the policy induced by DPP at iteration $k \geq 0$:*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma\left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta}\right)}{(1-\gamma)^2(k+1)}.$$

Proof See Appendix B.1. ■

Note that the DPP algorithm converges to the optimal policy for every $\eta > 0$ and choosing a different η only changes the rate of convergence. The best rate of convergence is achieved by setting $\eta = \infty$, for which the softmax policy and the softmax operator $\mathcal{M}_\eta$ are replaced with the greedy policy and the max-operator $\mathcal{M}$, respectively. Therefore, for $\eta = +\infty$ the DPP recursion is re-expressed as

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) - (\mathcal{M}\Psi_k)(x) + r(x,a) + \gamma(P\mathcal{M}\Psi_k)(x,a).$$

We must point out that the choice of $\eta < +\infty$ may be still useful in the presence of function approximation, where the greedy update rule can be unstable due to the non-differentiability of the max operator. In fact, our numerical results in Section 5.2 suggests that the performance of DPP in the presence of function approximation is optimized for some finite value of η rather than $\eta = +\infty$ (see Section 5.2 for more details).

As an immediate consequence of Theorem 2, we obtain the following result:

Corollary 3 *The following relation holds in limit:*

$$\lim_{k \to +\infty} Q^{\pi_k}(x,a) = Q^*(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

In words, the policy induced by DPP asymptotically converges to the optimal policy π^*. The following corollary shows that there exists a unique limit for the action preferences in infinity if the optimal policy π^* is unique.

Corollary 4 *Let Assumption 1 hold and k be a non-negative integer. Assume that the optimal policy π^* is unique and let $\Psi_k(x,a)$, for all $(x,a) \in \mathcal{Z}$, be the action preference after k iteration of DPP. Then, we have:*

$$\lim_{k \to +\infty} \Psi_k(x,a) = \begin{cases} V^*(x) & a = a^*(x) \\ -\infty & \text{otherwise} \end{cases}, \qquad \forall x \in \mathcal{X}.$$

Proof See Appendix B.2. ■

Notice that the assumption on the *uniqueness* of the optimal policy π^* is not required for the main result of this section (Theorem 2). Also, the fact that in Corollary 4 the action preferences of sub-optimal actions tend to $-\infty$ is the natural consequence of the convergence of π_k to the optimal policy π^*, which forces the probability of the sub-optimal actions to be 0.

4. Dynamic Policy Programming with Approximation

Algorithm 1 (DPP) only applies to small problems with a few states and actions. Also, to compute the optimal policy by DPP an explicit knowledge of the model is required. In many real world problems, this information is not available. Instead it may be possible to simulate the state transition by Monte-Carlo sampling and then *estimate* the optimal policy using these samples. In this section, we first prove some general bounds on the performance of DPP in the presence of approximation/estimation error and compare these bounds with those of AVI and API. We then present new approximate algorithms for implementing DPP with Monte-Carlo sampling (DPP-RL) and linear function approximation (SADPP). For both DPP-RL and SADPP we assume that we have access to the generative model of MDP, that is, an oracle can generate the next sample y from $P(\cdot|x,a)$ for every state-action pair $(x,a) \in \mathcal{Z}$ on the request of the learner.

4.1 The ℓ_∞-Norm Performance-Loss Bounds for Approximate DPP

Let us consider a sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2, \dots\}$ such that, at round k, the action preferences function Ψ_{k+1} is the result of approximately applying the DPP operator by the means of function approximation or Monte-Carlo simulation, that is, for all $(x,a) \in \mathcal{Z}$: $\Psi_{k+1}(x,a) \approx \mathcal{O}\Psi_k(x,a)$. The error ε_k is defined as the difference of $\mathcal{O}\Psi_k$ and its approximation:

$$\varepsilon_k(x,a) \triangleq \Psi_{k+1}(x,a) - \mathcal{O}\Psi_k(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{6}$$

Note that this definition of ε_k is rather general and does not specify the approximation technique used to compute Ψ_{k+1}. In the following subsections, we provide specific update rules to approximate Ψ_{k+1} for both DPP-RL and SADPP algorithms which also makes the definition of ε_k more specific.

The approximate DPP update rule then takes the following forms:

$$\begin{aligned}\Psi_{k+1}(x,a) &= \mathcal{O}\Psi_k(x,a) + \varepsilon_k(x,a) \\ &= \Psi_k(x,a) + r(x,a) + \gamma P \mathcal{M}_\eta \Psi_k(x,a) - \mathcal{M}_\eta \Psi_k(x,a) + \varepsilon_k(x,a) \\ &= \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k \Psi_k(x,a) + \varepsilon_k(x,a),\end{aligned} \tag{7}$$

where π_k is given by Equation 4.

We begin by the finite-iteration analysis of approximate DPP. The following theorem establishes an upper-bound on the performance loss of DPP in the presence of approximation error. The proof is based on generalization of the bound that we established for DPP by taking into account the error ε_k:

Theorem 5 (Finite-iteration performance loss bound of approximate DPP) *Let Assumption 1 hold. Assume that k is a non-negative integer and Ψ_0 is bounded by $V_{\max}$. Further, define ε_k for all k by Equation 6 and the accumulated error E_k as*

$$E_k(x,a) \triangleq \sum_{j=0}^{k} \varepsilon_j(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

Then the following inequality holds for the policy induced by approximate DPP at round k:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{1}{(1-\gamma)(k+1)} \left[\frac{2\gamma\left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta}\right)}{(1-\gamma)} + \sum_{j=0}^{k} \gamma^{k-j} \|E_j\| \right].$$

Proof See Appendix C. ■

Taking the upper-limit yields corollary 6.

Corollary 6 (Asymptotic performance-loss bound of approximate DPP) *Define $\bar{\varepsilon} \triangleq \limsup_{k\to\infty} \|E_k\| \,/(k+1)$. Then, the following inequality holds:*

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \bar{\varepsilon}. \tag{8}$$

The asymptotic bound is similar to the existing results of AVI and API (Thiery and Scherrer, 2010; Bertsekas and Tsitsiklis, 1996, Chapter 6):

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \varepsilon_{\max},$$

where $\varepsilon_{\max} = \limsup_{k\to\infty} \|\varepsilon_k\|$. The difference is that in Equation 8 the supremum norm of error $\varepsilon_{\max}$ is replaced by the supremum norm of the average error $\bar{\varepsilon}$. In other words, unlike AVI and API, the size of error at each iteration is not a critical factor for the performance of DPP and as long as the size of average error remains close to 0, DPP is guaranteed to achieve a near-optimal performance even when the individual errors ε_k are large

As an *example*: Consider a case in which, for both DPP and AVI/API, the sequence of errors $\{\varepsilon_0, \varepsilon_1, \varepsilon_2, \dots\}$ are some i.i.d. zero-mean random variables bounded by $0 < U < \infty$. Corollary 6 combined with the law of large numbers then leads to the following asymptotic bound for approximate DPP:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\bar{\varepsilon} = 0, \qquad \text{w.p. (with probability) } 1,$$

whilst for API and AVI we have

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}U.$$

In words, approximate DPP manages to cancel i.i.d. noise and asymptotically converges to the optimal policy whereas there is no guarantee, in this case, for the convergence of API and AVI to the optimal solution. This example suggests that DPP, in general, may average out some of the simulation noise caused by Monte-Carlo sampling and eventually achieve a better performance than AVI and API in the presence of sampling error.

Remark 7 *The i.i.d. assumption may be replaced by some weaker and more realistic assumption that only requires the error sequence* $\{\varepsilon_0, \varepsilon_1, \dots, \varepsilon_k\}$ *to be a sequence of martingale differences, that is, the errors do not need to be independent as long as the expected value of* ε_k*, conditioned on the past observations, is* 0*. We prove, in the next subsection, that DPP-RL satisfies this assumption and, therefore, asymptotically converges to the optimal policy (see Theorem 9).*

4.2 Reinforcement Learning with Dynamic Policy Programming

To compute the optimal policy by DPP one needs an explicit knowledge of the model. In many problems, we do not have access to this information but instead we can generate samples by simulating the model. The optimal policy can then be *learned* using these samples. In this section, we introduce a new RL algorithm, called DPP-RL, which relies on a sampling-based variant of DPP to update the policy. The update rule of DPP-RL is very similar to Equation 5. The only difference is that we replace the Bellman operator $\mathcal{T}^{\pi}\Psi(x,a)$ with its sample estimate $\mathcal{T}_k^{\pi}\Psi(x,a) \triangleq r(x,a) + \gamma(\pi\Psi)(y_k)$, where the next sample y_k is drawn from $P(\cdot|x,a)$:

$$\Psi_{k+1}(x,a) \triangleq \Psi_k(x,a) + \mathcal{T}_k^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \tag{9}$$

Based on Equation 9, we estimate the optimal policy by iterating some initial Ψ_0 through the DPP-RL update rule, where at each iteration we draw y_k for every $(x,a) \in \mathcal{Z}$. From Equation 6, the estimation error of the k^{th} iterate of DPP-RL is then defined as the difference between the Bellman operator $\mathcal{T}^{\pi_k}\Psi_k(x,a)$ and its sample estimate:

$$\varepsilon_k(x,a) = \mathcal{T}_k^{\pi_k}\Psi_k(x,a) - \mathcal{T}^{\pi_k}\Psi_k(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

The DPP-RL update rule can then be considered as a special case of the more general approximate DPP update rule of Equation 7.

Equation 9 is just an approximation of the DPP update rule of Equation 5. Therefore, the convergence result of Corollary 3 does not hold for DPP-RL. However, the new algorithm still converges to the optimal policy since one can show that the errors associated with approximating Equation 5 are asymptotically *averaged out* by DPP-RL, as postulated by Corollary 6. To prove this result we need the following lemma, which bounds the estimation error ε_k.

Lemma 8 (Boundedness of ε_k) *Let Assumption 1 hold and assume that the initial action-preference function Ψ_0 is uniformly bounded by $V_{\max}$, then we have, for all $k \geq 0$,*

$$\left\|\mathcal{T}_k^{\pi_k}\Psi_k\right\| \leq \frac{2\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + V_{\max}, \qquad \|\varepsilon_k\| \leq \frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + 2V_{\max}.$$

Proof See Appendix D. ■

Lemma 8 is an interesting result, which shows that, despite the fact that Ψ_k tends to $-\infty$ for the sub-optimal actions, the error ε_k is uniformly bounded by some finite constant. Note that $\varepsilon_k = \mathcal{T}^{\pi_k}\Psi_k - \mathcal{T}_k^{\pi_k}\Psi_k$ can be expressed in terms of the soft-max $\mathcal{M}_\eta\Psi_k$, which unlike Ψ_k, is always bounded by a finite constant, for every $\eta > 0$.

The following theorem establishes the asymptotic convergence of DPP-RL to the optimal policy.

Theorem 9 (Asymptotic convergence of DPP-RL) *Let Assumption 1 hold. Assume that the initial action-value function Ψ_0 is uniformly bounded by $V_{\max}$ and π_k is the policy induced by Ψ_k after k iteration of DPP-RL. Then, w.p. 1, the following holds:*

$$\lim_{k\to\infty} Q^{\pi_k}(x,a) = Q^*(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

Proof See Appendix D.1. ■

We also prove the following result on the converge rate of DPP-RL to the optimal policy by making use of the result of Theorem 5:

Theorem 10 (Finite-time high-probability loss-bound of DPP-RL) *Let Assumption 1 hold and k be a positive integer and $0 < \delta < 1$. Then, at iteration k of DPP-RL with probability at least $1-\delta$, we have*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{4(\gamma\log(|\mathcal{A}|)/\eta + 2R_{\max})}{(1-\gamma)^3}\left[\frac{1}{k+1} + \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}\right].$$

Proof See Appendix D.2. ■

Theorem 5 implies that, regardless of the value of η and γ, DPP-RL always converges with the rate of $1/\sqrt{k}$.

We can optimize the bound of Theorem 10 w.r.t. η which leads to the following corollary:

Corollary 11 *Let Assumption 1 hold and k be a positive integer, also set the inverse temperature* $\eta = +\infty$*, Then, at iteration k of DPP-RL with probability at least* $1-\delta$*, we have*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{8R_{\max}}{(1-\gamma)^3}\left[\frac{1}{k+1} + \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}\right].$$

This result implies that, in order to achieve the best rate of convergence, one can set the value of η to $+\infty$, that is, to replace the soft-max $\mathcal{M}_\eta$ with the max operator $\mathcal{M}$:

$$\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}_k\Psi_k(x,a) - \mathcal{M}\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z},$$

where $\mathcal{T}_k\Psi(x,a) \triangleq r(x,a) + \gamma(\mathcal{M}\Psi)(y_k)$ for all $(x,a) \in \mathcal{Z}$. The pseudo-code of DPP-RL algorithm, which sets $\eta = +\infty$, is shown in Algorithm 2.

Algorithm 2: (DPP-RL) Reinforcement learning with DPP

Input: Initial action preferences $\Psi_0(\cdot,\cdot)$, discount factor γ and number of steps T

for $k = 1,2,3,\ldots,K-1$ **do** `// main loop`
  foreach $(x,a) \in \mathcal{Z}$ **do** `// update` $\Psi_k(\cdot,\cdot)$ `for all state-action pairs`
    $y_k \sim P(\cdot|x,a)$; `// generate the next sample`
    $\mathcal{T}_k\Psi_k(x,a) := r(x,a) + \gamma\mathcal{M}\Psi_k(y_k)$; `// empirical Bellman operator`
    $\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}_k\Psi_k(x,a) - \mathcal{M}\Psi_k(x)$; `// DPP update rule`
  end
  foreach $x \in \mathcal{X}$ **do** `// compute the control policy`
    $a_{\max} := \arg\max_{a\in\mathcal{A}} \Psi_{k+1}(x,a)$;
    $\pi(\cdot|x) := 0$;
    $\pi_{k+1}(a_{\max}|x) := 1$;
  end
end
return π_K

Furthermore, the following PAC bound which determines the number of steps k required to achieve the error $\varepsilon > 0$ in estimating the optimal policy, w.p. $1-\delta$, is an immediate consequence of Theorem 10.

Corollary 12 (Finite-time PAC bound of DPP-RL) *Let Assumption 1 hold. Then, for any* $\varepsilon > 0$*, after*

$$k = \frac{256R_{\max}^2 \log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{(1-\gamma)^6\varepsilon^2}.$$

steps of Algorithm 2, the uniform approximation error $\|Q^* - Q^{\pi_k}\| \leq \varepsilon$*, w. p.* $1-\delta$*.*

4.3 Approximate Dynamic Policy Programming with Linear Function Approximation

In this subsection, we consider DPP with *linear function approximation* (LFA) and *least-squares regression*. LFA is commonly used in many RL algorithms (Szepesvári, 2010, Section 3.2). Given

a set of basis functions $\mathcal{F}_\phi = \{\phi_1, \ldots, \phi_m\}$, where each $\phi_i : \mathcal{Z} \to \mathbb{R}$ is a bounded real valued function, the sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2 \ldots\}$ are defined as a linear combination of these basis functions: $\Psi_k = \theta_k^\mathsf{T} \Phi$, where Φ is a $m \times 1$ column vector with the entries $\{\phi_i\}_{i=1:m}$ and $\theta_k \in \mathbb{R}^m$ is a $m \times 1$ vector of parameters.

The action preference function Ψ_{k+1} is an approximation of the DPP operator $\mathcal{O}\Psi_k$. In the case of LFA the common approach to approximate DPP operator is to find a vector θ_{k+1} that projects $\mathcal{O}\Psi_k$ on the column space spanned by Φ by minimizing the loss function:

$$J_k(\theta; \Psi) \triangleq \left\| \theta^\mathsf{T} \Phi - \mathcal{O}\Psi_k \right\|_{2,\mu}^2,$$

where μ is a probability measure on $\mathcal{Z}$. The best solution, which minimizes J, is called the least-squares solution:

$$\theta_{k+1} = \arg\min_{\theta \in \mathbb{R}^m} J_k(\theta; \Psi) = \left[\mathbb{E}\left(\Phi \Phi^\mathsf{T}\right)\right]^{-1} \mathbb{E}(\Phi \mathcal{O}\Psi_k),$$

where the expectation is taken w.r.t. $(x,a) \sim \mu$. In principle, to compute the least squares solution equation one needs to compute $\mathcal{O}\Psi_k$ for all states and actions. For large scale problems this becomes infeasible. Instead, one can make a sample estimate of the least-squares solution by minimizing the empirical loss $\widetilde{J}_k(\theta; \Psi)$ (Bertsekas, 2007, Chapter 6.3):

$$\widetilde{J}_k(\theta; \Psi) \triangleq \frac{1}{N} \sum_{n=1}^{N} (\theta^\mathsf{T} \Phi(X_n, A_n) - \mathcal{O}_n \Psi_k)^2 + \alpha \theta^\mathsf{T} \theta,$$

where $\{(X_n, A_n)\}_{n=1:N}$ is a set of N i.i.d. samples drawn from the distribution μ. Also, $\mathcal{O}_n \Psi_k$ denotes a single sample estimate of $\mathcal{O}\Psi_k(X_n, A_n)$ defined by $\mathcal{O}_n \Psi_k \triangleq \Psi_k(X_n, A_n) + r(X_n, A_n) + \gamma \mathcal{M}_\eta \Psi_k(Y_n) - \mathcal{M}_\eta \Psi_k(X_n)$, where $Y_n \sim P(\cdot | X_n, A_n)$. Further, to avoid over-fitting due to the small number of samples, one adds a quadratic regularization term to the loss function. The empirical least-squares solution which minimizes $\widetilde{J}_k(\theta; \Psi)$ is given by

$$\widetilde{\theta}_{k+1} = \left[\sum_{n=1}^{N} \Phi(X_n, A_n) \Phi(X_n, A_n)^\mathsf{T} + \alpha N \mathbf{I} \right]^{-1} \sum_{n=1}^{N} \mathcal{O}_n \Psi_k \Phi(X_n, A_n), \tag{10}$$

and $\Psi_k(x,a) = \widetilde{\theta}_{k+1} \Phi(x,a)$. This defines a sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2, \ldots\}$ and the sequence of approximation error through Equation 6.

Algorithm 3 presents the *sampling-based approximate dynamic policy programming* (SADPP) in which we rely on Equation 10 to approximate DPP operator at each iteration.

5. Numerical Results

In this section, we illustrate empirically the theoretical performance guarantee introduced in the previous sections for both variants of DPP: the exact case (DPP-RL) and the approximate case (SADPP). In addition, we compare with existing algorithms for which similar theoretical results have been derived.

We first examine the convergence properties of DPP-RL (Algorithm 2) on several discrete state-action problems with large state spaces. We compare it with a synchronous variant of Q-learning

Algorithm 3: (SADPP) Sampling-based approximate dynamic policy programming

Input: $\widetilde{\theta}_0$, η, γ, α, K and N

for $k = 0, 1, 2, \ldots, K-1$ **do** // main loop

  $\{(X_n, A_n)\}_{n=1:N} \sim \mu(\cdot,\cdot)$; // generate n i.i.d. samples from $\mu(\cdot,\cdot)$

  $\{Y_n\}_{n=1:N} \sim P(\cdot|\{(X_n, A_n)\}_{n=1:N})$; // generate next states from $P(\cdot|\cdot)$

  foreach $n = 1, 2, 3, \ldots, N$ **do**

    foreach $a \in \mathcal{A}$ **do** // compute Ψ_k for every action of states X_n, Y_n

      $\Psi_k(X_n, a) = \widetilde{\theta}_k^{\mathsf{T}} \Phi(X_n, a)$;

      $\Psi_k(Y_n, a) = \widetilde{\theta}_k^{\mathsf{T}} \Phi(Y_n, a)$;

    end

    $\mathcal{M}_\eta \Psi_k(X_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(X_n,a)) \Psi_k(X_n,a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(X_n,b)}$;

    $\mathcal{M}_\eta \Psi_k(Y_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(Y_n,a)) \Psi_k(Y_n,a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(Y_n,b)}$; // soft-max $\mathcal{M}_\eta \Psi_k$ for X_n and Y_n

    // empirical DPP operator

    $\mathcal{O}_n \Psi_k = \Psi_k(X_n, A_n) - r(X_n, A_n) - \gamma (\mathcal{M}_\eta \Psi_k)(Y_n) + (\mathcal{M}_\eta \Psi_k)(X_n)$;

  end

  // SADPP update rule

  $\widetilde{\theta}_{k+1} = \left[\sum_{n=1}^N \Phi(X_n, A_n) \Phi(X_n, A_n)^{\mathsf{T}} + \alpha N \mathbf{I}\right]^{-1} \sum_{n=1}^N \mathcal{O}_n \Psi_k \Phi(X_n, A_n)$;

end

return $\widetilde{\theta}_K$

(Even-Dar and Mansour, 2003) (QL) and a model-based Q-value iteration (VI) (Kearns and Singh, 1999). Next, we investigate the finite-time performance of SADPP (Algorithm 3) in the presence of function approximation and a limited sampling budget per iteration. In this case, we compare SADPP with regularized least-squares fitted Q-iteration (RFQI) (Farahmand et al., 2008) and regularized least-squares policy iteration (REG-LSPI) (Farahmand et al., 2009), two algorithms that, like SADPP, control the complexity of the solution using regularization.[3]

5.1 DPP-RL

To illustrate the performance of DPP-RL, we consider the following MDPs:

Linear MDP: this problem consists of states $x_k \in \mathcal{X}, k = \{1, 2, \ldots, 2500\}$ arranged in a one-dimensional chain (see Figure 1). There are two possible actions $\mathcal{A} = \{-1, +1\}$ (left/right) and every state is accessible from any other state except for the two ends of the chain, which are absorbing states. A state $x_k \in \mathcal{X}$ is called absorbing if $P(x_k|x_k, a) = 1$ for all $a \in \mathcal{A}$ and $P(x_l|x_k, a) = 0, \forall l \neq k$. The state space is of size $|\mathcal{X}| = 2500$ and the joint action state space is of size $|\mathcal{Z}| = 5000$. Note that naive storing of the model requires $\mathcal{O}(10^7)$ memory.

The transition probability from an interior state x_k to any other state x_l is inversely proportional to the distance in the direction of the selected action. Formally, consider the following quantity $n(x_l, a, x_k)$ assigned to all non-absorbing states x_k and to every $(x_l, a) \in \mathcal{Z}$:

3. The source code of all tested algorithms is available in `http://www.mbfys.ru.nl/~mazar/Research_Top.html`.

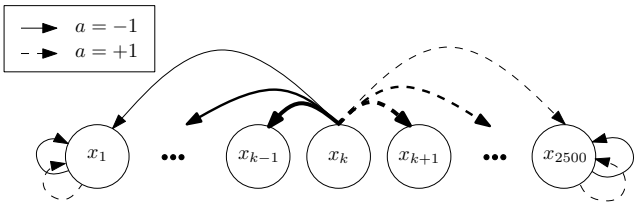


Figure 1: Linear MDP: Illustration of the linear MDP problem. Nodes indicate states. States x_1 and x_{2500} are the two absorbing states and state x_k is an example of interior state. Arrows indicate possible transitions of **these three nodes only**. From x_k any other node is reachable with transition probability (arrow thickness) proportional to the inverse of the distance to x_k (see the text for details).

$$n(x_l, a, x_k) = \begin{cases} \dfrac{1}{|l-k|} & \text{for } (l-k)a > 0 \\ 0 & \text{otherwise} \end{cases}.$$

We can write the transition probabilities as

$$P(x_l|x_k, a) = \frac{n(x_l, a, x_k)}{\sum_{x_m \in \mathcal{X}} n(x_m, a, x_k)}.$$

Transitions to an absorbing state have associated reward 1 and transitions to any interior state has associated reward -1.

The optimal policy corresponding to this problem is to reach the closest absorbing state as soon as possible.

Combination lock: the combination lock problem considered here is a stochastic variant of the reset state space models introduced in Koenig and Simmons (1993), where more than one reset state is possible (see Figure 2).

In our case we consider, as before, a set of states $x_k \in \mathcal{X}, k \in \{1, 2, \ldots, 2500\}$ arranged in a one-dimensional chain and two possible actions $\mathcal{A} = \{-1, +1\}$. In this problem, however, there is only one absorbing state (corresponding to the state *lock-opened*) with associated reward of 1. This state is reached if the all-ones sequence $\{+1, +1, \ldots, +1\}$ is entered correctly. Otherwise, if at some state x_k, $k < 2500$, action -1 is taken, the lock automatically resets to some previous state x_l, $l < k$ randomly (in the original problem, the reset state is always the initial state x_1).

For every intermediate state, the rewards of actions -1 and $+1$ are set to 0 and -0.01, respectively. The transition probability upon taking the wrong action -1 from state x_k to state x_l is $P(x_l|x_k, -1)$, as before, inversely proportional to the distance of the states. That is

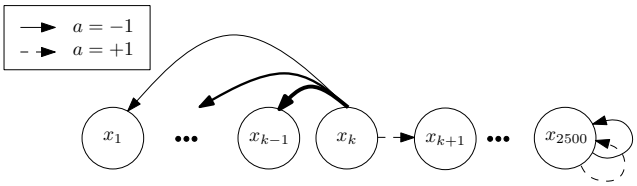


Figure 2: Combination lock: illustration of the combination lock MDP problem. Nodes indicate states. State x_{2500} is the goal (absorbing) state and state x_k is an example of interior state. Arrows indicate possible transitions of **these two nodes only**. From x_k any previous state is reachable with transition probability (arrow thickness) proportional to the inverse of the distance to x_k. Among the future states only x_{k+1} is reachable (arrow dashed).

$$n(x_k, x_l) = \begin{cases} \dfrac{1}{k-l} & \text{for } l < k \\ 0 & \text{otherwise} \end{cases}, \qquad P(x_l | x_k, -1) = \frac{n(x_k, x_l)}{\sum\limits_{x_m \in \mathcal{X}} n(x_k, x_m)}.$$

Note that this problem is more difficult than the linear MDP since the goal state is only reachable from one state, x_{2499}.

Grid world: this MDP consists of a grid of 50×50 states. A set of four actions {RIGHT, UP, DOWN, LEFT} is assigned to every state $x \in \mathcal{X}$. Although the state space of the grid world is of the same size as the previous two problems, $|\mathcal{X}| = 2500$, the joint action state space is larger, $|\mathcal{Z}| = 10^4$.

The location of each state x of the grid is determined by the coordinates $c_x = (h_x, v_x)$, where h_x and v_x are some integers between 1 and 50. There are 196 absorbing *wall states* surrounding the grid and another one at the center of grid, for which a reward -1 is assigned. The reward for the walls is

$$r(x, a) = -\frac{1}{\|c_x\|_2}, \qquad \forall a \in \mathcal{A}.$$

Also, we assign reward 0 to all of the remaining (non-absorbing) states.

This means that both the top-left absorbing state and the central state have the least possible reward (-1), and that the remaining absorbing states have reward which increases proportionally to the distance to the state in the bottom-right corner (but are always negative).

The transition probabilities are defined in the following way: taking action a from any non-absorbing state x results in a one-step transition in the direction of action a with probability 0.6, and a random move to a state $y \neq x$ with probability inversely proportional to their Euclidean distance $1/\|c_x - c_y\|_2$.

This problem is interesting because of the presence of the absorbing walls, which prevent the agent to escape and because of the high level of noise: from a non-absorbing state, many states are reachable with significant probability.

The resulting optimal policy is to *survive* in the grid as long as possible by avoiding both the absorbing walls and the center of the grid. Note that because of the difference between the cost of walls, the optimal control prefers the states near the bottom-right corner of the grid, thus avoiding absorbing states with higher cost.

5.1.1 EXPERIMENTAL SETUP AND RESULTS

For consistency with the theoretical results, we evaluate the performance of all algorithms in terms of ℓ_∞-norm error of the action-value function $\|Q^* - Q^{\pi_k}\|$ obtained by policy π_k induced at iteration k. The discount factor γ is fixed to 0.995 and the optimal action-value function Q^* is computed with high accuracy through value iteration.

We compare DPP-RL with two other algorithms:

Q-learning (QL): we consider a synchronous variant of Q-learning for which convergence results have been derived in Even-Dar and Mansour (2003). Since QL is sensitive to the learning step, we consider QL with polynomial learning step $\alpha_k = 1/(k+1)^\omega$ where $\omega \in \{0.51, 0.75, 1.0\}$. It is known that ω needs to be larger than 0.5, otherwise QL may not asymptotically converge (see Even-Dar and Mansour, 2003, for the proof).

Model-based Q-value iteration (VI): The VI algorithm (Kearns and Singh, 1999) first estimates a model using all the data samples and then performs value iteration on the learned model. Therefore, unlike QL and DPP, VI is a model-based algorithm and requires the algorithm to store the model.

Comparison between VI and both DPP-RL and QL is especially problematic: first, the number of computations per iteration is different. Whereas DPP-RL and QL require $|\mathcal{Z}|$ computations per iteration, VI requires $|\mathcal{Z}||\mathcal{X}|$. Second, VI requires to estimate the model initially (using a given number of samples) and then iterates until convergence. This latter aspect is also different from DPP-RL and QL, which use one sample per iteration. Therefore, the number of samples determines the number of iterations for DPP-RL and QL, but not for VI.

For consistency with the theoretical results, we use as error measure, the distance between the optimal action-value function and the value function of the policy induced by the algorithms. instead of the more popular average accumulated reward, which is usually used when the RL algorithm learns from a stream of samples.

Simulations are performed using the following procedure: at the beginning of each run **(i)** the action-value function and the action preferences are randomly initialized in the interval $[-V_{\max}, V_{\max}]$, and **(ii)** a set of 10^5 samples is generated from $P(\cdot|x,a)$ for all $(x,a) \in \mathcal{Z}$. As mentioned before, this fixes the maximum number of iterations for DPP-RL and QL to 10^5, but not for VI. We run VI until convergence. We repeat this procedure 50 times and compute the average error in the end. Using significantly fewer samples leads to a dramatic decrease of the quality of the solutions using all approaches and no qualitative differences in the comparison.

To compare the methods using equivalent logical units independently of the particular implementation, we rescale their number of iterations by the number of steps required in one iteration. For the case of VI, the *step* units are the number of iterations times $|\mathcal{Z}||\mathcal{X}|$ and for DPP-RL and QL, the number of iterations times $|\mathcal{Z}|$.

Figure 3 shows the error as a function of the number of steps. First, in agreement with the theoretical results, we observe that the DPP-error decays very fast in the beginning and keeps decreasing

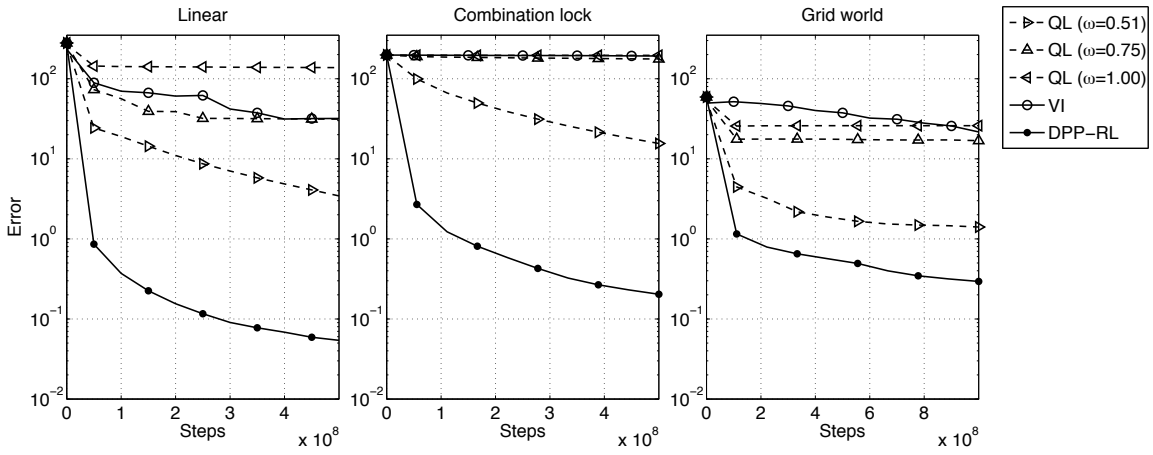


Figure 3: Comparison between DPP-RL, QL and VI in terms of **number of steps**, defined as the number of iterations times the number of computations per iteration of the particular algorithm. Each plot shows the averaged error of the induced policies over 50 different runs (see the text for details).

at a smaller rate afterwards. We also observe that DPP-RL performs significantly better than QL. The improvement is about two orders of magnitude in both the linear MDP and the combination lock problems and more than four times better in the Grid world. QL shows the best performance for $\omega = 0.51$ and the quality degrades as a function of ω.

Although the performance of VI looks poor for the number of steps shown in Figure 3, we observe that VI reaches an average error of 0.019 after convergence ($\approx 2 \cdot 10^{10}$ steps) for the linear MDP and the combination lock and an error of 0.10 after $\approx 4 \cdot 10^{10}$ steps for the grid problem. This means for a fixed number of samples, the asymptotic solution of VI is better than the one of DPP-RL, at the cost of much larger number of steps.

To illustrate the performance of the methods using a limited CPU time budget, we also compare the average and standard deviations of the errors in terms of elapsed CPU time by running the algorithms until a maximum allowed time is reached. We choose 30 seconds in the case of linear MDP and combination lock and 60 seconds for the grid world, which has twice as many actions as the other benchmarks. To minimize the implementation dependent variability, we coded all three algorithms in C++ and ran them on the same processor. CPU time was acquired using the system function `times()` which provides process-specific CPU time. Sampling time was identical for all methods and not included in the analysis.

Table 1 shows the final average errors (standard deviations between parenthesis) in the CPU time comparison. As before, we observe that DPP-RL converges very fast, achieving near optimal performance after a few seconds. The small variance of estimation of DPP-RL suggests that, as derived in Theorems 9 and 5, DPP-RL manages to average out the simulation noise caused by sampling and converges to a near optimal solution, which is very robust.

Overall, these results complement the theory presented in previous sections. We can conclude that for the chosen benchmarks DPP-RL converges significantly faster than VI and QL. However,

Benchmark		Linear MDP		Combination lock		Grid world	
Run Time		30 sec.		30 sec.		60 sec.	
DPP-RL		**0.05**	(**0.02**)	**0.20**	(**0.09**)	**0.32**	(**0.03**)
VI		16.60	(11.60)	69.33	(15.38)	5.67	(1.73)
QL	$\omega = 0.51$	4.08	(3.21)	18.18	(4.36)	1.46	(0.12)
	$\omega = 0.75$	31.41	(12.77)	176.13	(25.68)	17.21	(7.31)
	$\omega = 1.00$	138.01	(146.28)	195.74	(5.73)	25.92	(20.13)

Table 1: Comparison between DPP-RL, QL and VI given a fixed computational and sampling budget. Table 1 shows error means and standard deviations (between parenthesis) at the end of the simulations for three different algorithms (columns) and three different benchmarks (rows).

for a fixed number of samples, VI obtains a better solution than DPP-RL requiring significantly more computation.

5.2 SADPP

In this subsection, we illustrate the performance of the SADPP algorithm in the presence of function approximation and limited sampling budget per iteration. The purpose of this subsection is to analyze numerically the sample complexity, that is, the number of samples required to achieve a near optimal performance with low variance.

We compare SADPP with ℓ_2-regularized versions of the following two algorithms:

Regularized fitted Q-iteration (RFQI) (Farahmand et al., 2008):

> RFQI performs value iteration to approximate the optimal action value function. See also Antos et al. (2008) and Ernst et al. (2005).

Regularized Least Squares Policy Iteration (REG-LSPI) (Farahmand et al., 2009):

> It can be regarded as a Monte-Carlo sampling implementation of approximate policy iteration (API) with action-state representation (see also Lagoudakis and Parr, 2003).

The benchmark we consider is a variant of the *optimal replacement problem* presented in Munos and Szepesvári (2008).

5.2.1 OPTIMAL REPLACEMENT PROBLEM

This problem is an infinite-horizon, discounted MDP. The state measures the accumulated use of a certain product and is represented as a continuous, one-dimensional variable. At each time-step t, either the product is kept $a(t) = 0$ or replaced $a(t) = 1$. Whenever the product is replaced by a new one, the state variable is reset to zero $x(t) = 0$, at an additional cost C. The new state is chosen according to an exponential distribution, with possible values starting from zero or from the current state value, depending on the latest action:

$$p(y|x,a=0) = \begin{cases} \beta e^{\beta(y-x)} & \text{if } y \geq x \\ 0 & \text{if } y < 0 \end{cases} \qquad p(y|x,a=1) = \begin{cases} \beta e^{\beta} y & \text{if } y \geq 0 \\ 0 & \text{if } y < 0 \end{cases}.$$

The reward function is a monotonically decreasing function of the state x if the product is kept $r(x,0) = -c(x)$ and constant if the product is replaced $r(x,1) = -C - c(0)$, where $c(x) = 4x$.

The optimal action is to keep as long as the accumulated use is below a threshold or to replace otherwise:

$$a^*(x) = \begin{cases} 0 & \text{if } x \in [0, \bar{x}] \\ 1 & \text{if } x > \bar{x} \end{cases}. \tag{11}$$

Following Munos and Szepesvári (2008), $\bar{x}$ can be obtained exactly via the Bellman equation and is the unique solution to

$$C = \int_0^{\bar{x}} \frac{c'(y)}{1-\gamma} \left(1 - \gamma e^{-\beta(1-\gamma)y}\right) dy.$$

5.2.2 EXPERIMENTAL SETUP AND RESULTS

For all algorithms we map the state-action space using twenty radial basis functions (ten for the continuous one-dimensional state variable x, spanning the state space $\mathscr{X}$, and two for the two possible actions). Other parameter values where chosen to be the same as in Munos and Szepesvári (2008), that is, $\gamma = 0.6, \beta = 0.5, C = 30$, which results in $\bar{x} \simeq 4.8665$. We also fix an upper bound for the states, $x_{\max} = 10$ and modify the problem definition such that if the next state y happens to be outside of the domain $[0, x_{\max}]$ then the product is replaced immediately, and a new state is drawn as if action $a = 1$ were chosen in the previous time step.

We measure the performance loss of the algorithms in terms of the difference between the optimal action a^* and the action selected by the algorithms. We use this performance measure since it is easy to compute as we know the analytical solution of the optimal control in the optimal replacement problem (see Equation 11). We discretize the state space in $K = 100$ and compute the error as follows:

$$\text{Error} = \frac{1}{K} \sum_{k=1}^{K} |a^*(x_k) - \hat{a}(x_k)|, \tag{12}$$

where $\hat{a}$ is the action selected by the algorithm. Note that, unlike RFQI and REG-LSPI, SADPP induces a stochastic policy, that is, a distribution over actions. We select $\hat{a}$ for SADPP by choosing the most probable action from the induced soft-max policy, and then use this to compute Equation 12. RFQI and REG-LSPI select the action with highest action-value function.

Simulations are performed using the same following procedure for all three algorithms: at the beginning of each run, the vector $\widetilde{\theta}_0$ is initialized in the interval $[-1,1]$. We then let the algorithm run for 10^3 iterations for 200 different runs. A new independent set of samples is generated at each iteration.

For each of the algorithms and each N, we optimize their parameters for the best asymptotic performance. Note that SADPP, in addition to the regularizer parameter α, has an extra degree of freedom η. Empirically, we observe that the optimal performance of SADPP is attained for finite η. This differs from DPP-RL, for which the convergence rate is optimized for $\eta = \infty$. This difference may be related to the observation that replacing the non-differentiable max-operator ($\eta = +\infty$) with a differentiable soft-max operator ($\eta < +\infty$) can improve the convergence behavior of the algorithm, as shown in Perkins and Precup (2003); de Farias and Van Roy (2000).

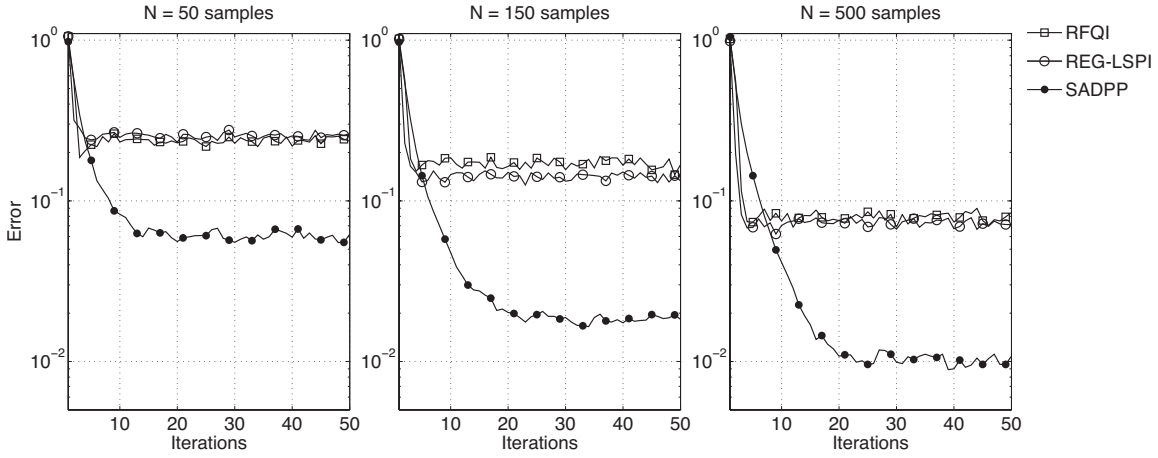


Figure 4: Numerical results for the optimal replacement problem. Each plot shows the error of RFQI, REG-LSPI and SADPP for certain number of samples N. Error is defined as in Equation 12 and averaged over 200 repetitions (see the text for details).

Num. samples	50		150		500	
SADPP	**0.07**	(**0.06**)	**0.02**	(**0.01**)	**0.01**	(**0.01**)
RFQI	0.24	(0.19)	0.17	(0.12)	0.08	(0.07)
REG-LSPI	0.26	(0.16)	0.13	(0.10)	0.07	(0.06)

Table 2: Comparison between SADPP, RFQI and REG-LSPI for the optimal replacement problem. Table shows error means and standard deviations (between parenthesis) at the end of the simulations (after 10^3 iterations) for the three different algorithms (columns) and three different number of samples (rows).

We are interested in the behavior of the error as a function of the iteration number for different number of samples N per iteration. Figure 4 and Table 2 show the performance results of the three different algorithms for $N \in \{50, 150, 500\}$ for the first 50 iterations and the total 10^3 iterations respectively. We observe that after an initial transient, all algorithms reach a nearly optimal solution after 50 iterations.

First, we note that SADPP asymptotically outperforms RFQI and REG-LSPI on average in all cases. Interestingly, there is no significant difference between the performance of RFQI and REG-LSPI. The performance of all algorithms improve for larger N. We emphasize that SADPP using only 50 samples shows comparable results to both RFQI and REG-LSPI using ten times more samples.

A comparison of the variances after the transient (see Table 2) shows that the sample complexity of SADPP is significantly smaller than RFQI and REG-LSPI. The variance of SADPP using again only 50 samples is comparable to the one provided by the other two methods using $N = 500$ samples.

Globally, we can conclude that SADPP has positive effects in reducing the effect of simulation noise, as postulated in Section 4. We can also conclude that, for our choice of settings, SADPP outperforms RFQI and REG-LSPI.

6. Related Work

In this section, we review some previous RL methods and compare them with DPP.

Policy-gradient actor-critic methods: As we explained earlier in Section 1, actor-critic method is a popular incremental RL algorithm (Sutton and Barto, 1998; Barto et al., 1983, Chapter 6.6), which makes use of a separate structure to store the value function (critic) and the control policy (actor). An important extension of AC, the *policy-gradient actor critic* (PGAC), extends the idea of AC to problems of practical scale (Sutton et al., 2000; Peters and Schaal, 2008). In PGAC, the actor updates the parameterized policy in the direction of the (natural) gradient of performance, provided by the critic. The gradient update ensures that PGAC asymptotically converges to a local maximum, given that an unbiased estimate of the gradient is provided by the critic (Maei et al., 2010; Bhatnagar et al., 2009; Konda and Tsitsiklis, 2003; Kakade, 2002). The parameter η in DPP is reminiscent of the learning step β in PGAC methods, since it influences the rate of change of the policy and in this sense may play a similar role as the learning step β in PGAC (Konda and Tsitsiklis, 2003; Peters and Schaal, 2008). However, it is known that in the presence of sampling error, asymptotic convergence to a local maximum is only attained when β asymptotically decays to zero (Konda and Tsitsiklis, 2003; Baxter and Bartlett, 2001), whereas the parameter η in DPP, and DPP-RL, can be an arbitrary constant.

Q-learning: DPP is not the only method which relies on an incremental update rule to control the sampling error. There are other incremental RL methods which aim to address the same problem (see, e.g., Maei et al., 2010; Singh et al., 2000; Watkins and Dayan, 1992).

One of the most well-known algorithms of this kind is Q-learning (QL) (Watkins and Dayan, 1992), which controls the sampling error by introducing a decaying learning step to the update rule of value iteration. QL has been shown to converge to the optimal value function in tabular case (Bertsekas and Tsitsiklis, 1996; Jaakkola et al., 1994). Also, there are some studies in the literature concerning the asymptotic convergence of Q-learning in the presence of function approximation (Melo et al., 2008; Szepesvári and Smart, 2004). However, the convergence rate of QL is very sensitive to the choice of learning step, and a bad choice of the learning step may lead to a slow rate of convergence (Even-Dar and Mansour, 2003). For instance, the convergence rate of QL with a linearly decaying learning step is of order $(1/k)^{1-\gamma}$, which makes the Q-learning algorithm extremely slow for $\gamma \approx 1$ (Szepesvári, 1998). This is in contrast to our previously mentioned result on the convergence of DPP-RL in Theorem 10 which guarantees that, regardless of the value of η and γ, DPP-RL always converges to the optimal policy with a rate of order $1/\sqrt{k}$. The numerical results of Section 5.1 confirm the superiority of DPP-RL to QL in terms of the rate of convergence.

One can also compare the finite-time behavior of DPP-RL and QL in terms of the PAC sample complexity of these methods. We have proven a sample-complexity PAC bound of order $O(1/(1-\gamma)^6)$ for DPP-RL in Section 4.2, whereas the best existing PAC bound for standard QL, to find an ε-optimal policy, is of order $O(1/(1-\gamma)^7)$ (Even-Dar and Mansour, 2003;

Azar et al., 2012, Section 3.3.1).[4] This theoretical result suggests that DPP-RL is superior to QL in terms of sample complexity of the estimating the optimal policy, especially, when γ is close to 1.

There is an on-policy version of Q-learning algorithm called SARSA (see, e.g., Singh et al., 2000) which also guarantees the asymptotic convergence to the optimal value function. However little is known about the rate of convergence and the finite-time behavior of this algorithm.

Very recently, Azar et al. (2012) propose a new variant of Q-learning algorithm, called speedy Q-learning (SQL), which makes use of a different update rule than standard Q-learning of Watkins and Dayan (1992). Like DPP-RL, SQL converges to the optimal policy with the rate of convergence of order $1/\sqrt{k}$. However, DPP-RL is superior to SQL in terms of memory space requirement, since SQL needs twice as much space as DPP-RL does.

Relative-entropy methods: The DPP algorithm is originally motivated (see Appendix A) by the work of Kappen (2005) and Todorov (2007), who formulate a stochastic optimal control problem to find a conditional probability distribution $p(y|x)$ given an uncontrolled dynamics $\bar{p}(y|x)$. The control cost is the relative entropy between $p(y|x)$ and $\bar{p}(y|x)\exp(r(x))$. The difference is that in their work a restricted class of control problems is considered for which the optimal solution p can be computed directly in terms of $\bar{p}$ without requiring Bellman-like iterations. Instead, the present approach is more general, but does require Bellman-like iterations. Likewise, our formalism is superficially similar to PoWER (Kober and Peters, 2009) and SAEM (Vlassis and Toussaint, 2009), which rely on EM algorithm to maximize a lower bound for the expected return in an iterative fashion. This lower-bound also can be written as a KL-divergence between two distributions. Also, the natural policy gradient method can be seen as a relative entropy method, in which the second-order Taylor expansion of the relative-entropy between the distribution of the states is considered as the metric for policy improvement (Bagnell and Schneider, 2003). Another relevant study is *relative entropy policy search* (REPS) (Daniel et al., 2012; Peters et al., 2010) which relies on the idea of minimizing the relative entropy to control the size of policy update. However there are some differences between REPS and DPP. **(i)** In REPS the inverse temperature η needs to be optimized while DPP converges to the optimal solution for any inverse temperature η, and **(ii)** unlike DPP, no convergence analysis is presented REPS.

7. Discussion and Future Work

We have presented a new approach, dynamic policy programming (DPP), to compute the optimal policy in infinite-horizon discounted-reward MDPs. We have theoretically proven the convergence of DPP to the optimal policy for the tabular case. We have also provided performance-loss bounds for DPP in the presence of approximation. The bounds have been expressed in terms of supremum

4. Note that Even-Dar and Mansour (2003) make use of a slightly different performance measure than the one we use in this paper: The optimized result of Even-Dar and Mansour (2003), which is of order $O(1/(1-\gamma)^5)$, is a bound on the sample complexity of estimating Q^* with ε precision, whereas in this paper we consider the sample complexity of finding an ε-optimal policy. However, the latter can be easily derived for QL from the inequality $\|Q^* - Q^{\pi_k}\| \leq 1/(1-\gamma)\|Q^* - Q_k\|$, where π_k is the greedy policy w.r.t. Q_k and Q_k is the estimate of action-value function at iteration k. This inequality combined with the result of Even-Dar and Mansour (2003) implies a sample complexity bound of order $O(1/(1-\gamma)^7)$ for QL.

norm of average accumulated error as opposed to the standard bounds which are expressed in terms of supremum norm of the errors. We have then introduced a new incremental RL algorithm, called DPP-RL, which relies on a sample estimate instance of the DPP update rule to estimate the optimal policy. We have proven that DPP-RL converges to the optimal policy with the rate of $1/\sqrt{k}$.

We have also compared numerically the finite-time behavior of DPP-RL with similar RL methods. Experimental results have shown a better performance of DPP-RL when compared to QL and VI in terms of convergence rate. In these problems, for equal number of samples, VI converged to a better solution than DPP-RL, at the cost of many more steps. When compared to VI, DPP-RL does not need to store the model dynamics, resulting in significantly less memory requirements for large-scale MDPs. This statement is general and holds when comparing DPP-RL to any model-based method.

We have proposed SADPP as a variant of DPP which makes use of linear function approximation and regularization. SADPP has been shown to perform better than two other regularized methods, RFQI and REG-LSPI. We think that this is mainly due to the reduction of the effect of simulation noise (Section 4). At the same time, we admit that the existence of an additional parameter η favors SADPP since SADPP performs best for a finite-value of η. Therefore, it is interesting to consider soft-max variants of RFQI and LSPI which also make use of the inverse temperature η. In these cases, η should be initialized at a finite value and would gradually grow to $+\infty$.

The empirical comparison with those methods that do not make use of generative model assumption is outside of the scope of the current work and is left for future research. These methods include, for instance, PGAC methods that use sequences of samples to learn the value function of the current policy (Peters and Schaal, 2008; Konda and Tsitsiklis, 2003; Sutton et al., 2000), or upper-confidence bounds methods which address the exploration-exploitation dilemma (Jaksch et al., 2010; Szita and Szepesvári, 2010; Bartlett and Tewari, 2009; Strehl et al., 2009).

Another interesting line of future research is to devise finite-sample PAC bounds for SADPP in the spirit of previous theoretical results available for fitted value iteration and fitted Q-iteration (Munos and Szepesvári, 2008; Antos et al., 2008; Munos, 2005). This would require extending the error propagation result of Theorem 5 to an ℓ_2-norm analysis and combining it with the standard regression bounds.

Finally, an important extension of our results would be to apply DPP to large-scale action problems. This would require an efficient way to approximate $\mathcal{M}_\eta \Psi_k(x)$ in the update rule of Equation 5, since computing the exact summations becomes expensive. One idea is to sample estimate $\mathcal{M}_\eta \Psi_k(x)$ using Monte-Carlo simulation (MacKay, 2003, Chapter 29), since $\mathcal{M}_\eta \Psi_k(x)$ is the expected value of $\Psi_k(x,a)$ under the soft-max policy π_k.

Acknowledgments

We acknowledge A. Farahmand, R. Munos, N. Vlassis, M. Ghavamzadeh, J. Peters, W. Wiegerinck and N. de Freitas for the useful comments and discussions. We also thank the action editor Ronald Parr and the anonymous reviewers.

Appendix A. From Bellman Equation to DPP Recursion

In this appendix, we give an informal *derivation* of the DPP equation. This is only for helping the reader to understand the origin of the DPP equation and it is in no way meant as a justification of DPP. The theoretical analysis and the proof of convergence of DPP is provided in Section 3.2.

Let $\bar{\pi}$ be a stochastic policy, that is, $\bar{\pi}(a|x) > 0$ for all $(x,a) \in \mathcal{Z}$. Consider the relative entropy between the policy π and some baseline policy $\bar{\pi}$:

$$g^{\pi}_{\bar{\pi}}(x) \triangleq \mathrm{KL}\left(\pi(\cdot|x)\|\bar{\pi}(\cdot|x)\right) = \sum_{a\in\mathcal{A}} \pi(a|x) \log\left[\frac{\pi(a|x)}{\bar{\pi}(a|x)}\right], \qquad \forall x \in \mathcal{X}.$$

Note that $g^{\pi}_{\bar{\pi}}(x)$ is a positive function of x which is also bounded from above due to the assumption that $\bar{\pi}$ is a stochastic policy. We define a new value function $V^{\pi}_{\bar{\pi}}$, for all $x \in \mathcal{X}$, which incorporates g as a penalty term for deviating from the base policy $\bar{\pi}$ and the reward under the policy π:

$$V^{\pi}_{\bar{\pi}}(x) \triangleq \lim_{n\to\infty} \mathbb{E}\left[\sum_{k=0}^{n} \gamma^k \left(r_{t+k} - \frac{1}{\eta} g^{\pi}_{\bar{\pi}}(x_{t+k})\right) \middle| x_t = x\right],$$

where η is a positive constant and r_{t+k} is the reward at time $t+k$. Also, the expected value is taken w.r.t. the state transition probability distribution P and the policy π. The optimal value function $V^{*}_{\bar{\pi}}(x) \triangleq \sup_{\pi} V^{\pi}_{\bar{\pi}}(x)$ then exists and is bounded by some finite constant $c > 0$. Also, the value function $V^{*}_{\bar{\pi}}(x)$ satisfies the following Bellman equation for all $x \in \mathcal{X}$:

$$V^{*}_{\bar{\pi}}(x) = \sup_{\pi(\cdot|x)} \sum_{a\in\mathcal{A}} \pi(a|x) \left[r(x,a) - \frac{1}{\eta} \log \frac{\pi(a|x)}{\bar{\pi}(a|x)} + \gamma (P V^{*}_{\bar{\pi}})(x,a)\right]. \tag{13}$$

Equation 13 is a modified version of Equation 2 where, in addition to maximizing the expected reward, the optimal policy $\bar{\pi}^*$ also minimizes the distance with the baseline policy $\bar{\pi}$. The maximization in Equation 13 can be performed in closed form. Following Todorov (2007), we state Proposition 1 (closely related results to Proposition 1 can be found in the recent works of Still and Precup, 2012; Peters et al., 2010):

Proposition 1 *Let η be a positive constant, then for all $x \in \mathcal{X}$ the optimal value function $V^{*}_{\bar{\pi}}(x)$ and for all $(x,a) \in \mathcal{Z}$ the optimal policy $\bar{\pi}^*(a|x)$, respectively, satisfy:*

$$V^{*}_{\bar{\pi}}(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta\left(r(x,a) + \gamma (P V^{*}_{\bar{\pi}})(x,a)\right)\right],$$
$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x) \exp\left[\eta\left(r(x,a) + \gamma (P V^{*}_{\bar{\pi}})(x,a)\right)\right]}{\exp\left(\eta V^{*}_{\bar{\pi}}(x)\right)}. \tag{14}$$

Proof We must optimize π subject to the constraints $\sum_{a\in\mathcal{A}} \pi(a|x) = 1$ and $0 < \pi(a|x) < 1$. We define the Lagrangian function $\mathcal{L}(x;\lambda_x) : \mathcal{X} \to \Re$ by adding the term $\lambda_x\left[\sum_{a\in\mathcal{A}} \pi(a|x) - 1\right]$ to the RHS of Equation 13. Because $\bar{\pi}$ is strictly positive, minimizing $\mathcal{L}$ ensures that the solution is positive and the constraints $0 < \pi(a|x) \leq 1$ are automatically satisfied. Note that the KL-divergence is well-defined when both $\bar{\pi}$ and π are positive.

$$\mathcal{L}(x;\lambda_x) = \sum_{a\in\mathcal{A}} \pi(a|x)\left[r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right] - \frac{1}{\eta}\mathrm{KL}\left(\pi(\cdot|x)\|\bar{\pi}(\cdot|x)\right) - \lambda_x\left[\sum_{a\in\mathcal{A}}\pi(a|x) - 1\right].$$

The maximization in Equation 13 can be expressed as maximizing the Lagrangian function $\mathcal{L}(x,\lambda_x)$. The necessary condition for the extremum with respect to $\pi(\cdot|x)$ is:

$$0 = \frac{\partial\mathcal{L}(x,\lambda_x)}{\partial\pi(a|x)} = r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a) - \frac{1}{\eta} - \frac{1}{\eta}\log\left(\frac{\pi(a|x)}{\bar{\pi}(a|x)}\right) - \lambda_x,$$

which leads to

$$\bar{\pi}^*(a|x) = \bar{\pi}(a|x)\exp\left(-\eta\lambda_x - 1\right)\exp\left[\eta\left(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right)\right], \qquad \forall x\in\mathcal{X}. \tag{15}$$

The Lagrange multipliers can then be solved from the constraints:

$$1 = \sum_{a\in\mathcal{A}}\bar{\pi}^*(a|x) = \exp\left(-\eta\lambda_x - 1\right)\sum_{a\in\mathcal{A}}\bar{\pi}(a|x)\exp\left[\eta\left(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right)\right],$$

$$\lambda_x = \frac{1}{\eta}\log\sum_{a\in\mathcal{A}}\bar{\pi}(a|x)\exp\left[\eta\left(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right)\right] - \frac{1}{\eta}. \tag{16}$$

By plugging Equation 16 into Equation 15 we deduce

$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x)\exp\left[\eta\left(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right)\right]}{\sum\limits_{a\in\mathcal{A}}\bar{\pi}(a|x)\exp\left[\eta\left(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a)\right)\right]}, \qquad \forall(x,a)\in\mathcal{Z}. \tag{17}$$

The results then follows by substituting Equation 17 in Equation 13. ■

The optimal policy $\bar{\pi}^*$ is a function of the base policy, the optimal value function $V^*_{\bar{\pi}}$ and the state transition probability P. One can first obtain the optimal value function $V^*_{\bar{\pi}}$ through the following fixed-point iteration:

$$V^{k+1}_{\bar{\pi}}(x) = \frac{1}{\eta}\log\sum_{a\in\mathcal{A}}\bar{\pi}(a|x)\exp\left[\eta\left(r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a)\right)\right], \tag{18}$$

and then compute $\bar{\pi}^*$ using Equation 14. $\bar{\pi}^*$ maximizes the value function $V^{\pi}_{\bar{\pi}}$. However, we are not, in principle, interested in quantifying $\bar{\pi}^*$, but in solving the original MDP problem and computing π^*. The idea to further improve the policy towards π^* is to replace the base-line policy with the just newly computed policy of Equation 14. The new policy can be regarded as a *new base-line policy*, and the process can be repeated again. This leads to a double-loop algorithm to find the optimal policy π^*, where the outer-loop and the inner-loop would consist of a policy update, Equation 14, and a value function update, Equation 18, respectively.

We then follow the following steps to derive the final DPP algorithm: **(i)** We introduce some extra smoothness to the policy update rule by replacing the double-loop algorithm by direct optimization of both value function and policy simultaneously using the following fixed point iterations:

$$V_{\bar{\pi}}^{k+1}(x) = \frac{1}{\eta} \log \sum_{a \in \mathcal{A}} \bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a))\big], \tag{19}$$

$$\bar{\pi}_{k+1}(a|x) = \frac{\bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a))\big]}{\exp\big(\eta V_{\bar{\pi}}^{k+1}(x)\big)}. \tag{20}$$

Further, **(ii)** we define the action preference function Ψ_k (Sutton and Barto, 1998, Chapter 6.6), for all $(x,a) \in \mathcal{Z}$ and $k \geq 0$, as follows:

$$\Psi_{k+1}(x,a) \triangleq \frac{1}{\eta} \log \bar{\pi}_k(a|x) + r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a). \tag{21}$$

By comparing Equation 21 with Equation 20 and Equation 19, we deduce

$$\bar{\pi}_k(a|x) = \frac{\exp(\eta \Psi_k(x,a))}{\sum_{a' \in \mathcal{A}} \exp(\eta \Psi_k(x,a'))}, \tag{22}$$

$$V_{\bar{\pi}}^k(x) = \frac{1}{\eta} \log \sum_{a \in \mathcal{A}} \exp(\eta \Psi_k(x,a))). \tag{23}$$

Finally, **(iii)** by plugging Equation 22 and Equation 23 into Equation 21 we derive

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) - \mathcal{L}_\eta \Psi_k(x) + r(x,a) + \gamma(P\mathcal{L}_\eta \Psi_k)(x,a), \tag{24}$$

with $\mathcal{L}_\eta$ operator being defined by $\mathcal{L}_\eta \Psi(x) \triangleq 1/\eta \log \sum_{a \in \mathcal{A}} \exp(\eta \Psi(x,a))$. Equation 24 is one form of the DPP equations. There is an analytically more tractable version of the DPP equation, where we replace $\mathcal{L}_\eta$ by the Boltzmann soft-max $\mathcal{M}_\eta$ defined by $\mathcal{M}_\eta \Psi(x) \triangleq \sum_{a \in \mathcal{A}} \big[\exp(\eta \Psi(x,a)) \Psi(x,a) / \sum_{a' \in \mathcal{A}} \exp(\eta \Psi(x,a'))\big]$.[5] In principle, we can provide formal analysis for both versions. However, the proof is somewhat simpler for the $\mathcal{M}_\eta$ case, which we make use of it in the rest of this paper. By replacing $\mathcal{L}_\eta$ with $\mathcal{M}_\eta$ we deduce the DPP recursion:

$$\begin{aligned} \Psi_{k+1}(x,a) = \mathcal{O}\Psi_k(x,a) &\triangleq \Psi_k(x,a) + r(x,a) + \gamma P \mathcal{M}_\eta \Psi_k(x,a) - \mathcal{M}_\eta \Psi_k(x) \\ &= \Psi_k(x,a) + \mathcal{T}^{\pi_k} \Psi_k(x,a) - \pi_k \Psi_k(x) \end{aligned}, \qquad \forall (x,a) \in \mathcal{Z},$$

where $\mathcal{O}$ is an operator defined on the action preferences Ψ_k and π_k is the soft-max policy associated with Ψ_k:

$$\pi_k(a|x) \triangleq \frac{\exp(\eta \Psi_k(x,a))}{\sum_{a' \in \mathcal{A}} \exp(\eta \Psi_k(x,a'))}.$$

5. Replacing $\mathcal{L}_\eta$ with $\mathcal{M}_\eta$ is motivated by the following relation between these two operators:

$$|\mathcal{L}_\eta \Psi(x) - \mathcal{M}_\eta \Psi(x)| = 1/\eta H_\pi(x) \leq \frac{\log(|\mathcal{A}|)}{\eta}, \qquad \forall x \in \mathcal{X}, \tag{25}$$

with $H_\pi(x)$ is the entropy of the policy distribution π obtained by plugging Ψ into Equation A. In words, $\mathcal{M}_\eta \Psi(x)$ is close to $\mathcal{L}_\eta \Psi(x)$ up to the constant $\log(|\mathcal{A}|)/\eta$. Also, both $\mathcal{L}_\eta \Psi(x)$ and $\mathcal{M}_\eta \Psi(x)$ converge to $\mathcal{M}\Psi(x)$ when η goes to $+\infty$. For the proof of Equation 25 and further readings see MacKay (2003, Chapter 31).

Appendix B. The Proof of Convergence of DPP—Theorem 2 and Theorem 4

In this section, we provide a formal analysis of the convergence behavior of DPP.

B.1 Proof of Theorem 2

In this subsection we establish a rate of convergence for the value function of the policy induced by DPP. The main result is in the form of following finite-iteration performance-loss bound, for all $k \geq 0$:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma\left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta}\right)}{(1-\gamma)^2(k+1)}. \tag{26}$$

Here, Q^{π_k} is the action-values under the policy π_k and π_k is the policy induced by DPP at step k.

To derive Equation 26 one needs to relate Q^{π_k} to the optimal Q^*. Unfortunately, finding a direct relation between Q^{π_k} and Q^* is not an easy task. Instead, we relate Q^{π_k} to Q^* via an auxiliary action-value function Q_k, which we define below. In the remainder of this Section we take the following steps: **(i)** we express Ψ_k in terms of Q_k in Lemma 13. **(ii)** we obtain an upper bound on the normed error $\|Q^* - Q_k\|$ in Lemma 14. Finally, **(iii)** we use these two results to derive a bound on the normed error $\|Q^* - Q^{\pi_k}\|$. For the sake of readability, we skip the formal proofs of the lemmas in this section since we prove a more general case in Section C.

Now let us define the auxiliary action-value function Q_k. The sequence of auxiliary action-value functions $\{Q_0, Q_1, Q_2, \dots\}$ is obtained by iterating the initial $Q_0 = \Psi_0$ from the following recursion:

$$Q_k = \frac{k-1}{k}\mathcal{T}^{\pi_{k-1}}Q_{k-1} + \frac{1}{k}\mathcal{T}^{\pi_{k-1}}Q_0, \tag{27}$$

where π_k is the policy induced by the k^{th} iterate of DPP.

Lemma 13 relates Ψ_k with Q_k:[6]

Lemma 13 *Let k be a positive integer. Then, we have*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0). \tag{28}$$

The following lemma relates Q_k and Q^*:

Lemma 14 *Let Assumption 1 hold and k be a positive integer. Also assume that* $\|\Psi_0\| \leq V_{\max}$. *Then the following inequality holds:*

$$\|Q^* - Q_k\| \leq \frac{\gamma\left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta}\right)}{(1-\gamma)k}.$$

Lemma 14 provides an upper bound on the normed-error $\left\|Q_k - Q^*\right\|$. We make use of Lemma 14 to prove the main result of this Subsection:

6. We make use of this lemma in the proof of convergence of DPP in Section B.2.

$$\begin{aligned}\|Q^* - Q^{\pi_k}\| &= \|Q^* - Q_{k+1} + Q_{k+1} - \mathcal{T}^{\pi_k} Q^* + \mathcal{T}^{\pi_k} Q^* - Q^{\pi_k}\| \\ &\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q^{\pi_k}\| \\ &\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \gamma \|Q^* - Q^{\pi_k}\|.\end{aligned}$$

By collecting terms we obtain

$$\begin{aligned}\|Q^* - Q^{\pi_k}\| &\leq \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\|\right) \\ &= \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \left\| \frac{k}{k+1} \mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1} \mathcal{T}^{\pi_k} Q_0 - \mathcal{T}^{\pi_k} Q^* \right\| \right) \\ &\leq \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \frac{k}{k+1} \|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_k\| + \frac{1}{k+1} \|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_0\| \right) \\ &\leq \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{\gamma}{k+1} \|Q^* - Q_0\| \right) \\ &\leq \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{2\gamma V_{\max}}{k+1} \right) \\ &\leq \frac{1}{1-\gamma} \left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{\gamma(4V_{\max} + \log(|\mathcal{A}|)/\eta)}{k+1} \right).\end{aligned}$$

This combined with Lemma 14 completes the proof.

B.2 Proof of Corollary 4

First, we note that Q_k converges to Q^* (Lemma 14) and π_k converges to π^* by Corollary 3. Therefore, there exists a limit for Ψ_k since Ψ_k is expressed in terms of Q_k, Q_0 and π_{k-1} (Lemma 13).

Now, we compute the limit of Ψ_k. Q_k converges to Q^* with a linear rate from Lemma 14. Also, we have $V^* = \pi^* Q^*$ by definition of V^* and Q^*. Then, by taking the limit of Equation 28 we deduce

$$\begin{aligned}\lim_{k\to\infty} \Psi_k(x,a) &= \lim_{k\to\infty} \left[k Q^*(x,a) + Q_0(x,a) - (k-1) V^*(x) - (\pi^* Q_0)(x)\right] \\ &= \lim_{k\to\infty} k(Q^*(x,a) - V^*(x)) \\ &\quad + Q_0(x,a) - (\pi^* Q_0)(x) + V^*(x).\end{aligned}$$

We then deduce, for all $(x,a) \in \mathcal{Z}$,

$$\lim_{k\to\infty} \Psi_k(x,a) = \begin{cases} Q_0(x,a) - (\pi^* Q_0)(x) + V^*(x) & a = a^*(x) \\ -\infty & a \neq a^*(x) \end{cases},$$

where $a^*(x) = \max_{a\in\mathcal{A}}(Q^*(x,a))$. This combined with the assumption that the optimal policy is unique completes the proof.

Appendix C. Proof of Theorem 5

This section provides a formal theoretical analysis of the performance of dynamic policy programming in the presence of approximation.

Consider a sequence of the action preferences $\{\Psi_0, \Psi_1, \Psi_2, \dots\}$ as the iterates of Equation 7. Our goal is to establish an ℓ_∞-norm performance loss bound of the policy induced by approximate DPP. The main result is that at iteration $k \geq 0$ of approximate DPP, we have

$$\|Q^* - Q^{\pi_k}\| \leq \frac{1}{(1-\gamma)(k+1)} \left[\frac{2\gamma \left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right)}{(1-\gamma)} + \sum_{j=1}^{k+1} \gamma^{k-j+1} \|E_{j-1}\| \right], \tag{29}$$

where $E_k = \sum_{j=0}^{k} \varepsilon_k$ is the cumulative approximation error up to step k. Here, Q^{π_k} denotes the action-value function of the policy π_k and π_k is the soft-max policy associated with Ψ_k.

As in the proof of Theorem 2, we relate Q^* with Q^{π_k} via an auxiliary action-value function Q_k. In the rest of this section, we first express Ψ_k in terms of Q_k in Lemma 15. Then, we obtain an upper bound on the normed error $\|Q^* - Q_k\|$ in Lemma 19. Finally, we use these two results to derive Equation 29.

Now, let us define the auxiliary action-value function Q_k. The sequence of auxiliary action-value functions $\{Q_0, Q_1, Q_2, \dots\}$ is resulted by iterating the initial action-value function $Q_0 = \Psi_0$ from the following recursion:

$$Q_k = \frac{k-1}{k} \mathcal{T}^{\pi_{k-1}} Q_{k-1} + \frac{1}{k} (\mathcal{T}^{\pi_{k-1}} Q_0 + E_{k-1}), \tag{30}$$

where Equation 30 may be considered as an approximate version of Equation 27. Lemma 15 relates Ψ_k with Q_k:

Lemma 15 *Let k be a positive integer and π_k denotes the policy induced by the approximate DPP at iteration k. Then we have*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1} \left((k-1)Q_{k-1} + Q_0 \right). \tag{31}$$

Proof We rely on induction for the proof of this theorem. The result holds for $k = 1$ since one can easily show that Equation 31 reduces to Equation 7. We then show that if Equation 31 holds for k then it also holds for $k+1$. From Equation 7 we have

$$\begin{aligned}
\Psi_{k+1} &= \Psi_k + \mathcal{T}^{\pi_k} \Psi_k - \pi_k \Psi_k + \varepsilon_k \\
&= kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0) + \mathcal{T}^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) \\
&\quad - \pi_k(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) + \varepsilon_k \\
&= kQ_k + Q_0 + \mathcal{T}^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) - \pi_k(kQ_k + Q_0) \\
&\quad + E_k - E_{k-1} \\
&= kQ_k + Q_0 + r + \gamma P^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) - \pi_k(kQ_k + Q_0) \\
&\quad + E_k - E_{k-1},
\end{aligned}$$

in which we rely on Equation 3. We then deduce

$$\begin{aligned}\Psi_{k+1} &= kQ_k + Q_0 + k(r+\gamma P^{\pi_k}Q_k) + r + \gamma P^{\pi_k}Q_0 - (k-1)(r+\gamma P^{\pi_{k-1}}Q_{k-1}) \\ &\quad - (r+\gamma P^{\pi_{k-1}}Q_0) + \pi_{k-1}((k-1)Q_{k-1}+Q_0)) - \pi_k(kQ_k+Q_0) + E_k - E_{k-1} \\ &= kQ_k - (k-1)\mathcal{T}^{\pi_{k-1}}Q_{k-1} - \mathcal{T}^{\pi_{k-1}}Q_0 - E_{k-1} + k\mathcal{T}^{\pi_k}Q_k + \mathcal{T}^{\pi_k}Q_0 + E_k \\ &\quad + Q_0 - \pi_k(kQ_k+Q_0) \\ &= (k+1)Q_{k+1} + Q_0 - \pi_k(kQ_k+Q_0),\end{aligned}$$

where in the last line we make use of Equation 30.

Thus Equation 31 holds for $k+1$, and is thus true for all $k \geq 1$. ■

Based on Lemma 15, one can express the policy induced by DPP, π_k, in terms of Q_k and Q_0:

Lemma 16 *For all $(x,a) \in \mathcal{Z}$:*

$$\pi_k(a|x) = \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a)\right)\right)}{\sum_{b\in\mathcal{A}} \exp\left(\eta\left(kQ_k(x,b) + Q_0(x,b)\right)\right)}.$$

Proof

$$\begin{aligned}\pi_k(a|x) &= \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a) - \pi_{k-1}((k-1)Q_{k-1}+Q_0)(x)\right)\right)}{Z(x)} \\ &= \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a)\right)\right)}{Z'(x)},\end{aligned}$$

where $Z(x)$ and $Z'(x) = Z(x)\exp\left(\eta\pi_{k-1}((k-1)Q_{k-1}+Q_0)(x)\right)$ are the normalization factors. ■

In an analogy to Lemma 14 we establish a bound on $\|Q^* - Q_k\|$ for which we make use of the following technical results:

Lemma 17 *Let $\eta > 0$ and $\mathcal{Y}$ be a finite set with cardinality L. Also assume that $\mathcal{F}$ denotes the space of real-valued functions on $\mathcal{Y}$. Then the following inequality holds for all $f \in \mathcal{F}$:*

$$\max_{y\in\mathcal{Y}} f(y) - \sum_{y\in\mathcal{Y}} \frac{\exp(\eta f(y))f(y)}{\sum_{y'\in\mathcal{Y}}\exp(\eta f(y'))} \leq \frac{\log(L)}{\eta}.$$

Proof For any $f \in \mathcal{F}$ we have

$$\max_{y\in\mathcal{Y}} f(y) - \sum_{y\in\mathcal{Y}} \frac{\exp(\eta f(y))f(y)}{\sum_{y'\in\mathcal{Y}}\exp(\eta f(y'))} = \sum_{y\in\mathcal{Y}} \frac{\exp(-\eta g(y))g(y)}{\sum_{y'\in\mathcal{Y}}\exp(-\eta g(y'))},$$

with $g(y) = \max_{y\in\mathcal{Y}} f(y) - f(y)$. According to MacKay (2003, Chapter 31):

$$\sum_{y\in\mathcal{Y}} \frac{\exp(-\eta g(y))g(y)}{\sum_{y'\in\mathcal{Y}}\exp(-\eta g(y'))} = -\frac{1}{\eta}\log\sum_{y\in\mathcal{Y}}\exp(-\eta g(y)) + \frac{1}{\eta}H_p,$$

where H_p is the entropy of probability distribution p defined by

$$p(y) = \frac{\exp(-\eta g(y))}{\sum_{y' \in \mathcal{Y}} \exp(-\eta g(y'))}.$$

Define $\mathcal{Y}^f_{\max} \subset \mathcal{Y}$ as the set of all entries of $\mathcal{Y}$ which maximizes $f \in \mathcal{F}$. The following steps complete the proof:

$$\begin{aligned}
\sum_{y \in \mathcal{Y}} \frac{\exp(-\eta g(y)) g(y)}{\sum_{y' \in \mathcal{Y}} \exp(-\eta g(y'))} &= -\frac{1}{\eta} \log \sum_{y \in \mathcal{Y}} \exp(-\eta g(y)) + \frac{1}{\eta} H_p \\
&\leq -\frac{1}{\eta} \log \left[1 + \sum_{y \notin \mathcal{Y}^f_{\max}} \exp(-\eta g(y))) \right] + \frac{1}{\eta} H_p \\
&\leq \frac{1}{\eta} H_p \leq \frac{\log(L)}{\eta},
\end{aligned}$$

in which we make use of $-\frac{1}{\eta} \log \left[1 + \sum_{y \notin \mathcal{Y}^f_{\max}} \exp(-\eta g(y))) \right] \leq 0$. ■

Lemma 18 *Let $\eta > 0$ and k be a positive integer. Assume $\|Q_0\| \leq V_{\max}$. Then, the following holds:*

$$\|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k}Q_k - \mathcal{T}^{\pi_k}Q_0\| \leq \gamma \left(2V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right).$$

Proof We have, by definition of operator $\mathcal{T}$,

$$\begin{aligned}
\|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k}Q_k - \mathcal{T}^{\pi_k}Q_0\| &\leq \gamma \|kP\mathcal{M}Q_k + P\mathcal{M}Q_0 - kP^{\pi_k}Q_k - P^{\pi_k}Q_0\| \\
&= \gamma \|P(\mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k + Q_0))\| \\
&\leq \gamma \|\mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k + Q_0)\| \qquad (32) \\
&\leq \gamma \|2\mathcal{M}Q_0 + \mathcal{M}(kQ_k + Q_0) - \pi_k(kQ_k + Q_0)\| \\
&\leq \gamma (2\|Q_0\| + \|\mathcal{M}(kQ_k + Q_0) - \mathcal{M}_\eta(kQ_k + Q_0)\|),
\end{aligned}$$

where in the last line we make use of Lemma 16. The result then follows by comparing Equation 32 with Lemma 17. ■

Now, we prove a bound on $\|Q^* - Q_k\|$:

Lemma 19 *Let Assumption 1 hold. Define Q_k by Equation 30. Let k be a non-negative integer, also, assume that $\|\Psi_0\| \leq V_{\max}$, then the following inequality holds:*

$$\|Q^* - Q_k\| \leq \frac{\gamma \left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right)}{(1-\gamma)k} + \frac{1}{k} \sum_{j=1}^{k} \gamma^{k-j} \|E_{j-1}\|. \qquad (33)$$

Proof We rely on induction for the proof of this lemma. Obviously the result holds for $k = 0$. Then we need to show that if Equation 33 holds for k it also holds for $k+1$:

$$\begin{aligned}
\|Q^* - Q_{k+1}\| &= \left\| \mathcal{T}Q^* - \left(\frac{k}{k+1} \mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1} (\mathcal{T}^{\pi_k} Q_0 + E_k) \right) \right\| \\
&= \left\| \frac{1}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_0) + \frac{k}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_k) - \frac{1}{k+1} E_k \right\| \\
&= \frac{1}{k+1} \|\mathcal{T}Q^* - \mathcal{T}Q_0 + \mathcal{T}Q_0 - \mathcal{T}^{\pi_k} Q_0 + k(\mathcal{T}Q^* - \mathcal{T}Q_k + \mathcal{T}Q_k - \mathcal{T}^{\pi_k} Q_k)\| \\
&\quad + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} [\|\mathcal{T}Q^* - \mathcal{T}Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\|] \\
&\quad + \frac{k}{k+1} \|\mathcal{T}Q^* - \mathcal{T}Q_k\| + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} [\gamma \|Q^* - Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\|] \\
&\quad + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{1}{k+1} \|E_k\| .
\end{aligned} \tag{34}$$

Now based on Lemma 18 and by plugging Equation 33 into Equation 34 we have

$$\begin{aligned}
\|Q^* - Q_{k+1}\| &\leq \frac{\gamma}{k+1} \left[4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right] \\
&\quad + \frac{\gamma k}{k+1} \left[\frac{\gamma \left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right)}{k(1-\gamma)} + \frac{1}{k} \sum_{j=1}^{k} \gamma^{k-j} \|E_{j-1}\| \right] + \frac{1}{k+1} \|E_k\| \\
&= \frac{\gamma \left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta} \right)}{(1-\gamma)(k+1)} + \frac{1}{k+1} \sum_{j=1}^{k+1} \gamma^{k-j+1} \|E_{j-1}\|.
\end{aligned}$$

The result then follows, for all $k \geq 0$, by induction. ■

Lemma 19 provides an upper-bound on the normed-error $\left\|Q^* - Q_k\right\|$. We make use of this result to derive a bound on the performance loss $\|Q^* - Q^{\pi_k}\|$:

$$\begin{aligned}
\|Q^* - Q^{\pi_k}\| &= \|Q^* - Q_{k+1} + Q_{k+1} - \mathcal{T}^{\pi_k} Q^* + \mathcal{T}^{\pi_k} Q^* - Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \gamma \|Q^* - Q^{\pi_k}\| .
\end{aligned}$$

By collecting terms we obtain

$$
\begin{aligned}
\|Q^* - Q^{\pi_k}\| &\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\|\right) \\
&= \frac{1}{1-\gamma}\left[\|Q^* - Q_{k+1}\| + \left\|\frac{k}{k+1}\mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1}(\mathcal{T}^{\pi_k} Q_0 + E_k) - \mathcal{T}^{\pi_k} Q^*\right\|\right] \\
&\leq \frac{1}{1-\gamma}\left[\|Q^* - Q_{k+1}\| + \frac{k}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_k\| + \frac{1}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_0\|\right] \\
&\quad + \frac{1}{(1-\gamma)(k+1)}\|E_k\| \\
&\leq \frac{1}{1-\gamma}\left[\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{\gamma}{k+1}\|Q^* - Q_0\|\right] \\
&\leq \frac{1}{1-\gamma}\left[\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{2\gamma V_{\max}}{k+1}\right] \\
&\leq \frac{1}{1-\gamma}\left[\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{\gamma(4V_{\max} + \frac{\log|\mathcal{A}|}{\eta})}{k+1}\right]
\end{aligned}
$$

This combined with the result of Lemma 19 completes the proof.

Appendix D. The Proof of Convergence of DPP-RL—Theorem 9 and Theorem 10

We begin the analysis by introducing some new notation. Let us define $\mathcal{F}_k$ as the filtration generated by the sequence of all random variables $\{y_1, y_2, y_3, \ldots, y_k\}$ drawn from the distribution $P(\cdot|x,a)$ for all $(x,a) \in \mathcal{Z}$. We know, by the definition of ε_k, that $\mathbb{E}(\varepsilon_k(x,a)|\mathcal{F}_{k-1}) = 0$, which means that for all $(x,a) \in \mathcal{Z}$ the sequence of estimation errors $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k\}$ is a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$. Now, we provide the proof of Lemma 8, on which we rely for the analysis of both Theorem 9 and Theorem 10:

Proof of Lemma 8 We first prove that $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq \frac{2\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + V_{\max}$ by induction. Let us assume that the bound $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq \frac{2\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + V_{\max}$ holds. Thus

$$\begin{aligned}
\|\mathcal{T}_{k+1}^{\pi_k}\Psi_{k+1}\| &\leq \|r\| + \gamma\|P^{\pi_k}\Psi_{k+1}\| \leq \|r\| + \gamma\|\mathcal{M}_\eta\Psi_{k+1}\| \\
&= \|r\| + \gamma\left\|\mathcal{M}_\eta\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k\right)\right\| \\
&\leq \|r\| + \gamma\left\|\mathcal{M}_\eta\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k\right) - \mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k\right)\right\| \\
&\quad + \gamma\left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k\right)\right\| \\
&\leq \|r\| + \frac{\gamma\log(|\mathcal{A}|)}{\eta} + \gamma\left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k\right)\right\| \\
&= \|r\| + \frac{\gamma\log(|\mathcal{A}|)}{\eta} + \gamma\left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta\Psi_k + \mathcal{M}\Psi_k - \mathcal{M}\Psi_k\right)\right\| \\
&\leq \|r\| + \frac{\gamma\log(|\mathcal{A}|)}{\eta} + \gamma\|\mathcal{M}(\mathcal{M}\Psi_k - \mathcal{M}_\eta\Psi_k)\| + \gamma\|\mathcal{M}\left(\Psi_k - \mathcal{M}\Psi_k\right)\| \\
&\quad + \gamma\left\|\mathcal{M}\mathcal{T}_k^{\pi_k}\Psi_k\right\| \\
&\leq \|r\| + \frac{2\gamma\log(|\mathcal{A}|)}{\eta} + \gamma\left\|\mathcal{T}_k^{\pi_k}\Psi_k\right\| \leq \|r\| + \frac{2\gamma\log(|\mathcal{A}|)}{\eta} + \frac{2\gamma^2\log(|\mathcal{A}|)}{\eta(1-\gamma)} + \gamma V_{\max} \\
&\leq \frac{2\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + R_{\max} + \gamma V_{\max} = \frac{2\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)} + V_{\max},
\end{aligned}$$

where we make use of Lemma 17 to bound the difference between the max operator $\mathcal{M}(\cdot)$ and the soft-max operator $\mathcal{M}_\eta(\cdot)$. Now, by induction, we deduce that for all $k \geq 0$, $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq 2\gamma\log(|\mathcal{A}|)/(\eta(1-\gamma)) + V_{\max}$. The bound on ε_k is an immediate consequence of this result. ■

D.1 Proof of Theorem 9

In this subsection, we provide the proof of Theorem 9 which guarantees that DPP-RL asymptotically converges to the optimal policy w.p. 1.

We make use of the result of Lemma 8 and Corollary 6 to prove the theorem. We begin by recalling the result of Corollary 6:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \lim_{k\to\infty} \frac{1}{k+1} \|E_k\| .$$

Therefore, to prove the convergence of DPP-RL, one only needs to prove that $1/(k+1)\,\|E_k\|$ asymptotically converges to 0 w.p. 1. For this we rely on the strong law of large numbers for martingale differences (Hoffmann-Jørgensen and Pisier, 1976), which states that the average of a sequence of martingale differences asymptotically converges, almost surely, to 0 if the second moments of all entries of the sequence are bounded by some $0 \leq U \leq \infty$. This is the case for the sequence of martingales $\{\varepsilon_1, \varepsilon_2, \dots\}$ since we already have proven the boundedness of $\|\varepsilon_k\|$ in Lemma 8. Thus, we deduce

$$\lim_{k\to\infty} \frac{1}{k+1} |E_k(x,a)| = 0, \qquad \text{w.p. 1.}$$

Thus

$$\lim_{k\to\infty} \frac{1}{k+1} \|E_k\| = 0, \qquad \text{w.p. } 1. \tag{35}$$

The result then follows by combining Equation 35 with Corollary 6.

D.2 Proof of Theorem 10

In this subsection, we prove Theorem 10, for which we rely on a maximal Azuma's inequality (see, e.g., Cesa-Bianchi and Lugosi, 2006, Appendix, pg. 359):

Lemma 20 (Azuma, 1967) *Let $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_K\}$ be a martingale difference sequence w.r.t. a sequence of random variables $\{X_1, X_2, \ldots, X_K\}$, that is, $\mathbb{E}(Y_{k+1}|X_1, \ldots X_k) = 0$ for all $0 < k \leq K$. Also, let $\mathcal{Y}$ be uniformly bounded by $U > 0$. Define $S_k = \sum_{i=1}^{k} Y_i$. Then, for any $\varepsilon > 0$, we have*

$$\Pr\left(\max_{1\leq k\leq K} S_k > \varepsilon\right) \leq \exp\left(\frac{-\varepsilon^2}{2KU^2}\right).$$

We recall the result of Theorem 5 at iteration k:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{\gamma\left(4V_{\max} + \frac{\log(|\mathcal{A}|)}{\eta}\right)}{(1-\gamma)^2(k+1)} + \frac{1}{(1-\gamma)(k+1)} \sum_{j=0}^{k} \gamma^{k-j} \|E_j\|.$$

Note that the main difference between this bound and the result of Theorem 10 is just in the second term. So, to prove Theorem 10 we need to show that the following inequality holds, with probability at least $1-\delta$:

$$\frac{1}{k+1} \sum_{j=0}^{k} \gamma^{k-j} \left\|E_j\right\| \leq \frac{4(\gamma \log(|\mathcal{A}|)/\eta + 2R_{\max})}{(1-\gamma)^2} \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}. \tag{36}$$

We first notice that

$$\frac{1}{k+1} \sum_{j=0}^{k} \gamma^{k-j} \left\|E_j\right\| \leq \frac{1}{k+1} \sum_{k=0}^{j} \gamma^{k-j} \max_{0\leq j\leq k} \|E_j\| \leq \frac{\max_{0\leq j\leq k} \|E_j\|}{(1-\gamma)(k+1)}. \tag{37}$$

Therefore, in order to prove Equation 36 it is sufficient to bound $\max_{0\leq j\leq k} \|E_j\| = \max_{(x,a)\in\mathcal{Z}} \max_{0\leq j\leq k} |E_{k-1}(x,a)|$ in high probability.

We begin by proving high probability bound on $\max_{0\leq j\leq k} |E_j(x,a)|$ for a given (x,a). We first notice that

$$\begin{aligned}
\Pr\left(\max_{0\leq j\leq k} |E_j(x,a)| > \varepsilon\right) &= \Pr\left(\max\left[\max_{0\leq j\leq k}(E_j(x,a)), \max_{0\leq j\leq k}(-E_j(x,a))\right] > \varepsilon\right) \\
&= \Pr\left(\left\{\max_{0\leq j\leq k}(E_j(x,a)) > \varepsilon\right\} \bigcup \left\{\max_{0\leq j\leq k}(-E_j(x,a)) > \varepsilon\right\}\right) \\
&\leq \Pr\left(\max_{0\leq j\leq k}(E_j(x,a)) > \varepsilon\right) + \Pr\left(\max_{0\leq j\leq k}(-E_j(x,a)) > \varepsilon\right),
\end{aligned} \tag{38}$$

The sequence of random variables $\{\varepsilon_0(x,a),\varepsilon_1(x,a),\ldots,\varepsilon_k(x,a)\}$ is a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$ (generated by the random samples $\{y_0,y_1,\ldots,y_k\}(x,a)$ for all (x,a)), that is, $\mathbb{E}[\varepsilon_k(x,a)|\mathcal{F}_{k-1}]=0$. It follows from Lemma 20 and Lemma 8 that for any $\varepsilon>0$ we have

$$\begin{aligned}\Pr\left(\max_{0\le j\le k}(E_j(x,a))>\varepsilon\right)&\le\exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right)\\ \Pr\left(\max_{0\le j\le k}(-E_j(x,a))>\varepsilon\right)&\le\exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right).\end{aligned}\tag{39}$$

By combining Equation 39 with Equation 38 we deduce that

$$\Pr\left(\max_{0\le j\le k}|E_j(x,a)|>\varepsilon\right)\le 2\exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right),$$

and a union bound over the state-action space leads to

$$\Pr\left(\max_{0\le j\le k}\left\|E_j\right\|>\varepsilon\right)\le 2|\mathcal{X}||\mathcal{A}|\exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right).$$

For any $0<\delta<1$, this bound can be re-expressed as

$$\Pr\left(\max_{0\le j\le k}\left\|E_j\right\|\le\left(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max}\right)\sqrt{2(k+1)\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}\right)\ge 1-\delta.$$

This combined with Equation 37 proves Equation 36 and Theorem 10.

References

A. Antos, R. Munos, and C. Szepesvári. Fitted Q-iteration in continuous action-space MDPs. In *Advances in Neural Information Processing Systems 20*, pages 9–16. MIT Press, 2008.

M. Gheshlaghi Azar, R. Munos, M. Ghavamzadeh, and H. J. Kappen. Speedy Q-learning. In *Advances in Neural Information Processing Systems 24*, pages 2411–2419. MIT Press, 2012.

J. A. Bagnell and J. G. Schneider. Covariant policy search. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1019–1024. Morgan Kaufmann, 2003.

P. L. Bartlett and A. Tewari. REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 35–42. AUAI Press, 2009.

A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC–13(5): 834–846, 1983.

J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15(1):319–350, 2001.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, third edition, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

C. Daniel, G. Neumann, and J. Peters. Hierarchical relative entropy policy search. *Journal of Machine Learning Research - Proceedings Track*, 22:273–281, 2012.

D. P. de Farias and B. Van Roy. On the existence of fixed points for approximate value iteration and temporal-difference learning. *Journal of Optimization Theory and Applications*, 105(3):589–608, 2000.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

E. Even-Dar and Y. Mansour. Learning rates for Q-learning. *Journal of Machine Learning Research*, 5(Dec):1–25, 2003.

A. Farahmand, M. Ghavamzadeh, Cs. Szepesvári, and S. Mannor. Regularized fitted Q-iteration: Application to planning. In *European Workshop on Reinforcement Learning*, Lecture Notes in Computer Science, pages 55–68. Springer, 2008.

A. Farahmand, M. Ghavamzadeh, Cs. Szepesvári, and S. Mannor. Regularized policy iteration. In *Advances in Neural Information Processing Systems 21*, pages 441–448. Curran Associates, Inc., 2009.

A. Farahmand, R. Munos, and Cs. Szepesvári. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems 23*, pages 568–576. MIT Press, 2010.

J. Hoffmann-Jørgensen and G. Pisier. The law of large numbers and the central limit theorem in banach spaces. *The Annals of Probability*, 4(4):587–599, 1976.

T. Jaakkola, M. I. Jordan, and S. Singh. On the convergence of stochastic iterative dynamic programming. *Neural Computation*, 6(6):1185–1201, 1994.

T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

S. Kakade. Natural policy gradient. In *Advances in Neural Information Processing Systems 14*, pages 1531–1538. MIT Press, 2002.

H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Statistical Mechanics*, 2005(11):P11011, 2005.

M. Kearns and S. Singh. Finite-sample convergence rates for Q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems 12*, pages 996–1002. MIT Press, 1999.

J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Advances in Neural Information Processing Systems 21*, pages 849–856. MIT Press, 2009.

S. Koenig and R. G. Simmons. Complexity analysis of real-time reinforcement learning. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 99–105. AAAI Press, 1993.

V. Konda and J. N. Tsitsiklis. On actor-critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2003.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4(Dec):1107–1149, 2003.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

H. Maei, Cs. Szepesvári, S. Bhatnagar, and R. S. Sutton. Toward off-policy learning control with function approximation. In *Proceedings of the 27th International Conference on Machine Learning*, pages 719–726. Omnipress, 2010.

F. Melo, S. Meyn, and I. Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th International Conference on Machine Learning*, pages 664–671. ACM Press, 2008.

R. Munos. Error bounds for approximate value iteration. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, pages 1006–1011. AAAI Press, 2005.

R. Munos and Cs. Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

T. J. Perkins and D. Precup. A convergent form of approximate policy iteration. In *Advances in Neural Information Processing Systems 15*, pages 1595–1602. MIT Press, 2003.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7–9):1180–1190, 2008.

J. Peters, K. Mülling, and Y. Altun. Relative entropy policy search. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1607–1612. AAAI Press, 2010.

S. Singh, T. Jaakkola, M.L. Littman, and Cs. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

S. Still and D. Precup. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3):139–148, 2012.

A. L. Strehl, L. Li, and M. L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press, 2000.

Cs. Szepesvári. The asymptotic convergence-rate of Q-learning. In *Advances in Neural Information Processing Systems 10*, pages 1064–1070. MIT Press, 1998.

Cs. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

Cs. Szepesvári and W. Smart. Interpolation-based Q-learning. In *Proceedings of 21st International Conference on Machine Learning*, pages 791–798. ACM Press, 2004.

I. Szita and Cs. Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1031–1038. Omnipress, 2010.

C. Thiery and B. Scherrer. Least-squares lambda policy iteration: Bias-variance trade-off in control problems. In *Proceedings of the 27th International Conference on Machine Learning*. Omnipress, 2010.

E. Todorov. Linearly-solvable Markov decision problems. In *Advances in Neural Information Processing Systems 19*, pages 1369–1376. MIT Press, 2007.

N. Vlassis and M. Toussaint. Model-free reinforcement learning as mixture learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1081–1088. ACM Press, 2009.

T. Wang, M. Bowling, and D. Schuurmans. Dual representations for dynamic programming and reinforcement learning. In *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 44–51. IEEE Press, 2007.

T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans. Stable dual dynamic programming. In *Advances in Neural Information Processing Systems 20*, pages 1569–1576. MIT Press, 2008.

C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 3(8):279–292, 1992.

Journal of Machine Learning Research 13 (2012) 3247-3251 Submitted 5/12; Revised 8/12; Published 11/12

Sally: A Tool for Embedding Strings in Vector Spaces

Konrad Rieck KONRAD.RIECK@UNI-GOETTINGEN.DE
University of Göttingen
Goldschmidtstraße 7
37077 Göttingen, Germany

Christian Wressnegger WRESSNEGGER@IDALAB.DE
idalab GmbH
Adalbertstraße 20
10977 Berlin, Germany

Alexander Bikadorov ABIKU@CS.TU-BERLIN.DE
Technische Universität Berlin
Franklinstraße 28/29
10587 Berlin, Germany

Editor: Antti Honkela

Abstract

Strings and sequences are ubiquitous in many areas of data analysis. However, only few learning methods can be directly applied to this form of data. We present Sally, a tool for embedding strings in vector spaces that allows for applying a wide range of learning methods to string data. Sally implements a generalized form of the bag-of-words model, where strings are mapped to a vector space that is spanned by a set of string features, such as words or n-grams of words. The implementation of Sally builds on efficient string algorithms and enables processing millions of strings and features. The tool supports several data formats and is capable of interfacing with common learning environments, such as Weka, Shogun, Matlab, or Pylab. Sally has been successfully applied for learning with natural language text, DNA sequences and monitored program behavior.

Keywords: string embedding, bag-of-words models, learning with sequential data

1. Introduction

Strings and sequences are a common representation of data and many applications of machine learning center on analyzing strings, for example, for discovering topics in natural language text, identifying genes in DNA, or filtering spam messages. However, the vast majority of learning methods can not be directly applied to string data, as these methods usually operate in vector spaces and require numerical vectors as input for learning. While a large body of research has studied techniques for learning with strings—most notably work on mapping strings to vectors (see Salton et al., 1975; Damashek, 1995) and string kernels (see Sonnenburg et al., 2007; Shawe-Taylor and Cristianini, 2004)—only few software tools have been made available to the community so far.

In this article, we present Sally, a general-purpose tool for mapping a set of strings to a set of vectors. This mapping is referred to as *embedding of strings* and allows for applying a wide range of learning methods to string data. Sally implements a generalized form of the bag-of-words model, where strings are mapped to a high-dimensional vector space that is spanned by a set of string features. Different types of features are supported for this embedding, which range from words

delimited by whitespace characters to positional and sorted n-grams. Sally proceeds by counting the occurrences of these features in each string and generating a sparse vector of frequency values. The implementation of Sally builds on string algorithms with linear run-time and space complexity, which enables processing millions of strings and features.

Sally is not the only tool for learning with strings; some learning toolboxes also provide support for extracting features from strings or computing string kernels. The respective implementations are often tightly coupled with the toolboxes and restricted to specific applications. By contrast, Sally can be thought of as a "Swiss Army Knife" for embedding strings: Instead of targeting a single toolbox or application, Sally provides a generic link between string data and learning methods. The tool supports several data formats and is capable of interfacing with common learning environments, such as Weka, Shogun, Matlab, or Pylab. Sally has been successfully applied in diverse learning settings, including text categorization, intrusion detection, clustering of malicious software, and analysis of electricity consumption (cf. Rieck and Laskov, 2008; Wahl et al., 2009; Rieck et al., 2011; Jawurek et al., 2011).

2. Embedding of Strings

Sally implements a generalized form of the classic bag-of-words model (Salton et al., 1975). A string is represented by a set of string features ("the bag") and mapped to a vector space whose dimensions are associated with the occurrences of these features. This association is created using a hash function, where the hash value of each feature defines its dimension. Moreover, the tool extends the original bag-of-words model by supporting features derived from string kernels, such as the spectrum kernel (Leslie et al., 2002), the weighted-degree kernel (Sonnenburg et al., 2007) and the word kernel (Lodhi et al., 2002).

2.1 String Features

Sally supports three basic types of string features for constructing a bag-of-words model. These types are defined implicitly by specifying delimiters (Configuration: `ngram_delim`) and the number of consecutive bytes/words to consider (Configuration: `ngram_len`).

1. *Words.* The strings are partitioned into substrings using a set of delimiter characters D. Such partitioning is typical for natural language processing, where the delimiters are usually defined as whitespace and punctuation characters (Configuration: `ngram_delim =` D`;` and `ngram_len = 1;`).

2. *Byte n-grams.* The strings are characterized by overlapping byte sequences of length n. These features are frequently used, if only little information about the structure of the strings is known, such as in bioinformatics and computer security (Configuration: `ngram_delim = "";` and `ngram_len =` n`;`).

3. *Word n-grams.* The strings are described by overlapping word sequences of length n. These features require the definition of delimiters D and a length n. They are often used as a coarse way for capturing structure in text and tokens (Configuration: `ngram_delim =` D`;` and `ngram_len =` n`;`).

Additionally to these basic types, Sally supports extensions for refining the set of string features. For instance, inspired by the weighted-degree kernel (Sonnenburg et al., 2007), Sally supports the

extraction of *positional features*. Each string feature is extracted along with its position in the string (Configuration: `ngram_pos = 1;`). As an example, the string `abab` then contains the positional 2-grams $\texttt{ab}_1$, $\texttt{ba}_2$ and $\texttt{ab}_3$.

Moreover, as with any data, strings can suffer from noise. Words may be swapped in text and DNA bases flip positions due to mutations. Such noise can be compensated by the extension of *sorted n-grams* (Configuration: `ngram_sort = 1;`). After the extraction of an n-gram, its elements are sorted, thereby removing the local ordering of the data. This removal may improve performance, if the strings suffer from local perturbations.

Finally, Sally also supports the use of stop words (Configuration: `stopword_file`) and the thresholding of values (Configuration: `thres_low` and `thres_high`). When analyzing natural language text, these extensions allow for filtering irrelevant and (in)frequent words from the resulting feature vectors.

2.2 Embedding Function

After the extraction of string features, Sally proceeds to construct a feature vector for each string in the defined output format. This construction can be formally defined as a mapping function Φ (see Rieck and Laskov, 2008),

$$\Phi : \mathcal{X} \longrightarrow \mathbb{R}^{|\mathcal{S}|}, \quad \Phi : x \longmapsto (\phi_s(x))_{s \in \mathcal{S}},$$

where $\mathcal{X}$ corresponds to the domain of strings and $\mathcal{S}$ to the set of (hashed) string features. Depending on the configuration, the inner function ϕ either returns the number of occurrences of the feature s in the string x, a binary flag for the occurrence of s or an TFIDF weighting of s. Furthermore, different normalizations can be applied to the resulting vectors (Configuration: `vect_embed =` *embed* and `vect_norm =` *norm*).

The size of the vector space can be controlled using the number of bits for hashing the string features (Configuration: `hash_bits`), where for k bits the vector space may span up to 2^k dimensions. While such high dimensionality is favorable for constructing an expressive representation of strings, it requires an efficient implementation to guarantee a tractable run-time. Fortunately, the number of features extracted by Sally is linear in length of each string and the resulting vectors are extremely sparse. This sparsity enables Sally to embed the strings in linear time and space, irrespective of the dimensionality of the vector space (cf. Rieck and Laskov, 2008; Shi et al., 2009).

3. Run-time Evaluation

To illustrate the efficient implementation of Sally, we conduct a run-time evaluation with data sets from four application domains: DNA sequences (ARTS; Sonnenburg et al., 2006), protein sequences (SPROT; O'Donovan et al., 2002), email messages (ENRON; Klimt and Yang, 2004) and text documents (RFC; www.ietf.org/rfc.html). Statistics of the data sets are shown in Table 1. As a baseline, we consider a typical Matlab and Python script for embedding strings. Both scripts are 60–70 lines long and make use of hashing for efficiently mapping strings to vectors. For each data set and implementation, we measure the run-time for embedding strings using byte/word 5-grams as features. We set the size of the feature hashing to 24 bits ($\approx$ 16.7 million dimensions). The evaluation is conducted on an Intel Xeon CPU with 2.6GHz with 4 GB of memory.

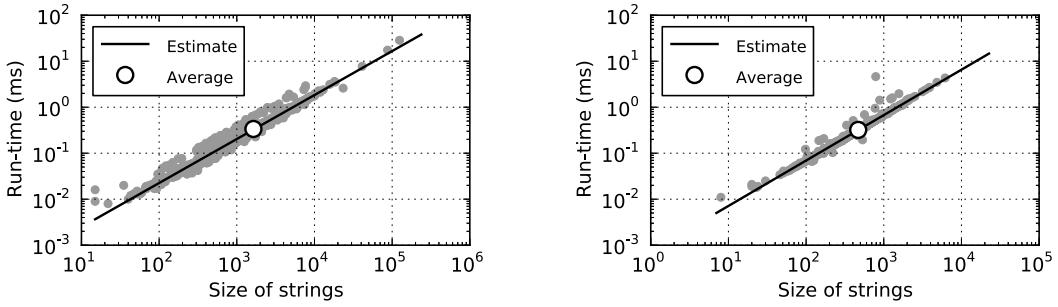
Results for the evaluation are shown in Table 1. All implementations are able to embed the strings in reasonable time (less than 10 minutes). However, Sally consistently outperforms the

other implementations. In comparison with the Python script, Sally embeds the strings 2.5× faster on average, where for Matlab a speedup of 9.5× is attained. This performance demonstrates the efficiency of Sally on different types of data, rendering it the tool of choice in applications where time matters, such as in large-scale and on-line learning.

Data sets	**ARTS**	**SPROT**	**ENRON**	**RFC**
Data set size	10^8 DNA bases	10^7 proteins	10^6 words	10^7 words
Number of strings	46,794	150,807	33,702	4,590
String features	byte 5-grams	byte 5-grams	word 5-grams	word 5-grams
Number of features	1,024 ($=4^5$)	2,800,728	4,070,853	12,136,059
Run-time performance				
Matlab script	528 s (9.6×)	381 s (7.3×)	158 s (11.4×)	530 s (9.6×)
Python script	140 s (2.5×)	183 s (3.5×)	30 s (2.1×)	113 s (2.1×)
Sally	55 s —	52 s —	14 s —	55 s —

Table 1: Run-time performance of Sally and typical scripts for embedding strings.

A fine-grained analysis of the run-time of Sally is shown in Figure 1. The embedding shows a linear run-time complexity even for large strings. On average, Sally is able to map a string to a vector within 0.3 ms including reading and writing of data, which amounts to an overall throughput of 3,000 strings per second.



(a) Email messages (word 5-grams)

(b) Protein sequences (byte 5-grams)

Figure 1: Detailed run-time analysis of Sally. The run-time per string is measured on the email messages of ENRON and the protein sequences of SPROT.

4. Conclusions

Sally provides a generic link between the wealth of string data and the available machinery of learning methods. Instead of targeting a specific application domain, the tool implements a generalized form of the bag-of-words model, which allows for learning with various types of string data, such as natural language text (Rieck and Laskov, 2008), payloads of network packets (Wahl et al., 2009) or even traces of electricity consumption (Jawurek et al., 2011). Moreover, by supporting different input and output formats, the tool can easily interface with common learning environments. Sally is open source software and available at the webpage *http://mlsec.org/sally*.

References

M. Damashek. Gauging similarity with *n*-grams: language-independent categorization of text. *Science*, 267(5199):843–848, 1995.

M. Jawurek, M. Johns, and K. Rieck. Smart metering de-pseudonymization. In *Proc. of Annual Computer Security Applications Conference (ACSAC)*, pages 227–236, Dec. 2011.

B. Klimt and Y. Yang. The Enron corpus: a new dataset for email classification research. In *Proc. of Conference on Email and Anti-Spam (CEAS)*, 2004.

C. Leslie, E. Eskin, and W. Noble. The spectrum kernel: a string kernel for SVM protein classification. In *Proc. of Pacific Symposium on Biocomputing (PSB)*, pages 564–575, 2002.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

C. O'Donovan, M. Martin, A. Gattiker, E. Gasteiger, A. Bairoch, and R. Apweiler. High-quality protein knowledge resource: SWISS-PROT and TrEMBL. *Briefings in Bioinformatics*, 3(3): 275–284, 2002.

K. Rieck and P. Laskov. Linear-time computation of similarity measures for sequential data. *Journal of Machine Learning Research*, 9(Jan):23–48, Jan. 2008.

K. Rieck, P. Trinius, C. Willems, and T. Holz. Automatic analysis of malware behavior using machine learning. *Journal of Computer Security (JCS)*, 19(4):639–668, June 2011.

G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, and S. Vishwanathan. Hash kernels for structured data. *Journal of Machine Learning Research*, 10(Nov):2615–2637, 2009.

S. Sonnenburg, A. Zien, and G. Rätsch. ARTS: accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–e480, 2006.

S. Sonnenburg, G. Rätsch, and K. Rieck. Large scale learning with string kernels. In *Large Scale Kernel Machines*, pages 73–103. MIT Press, Sept. 2007.

S. Wahl, K. Rieck, P. Laskov, P. Domschitz, and K.-R. Müller. Securing IMS against novel threats. *Bell Labs Technical Journal*, 14(1):243–257, May 2009.

Journal of Machine Learning Research 13 (2012) 3253-3295 Submitted 10/11, Revised 9/12; Published 11/12

Linear Fitted-Q Iteration with Multiple Reward Functions

Daniel J. Lizotte DLIZOTTE@UWATERLOO.CA
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON N2L 3G1, Canada

Michael Bowling BOWLING@CS.UALBERTA.CA
Department of Computing Science
University of Alberta
Edmonton, AB T6G 2E8, Canada

Susan A. Murphy SAMURPHY@UMICH.EDU
Department of Statistics
University of Michigan
Ann Arbor, MI 48109-1107, USA

Editor: Sridhar Mahadevan

Abstract

We present a general and detailed development of an algorithm for finite-horizon fitted-Q iteration with an arbitrary number of reward signals and linear value function approximation using an arbitrary number of state features. This includes a detailed treatment of the 3-reward function case using triangulation primitives from computational geometry and a method for identifying globally dominated actions. We also present an example of how our methods can be used to construct a real-world decision aid by considering symptom reduction, weight gain, and quality of life in sequential treatments for schizophrenia. Finally, we discuss future directions in which to take this work that will further enable our methods to make a positive impact on the field of evidence-based clinical decision support.

Keywords: reinforcement learning, dynamic programming, decision making, linear regression, preference elicitation

1. Introduction

Within the field of personalized medicine, there is increasing interest in investigating the role of sequential decision making for managing chronic disease (Weisz et al., 2004; McKay, 2009; Kuk et al., 2010). Reinforcement learning methods (Szepesvári, 2010; Sutton and Barto, 1998) are already being used (Pineau et al., 2007; Murphy et al., 2007; Zhao et al., 2009) to analyze Sequential Multiple Assignment Randomized Trials (SMART) (Murphy, 2005). A patient's progression through a SMART is divided into stages, each of which consists of a (typically uniform) random assignment to a treatment, followed by monitoring and recording data on the patient's condition. The patient data collected during each stage are very rich and commonly include several continuous variables related to symptoms, side-effects, treatment adherence, quality of life, and so on. For the ith patient in the trial, we obtain a *trajectory* of *observations* and *actions* of the form

$$o_1^i, a_1^i, o_2^i, a_2^i, \ldots, o_T^i, a_T^i, o_{T+1}^i.$$

Here, a_t^i represents the action (treatment) at time t, and o_t^i represents measurements made of patient i *after* action a_{t-1}^i and *before* action a_t^i. The first observations o_1^i are baseline measurements made before any actions are taken.

To analyze these data using reinforcement learning methods, we must define two functions $s_t(o_1, a_1, ..., o_t)$ and $r_t(s_t, a_t, o_{t+1})$ which map the patient's current history to a state representation and a scalar reward signal, respectively. Applying these functions to the data from the ith patient gives a trajectory

$$s_1^i, a_1^i, r_1^i, s_2^i, a_2^i, r_2^i, ..., s_T^i, a_T^i, r_T^i.$$

These redefined data are treated as sample trajectories from a known policy which is typically uniformly random over possible actions. Once we have these, we will view ongoing patient care as a Markov decision process (MDP) (Bertsekas and Tsitsiklis, 1996), and apply batch off-policy reinforcement learning methods to learn an optimal policy that takes a patient state and indicates which action appears to be best in view of the data available. In an MDP, both the state transition dynamics and the reward distributions are assumed to have the *Markov property*. That is, given the value s_t of the current state, the distribution of next state S_{t+1} and current reward R_t is conditionally independent of s_j, a_j, r_j for all $j < t$. Clearly this can be achieved by including past history in s_t, but this may not be feasible. For this work, we will assume that practitioners can suggest state features (which may be summaries of history) that are "as good as a complete history" in terms of making predictions about future states and rewards: we want features that are rich enough to provide good predictions about action values, but that are simple enough to allow us to learn from a limited amount of data. In medical domains, we may additionally want the learned policy to be easily interpreted and implemented. The interplay between predictiveness, learnability, and interpretability makes the definition of s_t a challenging problem that requires a great deal of further investigation, particularly into the consequences of a non-Markov definition of state. However, the question of how s_t should be defined can be answered at least in part by the data themselves together with expert knowledge and feature/model selection techniques analogous to those used in supervised learning settings (Keller et al., 2006) *if we have an adequate definition of* r_t.

A major difficulty with using trial data in this way is that there is often no obviously correct way to define r_t. Indeed, any definition of r_t is an attempt to answer the question "*What is the right quantity to optimize?*"—a question that is driven by the objectives of individual decision makers and *cannot be answered by the data alone.* There are many reasonable reward functions one could define, since each patient record includes a multi-dimensional measurement of that patient's overall well-being. For example, data often include a measure of the severity of the symptoms the patient is experiencing, as well as a measure of the severity of the side-effects caused by the current treatment. These different dimensions are typically better addressed by some treatments than by others, and therefore the choice of which dimension to use as the reward will affect the resulting learned policy. For example, a policy that minimizes expected symptom level will tend to choose more aggressive drugs that are very effective but that have a more severe side-effect profile. On the other hand, a policy that minimizes expected side-effect measurements will choose drugs that are less effective but that have milder side-effects.

In clinical practice, doctors, patients, and families decide on a treatment by weighing different measures of well-being, like symptoms and side-effects, according to subjective preferences that are not known to us at the time of data analysis. Continuing our example, these preferences may lean more toward symptom reduction or side-effect reduction, depending on the individual decision makers involved, and an appropriate reward function definition should reflect these preferences. In

principle, one could elicit these preferences, use them to define the reward function of each individual decision maker, and then learn a policy for that reward function; however accurate preference elicitation can be difficult to achieve, and even when it is possible it can be a time-consuming process for the decision maker. Moreover, this approach is problematic because it does not give a complete picture of the quality of the available actions under different reward choices. Indeed, the decision maker will not know when very small changes in preferences might lead to different actions, or when one action is optimal for a broad range of preferences, or when another action is not optimal for any preference.

Rather than eliciting preference and producing a policy that recommends a single action per state, our "inverse preference elicitation" approach is to first consider *all* of the actions available at the current state. For each of the actions, we answer the question, *"What range of preferences makes this action a good choice?"* This provides much richer information about the possible actions at each stage. Furthermore, even if a preference is specified somehow, our methods allow the maker to immediately see if his or her preferences are near a "boundary"—that is, whether a small change in preference can lead to a different recommended action. In this case, according to the data analysis two or more actions perform comparably well, and therefore the final decision could be based on other less crucial considerations such as dosing schedule and difference in cost. We are interested in efficient algorithms that can exactly compute the optimal policy for a range of reward functions to investigate how our choice of reward function influences the optimal policy, and in turn how we can offer more flexible choices among good actions.

2. Related Applications, Existing Methods, and Our Contributions

Our approach can help explore trade-offs in different application domains besides sequential medical decision making as well. In e-commerce, one may wish to trade off short-term profits with increased brand-visibility. In robotics, one may wish to trade-off rapid task completion against wear-and-tear on equipment. Researchers are already considering trading off water reserves versus flood risk in water reservoir control (Castelletti et al., 2010); our approach could provide further insight here. Even within RL itself, our approach could provide a new perspective on trading off achieving high expected reward with avoiding risk, an issue explored by Mannor and Tsitsiklis (2011). Any problem for which it is difficult or undesirable to formulate a single scalar reward to drive decision making could benefit from our approach.

There is wide interest in making use of multiple reward signals for sequential decision making. Gábor et al. (1998) demonstrated that an MDP with multiple reward signals can be well-defined and solved so long as we are given a fixed partial ordering on reward vectors. Mannor and Shimkin (2004) offer a formalism where actions are chosen to ensure that the long-term average reward vector approaches a "target set". The target set induces an ordering (closeness) on reward vectors which drives the agent's actions. Natarajan and Tadepalli (2005) assume that a scalar reward function is constructed by taking a weighted sum of a reward vector, just as we will. They assume that the weights are given, and that the weights will change over time. Their strategy is to learn a dictionary of policies for different weight vectors that should eventually contain policies that work well for many different preferences. They note that "An interesting direction for future research is to investigate the number of different weight vectors needed to learn all the optimal policies within a desired degree of accuracy," which we will address as part of this work. Early work in this direction (Barrett and Narayanan, 2008) explored the problem of simultaneously computing optimal policies for a

class of reward functions over a small, finite state space in a framework where the model is known. Subsequent developments were made that focussed on the infinite-horizon discounted setting and black-box function approximation techniques (Castelletti et al., 2010; Vamplew et al., 2011). We extended the approach of Barrett and Narayanan (2008) to the setting with real-valued state features and *linear* function approximation, which is a more appropriate framework for analyzing trial data (Lizotte et al., 2010). We also introduced an algorithm that is asymptotically more time- and space-efficient than the Barrett & Narayanan approach, and described how it can be directly applied to batch data. We also gave an algorithm for finding the set of all non-dominated actions in the single-variable continuous state setting. This paper builds on our previous work by contributing:

- A general and detailed development of finite-horizon fitted-Q iteration with an arbitrary number of reward signals and linear approximation using an arbitrary number of state features
- A detailed treatment of 3-reward function case using triangulation algorithms from computational geometry that has the same asymptotic time complexity as the 2-reward function case
- A more concise solution for identifying globally dominated actions under linear function approximation, and method for solving this problem in higher dimensions
- A real-world decision aid example that considers symptom reduction, weight gain, and quality of life when choosing treatments for schizophrenia

3. Background

We begin by defining the mathematical framework for our problem and describing its relationship to the usual MDP formulation. We then discuss how two existing formalisms, *Inverse Reinforcement learning* and *POMDP Planning*, relate to our approach.

3.1 Problem Framework

For each patient, we assume we will choose a treatment action at each timepoint $t = 1, 2, ..., T$, after which they are no longer under our care. In this finite-horizon sequential decision making setting, the optimal policy in general depends on t (Bertsekas and Tsitsiklis, 1996), so we explicitly maintain separate r_t, Q_t, and V_t functions for each timepoint, and we define $Q_T \equiv r_T$. Furthermore, it is convenient for our purposes to allow the set of possible states $\mathcal{S}_t$ and the set of possible actions $\mathcal{A}_t$ to depend on time. We then designate the learned policy at a particular time point by $\pi_t : \mathcal{S}_t \to \mathcal{A}_t$.

We consider sets of MDPs that all have the same $\mathcal{S}_t$, $\mathcal{A}_t$, and state transition dynamics, but whose expected reward functions $r_t(s_t, a_t, \delta)$ have an additional parameter δ. One may think of δ as a special part of state that: i) does not evolve with time, and ii) does not influence transition dynamics. Each fixed δ identifies a single MDP by fixing a reward function, which has a corresponding optimal[1] state-action value function defined in the usual way via the Bellman equation:

$$Q_t(s_t, a_t, \delta) = r_t(s_t, a_t, \delta) + E_{S_{t+1}|s_t,a_t}[\max_{a \in \mathcal{A}_{t+1}} Q_{t+1}(S_{t+1}, a, \delta)].$$

1. In this work, most Q- and V-functions are either optimal or estimates of optimal. We omit the usual * superscript in most cases, and mark estimates with a hat $\hat{}$.

We will also refer to the optimal state value function $V_t(s_t,\delta) = \max_{a\in\mathcal{A}_t} Q_t(s_t,a,\delta)$, and the[2] optimal deterministic policy $\pi_t(s_t,\delta) \in \text{argmax}_{a\in\mathcal{A}_t} Q_t(s_t,a,\delta)$. The purpose of δ is to represent the *preferences* of a decision maker: we presume that a decision maker would like to follow an optimal policy $\pi_t(s_t,\delta)$ for the MDP indexed by the value of δ that represents their preferences, that is, the value of δ for which $r_t(s_t,a_t,\delta)$ is a reflection of their reward function.

In order to mathematize the relationship between preference and δ, we define the structure of $r_t(s_t,a_t,\delta)$ to be

$$\delta = (\delta_{[0]},\delta_{[1]},...,\delta_{[D-1]}), \tag{1}$$

$$r_t(s_t,a_t,\delta) = \delta_{[0]} r_{t[0]}(s_t,a_t) + \delta_{[1]} r_{t[1]}(s_t,a_t) + ... + (1 - \sum_{d=0}^{D-2} \delta_{[d]}) r_{t[D-1]}(s_t,a_t). \tag{2}$$

where $\sum_{d=0}^{D-1} \delta_{[d]} = 1$. Thus $r_t(s_t,a_t,\delta)$ is a convex combination of the *basis rewards* $r_{t[0]},...,r_{t[D-1]}$, identified by a vector δ of length D that identifies points on the $(D-1)$-simplex. The vector δ represents a preference that assigns weight to each of the basis rewards. For example, if $\delta_{[d]} = 1$ for some index d, then $r_t(s_t,a_t,\delta) = r_{t[d]}(s_t,a_t)$, and the optimal policy $\pi_t(s_t,\delta)$ will choose actions that optimize the expected sum of rewards as determined by $r_{t[d]}$. Intuitively, the magnitude of $\delta_{[d]}$ determines how much $\pi_t(s_t,\delta)$ "cares about" the dth basis reward. Preferences defined by non-linear combinations of reward have been considered in non-sequential settings (e.g., Thall 2008), but such approaches would be much more computationally challenging in the sequential decision making setting in addition to being much more challenging to interpret; we discuss this in Section 7.1. Throughout, we presume our data are trajectories where the ith one takes the form

$$s^i_1, a^i_1, r^i_{1[0]}, r^i_{1[1]}, ..., r^i_{1[D-1]}, s^i_2, a^i_2, r^i_{2[0]}, r^i_{2[1]}, ..., r^i_{2[D-1]}, ..., s^i_T, a^i_T, r^i_{T[0]}, r^i_{T[1]}, ..., r^i_{T[D-1]}.$$

3.2 Related Approaches

Inverse Reinforcement Learning (IRL) (e.g., see Ng and Russell, 2000) comprises a collection of methods that take as input trajectories acquired by observing an expert and then attempt to infer the reward function of that expert. While IRL methods also operate in a setting with unknown rewards, our goal is quite different since we explicitly assume that our data do *not* come from an expert—in fact actions are often chosen uniformly randomly. Furthermore, we do *not* attempt to recover the reward function of any particular agent; we will instead attempt to learn the optimal policy for a set of reward functions. IRL methods could be useful if one wanted to attempt to explicitly learn the preference (i.e., the δ) of a decision-maker under our proposed framework; we leave this as potential future work.

Partially Observable Markov Decision Process (POMDP) planning (Kaelbling et al., 1998) comprises a large collection of methods that are designed to learn policies in the face of *partial observability*, that is, when the current "nominal state"[3] of the system is not observed. In this framework, the agent maintains a *belief state* (i.e., distribution) over the current nominal state and defines a value function and policy over these belief states. In the simplest setting with k nominal states, the space of possible belief states is the set of vectors on the $(k-1)$-simplex, and value-based exact POMDP planners compute the optimal value function for all possible belief states. In effect,

2. The optimal policy may not be unique, but this does not concern us.
3. The term "nominal state" is used to denote the actual unobserved state of the system, so as to distinguish it from the "belief state," which comprises the agent's current beliefs about the system.

the POMDP planner defines and solves the *belief MDP* in which the belief states are considered to be the observed (continuous) state.

Our goal in this work is to learn the optimal value function and policy for all possible preferences δ, which also happen live on the simplex. The value functions we learn are similar in structure to those of the POMDP planning problem, but there are at least two important differences.

First and foremost, the value functions and policies we learn are functions of preference *and* of additional state (e.g., patient information) both of which are assumed to be observed. We will see that in our formulation, for any fixed state the optimal value function is piecewise linear in preference. The preference part of the value function has a structure similar to that of the value function of a belief MDP, which is piecewise linear in the belief state.

Second, in POMDP planning, value is always a *convex* function of belief state, and this property is crucial in the development of exact and approximate (e.g., Pineau et al. 2006, Wang et al. 2006) methods. However, we will show in Section 4.2.4 that because our approach estimates value functions using regression, the Q-functions in our problem are *not* convex in δ. Since we do not have convexity, we will develop alternative methods for representing value functions in Section 4.

Despite these two differences, it is possible to interpret our definition of preference as a "belief" that the agent/patient is in exactly one of D different hidden "preference states" each corresponding to a single basis reward. We will not approach the problem from this point of view since we prefer the interpretation that each agent (e.g., patient) has a true observable δ and a corresponding reward function given by (2), but there may be applications where the hidden "preference state" interpretation is preferable. In any case, the two differences mentioned above mean that even if we interpret δ as a belief over preference states, standard POMDP methods are not applicable.

4. Fitted-Q Iteration for Multiple Reward Functions

In order to illustrate the intuition behind our approach, we first describe an algorithm for learning policies for all possible δ in the simplest case: a finite state space with $D=2$ basis rewards. We then describe how to accomodate linear function approximation with an arbitrary number of features in the $D=2$ setting. We then give a generalization to arbitrary D, and we provide an explicit algorithm for the $D=3$ case based on methods from computational geometry.

4.1 Optimal Value Functions for All Tradeoffs: Finite State Space, $D=2$ Basis Rewards

To begin, we assume that the $\mathcal{S}_t$ are all finite, and that state transition probabilities $P(s_{t+1}|s_t,a_t)$ and expected rewards $r_{t[0]}(s_t,a_t),\ldots,r_{t[0]}(s_t,a_t)$ are estimated using empirical averages from the data set and "plugged in" where appropriate. From an algorithmic standpoint, in this setting there is no difference whether these quantities are known or estimated in this way. We therefore present our algorithm as though all expectations can be computed exactly.

First, we consider two basis rewards $r_{t[0]}$ and $r_{t[1]}$ and corresponding preferences $\delta=(\delta_{[0]},\delta_{[1]})$. In this setting, the range of possible reward functions can be indexed by a single scalar $\delta=\delta_{[1]}$ by defining

$$r_t(s_t,a_t,\delta)=(1-\delta)\cdot r_{t[0]}(s_t,a_t)+\delta\cdot r_{t[1]}(s_t,a_t).$$

We will show that the optimal state-action value function $V_t(s_t,\delta)$ is piecewise-linear in the tradeoff parameter δ. Where appropriate, we will use the notation $V_t(s_t,\cdot)$ to represent the function of

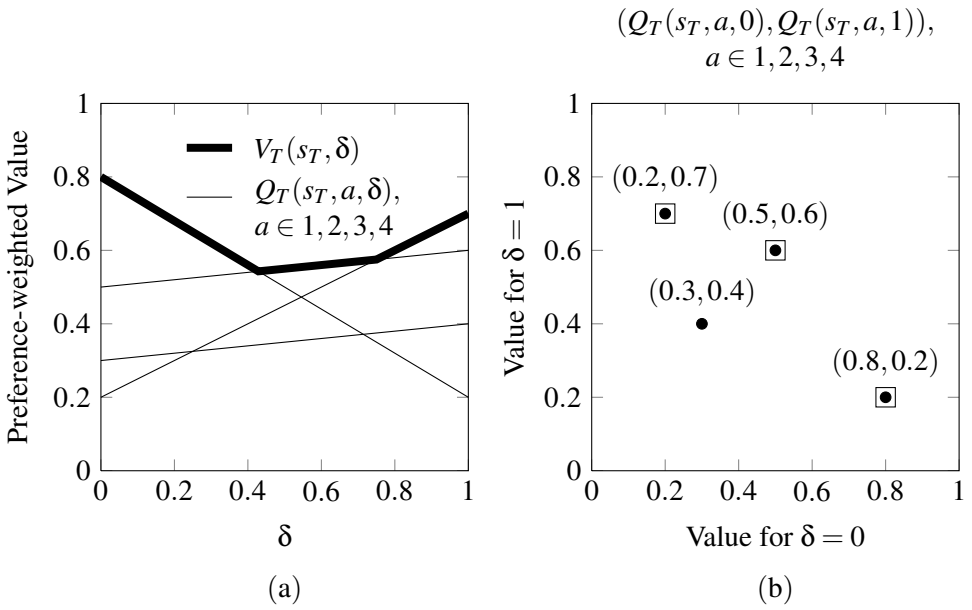


Figure 1: Computing V_T from Q_T for all δ by convex hull.

one argument (i.e., of δ) identified by fixing s_t. (We will use notation $Q_t(s_t,a_t,\cdot)$ and $r_t(s_t,a_t,\cdot)$ similarly.) We use an exact piecewise-linear representation of the functions $V_t(s_t,\cdot)$ for each state and timepoint, which allows us to exactly compute value backups for all δ more efficiently than the point-based representations of Barrett and Narayanan (2008). Our representation also allows identification of the set of dominated actions, that is, the actions that are not optimal for any (s_t,δ) pair. Value backups for finite state spaces require two operations: maximization over actions, and expectation over future states.

4.1.1 MAXIMIZATION

We begin at time $t = T$, the final time point,[4] and describe how to take a collection of functions $Q_T(s_T,a_T,\cdot)$ for all (s_T,a_T) and produce an explicit piecewise-linear representation of $V_T(s_T,\cdot)$ by maximizing over $a_T \in \mathcal{A}_T$. In Section 4.1.3, we show how this can be accomplished at earlier timepoints $t < T$ using a divide-and-conquer approach.

The Q-function for the last timepoint is equal to the terminal expected reward function r_T, which is linear in δ for each state-action pair as defined in (2), so we have $Q_T(s_T,a_T,\delta) = (1-\delta)\cdot r_{T[0]}(s_T,a_T) + \delta \cdot r_{T[1]}(s_T,a_T)$. To represent each $Q_T(s_T,a_T,\cdot)$, we maintain a list[5] of linear functions, one for each action, for each s_T. Figure 1(a) shows an example Q-function at a fixed state s_T at time T for four different actions, three of which are optimal for some δ and one which is not optimal for any δ. The linear function for each action can be represented by a list of tradeoff points (i.e., [0 1]) together with a list of their corresponding values (i.e., $[Q_T(s_T,a_T,0)\ Q_T(s_T,a_T,1)]$) at those tradeoff points. Each can also be represented by a point $(Q_T(s_T,a_T,0),Q_T(s_T,a_T,1))$ in the plane, as shown in Figure 1(b). These two equivalent representations offer an important concep-

4. We will write Q_T rather than $Q_{t=T}$ in this section, and similarly write $s_T, \mathcal{A}_T$, etc.
5. We denote an ordered list with objects a,b,c by $[a\ b\ c]$.

tual and computational insight that is well-established in the multi-criterion optimization literature (Ehrgott, 2005): the set of actions that are optimal for some $\delta \in [0,1]$ are exactly those actions whose line-representations lie on the upper convex envelope of the Q-functions, and equivalently, whose point-based representations lie on the upper-right convex hull of the set of points in the plane. In general, we can recover the actions that are optimal on any interval $[\delta, \delta']$ by finding the upper-right convex hull of the points $\{(Q_T(s_T,a,\delta), Q_T(s_T,a,\delta')) : a \in \{1...|A|\}\}$. This equivalence is important because the time complexity of the convex hull operation on n points in two or three dimensions is $O(n \log n)$ (de Berg et al., 2008)—as fast as sorting.

We make use of this equivalence to construct our piecewise-linear representation of $V_T(s,\cdot)$. Commonly-used convex hull routines produce output that is ordered, so it is easy to recover the list of actions that are optimal for some δ, along with the values of δ where the optimal action changes. These values are the "knots" in the piecewise-linear representation. We denote the list of knots of a piecewise-linear function $f(\cdot)$ by $\Delta(f(\cdot))$. The output of a convex hull algorithm is an ordered list of points, each of the form $(Q_T(s_T,a_T,0), Q_T(s_T,a_T,1))$. In this case, the list is $[(0.8,0.2)\ (0.5,0.6)\ (0.2,0.7)]$. We know from the order of this list that the second knot in $V_T(s,\cdot)$ (after $\delta = 0$) occurs where the lines represented by $(0.8,0.2)$ and $(0.5,0.6)$ intersect. Thus we can compute that the line represented by $(0.8,0.2)$ is maximal from $\delta = 0$ to $\delta = 0.43$, at which point it intersects the line represented by $(0.5,0.6)$. After finding the knots, we represent the piecewise-linear value function in Figure 1(a) by the ordered knot-list $\Delta(V_T(s_T,\cdot)) = [0.00\ 0.43\ 0.75\ 1.00]$ and value-list $[0.80\ 0.54\ 0.58\ 0.70]$, rather than by the list of points. To recover the policy at this point, we may also retain a list of lists containing the actions that are optimal at each knot: $[[1]\ [1\ 2]\ [2\ 4]\ [4]]$. This allows us to determine the action or actions that are optimal for any segment by taking the intersection of the action lists for the endpoints of the segment. Note that because $V_T(s_T,\cdot)$ is a point-wise maximum of convex[6] functions, it is convex.

Our representation allows us to evaluate $V_T(s_T,\delta) = \max_{a \in \mathcal{A}_T} Q_T(s_T,a_T,\delta)$ efficiently. Because our knot list and value list are ordered, we can use binary search to find the largest knot in $V_T(s_T,\cdot)$ that is less than δ. This tells us which linear piece is maximal for δ, so we only need to evaluate this single linear function. Thus computing $V_T(s_T,\delta)$ takes $O(\log|\Delta(V_T(s_T,\cdot))|)$ time, that is, the time for the cost of the search, rather than the $O(|\Delta(V_T(s_T,\cdot))|)$ time it would take to evaluate all of the linear functions at δ and then take the maximum.

4.1.2 EXPECTATION

We now show how we use our piecewise-linear representation of $V_T(s_T,\cdot)$ to efficiently compute a piecewise-linear representation of

$$Q_{T-1}(s_{T-1},a_{T-1},\cdot) = r_{T-1}(s_{T-1},a_{T-1},\cdot) + E_{S_T}[V_T(S_T,\cdot)|s_{T-1},a_{T-1}]$$

using the piecewise-linear representation of V_T. To do so, we must evaluate conditional expectations of V_T over possible future states.

Consider an example with two terminal states $s_T = 1$ and $s_T = 2$. Suppose that the probability of arriving in state j (conditioned on some (s_{T-1},a_{T-1})) is given by θ_j. Since each $V_T(j,\cdot)$ is linear over the intervals between $\Delta(V_T(j,\cdot))$, these two functions are *simultaneously* linear over the intervals between $\Delta(V_T(1,\cdot)) \cup \Delta(V_T(2,\cdot))$, and their weighted average is linear over the same

6. The $Q_T(s_T,a_T,\cdot)$ are each linear, and therefore convex.

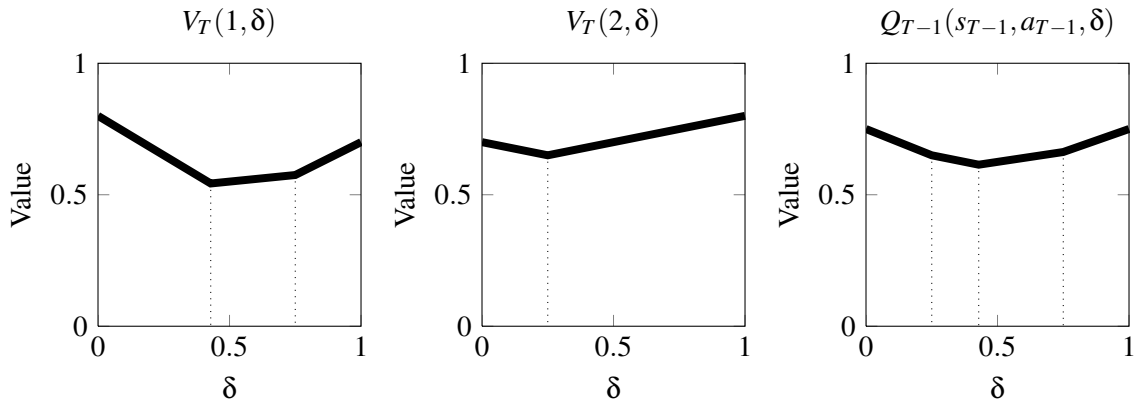


Figure 2: Computing expectations using unions of piecewise-linear representations. The graphs of $V_T(1,\delta)$ and $V_T(2,\delta)$ show the time T value function at two different states, $S_T = 1$ and $S_T = 2$. The graph of $Q_{T-1}(s_{T-1},a_{T-1},\delta) = 0.5 \cdot V_T(1,\delta) + 0.5V_T(2,\delta)$ shows the expected action-value function if the two states are each reached with probability 0.5 when starting from state $S_{T-1} = s_{T-1}$ and taking action $A_{T-1} = a_{T-1}$, and there is no immediate reward.

intervals. Therefore the expectation

$$E_{S_T}[V_T(S_T,\cdot)|s_{T-1},a_{T-1}] = \theta_1 \cdot V_T(1,\cdot) + \theta_2 \cdot V_T(2,\cdot)$$

is itself a piecewise-linear function of δ with knot-list $\Delta(V_T(1,\cdot)) \cup \Delta(V_T(2,\cdot))$. Since the function $r_{T-1}(s_{T-1},a_{T-1},\cdot)$ is linear, it does not contribute additional knots, so we can compute the piecewise-linear representation of $Q_{T-1}(s_{T-1},a_{T-1},\cdot)$ by computing its value-list at the aforementioned knots. The value list of $Q_{T-1}(s_{T-1},a_{T-1},\cdot)$ is

$$\left[\left(r_{T-1}(s_{T-1},a_{T-1},\delta) + \sum_j \theta_j V_T(j,\delta)\right) : \delta \in \Delta(V_T(1,\cdot)) \cup \Delta(V_T(2,\cdot))\right].$$

Let $k_j = |\Delta(V_T(s'_j,\cdot))|$. This construction uses $O(k_1+k_2)$ space and requires $O(k_1+k_2)$ evaluations of V_T. Note that because $Q_{T-1}(s_{T-1},a_{T-1},\cdot)$ is a positive weighted sum of convex functions, it is convex. Figure 2 illustrates this weighted sum operation.

We contrast the piecewise-linear representation approach with that of Barrett and Narayanan (2008). The expectation can also be computed using the point-based representation in Figure 1(b): let Ξ_i be the set of points in the point-based representation of $V_T(s_T,\cdot)$. One can compute the point-based representation of $Q_{T-1}(s_{T-1},a_{T-1},\cdot)$ by constructing a set of points

$$\begin{aligned}&\{(r_{(T-1)[0]}, r_{(T-1)[1]}) + \theta_1 \cdot (a_1,b_1) + \theta_2 \cdot (a_2,b_2)\},\\ &\text{where } r_{(T-1)[\delta]} = r_{T-1}(s_{T-1},a_{T-1},\delta)\\ &\text{for all } (a_1,b_1) \in \Xi_1,\ \ (a_2,b_2) \in \Xi_2.\end{aligned} \tag{3}$$

and then taking the upper-right portion of the convex hull of this set. Barrett and Narayanan (2008) advocate this procedure and prove its correctness; however, they note that the set given in (3) has

$|\Xi_1||\Xi_2|$ points that must be constructed and fed into the convex hull algorithm. Since $k_i = |\Xi_i| + 1$, computing the expectation in this way will take $O(k_1 k_2)$ space and $O(k_1 k_2 \log k_1 k_2)$ time, which is asymptotically much less efficient than our $O(k_1 + k_2)$ piecewise-linear representation based approach.

4.1.3 VALUE BACKUPS FOR $t < T-1$

The maximization procedure described in Section 4.1.1 relies on the linearity of $Q_T(s_T, a_T, \cdot)$. However, for $t < T$, we have shown that $Q_t(s_t, a_t, \cdot)$ is piecewise-linear. We now show how to compute V_t and Q_t from Q_{t+1} by first decomposing $Q_{t+1}(s_{t+1}, a_{t+1}, \cdot)$ into linear pieces and applying the expectation and maximization operations to each piece. Recall that

$$Q_t(s_t, a_t, \delta) = r_t(s_t, a_t, \delta) + E_{S_{t+1}}[V_{t+1}(S_{t+1}, \delta) | s_t, a_t].$$

We have shown by construction that $Q_{T-1}(s_{T-1}, a_{T-1}, \cdot)$ is convex and piecewise-linear. In general, $V_t(s_t, \cdot)$ is computed by taking a point-wise max over functions $Q_t(s_t, a_t, \cdot)$, and $Q_{t-1}(s_{t-1}, a_{t-1}, \cdot)$ is computed by taking a positive weighted sum of the convex functions $r_{t-1}(s_{t-1}, a_{t-1}, \cdot)$ and $V_t(s_t, \cdot)$. Since both of these operations preserve convexity and piecewise-linearity, it follows by induction that $Q_t(s_t, a_t, \cdot)$ is convex piecewise-linear for all $t \in 1, ..., T$. To compute $Q_t(s_t, a_t, \cdot)$, we first identify the knots in $E_{S_{t+1}}[V_{t+1}(S_{t+1}, \cdot) | s_t, a_t]$ and store them; this is done in the same way as for $t+1 = T$. We then compute the value-list as described above. To compute $V_t(s_t, \cdot) = \max_{a \in \mathcal{A}_t} Q_t(s_t, a, \cdot)$, we take the maximum over actions of these piecewise-linear Q-functions using Algorithm 2. First, we decompose the problem of finding $\max_{a \in \mathcal{A}_t} Q_t(s_t, a, \cdot)$ for $\delta \in [0, 1]$ into sub-problems of finding $\max_{a \in \mathcal{A}_t} Q_t(s_t, a, \cdot)$ over intervals of δ where we know the $Q_t(s_t, a, \cdot)$ are simultaneously linear. The ends of these intervals are given by $\bigcup_a \Delta(Q_t(s_t, a, \cdot))$. We then apply the convex hull algorithm to each of these intervals to recover any additional knots in $\max_{a \in \mathcal{A}_t} Q_t(s_t, a_t, \cdot)$.

The full backup procedure is described in Algorithm 1. In practice, we can avoid running the convex hull algorithm over every interval by checking each interval's end points: if for some action a_t^* we find that $Q_t(s_t, a_t^*, \cdot)$ is maximal at both ends of an interval in the current knot-list, then $\max_a Q_t(s_t, a, \cdot)$ has no knots inside the interval. Note that though we present our algorithms assuming the reward functions are linear, they will work for piecewise-linear reward functions as well.

4.1.4 COMPLEXITY OF $Q_t(s_t, a_t, \cdot)$ AND $V_t(s_t, \cdot)$

Suppose there are $|\mathcal{S}|$ states and $|\mathcal{A}|$ actions at each stage. For any fixed s_T, each function $Q_T(s_T, i, \cdot)$, $i = 1..|\mathcal{A}|$, has 2 knots, $\delta = 0$ and $\delta = 1$. Applying Algorithm 2 to produce $V_T(s_T, \cdot)$ from these functions generates at most $|\mathcal{A}| - 1$ new internal knots, and therefore each $V_T(s_T, \cdot)$ has at most $(|\mathcal{A}| - 1) + 2$ knots. To compute $Q_{T-1}(s_{T-1}, a_{T-1}, \cdot)$, we take the expectation of $V_T(s_T, \cdot)$ over states s_T. Since $V_T(s_T, \cdot)$ might have different internal knots for every s_T, $Q_{T-1}(s_{T-1}, a_{T-1}, \cdot)$ may have as many as $|\mathcal{S}|(|\mathcal{A}| - 1) + 2$ knots. However, the knots in $Q_{T-1}(s_{T-1}, a_{T-1}, \cdot)$ will be the same for all s_{T-1} and a_{T-1}. Computing $V_{T-1}(s_{T-1}, \cdot)$ using Algorithm 2 adds at most $|\mathcal{A}| - 1$ new knots between each pair of existing knots, for a total of $(|\mathcal{A}| - 1|)(|\mathcal{S}|(|\mathcal{A}| - 1) + 1) + 2$. In general, $Q_t(s_t, a_t, \cdot)$ may have up to $O(|\mathcal{S}|^{T-t}|\mathcal{A}|^{T-t})$ knots, and $V_t(s_t, \cdot)$ may have up to $O(|\mathcal{S}|^{T-t}|\mathcal{A}|^{(T-t)+1})$ knots.

To compute $Q_t(s_t, a_t, \cdot)$ from r_t and V_{t+1}, our approach requires $O(|\mathcal{S}|^{T-t}|\mathcal{A}|^{(T-t)+1})$ time for each state, for a total of $O(|\mathcal{S}|^{(T-t)+1}|\mathcal{A}|^{(T-t)+1})$ time. In contrast, the approach of Barrett &

Algorithm 1 Value Backup - Finite State Space

/* $A \overset{\cup}{\leftarrow} B$ means $A \leftarrow A \cup B$ */
$\forall (s_{T+1}, \delta)$, $V_{T+1}(s_{T+1}, \delta) \triangleq 0$. $\forall s_{T+1}$, $\Delta(V_{T+1}(s_{T+1}, \cdot)) \triangleq \{0,1\}$.
for $t = T$ **downto** 1 **do**
  for all $s_t \in \mathcal{S}_t$ **do**
    for all $a_t \in \mathcal{A}_t$ **do**
      $\Delta(Q_t(s_t, a_t, \cdot)) \leftarrow \{\}$
      for all $s_{t+1} \in \mathcal{S}_{t+1}$ **do**
        $\Delta(Q_t(s_t, a_t, \cdot)) \overset{\cup}{\leftarrow} \Delta(V_{t+1}(s_{t+1}, \cdot))$
      end for
      for all $\delta \in \Delta(Q_t(s_t, a_t, \cdot))$ **do**
        $Q_t(s_t, a_t, \delta) \leftarrow r(s_t, a_t, \delta) +$
          $\sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) \cdot V_{t+1}(s_{t+1}, \delta)$
      end for
    end for
    Compute $\Delta(V_t(s_t, \cdot))$ by applying Algorithm 2
    to $Q_t(s_t, a, \cdot)$, $a \in \mathcal{A}_t$
  end for
end for

Algorithm 2 Max of Piecewise-Linear Functions

/* $A \overset{\cup}{\leftarrow} B$ means $A \leftarrow A \cup B$ */
input piecewise-linear functions $f_i(\cdot)$, $i = 1..k$ defined on $[\delta_0, \delta_1]$.
$\Delta^{\text{all}} = \bigcup_{i=1}^{k} \Delta(f_i(\cdot))$
$\Delta^{\text{out}} = \Delta^{\text{all}}$
for $i = 2$ **to** $|\Delta^{\text{all}}|$ **do**
  if $\text{argmax}_j f_j(\Delta^{\text{all}}_{i-1}) \neq \text{argmax}_j f_j(\Delta^{\text{all}}_i)$ **then**
    $\Delta^{\text{out}} \overset{\cup}{\leftarrow} \Delta(\max_j f_j(\delta), \delta \in (\Delta^{\text{all}}_{i-i}, \Delta^{\text{all}}_i))$
  end if
end for

Narayanan requires $O(|\mathcal{S}|^{2\cdot(T-t)+1}|\mathcal{A}|^{2\cdot(T-t)+1} \log |\mathcal{S}|^{2\cdot(T-t)+1}|\mathcal{A}|^{2\cdot(T-t)+1})$ time for each of $\log_2 |\mathcal{S}|$ pairs of piecewise-linear functions.

4.2 Optimal Value Functions for All Tradeoffs: Linear Function Approximation, $D=2$ Basis Rewards

Here, we demonstrate how our previously developed algorithms for value backups over all tradeoffs can be extended to the case where we have arbitrary features of state variables and we use a linear approximation of the Q_t functions. Again, we first consider Q_T and V_T, which have the simplest form, and then describe how to compute Q_t and V_t at earlier time points. This treatment allows for linear function approximators based on an arbitrary number of state features rather than the single continuous state feature described in Lizotte et al. (2010).

Suppose the expected terminal rewards $Q_T(s_T, a_T, 0)$ and $Q_T(s_T, a_T, 1)$ are each linear functions of the form $Q_T(s_T, a_T, 0) = \phi_{s_T,a_T}^{\mathsf{T}} \beta_{T[0]}$ and $Q_T(s_T, a_T, 1) = \phi_{s_T,a_T}^{\mathsf{T}} \beta_{T[1]}$. Here, ϕ_{s_T,a_T} is a *feature vector*[7] that depends on state and action, and the weight vectors $\beta_{T[0]}$ and $\beta_{T[1]}$ define the linear relationships between the feature vector and the expectation of the two basis rewards at time T. From Equation (2), we have

$$\begin{aligned} Q_T(s_T, a_T, \delta) &= (1-\delta) \cdot \phi_{s_T,a_T}^{\mathsf{T}} \beta_{T[0]} + \delta \cdot \phi_{s_T,a_T}^{\mathsf{T}} \beta_{T[1]}, \\ &= \phi_{s_T,a_T}^{\mathsf{T}} \left[(1-\delta) \cdot \beta_{T[0]} + \delta \cdot \beta_{T[1]} \right]. \end{aligned} \tag{4}$$

A typical definition of ϕ_{s_T,a_T} might include a constant component for the intercept, the measurements contained in s_T, the discrete action a_T encoded as dummy variables, and the product of the measurements in s_T with the encoded a_T (Cook and Weisberg, 1999). One could also include other non-linear functions of s_T and a_T as features if desired. In particular, one could produce exactly the same algorithm described in Section 4.1 by using feature vectors of length $|\mathcal{S}| \times |\mathcal{A}|$ that consist of a separate indicator for each state-action pair. In this case the estimated parameters will be precisely the sample averages from Section 4.1. Note from (4) that regardless of the definition of ϕ_{a_T,s_T}, the function $Q_T(s_T, a_T, \delta)$ is linear in δ for any fixed s_T, a_T.

Recall that we have a set of trajectories of the form

$$s_1^i, a_1^i, r_{1[0]}^i, r_{1[1]}^i, s_2^i, a_2^i, r_{2[0]}^i, r_{2[1]}^i, ..., s_T^i, a_T^i, r_{T[0]}^i, r_{T[1]}^i,$$

for $i = 1...N$ with which to estimate the optimal Q functions. In order to estimate $Q_T(s_T, a_T, 0)$ and $Q_T(s_T, a_T, 1)$, we compute parameter estimates $\hat{\beta}_{T[0]}$ and $\hat{\beta}_{T[1]}$ using ordinary least-squares regression by first constructing a design matrix and regression targets

$$\Phi_T = \begin{bmatrix} \phi_{s_T^1,a_T^1}^{\mathsf{T}} \\ \phi_{s_T^2,a_T^2}^{\mathsf{T}} \\ \vdots \\ \phi_{s_T^N,a_T^N}^{\mathsf{T}} \end{bmatrix}, \quad \mathbf{r}_{T[0]} = \begin{bmatrix} r_{T[0]}^1 \\ r_{T[0]}^2 \\ \vdots \\ r_{T[0]}^N \end{bmatrix}, \quad \mathbf{r}_{T[1]} = \begin{bmatrix} r_{T[1]}^1 \\ r_{T[1]}^2 \\ \vdots \\ r_{T[1]}^N \end{bmatrix}.$$

We then compute parameter estimates

$$\begin{aligned} \hat{\beta}_{T[0]} &= (\Phi_T^{\mathsf{T}} \Phi_T)^{-1} \Phi_T^{\mathsf{T}} \mathbf{r}_{T[0]}, \\ \hat{\beta}_{T[1]} &= (\Phi_T^{\mathsf{T}} \Phi_T)^{-1} \Phi_T^{\mathsf{T}} \mathbf{r}_{T[1]}, \end{aligned}$$

using ordinary least squares.[8] These estimated parameters are then substituted into definition (4), giving $\hat{Q}_T(s_T, a_T, 0)$ and $\hat{Q}_T(s_T, a_T, 1)$. To construct an estimate $\hat{Q}_T(s_T, a_T, \delta)$ for arbitrary $\delta \in [0, 1]$, we could construct a scalar reward using $\mathbf{r}_{t[0]}$, $\mathbf{r}_{t[1]}$, and δ, and solve for the corresponding $\hat{\beta}_T(\delta)$, giving

$$\hat{\beta}_T(\delta) = (\Phi_T^{\mathsf{T}} \Phi_T)^{-1} \Phi_T^{\mathsf{T}} \left[(1-\delta) \mathbf{r}_{T[0]} + \delta \mathbf{r}_{T[1]} \right],$$

7. In statistical terms, $\phi_{s_T,a_T}^{\mathsf{T}}$ represents a row in the design matrix of the linear model.
8. We could also use ridge regression with a fixed ridge parameter. All of our techniques immediately apply in this case as well since the parameter estimates remain piecewise linear in δ.

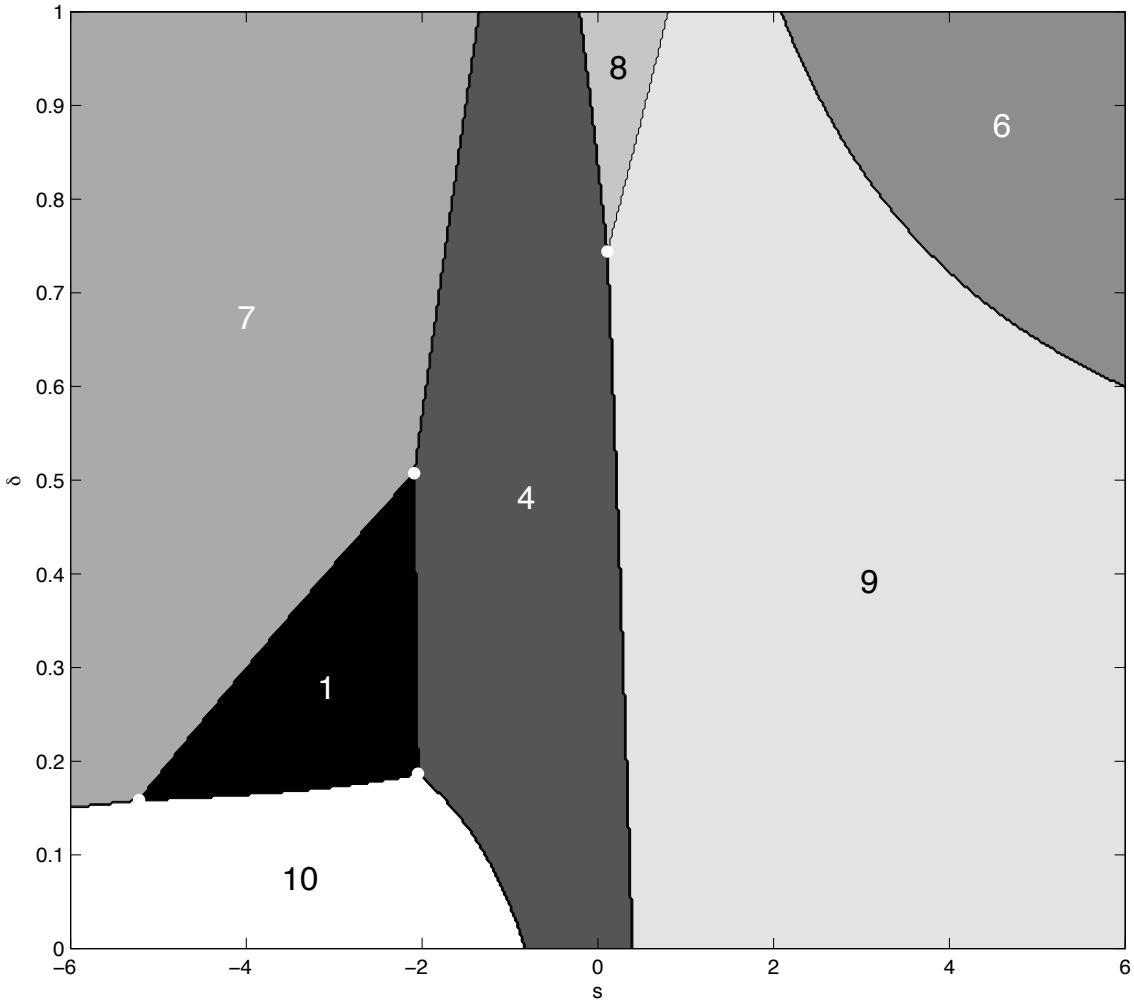


Figure 3: Diagram of the regions in (ψ_{s_T}, δ) space where different actions are optimal at time T. In this example, $\psi_{s_T} \in [-6, 6]$.

but by linearity,

$$\begin{aligned}\hat{\beta}_T(\delta) &= (1-\delta)\cdot(\Phi_T^{\mathsf{T}}\Phi_T)^{-1}\Phi_T^{\mathsf{T}}\mathbf{r}_{T[0]} + \delta\cdot(\Phi_T^{\mathsf{T}}\Phi_T)^{-1}\Phi_T^{\mathsf{T}}\mathbf{r}_{T[1]}, \\ &= (1-\delta)\cdot\hat{\beta}_{T[0]} + \delta\cdot\hat{\beta}_{T[1]}.\end{aligned}$$

Thus we only need to solve for $\hat{\beta}_{T[0]}$ and $\hat{\beta}_{T[1]}$, after which we compute $\hat{Q}_T(s_T, a_T, \delta) = \phi_{s_T,a_T}^{\mathsf{T}}\hat{\beta}_T(\delta)$ for any δ by taking convex combinations of these coefficient vectors. Therefore, for $t = T$, it is straightforward to exactly represent $\hat{Q}_T(s_T, a_T, \delta)$ for all δ.

4.2.1 MAXIMIZATION

For any fixed values of s_T and a_T, $\hat{Q}_T(s_T, a_T, \cdot)$ is a linear function. Therefore, we can use the convex hull method to identify the actions that maximize value at a given s_T, and use it to recover the knots in the piecewise-linear $\hat{V}_T(s_T, \cdot)$.

Figure 3 is an illustration of the pieces of a hypothetical $\hat{V}_T(s_T, \cdot)$ that is a maximization over 10 actions. In this example we define a scalar state feature ψ_{s_T} and we construct feature vectors

$$\phi_{s_T, a_T} = [[1 \ \psi_{s_T}] 1_{a_T=1} \ \ [1 \ \psi_{s_T}] 1_{a_T=2} \ \ \ldots \ \ [1 \ \psi_{s_T}] 1_{a_T=10}]^{\mathsf{T}}, \tag{5}$$

where $1_{a_T=k}$ is the indicator function that takes the value 1 if $a_T = k$ and 0 otherwise. Note that this choice of features is equivalent to defining $\hat{Q}$ using a separate linear regression for each action. Each number in Figure 3 marks the region where that action is optimal at time T. For example, a vertical slice at $\psi_{s_T} = -4$ of the value function has three linear pieces where actions 10, 1, and 7 are optimal.

In the finite state-space case, we explicitly represented $V_T(s_T, \cdot)$ separately for each nominal state s_T in the MDP in order to allow computation of expectations over terminal states. In contrast, in the linear regression setting, we represent $\hat{V}_T(s_T, \cdot)$ for each *observed* terminal state $s_T^1, \ldots, s_T^N$ in our data set. That is, we explicitly represent a one-dimensional slice of the value function for each of the s_T^i by applying Algorithm 2 to construct a piecewise-linear representation for $\hat{V}_T(s_T^i, \cdot)$.

4.2.2 REGRESSION ON STATE FEATURES

At stage $T-1$, the regression parameters of our estimate $\hat{Q}_{T-1}(s_{T-1}, a_{T-1}, \delta)$ are given by

$$\hat{\beta}_{T-1}(\delta) = (\Phi_{T-1}^{\mathsf{T}} \Phi_{T-1})^{-1} \Phi_{T-1}^{\mathsf{T}} \hat{\mathbf{y}}_{T-1}(\delta),$$

where, for $t \in \{1, \ldots, T-1\}$, we define

$$\hat{\mathbf{y}}_t(\delta) = ((1-\delta)\mathbf{r}_{t[0]} + \delta \mathbf{r}_{t[1]}) + \hat{\mathbf{v}}_{t+1}(\delta),$$

which are the one-step value estimates for time t, where

$$\hat{\mathbf{v}}_{t+1}(\delta) = \begin{bmatrix} \hat{V}_{t+1}(s_{t+1}^1, \delta) \\ \hat{V}_{t+1}(s_{t+1}^2, \delta) \\ \vdots \\ \hat{V}_{t+1}(s_{t+1}^N, \delta) \end{bmatrix}.$$

The components of the vector $\hat{\mathbf{y}}_{T-1}(\delta)$ are not linear in δ, so for $t < T$, solving the regression only for $\delta = 0$ and $\delta = 1$ does not completely determine $\hat{Q}_t(s_t, a_t, \delta)$. However, the components of $\hat{\mathbf{y}}_{T-1}(\delta)$ are each piecewise-linear in δ. We determine the intervals over which the components are simultaneously linear and then explicitly represent the state-value function at the knots $[\delta_1 \ \delta_2 \ \ldots \ \delta_K]$ between these intervals. The output accompanying this list of knots is a list of estimated parameter vectors $[\beta_{T-1}(\delta_1) \ \beta_{T-1}(\delta_2) \ \ldots \ \beta_{T-1}(\delta_K)]$, each given by $\hat{\beta}_{T-1}(\delta_k) = (\Phi_{T-1}^{\mathsf{T}} \Phi_{T-1})^{-1} \Phi_{T-1}^{\mathsf{T}} \hat{\mathbf{y}}_{T-1}(\delta_k)$. This collection of parameters is analogous to the *value list* in the finite state-space case, and completely defines $\hat{Q}_{T-1}(s_{T-1}, a_{T-1}, \cdot)$ for all s_{T-1} and a_{T-1}. As before, we can also retain a list of the optimal actions at each of the knots in order to later recover the policy.

Algorithm 3 Value Backup - Linear Function Approximation, $D=2$ Basis Rewards

$\forall(s,\delta)$, $\hat{V}_{T+1}(s,\delta) \triangleq 0$. $\forall s$, $\Delta(\hat{V}_{T+1}(s,\cdot)) \triangleq \{0,1\}$.
for $t = T$ **downto** 1 **do**
  $\Delta^{\hat{Q}_t} \leftarrow \{\}$
  for all $(s_t, a_t, s_{t+1}) \in \mathcal{D}$ **do**
    $\Delta^{\hat{Q}_t} \leftarrow \Delta^{\hat{Q}_t} \cup \Delta(\hat{V}_{t+1}(s_{t+1},\cdot))$
  end for
  for all $\delta \in \Delta^{\hat{Q}_t}$ **do**
    $\mathbf{y}_t^{(\delta)} = ((1-\delta)\mathbf{r}_{t[0]} + \delta\mathbf{r}_{t[1]}) + \hat{\mathbf{v}}_{t+1}(\delta)$
    $\hat{\beta}_t^{(\delta)} = (\Phi_t^\mathsf{T}\Phi_t)^{-1}\Phi_t^\mathsf{T}\mathbf{y}_t^{(\delta)}$
  end for
  for all $s_t \in \mathcal{D}$ **do**
    Compute $\Delta(\hat{V}_t(s_t,\cdot))$ by Algorithm 2
  end for
end for

4.2.3 VALUE BACKUPS FOR $t < T-1$

The procedure for computing the $\hat{V}_T(s_T^i,\cdot)$ relies on the linearity of $\hat{Q}_T(s_T^i,a_T,\cdot)$, but for $t<T$, $\hat{Q}_t(s_t,a_t,\cdot)$ is piecewise-linear in general. Thus to compute $\hat{V}_t(s_t^i,\cdot) = \max_a \hat{Q}_t(s_t^i,a,\cdot)$ for each s_t^i in our data set, we apply Algorithm 2 for each s_t^i, using regression to compute $Q_{t-1}(s_{t-1},a_{t-1},\cdot)$ from these functions then proceeds as we did for the $t=T-1$ case. The entire procedure is described in Algorithm 3.

4.2.4 NON-CONVEXITY OF $\hat{Q}_t(s_t,a_t,\cdot)$

For $t<T$, the resulting $\hat{Q}_t(s_t,a_t,\cdot)$ are not necessarily convex in the regression setting, as we alluded to in Section 3.2. To see this, recall that each element of $\hat{\beta}_{T-1}(\delta)$ is a weighted sum of the piecewise-linear $\hat{\mathbf{y}}_{T-1}(\cdot)$:

$$\begin{aligned}\hat{\beta}_{T-1}(\delta_k) &= (\Phi_{T-1}^\mathsf{T}\Phi_{T-1})^{-1}\Phi_{T-1}^\mathsf{T}\hat{\mathbf{y}}_{T-1}(\delta_k),\\ &= \mathbf{w}_{T-1}^\mathsf{T}\cdot\hat{\mathbf{y}}_{T-1}(\delta).\end{aligned}$$

Here, $\mathbf{w}_{T-1}$ is an $1\times N$ vector that depends on s and on the data, but does not depend on δ. Elements of $\mathbf{w}_{T-1}$ can be positive or negative, depending on the feature representation used and the particular data set on hand. Therefore, although each element of $\hat{\mathbf{y}}_{T-1}(\cdot)$ is a convex, piecewise-linear function, the $\hat{\beta}_t(\cdot)$, and therefore the $\hat{Q}_t(s_t,a_t,\cdot)$ may not be convex for $t<T$. One consequence of this non-convexity is that both the algorithm by Barrett and Narayanan (2008), as well as important algorithms from the POMDP literature (e.g., Pineau et al., 2003) that operate on convex piecewise-linear value functions, cannot represent the function $\hat{Q}_t(s_t,a_t,\cdot)$ for $t<T$.

4.2.5 COMPLEXITY OF $\hat{Q}_t(s_t,a_t,\cdot)$ AND $\hat{V}_t(s_t,\cdot)$

Suppose there are N trajectories and $|\mathcal{A}|$ actions at each time point. For any fixed s_T and a_T, the final learned Q-function $\hat{Q}_T(s_T,a_T,\cdot)$ has two knots, one at $\delta=0$ and one at $\delta=1$. The terminal value function $\hat{V}_T(s_T^i,\cdot)$ is constructed at each of N points in state space by applying Algorithm 2 to the

$\hat{Q}_T(s_T^i, a, \cdot)$ for each observed terminal state $s_T^1, s_T^2, ..., s_T^N$ in $\mathcal{D}$. Each resulting $\hat{V}_T(s_T^i, \cdot)$ has at most $|\mathcal{A}| - 1$ new internal knots, and therefore each has at most $(|\mathcal{A}| - 1) + 2$ knots in total. To compute $\hat{Q}_{T-1}(s_{T-1}, a, \cdot)$, we use regression with targets constructed from the N value function estimates $\hat{V}_T(s_T^i, \cdot)$. In general, the knots for each $\hat{V}_T(s_T^i, \cdot)$ will be unique. Thus each $\hat{Q}_{T-1}(s_{T-1}, a, \cdot)$, whose knots are the union of the knots of the $\hat{V}_T(s_T^i, \cdot)$, will have at most $N \cdot (|\mathcal{A}| - 1) + 2$ knots. Computing $\hat{V}_{T-1}(s_{T-1}^i, \cdot)$ using Algorithm 2 adds at most $|\mathcal{A}| - 1$ new knots between each pair of knots in the union, for a total of $(|\mathcal{A}| - 1|)(N \cdot (|\mathcal{A}| - 1) + 1) + 2$ knots. In general, $\hat{Q}_t(s, a, \cdot)$ may have up to $O(N^{T-t}|\mathcal{A}|^{T-t})$ knots, and $\hat{V}_t(s, \cdot)$ may have up to $O(N^{T-t}|\mathcal{A}|^{(T-t)+1})$ knots. To compute the expectation described in Section 4.2.2 at time t, our approach requires $O(N^{T-t}|\mathcal{A}|^{(T-t)+1})$ for each trajectory, for a total of $O(N^{(T-t)+1}|\mathcal{A}|^{(T-t)+1})$ time.

4.3 Optimal Value Functions for All Tradeoffs: Linear Function Approximation, $D > 2$ Basis Rewards

We have seen that, for $D = 2$ basis rewards, $\hat{Q}_t(s_t, a_t, \cdot) = \phi_{s_t, a_t}^{\mathsf{T}} \hat{\beta}_t(\cdot)$ is continuous and piecewise-linear, but not convex. This remains true for D reward functions and D tradeoffs $\delta = \delta_{[0]}, ..., \delta_{[D-1]}$, but as D increases, representing $\hat{Q}_t(s_t, a_t, \cdot)$ becomes more difficult. In the general case, $\hat{Q}_t(s_t, a_t, \cdot)$ and $\hat{\beta}_t(\cdot)$ are linear over pieces that are convex polytopes within the space of possible preferences. We prove this below and show how this insight can be used to develop representations of $\hat{Q}_t$ and $\hat{V}_t$. As in the $D = 2$ case, we can construct $\hat{Q}_t(s_t, a_t, \cdot)$ and $\hat{V}_t(s_t, \cdot)$ for all $t \leq T$ by taking pointwise maximums and pointwise weighted sums of piecewise-linear functions. All proofs are deferred to Appendix A.

Definition 1 (Linear functions over convex polytopes) *A function $f : \mathbb{R}^D \to \mathbb{R}$ is* linear *over a convex polytope $\mathcal{R} \subseteq \mathbb{R}^D$ if*

$$\exists \mathbf{w} \in \mathbb{R}^D : f(\delta) = \delta^{\mathsf{T}} \mathbf{w} \; \forall \delta \in \mathcal{R}.$$

Since $\mathcal{R}$ is a convex polytope, it can be decomposed into a finite collection of *simplices* $\mathcal{T}_i$, each with $\mathcal{D}$ vertices, such that $\mathcal{R} = \cup_i \mathcal{T}_i$ (Grünbaum, 1967). Each simplex is itself a convex polytope. For a simplex $\mathcal{T}$ with vertices $\delta^1, \delta^2, ..., \delta^D$, the weight vector $\mathbf{w}$ of a linear function $f(\delta) = \delta^{\mathsf{T}} \mathbf{w}$ defined over $\mathcal{T}$ can be computed from the values $y^1, y^2, ..., y^{D-1}$ that f takes on the vertices, together with the locations of the vertices themselves. This is accomplished by solving the system of linear equations for $\mathbf{w}$:

$$\begin{bmatrix} \delta_{[0]}^1 & \delta_{[1]}^1 & \cdots & \delta_{[D-1]}^1 \\ \delta_{[0]}^2 & \delta_{[1]}^2 & \cdots & \delta_{[D-1]}^2 \\ \vdots & \vdots & & \vdots \\ \delta_{[0]}^D & \delta_{[1]}^D & \cdots & \delta_{[D-1]}^D \end{bmatrix} \begin{bmatrix} w_{[0]} \\ w_{[1]} \\ \vdots \\ w_{[D-1]} \end{bmatrix} = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^D \end{bmatrix}. \tag{6}$$

Thus, a linear function over a convex polytope can be represented as a *piecewise-linear function* over simplices.[9]

Definition 2 (piecewise-linear functions over collections of convex polytopes)
A function $f : \mathbb{R}^D \to \mathbb{R}$ is piecewise-linear *over a collection $\mathcal{C}$ of convex polytopes if*

$$\forall \mathcal{R} \in \mathcal{C} \; \exists \mathbf{w}_{\mathcal{R}} \in \mathbb{R}^D : f(\delta) = \delta^{\mathsf{T}} \mathbf{w}_{\mathcal{R}} \; \forall \delta \in \mathcal{R}.$$

9. Equation (6) has a unique solution only if the determinant of the matrix in the equation is non-zero, that is, only if there are no collinearities in the vertices of $\mathcal{T}$.

Algorithm 4 Algorithm sketch for max

Identify the convex polytopes $\mathcal{R}_i$ where f_i is maximal. Each $\mathcal{R}_i$ is the intersection of $|\mathcal{A}_t|$ half-spaces.
Decompose each $\mathcal{R}_i$ into simplices
Evaluate $f_{\max}$ at each vertex in each resulting simplex
Recover the $\mathbf{w}$s as needed using Equation (6)

Algorithm 5 Algorithm sketch for sum

Identify the convex polytopes of the form $\mathcal{U} \cap \mathcal{V}$ over which the sum is linear
Decompose each of these polytopes into simplices
Evaluate f_{sum} at each vertex in each resulting simplex
Recover the $\mathbf{w}$s as needed using Equation (6).

Thus we can completely represent a piecewise-linear function as a collection of $(\mathcal{R}, \mathbf{w}_{\mathcal{R}})$ pairs.

Lemma 3 (Max of linear functions) *Given a set of functions $\{f_1, f_2, ..., f_N\}$ that are all linear over the same convex polytope $\mathcal{R}$, the function*

$$f_{\max}(\delta) = \max(f_1(\delta), f_2(\delta), ..., f_N(\delta))$$

is piecewise-linear over the collection of convex polytopes $\mathcal{C} = \{\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_N\}$ given by

$$\mathcal{R}_i = \mathcal{R} \cap \bigcap_{j=1}^{N} \left\{\delta \in \mathbb{R}^D : f_i(\delta) \geq f_j(\delta)\right\},\ i = 1..N.$$

Note further that $\bigcup_{i=1}^{N} \mathcal{R}_i = \mathcal{R}$, and that the function $f_{\max}$ is convex (and therefore continuous) on $\mathcal{R}$, since each f_i is convex. These properties immediately suggest a strategy for computing a representation of $f_{\max}$ described in Algorithm 4. Figure 4 gives a pictorial representation of this strategy, which allows us to perform the max-over-actions portion of a value iteration backup for $D > 2$. We now address how to perform the weighted-sum-over-states portion of the backup.

Lemma 4 *[Sum of piecewise-linear functions] Given two functions, g_1 which is piecewise-linear over a collection $\mathcal{C}_1$ of polytopes, and g_2 which is piecewise-linear over a collection $\mathcal{C}_2$ of polytopes, their linear combination*

$$f_{\text{sum}}(\delta) = \alpha_1 g_1(\delta) + \alpha_2 g_2(\delta)$$

is piecewise-linear over the collection

$$\mathcal{C}_{1+2} = \{\mathcal{U} \cap \mathcal{V}\},\ \text{s.t. } \mathcal{U} \in \mathcal{C}_1,\ \mathcal{V} \in \mathcal{C}_2.$$

This property suggests a strategy for representing f_{sum} described in Algorithm 5. Figure 5 gives a pictorial representation of this strategy, which allows us to perform the weighted-sum-over-states portion of a value iteration backup for $D > 2$.

We can now use these strategies to construct a complete algorithm for the $D > 2$ case. At time T, we have

$$\begin{aligned}\hat{Q}_T(s_T, a_T, \delta) &= \phi_{s_T, a_T}^{\mathsf{T}} \left(\delta_{[0]} \cdot \hat{\beta}_{t[0]} + \delta_{[1]} \cdot \hat{\beta}_{t[1]} + ... + \delta_{[D-1]} \cdot \hat{\beta}_{T(D-1)}\right), \\ &= \delta^{\mathsf{T}} \mathbf{w}_T,\end{aligned}$$

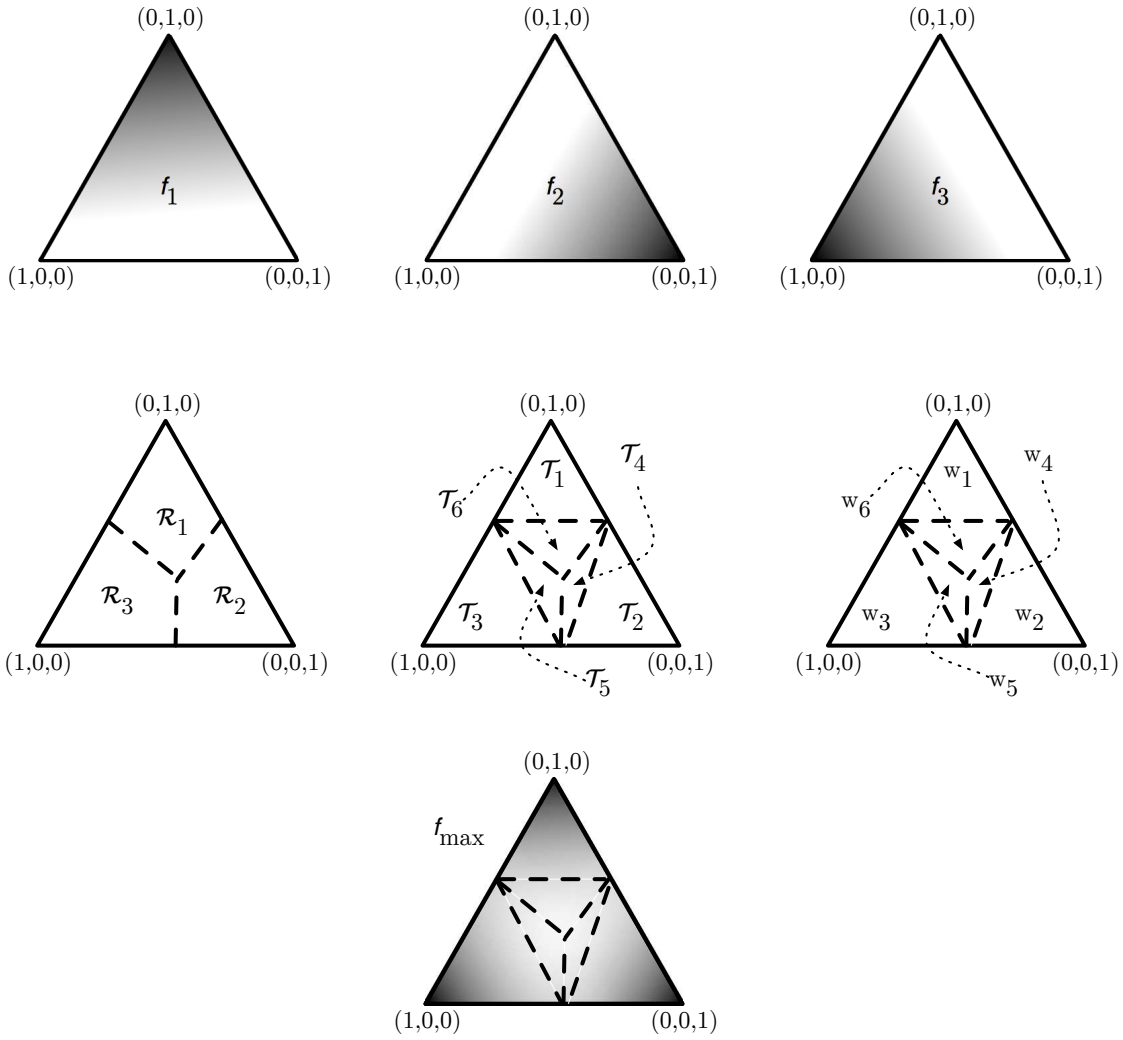


Figure 4: Pictorial representation of taking max of linear functions for $D=3$ basis rewards. The first row of triangles represents three linear functions f_1, f_2, and f_3; darker shading indicates higher function values. The second row shows the convex polytopes $\mathcal{R}_i$ over which f_i is maximal, the decomposition of each of these polytopes into simplices $\mathcal{T}_i$, and their corresponding weight vectors w_i. The continuous piecewise-linear function $f_{\max}$ is shown at the bottom.

where

$$\hat{\beta}_{T[d]} = (\Phi_T^\mathsf{T}\Phi_T)^{-1}\Phi_T^\mathsf{T}\mathbf{r}_{T[d]},$$
$$w_{T[d]} = \phi_{s_T,a_T}^\mathsf{T}\hat{\beta}_{T[d]}.$$

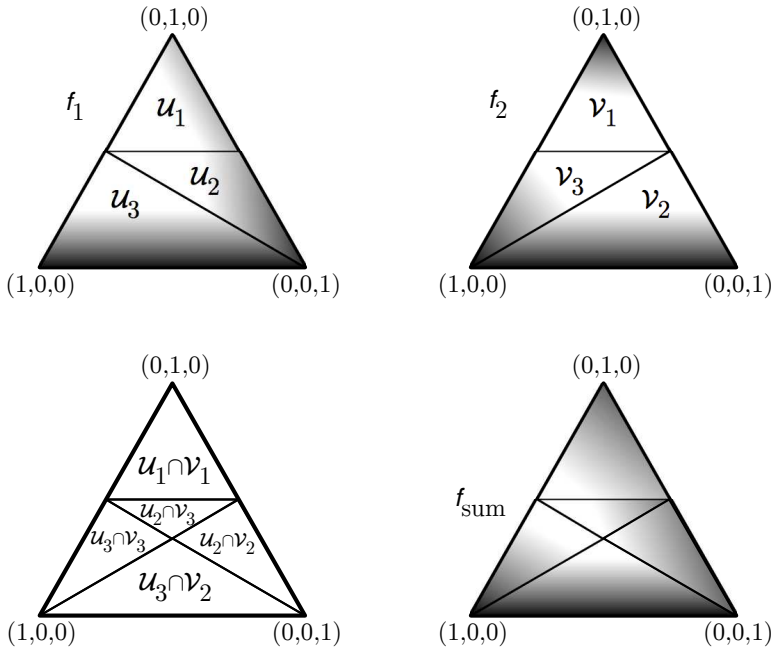


Figure 5: Pictorial representation of taking sum of piecewise-linear functions for $D = 3$ basis rewards. The top row shows the two functions to be added; darker shading indicates higher function values. In the second row, the left diagram shows the pieces over which the sum is linear. The right diagram of the second row shows the resulting continuous piecewise-linear function.

Thus for any s_T, a_T, the function $\hat{Q}_T(s_T, a_T, \cdot) = \phi_{s_T,a_T}^{\mathsf{T}} \hat{\beta}_T(\cdot)$ is linear over the piece $\mathcal{R} = \{\delta : \delta_{[d]} > 0\} \cap \{\delta : \sum_d \delta_{[d]} = 1\}$, which is the unit $(D-1)$-simplex. This is because each element of $\hat{\beta}_{T[d]}(\cdot)$ is linear in δ. It follows that $\hat{V}_T(s_T, \cdot) = \max_a \hat{Q}_T(s_T, a, \cdot)$ is piecewise-linear over the sets described in Lemma 3. To represent the stage T value functions $\hat{V}_T(s_T, \cdot)$, we apply Algorithm 4 to the Q-functions $\hat{Q}_T(s_T, a, \cdot)$ of each action a for each s_T in our data set. Given this value function at time T, we can compute $\hat{Q}_{T-1}(\cdot, \cdot, \hat{\beta}_{T-1}(\delta))$ by computing each element of $\hat{\beta}_{T-1}(\delta)$ as the weighted sum of $\hat{V}_T(s_T, \cdot)$ evaluated at the points s_T in our data set by repeated application of Algorithm 5. As in the $D = 2$ case, these weights are given by the columns of the matrix $(\Phi_T^{\mathsf{T}} \Phi_T)^{-1} \Phi_T^{\mathsf{T}}$. At this point, note that for any s_{T-1}, a_{T-1}, the function $\hat{Q}_{T-1}(s_{T-1}, a_{T-1}, \cdot)$ is piecewise-linear over the *same pieces*—they are the pieces identified in Lemma 4. Thus to compute $\hat{V}_{T-1}$ we can simply apply Algorithm 4 to each of these pieces. Backups to earlier timepoints proceed analogously.

4.3.1 COMPLEXITY

Note that the primitive operations required for fitted-Q iteration—pointwise max and pointwise weighted sum—are precisely the same as in the simpler settings discussed earlier, but the functions we are operating on are $(D-1)$-dimensional.

Suppose there are N trajectories and $|\mathcal{A}|$ actions at each time point. For any fixed s_T and a_T, the final learned Q-function $\hat{Q}_T(s_T, a_T, \cdot)$ has 1 piece $\mathcal{R}_1$ corresponding to the unit $(D-1)$-simplex. The terminal value function $\hat{V}_T(s_T^i, \cdot)$ is constructed at each of N points in state space by applying Algorithm 4 to the $\hat{Q}_T(s_T^i, a_T, \cdot)$ for each observed terminal state $s_T^1, s_T^2, ..., s_T^N$ in $\mathcal{D}$ and each action a_T. Each resulting $\hat{V}_T(s_T^i, \cdot)$ has at most $|\mathcal{A}|$ pieces $\mathcal{R}_1, ..., \mathcal{R}_{|\mathcal{A}|}$, supposing each action has a piece where it is optimal. To compute $\hat{Q}_{T-1}(s_{T-1}, a, \cdot)$, we use regression with targets constructed from the N value function estimates $\hat{V}_T(s_T^i, \cdot)$. In general, the pieces for each $\hat{V}_T(s_T^i, \cdot)$ may be unique. Thus each $\hat{Q}_{T-1}(s_{T-1}, a, \cdot)$ has pieces formed from all possible intersections between pieces of the N different $\hat{V}_T(s_T^i, \cdot)$, so there may be up to $|\mathcal{A}|^N$ such pieces. Applying Algorithm 4 again within each of these pieces means that each $\hat{V}_{T-1}(s_{T-1}^i, \cdot)$ may have $|\mathcal{A}|^{N+1}$ pieces. In general, $\hat{Q}_t(s, a, \cdot)$ may have up to $O(|\mathcal{A}|^{\sum_{i=1}^{T-t} N^i})$ pieces, and $\hat{V}_t(s, \cdot)$ may have up to $O(|\mathcal{A}|^{\sum_{i=0}^{T-t} N^i})$ pieces.

A more detailed complexity analysis would depend on how the pieces are represented, and on how Algorithms 4 and 5 are implemented using computational geometry primitives—we have already seen that for $D = 2$ basis rewards we can do much better than this worst-case bound. Intuitively this is because most of the intersections between pieces of the N different $\hat{V}_T(s_T^i, \cdot)$ are in fact empty. A general treatment of implementing Algorithms 4 and 5 is beyond the scope of this paper; however, we now present a detailed algorithm designed for the $D = 3$ case that is also much less computationally intensive than the above double-exponential bound suggests.

4.4 Optimal Value Functions for All Tradeoffs: Linear Function Approximation, $D = 3$ Basis Rewards

We now consider the $D = 3$ case specifically. The first "algorithmic trick" we will use is to represent functions of δ using two rather than three dimensions, that is,

$$r_t(s_t, a_t, \delta_{[0]}, \delta_{[1]}) = \delta_{[0]} r_{t[0]}(s_t, a_t) + \delta_{[1]} r_{t[1]}(s_t, a_t) + (1 - \delta_{[0]} - \delta_{[1]}) r_{t[2]}(s_t, a_t).$$

This follows from the constraint that $\sum_i \delta_{[i]} = 1$. Note that the set of points $(\delta_{[0]}, \delta_{[1]}) : \delta_{[0]} + \delta_{[1]} \leq 1,\ \delta_{[0]} \geq 0,\ \delta_{[1]} \geq 0$ is a convex polytope in $\mathbb{R}^2$. In fact it is a simplex, and therefore we can represent the linear function $Q_T(s_T, a_T, \cdot)$ by storing the corners of the simplex $\mathcal{T} = [(1,0)\ (0,1)\ (0,0)]$ together with the parameter vectors

$$\hat{\beta}_T(1,0) = (\Phi_T^\mathsf{T} \Phi_T)^{-1} \Phi_T^\mathsf{T} \mathbf{r}_{t[0]},$$
$$\hat{\beta}_T(0,1) = (\Phi_T^\mathsf{T} \Phi_T)^{-1} \Phi_T^\mathsf{T} \mathbf{r}_{t[1]},$$
$$\hat{\beta}_T(0,0) = (\Phi_T^\mathsf{T} \Phi_T)^{-1} \Phi_T^\mathsf{T} \mathbf{r}_{t[2]}.$$

We can compute a weight-vector representation of the function using Equation (6).

Consider two linear functions $\hat{Q}_T(s_T, 1, \cdot)$ and $\hat{Q}_T(s_T, 2, \cdot)$ over $\mathcal{T}$. To take their pointwise maximum, we must identify the pieces over which the maximum is linear, as described in Lemma 3. The boundary of these two pieces is a line in $\mathbb{R}^2$. If this line intersects $\mathcal{T}$, it will divide $\mathcal{T}$ into the two pieces. If it does not, then one function must be greater than the other over all of $\mathcal{T}$. Identifying the pieces can be accomplished by finding where (if anywhere) the dividing line given by $\hat{Q}_T(s_T, 1, \cdot) = \hat{Q}_T(s_T, 2, \cdot)$ intersects $\mathcal{T}$; this is illustrated in Figure 6. We represent $\hat{V}_T(s_T, \cdot)$ by recording the pieces $\mathcal{R}$ on either side of the dividing line. Each piece is identified by a set of vertices, along with the value of the max at each vertex. (Note that certain vertices will belong to both

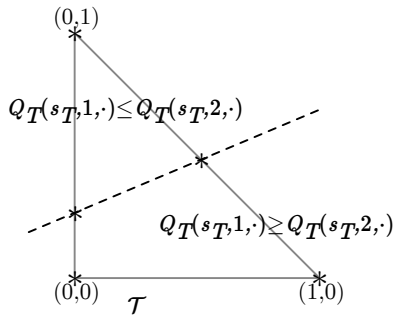


Figure 6: Identifying the pieces over which the max of two linear functions is linear.

pieces.) If there are more than 2 actions, we can take further maxes over each identified sub-piece, partitioning as necessary. This completes the max-over-actions step at time T.

To compute $\hat{Q}_{T-1}(s_{T-1}, a_{T-1}, \cdot)$, we compute each element of $\hat{\beta}_{T-1}(\cdot)$ at each vertex δ by taking a weighted sum over next states of $\hat{V}_T(s_T, \cdot)$, again with weights given by columns of $(\Phi_T^\intercal \Phi_T)^{-1}\Phi_T^\intercal$. From Lemma 4 we know that we need to identify all of the pieces formed by intersecting the linear pieces of the functions to be summed. Naïvely, one might compute the intersection of all pairs of pieces, but for $D=3$ basis rewards we can instead use a *constrained Delaunay triangulation* (CDT), which essentially gives us only the non-empty intersections and does so much more efficiently than enumerating all pairs. Figure 7 gives a schematic diagram of this procedure. The input to a standard Delaunay triangulation algorithm is a list of points in space. The output is a list of simplices (in this case triangles) that partition space and whose vertices come from the input points. The particular triangles chosen satisfy certain properties (de Berg et al., 2008), but the main appeal for our purposes is the algorithm's $O(n \log n)$ running time (Chew, 1987), where n is the number of points to be triangulated. A constrained version of the algorithm allows us to additionally specify edges between points that must be present in the output. The constrained version of the algorithm will add points as needed to satisfy this requirement; again Figure 7 illustrates this. The simplices (triangles) will form the pieces for the elements of $\hat{\beta}_{T-1}(\cdot)$, which will define our estimates $\hat{Q}_{T-1}$.

The output of the CDT algorithm is a set of pieces over which we know the sum of the piecewise-linear functions will be linear. There are in fact more pieces than are strictly necessary, because linear pieces that have more than three vertices (e.g., quadrilaterals) are divided up by the algorithm. Nonetheless, the output is convenient because we can determine the weight vector $\mathbf{w}$ for any simplex using Equation (6). Once we have determined these pieces and vertices, we evaluate $\hat{V}_T(s_T, \cdot)$ at each terminal state and each vertex. Each element of $\hat{\beta}_{T-1}(\cdot)$ is a piecewise-linear function whose pieces are given by the CDT algorithm, and whose values are given by the appropriate weighted sum of $\hat{V}_T(s_T, \cdot)$ evaluated at the vertices. This gives $\hat{Q}_{T-1}$. The max operation to obtain $\hat{V}_{T-1}$ can again be achieved by taking the max over each piece of $\hat{Q}_{T-1}$, and so on backward to $t = 1$. A complete description is given in Algorithm 6

The problem of finding intersections between lines and computing triangulations is well-studied in the field of computational geometry (de Berg et al., 2008). Though these problems may appear trivial, it is very important to avoid situations where there is "ill-conditioning." For example, if we were to use floating point arithmetic to define three lines that should intersect at the same point,

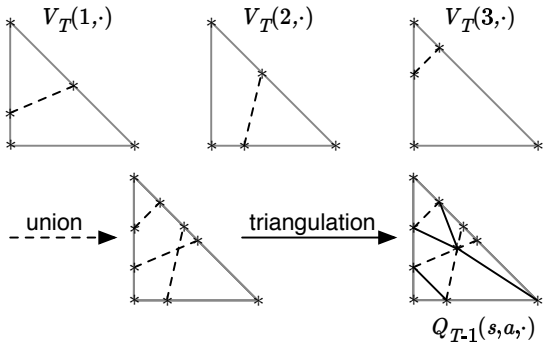


Figure 7: Computing the sum of three piecewise-linear functions. The three example value functions each have two linear pieces. The boundary between two pieces is shown by a dotted line. We take all of the vertices, plus the boundaries, and give them as input to a constrained Delaunay triangulation procedure. The output is shown.

we may find that the "intersection point" is different depending on which pair of lines is used to compute it. This can lead to many spurious points and edges being generated as we proceed with value iteration. We take advantage of CGAL, the Computational Geometry Algorithms Library (CGAL, 2011), which is designed specifically to avoid these problems.

Algorithm 6 Value Backup - Linear Function Approximation, $D=3$ Basis Rewards

$\forall(s,\delta)$, $\hat{V}_{T+1}(s,\delta) \triangleq 0$. $\forall s$, $\Delta(\hat{V}_{T+1}(s,\cdot)) \triangleq \{[(1,0),(0,1),(0,0)]\}$.
for $t = T$ **downto** 1 **do**
  $\Delta^{\hat{Q}_t} \leftarrow \{\}$
  for all $(s_t, a_t, s_{t+1}) \in \mathcal{D}$ **do**
    $\Delta^{\hat{Q}_t} \leftarrow \Delta^{\hat{Q}_t} \cup \Delta(\hat{V}_{t+1}(s_{t+1},\cdot))$
  end for
  $\Delta^{\hat{Q}_t} \leftarrow$ constrained_Delaunay_Triangulation$(\Delta^{\hat{Q}_t})$
  for all $\delta \in$ vertices$(\Delta^{\hat{Q}_t})$ **do**
    $\mathbf{y}_t(\delta) = \delta_{[0]}\mathbf{r}_{t[0]} + \delta_{[1]}\mathbf{r}_{t[1]} + (1-\delta_{[0]}-\delta_{[1]})\mathbf{r}_{t[2]} + \hat{\mathbf{v}}_{t+1}(\delta)$
    $\hat{\beta}_t(\delta) = (\Phi_t^{\mathsf{T}}\Phi_t)^{-1}\Phi_t^{\mathsf{T}}\mathbf{y}_t(\delta)$
  end for
  for all $s_t \in \mathcal{D}$ **do**
    Compute $\Delta(\hat{V}_t(s_t,\cdot))$ by Algorithm 4
  end for
end for

4.4.1 COMPLEXITY FOR $D = 3$

Any triangulation of n points in the plane contains $O(n)$ triangles (Brass, 2005), so the operation $\Delta^{\hat{Q}_t} \leftarrow$ constrained_Delaunay_Triangulation$(\Delta^{\hat{Q}_t})$ increases the size of $\Delta^{\hat{Q}_t}$ only linearly. It follows

that each $\hat{Q}_{T-1}(s_{T-1}, a, \cdot)$ has $O(N \cdot |\mathcal{A}|)$ pieces rather than the $|\mathcal{A}|^N$ given by the worst case analysis in Section 4.3.1. Therefore $\hat{Q}_t(s, a, \cdot)$ may have up to $O(N^{T-t}|\mathcal{A}|^{T-t})$ pieces, and $\hat{V}_t(s_t, \cdot)$ may have up to $O(N^{T-t}|\mathcal{A}|^{(T-t)+1})$ pieces. Note that these rates are the same as for the $D = 2$ special case discussed in Section 4.2.5. Intuitively this is because the triangulation of n points in d-dimensional space has $O(n^{\lceil d/2 \rceil})$, triangles (Brass, 2005), that is, the same asymptotic growth rate for $D = 2$ (one-dimensional preference space) and $D = 3$ (two-dimensional preference space).

5. Dominated Actions

The ability to compute $\hat{Q}$ and $\hat{V}$ for all preferences achieves our goal of informing the decision maker about the quality of available actions under different preferences, and of informing the decision maker about how the recommended policy changes with preference. In addition to achieving these primary goals, our representations of $\hat{Q}$ and $\hat{V}$ allow us to compute whether or not an action is *dominated* (not optimal for a given state no matter what the preference) and whether it is *globally dominated* (not optimal for *any* state-preference pair.) The notion of domination arises in POMDP planning as well, where certain actions may not be optimal for any belief state, but the notion of global domination has no direct analog in the POMDP setting since it is a property of additional observed state s that is not part of a typical POMDP.

The concept of domination is central to the field of multi-criterion optimization (Ehrgott, 2005), and is important in the medical decision making setting because it identifies treatment actions that are not appropriate no matter what the decision maker's preference is. One may also consider actions that are dominated for all patient states in a population of interest, that is, the actions that are globally dominated. Knowing this set of actions would be useful for developing a *formulary*—a list of treatments that should generally be made available to a patient population.

The general problem of analytically identifying the set of globally dominated actions is difficult, as we will illustrate, but we first provide solutions for low-dimensional problems and discuss the identification of globally dominated actions in higher dimensional settings. Our approach for determining if actions are dominated is to look for *certificates* of **non**-domination for each action. A point (s_t, a_t, δ) where $\hat{Q}_t(s_t, a_t, \delta) = \hat{V}_t(s_t, \delta)$ is a certificate that action a_t is not dominated at state s_t, and that action a_t is therefore not globally dominated.[10] All proofs are deferred to Appendix A.

5.1 Finite Case, $D = 2$ Basis Rewards

We showed in Section 4.1 how to exactly represent the $Q_t(s_t, a_t, \cdot)$ and $V_t(s_t, \cdot)$ functions for all s_t and a_t for $D = 2$ when the state space is finite by representing them as lists of knots (vertices) and knot-values (vertex-values). In addition to storing this information, we may also store the set of actions that are optimal at each knot, that is, we may store $\mathcal{A}^*(\delta) = \{a_t^* : Q_t(s_t, a_t^*, \delta) = V_t(s_t, \delta)\}$ for each δ in the knot list of $V_t(s_t, \cdot)$. Note that $\mathcal{A}^*(\delta)$ may contain more than one action. Suppose δ_k and δ_{k+1} are adjacent knots in $\Delta(V_t(s_t, \cdot))$. For all δ s.t. $\delta_k < \delta < \delta_{k+1}$, we have $\mathcal{A}^*(\delta) = \mathcal{A}^*(\delta_k) \cap \mathcal{A}^*(\delta_{k+1})$. Thus the set of actions that have a non-domination certificate at state s_t is given by

$$\bigcup_{k=1}^{|\Delta(V_t(s_t,\cdot))|} \mathcal{A}^*(\delta_k),$$

10. Note that in this work we determine which actions are *estimated* to be dominated, since we are using estimates $\hat{Q}_t$ and $\hat{V}_t$ to make this determination. Assessing our confidence that an action is truly non-dominated based on available data will require incorporation of appropriate statistical methods (Laber et al., 2009).

and any actions not in the above union are dominated at s_t. Note that recording this additional information does not increase the time complexity of the method. It also allows us to find every globally dominated action by computing the above union at each finite state, taking the union of those sets, and identifying actions not present in the union.

5.2 Regression Case, $D=2$ Basis Rewards, One State Feature

We now show how to identify all of the globally dominated actions in the linear function approximation setting. We first discuss the case with a single state feature ψ_{s_T}, $D=2$ basis rewards, and the last timepoint T. We also construct feature vectors ϕ_{s_T,a_T} so that the $\hat{Q}_t$ functions are built using separate regressions for each action; for example see (5). We can then define $\hat{\beta}_t^a(\delta)$ to be the 2×1 sub-vector of $\hat{\beta}_t(\delta)$ that aligns with the sub-vector of ϕ_{s_t,a_t} that is non-zero for $a_t = a$. We also define the matrix $B_T^a = \left[\begin{array}{cc} \hat{\beta}_T^a(0) & \hat{\beta}_T^a(1) \end{array}\right]$ for each action, so that

$$\hat{Q}_T(s_T, a_T, \delta) = \left[\begin{array}{cc} 1 & \psi_{s_T} \end{array}\right] B_T^a \left[\begin{array}{c} \delta \\ 1-\delta \end{array}\right]. \tag{7}$$

To find the globally dominated actions, we will search for certificates of non-domination in (ψ_{s_T}, δ) space and identify the actions that do not have a certificate. Figure 3 shows an example of this setting. In the example, actions 1, 4, 6, 7, 8, 9, and 10 have regions where they are optimal, and hence certificates of non-domination. Actions 2, 3, and 5 have no certificates, that is, they are not optimal for any combination of ψ_{s_T} and δ.

Each B_T^a is a constant given the data and the regression algorithm. The form of (7) clearly shows that $\hat{Q}_T(\cdot, a, \cdot)$ is a bilinear function of ψ_{s_T} and δ. To analytically identify the set of dominated actions, we analyze the boundaries between the regions where one action has higher value than another. These boundaries occur where $Q_T(\cdot, a_1, \cdot) = Q_T(\cdot, a_2, \cdot)$ for some actions a_1 and a_2, that is, where

$$\left[\begin{array}{cc} 1 & \psi_{s_T} \end{array}\right] B_T^{a_1} \left[\begin{array}{c} \delta \\ 1-\delta \end{array}\right] = \left[\begin{array}{cc} 1 & \psi_{s_T} \end{array}\right] B_T^{a_2} \left[\begin{array}{c} \delta \\ 1-\delta \end{array}\right],$$

which describes the hyperbola in δ and ψ_{s_T} given by

$$\left[\begin{array}{cc} 1 & \psi_{s_T} \end{array}\right] (B_T^{a_1} - B_T^{a_2}) \left[\begin{array}{c} \delta \\ 1-\delta \end{array}\right] = 0. \tag{8}$$

Along these boundaries, "triple-points" occur at (ψ_{s_T}, δ) points where three or more actions have exactly the same value. At these points, either all of the actions involved are optimal, or none of them are. We now show that if there exists a certificate of non-domination for action a, but there exists no certificate for a on the boundary of the domain of $V_T(s_T, \delta)$, then there exists a certificate for a at a triple-point.

Lemma 5 (Lizotte et al., 2010) *If action a is optimal at time T for some point (ψ_{s_T}, δ) but is not optimal for any (ψ_{s_T}, δ) on the boundary of the domain, then a is optimal for some (ψ_{s_T}, δ) that is a triple-point.*

From Lemma 5 we know that to find all actions that are optimal for some (ψ_{s_T}, δ) we need only check the boundaries and the triple points. The boundaries can be checked using Algorithm 2.

(Note that because $\hat{Q}_T(\cdot, a, \cdot)$ is bilinear in δ and in ψ_{s_T}, we can also use Algorithm 2 to identify for any fixed δ the actions that are optimal for some ψ_{s_T}.) We can then enumerate the $\binom{|\mathcal{A}|}{3}$ triple-points and check them to detect any regions that do not intersect the boundary of the domain, like for example the region where action 1 is optimal in Figure 3 where we have identified the triple-points with white dots. This procedure reveals all actions that are optimal for some (ψ_{s_T}, δ), and thereby identifies any actions that are not optimal for any (ψ_{s_T}, δ).

To compute the triple points, we must solve the following system of bilinear equations for ψ_{s_T} and δ:

$$\begin{bmatrix} 1 & \psi_{s_T} \end{bmatrix} (B_T^{a_1} - B_T^{a_2}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix} = 0,$$
$$\begin{bmatrix} 1 & \psi_{s_T} \end{bmatrix} (B_T^{a_1} - B_T^{a_3}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix} = 0.$$

There are many ways of interpreting this system of equations; it describes the intersection of two hyperbolas, as pointed out in our earlier work (Lizotte et al., 2010). We describe a more concise solution here. Note that any solution (ψ_{s_T}, δ) must have the property that the vector $[1 \ \psi_{s_T}]$ is orthogonal to the two vectors given by $(B_T^{a_1} - B_T^{a_2}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix}$ and $(B_T^{a_1} - B_T^{a_3}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix}$. Since $[1 \ \psi_{s_T}]$ is two-dimensional, this implies that these two vectors are collinear. Therefore the vector $\begin{bmatrix} \delta \\ 1-\delta \end{bmatrix}$ must satisfy

$$(B_T^{a_1} - B_T^{a_2}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix} = \lambda (B_T^{a_1} - B_T^{a_3}) \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix}. \tag{9}$$

Equation (9) describes the *generalized eigenvalue problem* (Golub and Van Loan, 1996). Common software packages can solve for λ and δ.[11] We have described the process of identifying globally dominated actions for bilinear functions; we can immediately extend this algorithm for piecewise bilinear functions by applying it between pairs of knots.

5.3 Regression Case, p State Features, Arbitrary D

Lizotte et al. (2010) conjectured that an analogue of Lemma 5 holds in higher dimensions, and that identifying all globally dominated actions for more state variables and/or reward functions would require computing intersections of surfaces in higher dimensions. We refine this conjecture, and we propose a solution method for finding globally dominated actions for

$$\hat{Q}_T(\psi_{s_T}, a, \delta) = \begin{bmatrix} 1 & \psi_{s_T}^{\mathsf{T}} \end{bmatrix} B_T^a \, \delta.$$

where ψ_{s_T} is p-dimensional, and δ is the D-dimensional vector of preferences defined as usual. We consider finding non-dominated actions over a "domain of interest," of the form $\mathcal{S} \times \mathcal{D}$, where $\mathcal{S}$ is a rectangle in $\mathbb{R}^p$, and $\mathcal{D}$ is a convex subset of valid preferences. To prove our method is correct, we require the following conjecture.

11. In practice one solves $(B_T^{a_1} - B_T^{a_2})x = \lambda(B_T^{a_1} - B_T^{a_3})x$ and then projects x onto the subspace $x_{[1]} = 1 - x_{[0]}$ by dividing it by $x_{[0]} + x_{[1]}$.

Conjecture 6 *If the system of polynomial equations of the form*

$$\begin{aligned}
\begin{bmatrix} 1 & \psi_{s_T}^{\mathsf{T}} \end{bmatrix} (B_T^{a_1} - B_T^{a_2})\,\delta = 0 \\
\begin{bmatrix} 1 & \psi_{s_T}^{\mathsf{T}} \end{bmatrix} (B_T^{a_1} - B_T^{a_3})\,\delta = 0 \\
\vdots \\
\begin{bmatrix} 1 & \psi_{s_T}^{\mathsf{T}} \end{bmatrix} (B_T^{a_1} - B_T^{a_k})\,\delta = 0
\end{aligned} \tag{10}$$

has a finite number of solution points, those points taken together are a continuous vector-valued function of the coefficients of the system.

Conjecture 6 is true for a single polynomial of one variable over the complex plain (Uherka and Sergott, 1977), and holds for the $D=2$ case. We believe the conjecture holds because it is known that systems of multivariate polynomials can be reduced to solving a collection of independent problems each involving a single polynomial of one variable. This reduction uses the methods of *elimination* and *extension* (Cox et al., 1997).

Proposition 7 *Assume Conjecture 6. If there exists a point (ψ_{s_T}, δ) in the interior of the domain of interest where action a is optimal, but a is not optimal at any point (ψ_{s_T}, δ) where $p+D+1$ actions are simultaneously optimal, then there exists a point (ψ_{s_T}, δ) on the boundary of the domain of interest where action a is optimal.*

We refer to a point where k actions are simultaneously optimal as a "k-tuple point." To find the set of globally non-dominated actions, we first solve (10) and check to see if any of the $(p+D+1)$-tuple points are optimal. If so, all of the point's associated actions not globally dominated. The system (10) of polynomial equations can be solved by computer algebra systems or using numerical approximation techniques (Cox et al., 1997; Sturmfels, 2002). By recursively applying the proposition to the boundaries of the original domain, we can ensure that we identify every action that is not globally dominated: first, we find $(D+p+1)$-tuple points inside the original $(D+p)$-dimensional domain of interest and check whether any of these are optimal. We then treat each of the $(D+p-1)$-dimensional boundaries as our new domains of interest, and look for $(D+p)$-tuple points in each of these, and so on until we check each of the 2^{D+p} zero-dimensional points at the corners of our original domain. Again, we have described the process of identifying globally dominated actions for functions linear in δ; we can immediately extend this algorithm to piecewise-linear functions by applying it within linear regions.

6. Application to Medical Decision Making

An important application of this work is the improvement of the use of sequential medical data for constructing clinical decision support systems. In this section, we briefly discuss how such systems are currently constructed, how preferences are currently addressed in the medical decision making community, and how the methods presented in this paper provide a novel and useful way of incorporating preferences in clinical decision support systems. We then present an example using real data that illustrates how our methods can be used to inform clinical decision making.

6.1 Clinical Decision Support, Evidence-Based Medicine, and Preferences

Currently, most clinical decision support systems are constructed using expert opinion (e.g., Working Group for the Canadian Psychiatric Association and the Canadian Alliance for Research on Schizophrenia, 1998; Miller et al., 1999). Although accumulated clinical experience is invaluable in making good treatment decisions, data-derived scientific evidence is playing an increasingly prominent role. Sackett (1996) state that "The practice of evidence based medicine means integrating individual clinical expertise with the best available external clinical evidence from systematic research."

In order to be effective, any evidence-based decision support system must leave room for individual clinical expertise to inform the final decision. The methods we have presented are able to do this by presenting treatment recommendations in a way that incorporates decision maker preferences. There is extensive literature on "preference elicitation," both within and outside the field of medical decision making. In the medical decision making field, however, preference elicitation is usually done at the population level and used to produce generic clinical guidelines, rather than to make recommendations tailored to individual patients (e.g., Bonnichsen, 2011; Davis et al., 2011). In other fields, preference elicitation is done before presenting any information about the available treatments (Boutilier, 2002; Thall, 2008). It is assumed that preference elicitation is able to reliably extract the preferences of the decision maker; in our setting, preference elicitation would attempt to find the δ that represents the preference of a decision maker, run fitted-Q iteration using $r_t(s_t, a_t, \delta)$, and recommend a single treatment. This approach leaves no room for individual clinical expertise.

Our methods provide a novel alternative to preference elicitation. Rather than trying to determine which of the uncountable number of possible preferences a user might have, we present, for each available action, the set of preferences for which that action is optimal. That is, we present the policy as a function of preference. We call this approach "inverse preference elicitation" because rather than eliciting a preference and recommending a treatment, we can easily and intuitively show for each treatment the set of preferences that are consistent with its recommendation. By using this approach, the time a user would have spent having his or her preference elicited is now spent directly considering the evidence for how preferences influence recommended treatment.

6.2 Example: CATIE Study

We illustrate inverse preference elicitation using data from the Clinical Antipsychotic Trials of Intervention Effectiveness (CATIE) study. The CATIE study was designed to compare sequences of antipsychotic drug treatments for the care of schizophrenia patients. The full study design is quite complex (Stroup et al., 2003; Swartz et al., 2003); we have therefore chosen a simplified subset of the CATIE data in order to more clearly illustrate the potential of the methods presented in this paper. CATIE was an 18-month study that was divided into two main phases of treatment. Upon entry into the study, most patients began "Phase 1," in which they were randomized to one of five possible treatments with equal probability: olanzapine, risperidone, quetiapine, ziprasidone, or perphenazine. As they progressed through the study, patients were given the opportunity at each monthly visit to discontinue their Phase 1 treatment and begin "Phase 2" on a new treatment. The set of possible Phase 2 treatments depended on the reason for discontinuing Phase 1 treatment. If the Phase 1 treatment was deemed to be ineffective at reducing symptoms, then their Phase 2 treatment was chosen randomly as follows: clozapine with probability $1/2$, or uniformly randomly from the set {olanzapine, risperidone, quetiapine} with probability $1/2$. If the Phase 1 treatment was deemed

to produce unacceptable side-effects, their Phase 2 treatment was chosen uniformly randomly from the set {olanzapine, risperidone, quetiapine, ziprasidone}.

In previous work, we used batch off-policy reinforcement learning to analyze data from this study using a single reward function (Shortreed et al., 2010). We now give two new analyses using the new methods we have presented to examine multiple rewards simultaneously. The basis rewards we consider are measures of symptoms, side-effects, and quality of life.

Symptoms: PANSS For our symptom measurement, we use the Positive and Negative Syndrome Scale (PANSS) which is a numerical representation of the level of psychotic symptoms experienced by a patient (Kay et al., 1987). A higher value of PANSS reflects the presence of more severe symptoms. PANSS is a well-established measure that we have used in previous work on the CATIE study (Shortreed et al., 2010), and is measured for each CATIE patient both at the beginning of the study and at several times over the course of the study. Since having larger PANSS is worse, for our first basis reward $r_{[0]}$ we use 100 minus the percentile of a patient's PANSS at the end of their time in the study. We use the distribution of PANSS at the beginning of the study as the reference distribution for the percentile.

Body Weight: BMI Weight gain is an important and problematic side-effect of many antipsychotic drugs (Allison et al., 1999). Patients in the CATIE study had their Body Mass Index (BMI) (National Institutes of Health., 1998) measured at study intake and several times over the course of the study. Since in this population having a larger BMI is worse, for our second basis reward $r_{[1]}$ we use 100 minus the percentile of a patient's BMI at the end of their time in the study. We use the distribution of BMI at the beginning of the study as the reference distribution for the percentile.

Quality of Life: HQLS Measures of quality of life are intended to assess to what degree a patient's disease is impacting his or her daily life, in terms of a patient's relationships with others, ability to work, emotional state, and ability to carry out daily activities (Cramer et al., 2000). Patients in CATIE were administered the Heinrichs-Carpenter Quality of Life (HQLS) (Heinrichs et al., 1984) scale at intake and repeatedly as they progressed through the study. Since having a higher HQLS is better, for our third basis reward $r_{[2]}$ we use the percentile of a patient's HQLS at the end of their time in the study. We use the distribution of HQLS at the beginning of the study as the reference distribution for the percentile.

6.3 Symptoms versus Weight Gain

We begin by presenting the output of our algorithm for $D = 2$, using PANSS as described above for $r_{[0]}$, and BMI for $r_{[1]}$. In Figures 8, 9 and 10 we will present plots of the piecewise linear value function $\hat{V}_t(s_t, \cdot)$ for $t = 1, 2$ and for various representative values of s_t. When we plot $\hat{V}_t(s_t, \delta)$ as a function of δ, we simultaneously show the learned optimal action using the style and colour of the plotted line. Thus from our plots one can see both the learned value and the learned policy as a function of δ, which enables us to easily see for each action the range of preferences for which that action looks best.

6.3.1 PHASE 2 ANALYSES

Following the approach of our previous work (Shortreed et al., 2010), we use PANSS at entry to Phase 2 as a continuous state variable s_2 so that we can allow symptom severity to influence optimal action choice. We convert the PANSS scores at entry to Phase 2 into percentiles just as we did for the PANSS reward signal. Furthermore, we learn value functions for the Phase 2 Efficacy patients and

the Phase 2 Tolerability patients separately, since these two groups have different sets of possible actions.

We have relatively little data for Phase 2 Efficacy subgroup of patients. Therefore for this subgroup, we combine the actions of giving {olanzapine, risperidone, or quetiapine} into one "not-clozapine" action: $\mathcal{A}_2^{\text{EFF}} = \{\text{CLOZ}, \text{not-CLOZ}\}$. The other three drugs are much more similar to each other than they are to clozapine, which is much more toxic and is currently considered a "last resort" for use when symptoms are not effectively managed by other treatments (McDonagh et al., 2010). The feature vectors we use for Stage 2 Efficacy patients are given by

$$\phi_{s_2,a_2}^{\text{EFF}} = [1,\ 1_{a_2=\text{CLOZ}},\ s_2,\ s_2 \cdot 1_{a_2=\text{CLOZ}},\ 1_{\text{TD}},\ 1_{\text{EX}},\ 1_{\text{ST1}},\ 1_{\text{ST2}},\ 1_{\text{ST3}},\ 1_{\text{ST4}}]^{\mathsf{T}}.$$

Here, s_2 is the PANSS percentile at entry to Phase 2. Feature $1_{a_2=\text{OLAN}}$ is an indicator that the action at the second stage was clozapine, as opposed to one of the other treatments. We also have other features that do not influence the optimal action choice but that are chosen by experts to improve the value estimates.[12] 1_{TD} is an indicator variable of whether the patient has had tardive dyskinesia (a motor-control side-effect), 1_{EX} indicates whether the patient has been recently hospitalized, and 1_{ST1} through 1_{ST4} indicate the type of facility at which the patient is being treated (e.g., hospital, specialist clinic)

For Phase 2 Tolerability patients, the possible actions are $\mathcal{A}_2^{\text{TOL}} = \{\text{OLAN}, \text{QUET}, \text{RISP}, \text{ZIP}\}$, and the feature vectors we use are given by

$$\begin{aligned}\phi_{s_2,a_2}^{\text{TOL}} = [&1,\ 1_{a_2=\text{OLAN}},\ 1_{a_2=\text{QUET}},\ 1_{a_2=\text{RISP}},\ s_2,\ s_2 1_{a_2=\text{OLAN}},\ s_2 1_{a_2=\text{QUET}},\ s_2 1_{a_2=\text{RISP}}, \ldots \\ &1_{\text{TD}},\ 1_{\text{EX}},\ 1_{\text{ST1}},\ 1_{\text{ST2}},\ 1_{\text{ST3}},\ 1_{\text{ST4}}]^{\mathsf{T}}.\end{aligned}$$

Here we have three indicator features for different treatments at Phase 2, $1_{a_2=\text{OLAN}}$, $1_{a_2=\text{RISP}}$, $1_{a_2=\text{QUET}}$, with ziprasidone represented by turing all of these indicators off. Again we include the product of each of these indicators with the PANSS percentile s_2. The remainder of the features are the same as for the Phase 2 Efficacy patients.

Figure 8 shows a plot of the piecewise linear value function $\hat{V}_2(s_2, \cdot)$ for patients who are in Phase 2 of the study because of a lack of efficacy of the Phase 1 treatment. We plot $\hat{V}_2(s_2, \cdot)$ for three fixed values of s_2 corresponding to having low PANSS, moderate PANSS, or high PANSS at entry to Phase 2. (These correspond to setting $s_2 = 25$, $s_2 = 50$ and $s_2 = 75$, respectively.) For all three states shown in the plot, the learned policy indicates that clozapine is the best action for a reward based only on PANSS (i.e., for $\delta = 0$), but not-clozapine (olanzapine or risperidone or quetiapine) is best for a reward based only on BMI (i.e., for $\delta = 1$.) We have indicated the values of δ at which the decision changes from one action to the other by dropping down a dotted line. We see that, except for those with a strong preference for controlling BMI, clozapine appears to be the best choice among patients who found their Phase 1 treatment to be ineffective at controlling symptoms. It is clear from the plot that neither action is globally dominated since neither is dominated at any of our example states.

Figure 9 shows a plot of the piecewise linear value function $\hat{V}_2(s_2, \cdot)$ for patients who are in phase 2 of the study because they could not tolerate the side-effects of their Phase 1 treatment. Again we plot $\hat{V}_2(s_2, \cdot)$ for three different Phase 2 entry percentiles of PANSS: $s_2 = 25$, $s_2 = 50$ and

12. See Section 4.2 by Shortreed et al. (2010) for a more thorough discussion of these kinds of features. When we display value functions and learned policies in our examples, we set all of these indicators to 0 since they are not needed by the learned policy to select actions in the future.

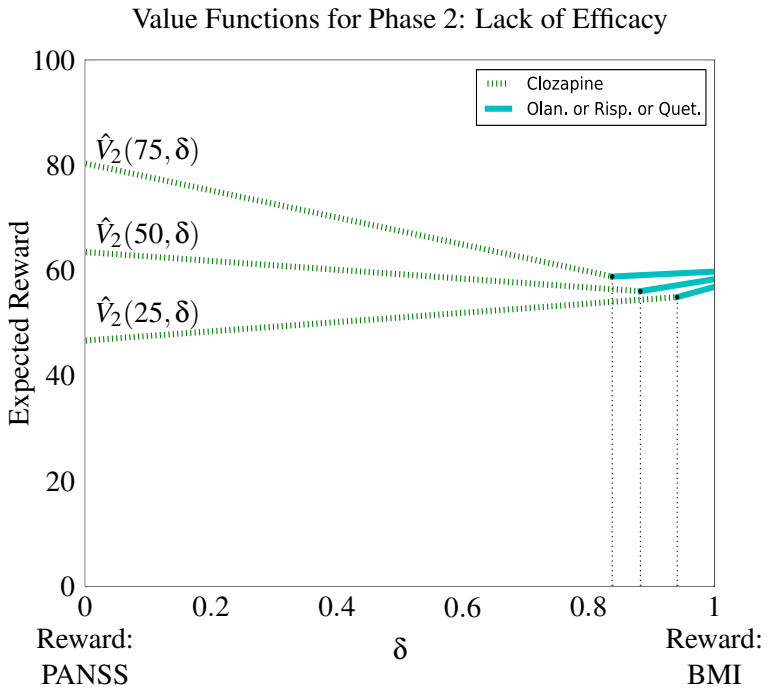


Figure 8: Multiple rewards analysis showing learned value function and associated learned policy for Phase 2 Efficacy patients. Three value functions are shown, with the associated action chosen by the learned policy, for $s_2 = 25$, $s_2 = 50$, and $s_2 = 75$.

$s_2 = 75$. Possible treatments are olanzapine, quetiapine, risperidone and ziprasidone. If we use a reward based only on PANSS (i.e., for $\delta = 0$), the learned policy indicates that olanzapine is the best action for those with high or moderate incoming PANSS, and that risperidone is best for those with lower incoming PANSS. Ziprasidone is best for a reward based only on BMI (i.e., for $\delta = 1$) independent of PANSS level. This result agrees with existing research on weight gain associated with these atypical antipsychotics (Allison et al., 1999). Again, we have indicated the values of δ at which the decision changes from one action to another by dropping down a dotted line. In this analysis, we found that quetiapine was globally dominated.

6.3.2 PHASE 1 ANALYSIS

For Phase 1 patients, the possible actions are $\mathcal{A}_1 = \{\text{OLAN}, \text{PERP}, \text{QUET}, \text{RISP}, \text{ZIP}\}$, and the feature vectors we use are given by

$$\begin{aligned}\phi^{\text{TOL}}_{s_1,a_1} = [&1,\ 1_{a_1=\text{OLAN}},\ 1_{a_1=\text{PERP}},\ 1_{a_1=\text{QUET}},\ 1_{a_1=\text{RISP}}, \cdots \\ &s_1,\ s_1 1_{a_1=\text{OLAN}},\ s_1 1_{a_1=\text{PERP}},\ s_1 1_{a_1=\text{QUET}},\ s_1 1_{a_1=\text{RISP}}, \cdots \\ &1_{\text{TD}},\ 1_{\text{EX}},\ 1_{\text{ST1}},\ 1_{\text{ST2}},\ 1_{\text{ST3}},\ 1_{\text{ST4}}]^{\mathsf{T}}.\end{aligned}$$

We have four indicator features for different treatments at Phase 2, $1_{a_1=\text{OLAN}}$, $1_{a_1=\text{PERP}}$, $1_{a_1=\text{QUET}}$, and $1_{a_1=\text{RISP}}$, with ziprasidone represented by turing all of these indicators off. We include the

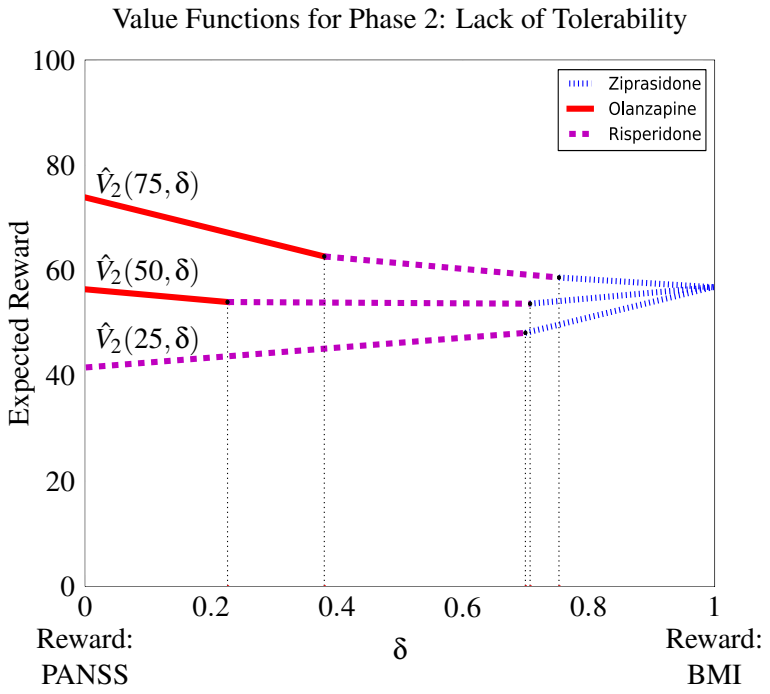


Figure 9: Multiple rewards analysis showing learned value function and associated learned policy for Phase 2 Tolerability patients. Three value functions are shown, with the associated action chosen by the learned policy, for $s_2 = 25$, $s_2 = 50$, and $s_2 = 75$.

product of each of these indicators with the PANSS percentile s_1 at entry to the study, and the remainder of the features are the same as for the Phase 2 feature vectors. (These are collected before the study begins and are therefore available at Phase 1 as well.)

Figure 10 shows a plot of the piecewise linear value function $\hat{V}_1(s_1,\cdot)$ for patients entering Phase 1 (the beginning) of the study. Again we plot $\hat{V}_1(s_1,\cdot)$ for three fixed values of $s_1 = 25$, $s_1 = 50$ and $s_1 = 75$. Possible treatments are perphenazine, olanzapine, quetiapine, risperidone and ziprasidone. For all three states shown in the plot, the learned policy indicates that olanzapine is the best action for a reward based only on PANSS (i.e., for $\delta = 0$). Ziprasidone is best for a reward based only on BMI (i.e., for $\delta = 1$), also independent of PANSS level. Again, the result agrees well with existing research (Allison et al., 1999). In this analysis, we found perphenazine and quetiapine to be globally dominated.

6.4 Symptoms vs. Weight Gain vs. Quality of Life

We now present the output of our algorithm for $D = 3$, using PANSS for $r_{[0]}$, BMI for $r_{[1]}$, and HQLS for $r_{[2]}$. We use the methods described in Section 4.4 to compute the value functions which map a state s_t and a three-element preference vector δ to an estimated value. Rather than display the shape of this value function using a surface or contour plot, we have elected to show only the regions of preference space where each action is optimal (i.e., the learned policy) mapped onto an equilateral triangle. This simplifies the presentation, but still allows us to easily see for each action

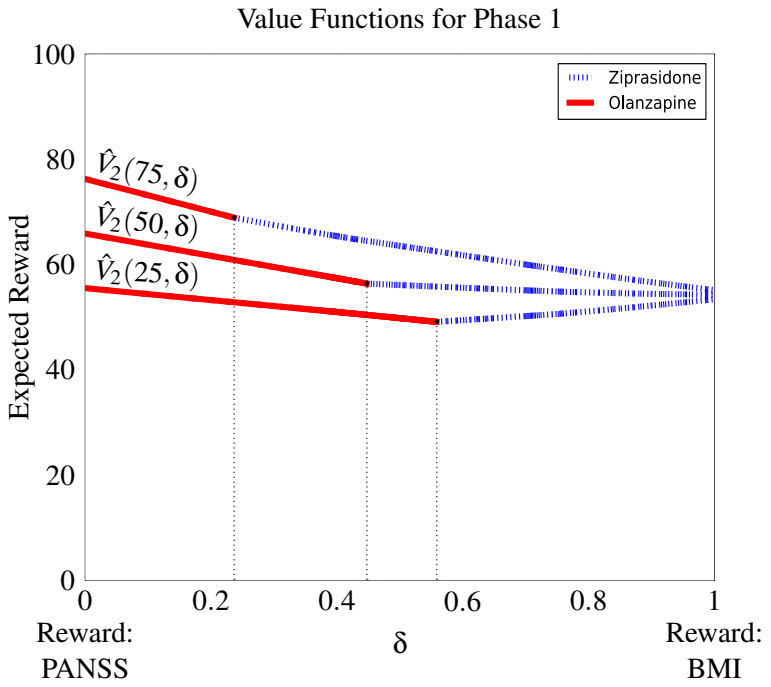


Figure 10: Multiple rewards analysis showing learned value function and associated learned policy for Phase 1 patients. Three value functions are shown, with the associated action chosen by the learned policy, for $s_2 = 25$, $s_2 = 50$, and $s_2 = 75$.

the set of preferences for which that action looks best.[13] In all examples, we show the policy for PANSS percentile $s_t = 50$.

6.4.1 PHASE 2 ANALYSES

We use the same state representation as for the $D = 2$ example. Because we are using the exact same $r_{[0]}$ and $r_{[1]}$ as we did for the $D = 2$ example as well, we can exactly recover the learned policy of our previous $D = 2$ analysis from our $D = 3$ analysis simply by considering all preferences of the form $\delta = (\delta, 1-\delta, 0)$, that is, the preferences along the upper-left edge of the triangle.

Figure 11 shows the learned policy for patients with $s_2 = 50$ whose Phase 1 treatment was not efficacious. As in the $D = 2$ case, we combine the actions of giving {olanzapine, risperidone, or quetiapine} into one "not-clozapine" action. We see that clozapine appears best if the reward is based only on PANSS or on HQLS, and "not-clozapine" appears best only if the preference assigns a relatively large weight to BMI. If we consider the upper-left edge of the triangle where the preferences assign zero weight to HQLS, we get precisely the same policy shown in Figure 8. We also see that clozapine appears best for all preferences that consider only PANSS and HQLS (bottom edge) and for most preferences that consider only HQLS and BMI (upper-right edge.) We hypothesize that this is because there is a strong association between control of schizophrenia

13. In addition to the policy, we have indicated the linear regions produced by the Delaunay triangulations using faint lines in order to give a sense of their complexity.

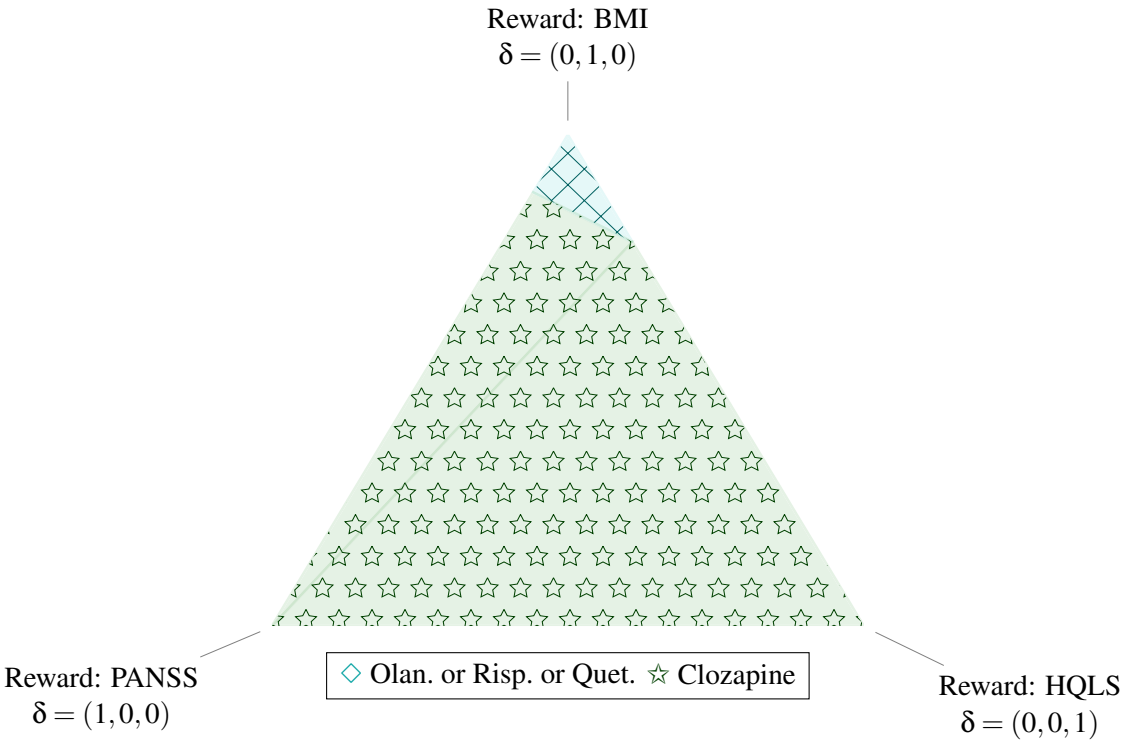


Figure 11: Multiple rewards analysis using PANSS, BMI, and HQLS, showing learned policy for Phase 2 Efficacy patients with $s_2 = 50$.

symptoms and quality of life; thus treatments that work well for PANSS should also work somewhat well for HQLS. We note however that clozapine occupies a narrower range on the upper-right edge than it does on the upper-left edge. In this analysis it is clear that neither action is globally dominated because neither is dominated at state $s_2 = 50$.

Figure 12 shows the learned policy for patients with $s_2 = 50$ whose Phase 1 treatment was not tolerable due to side-effects. Here, we see that olanzapine appears best if the reward is based only on PANSS or on HQLS, and ziprasidone appears best if the preference assigns a relatively large weight to BMI. For intermediate preferences, risperidone appears best. Again if we consider the upper-left edge of the triangle where the preferences assign zero weight to HQLS, we get precisely the same policy shown in Figure 9. We also see that olanzapine appears best for all preferences that consider only PANSS and HQLS (bottom edge.) Note that horizontal lines in the triangle represent sets of preferences where the weight on BMI is held constant. Over much of preference space, these horizontal lines are completely contained within one treatment's optimal region, meaning that given a weight for BMI, the policy usually does not depend on the relative preference for PANSS versus HQLI. We hypothesize again that this is because there is a strong association between symptom control and quality of life. In this analysis, we found that quetiapine was globally dominated.

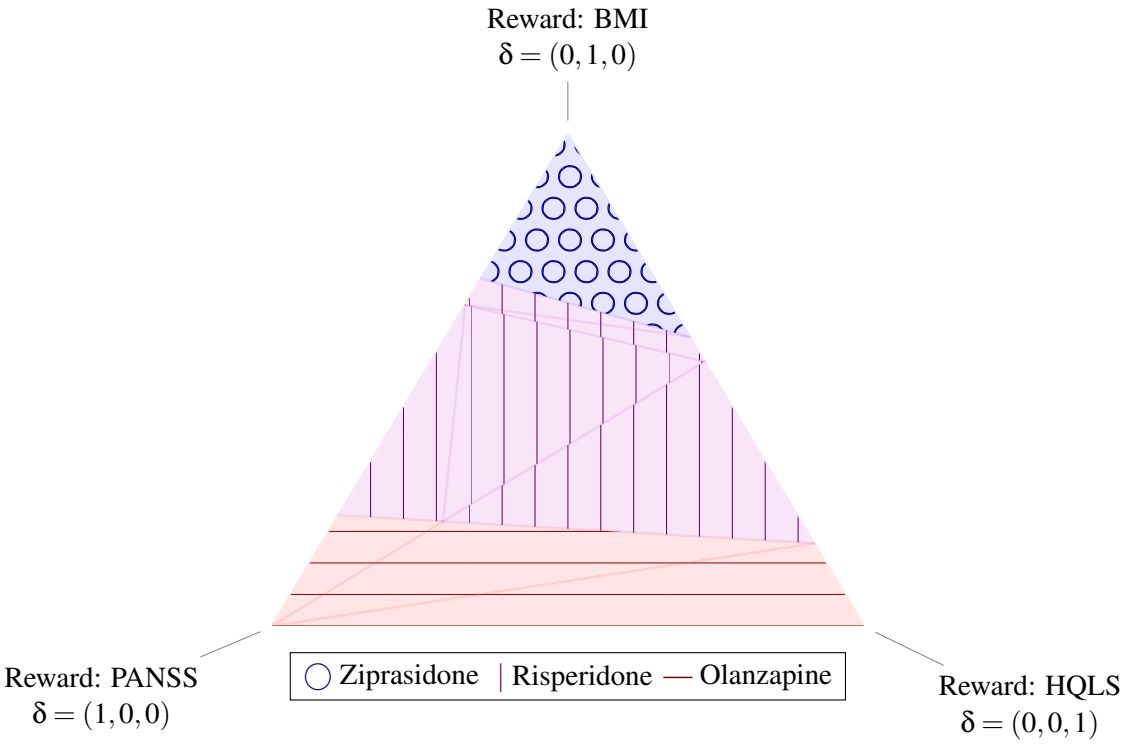


Figure 12: Multiple rewards analysis using PANSS, BMI, and HQLS, showing learned policy for Phase 2 Tolerability patients with $s_2 = 50$.

6.4.2 PHASE 1 ANALYSIS

Figure 13 shows the learned policy for patients with $s_1 = 50$. Again we see that ziprasidone appears best for preferences that assign a large importance to BMI, and olanzapine appears best for other preferences. Again if we consider the upper-left edge of the triangle where the preferences assign zero weight to HQLS, we get precisely the same policy shown in Figure 10. Interestingly, the region where ziprasidone appears best increases as we decrease the importance of PANSS, indicating it may be preferable for patients who are more concerned with weight control and quality of life than with very tight control of symptoms. In this analysis, we found that quetiapine, risperidone, and perphenazine were dominated at our example state $s_1 = 50$, but we found no action to be globally dominated.

6.5 Limitations

We note that unlike our previous work using this data, this analysis does not attempt to remove bias induced by missing data, nor does it provide measures of uncertainty for the learned policy (Shortreed et al., 2010). Both of these limitations indicate important directions for future work, as we discuss below.

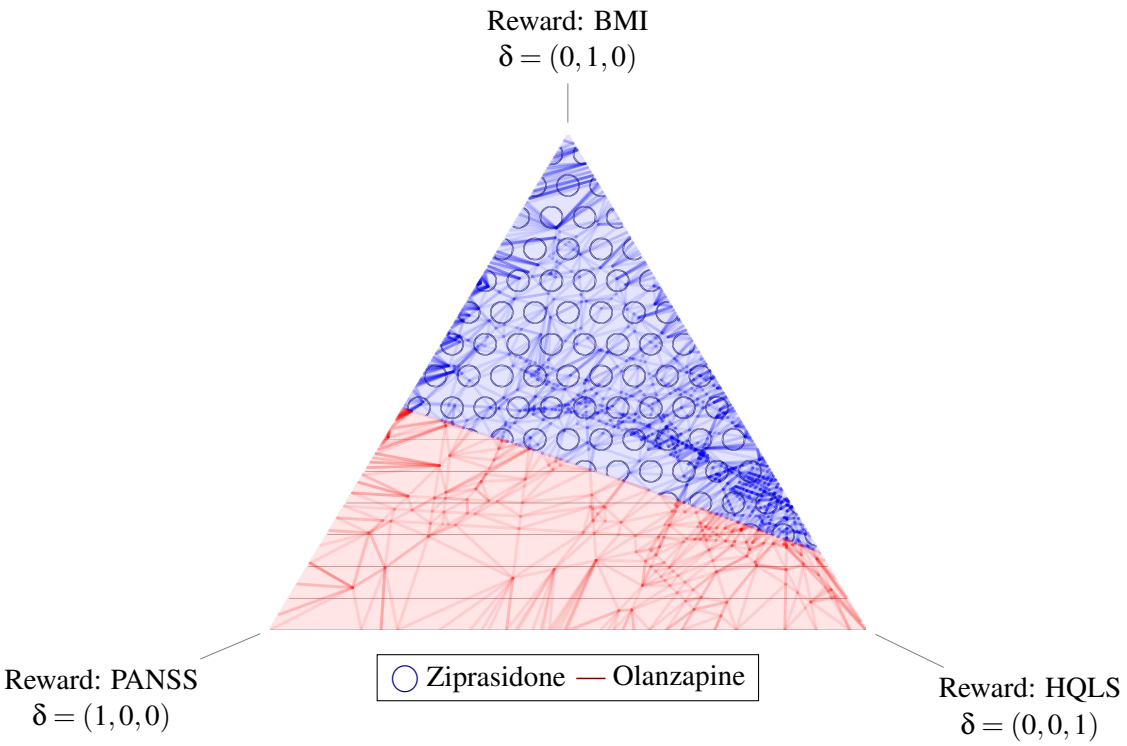


Figure 13: Multiple rewards analysis using PANSS, BMI, and HQLS, showing learned policy for Phase 1 patients with $s_1 = 50$.

7. Discussion and Future Work

The methods we have presented comprise a crucial first step towards a data analysis method that can be deployed in clinical decision support systems. However, there are challenges that remain to be addressed.

7.1 The Meaning of Rewards and the Effect of Scaling

Consider an analysis with $D = 2$ basis rewards at its final time point $t = T$. For a preference of $\delta = 0.5$, two actions a_1 and a_2 for which $0.5r_{[0]}(s_T, a_1) + 0.5r_{[1]}(s_T, a_1) = 0.5r_{[0]}(s_T, a_2) + 0.5r_{[1]}(s_T, a_2)$ are indistinguishable. One can think of the preference as setting an "exchange rate" for $r_{[0]}$ and $r_{[1]}$: in this case, the basis rewards can be exchanged at a one-to-one rate and our happiness with the overall result of an action remains the same. For $\delta = 0.75$, the two actions would be indistinguishable if $0.25r_{[0]}(s_T, a_1) + 0.75r_{[1]}(s_T, a_1) = 0.25r_{[0]}(s_T, a_2) + 0.75r_{[1]}(s_T, a_2)$, meaning that the loss of one unit of $r_{[1]}$ would have to be compensated by a gain of three units of $r_{[0]}$ in order for the actions to be considered equivalent. The stronger the preference for $r_{[1]}$, the more units of $r_{[0]}$ we need in order to "make up" for the loss a unit of $r_{[1]}$. Note that this interpretation would not be possible had we chosen to define reward as a non-linear function of preference.

In our example analysis, we chose to convert all of the rewards to percentiles before using them; thus we interpret a preference of $\delta = 0.5$ to mean that the "exchange rate" is one-to-one

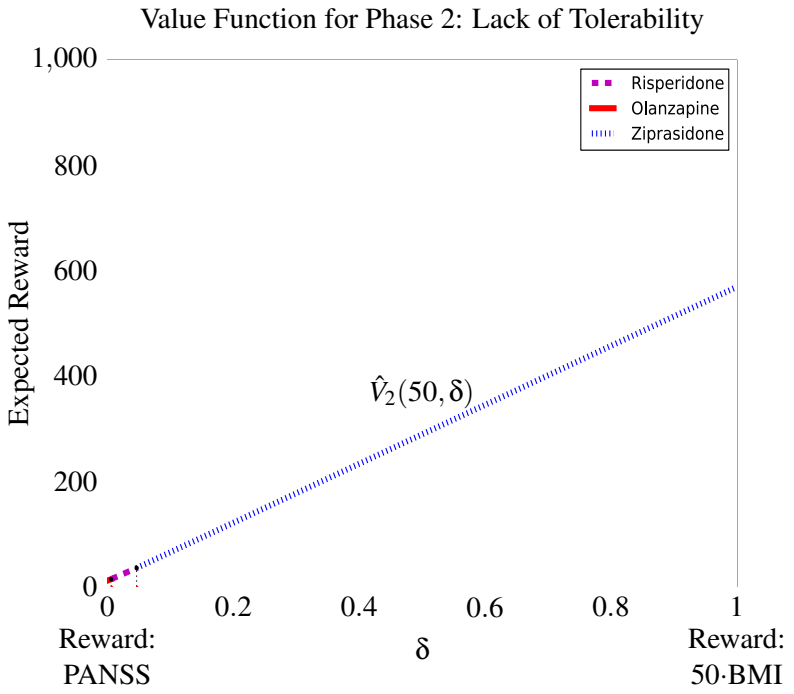


Figure 14: Multiple rewards analysis showing learned value function and associated learned policy for Phase 2 Tolerability patients, **using** 50·**BMI as one basis reward**. Note how scaling the BMI reward affects the regions where different actions are optimal. (Compare with Figure 9.)

for percentiles of PANSS and percentiles of BMI. The exchange rate at $\delta = 0.5$ could be shifted however by first multiplying one or both rewards by a constant factor before using them in our algorithm. If we fed $2 \cdot \text{BMI}$ into our algorithm as $r_{[1]}$, the exchange rate at $\delta = 0.5$ would be two percentiles of PANSS equals one percentile of BMI, and the preference at which the policy changes from one action to the other in Figure 9, for example, would shift to the left. The ordering of recommended treatments (olanzapine for lowest δ, risperidone for moderate δ, ziprasidone for high δ) would remain the same, however.

A more extreme version is illustrated in Figure 14, where we use $50 \cdot \text{BMI}$ as one basis reward. In this analysis, the "exchange rate" for $\delta = 0.5$ is one unit of BMI equals 50 units of PANSS. Note that there are still three non-dominated actions, but the regions where two of them are optimal are now very small and "compressed" into an area very near $\delta = 0$. This illustrates a potential pitfall: if the exchange rate represented by $\delta = 0.5$ is not "moderate," resulting decision aids will be at best unhelpful and at worst misleading. However, it also supports the use of the exact algorithms we have presented: even if the rewards are poorly scaled, the set of non-dominated actions remains the same, and they retain their ordering according to delta.

Note that if the region where an action is optimal is very small, a naïve grid-search over δ may not detect it. For example, if we ask try to determine in the Figure 14 example which treatments are optimal near a preference of $\delta = 0$ by checking nearby δ, we may miss risperidone. On the

other hand, the exact methods we have presented will correctly recognize that the risperidone is non-dominated. A practitioner using our methods might then wish to change the analysis by rescaling one or more of the basis rewards. The nature of this rescaling will of course depend on the application at hand; we intend to formalize this problem in future work.

7.2 Value Function and Policy Approximations

We have shown that the complexity of constructing the exact value function is potentially exponential in the time horizon of the problem. However, we have also shown in our example that although the value function may be very complex, the learned policy may still be very simple. Figure 10 illustrates this: each faint triangle in the figure represents a linear piece of the value function. Though there are many pieces, by and large adjacent pieces recommend the same action. This reflects a large-scale smoothness in the Q-functions, and suggests that a simple, smooth function might approximate the piecewise-linear Q-functions very well while reducing computational cost. Some existing algorithms for POMDPs that approximate the value function (e.g., Pineau et al. 2006, Wang et al. 2006) may be useful, but novel modifications will be needed to use these approximations in our setting. Another class of approximations introduced by Poupart and Boutilier (2002) focuses on compressing the state space of the POMDP. As we discussed in Section 3.2, the number of states in a POMDP roughly corresponds to the number of basis rewards considered by our method. Thus, these methods may lead to a way of computing a simplified or "compressed" view of preferences when the number of basis rewards is large, which could be used both to reduce computational cost and to help users better understand their preferences.

7.3 Measures of Uncertainty and Similar Q-values

In our example, almost all preferences are associated with exactly one optimal action. In practice, it may make more sense to recommend more than one action for a particular preference if the Q-values of those actions are very similar. In the medical setting, one may prefer to allow the physician or patient to break ties if outcomes under different treatments are deemed to be "close." We note two criteria for "closeness" that deserve further study.

Statistical Significance One reason for recommending a set of treatments arises when there is insufficient evidence that one action is actually superior to another. Ideally, one would like to know if an observed difference in Q-values for different actions is true for the population or if it is present in the data we have simply by chance. The methods we have presented do not provide uncertainty information about the learned value function or policy, and although the algorithm is based on linear regression which itself has well-established methods for statistical inference, it is known that even standard single-reward fitted-Q iteration requires specially tailored statistical methods in order to obtain valid confidence measures (Laber et al., 2009; Shortreed et al., 2010). These methods, based on the bootstrap data re-sampling procedure, can be very computationally intensive even for one reward function; thus it will be crucial to combine them with new approximations to the problem in order to produce analyses in a reasonable amount of time. Methods for mitigating the bias induced by having partially missing data can be computationally intensive as well (Shortreed et al., 2010), and should be investigated concurrently with methods for producing confidence information.

Practical Significance Even if we have strong statistical evidence that one action has a higher Q-value than another, we may still wish to recommend a set of actions if that difference is too small to be practically meaningful. Methods for mathematizing the idea of a "clinically meaningful

difference" are under investigation (Laber et al., 2012); we see promise for integrating them with our methods.

7.4 $D > 3$ Basis Rewards

To allow more than 3 basis rewards, we need methods that can represent and manipulate piecewise linear functions in higher dimensions. One avenue would be to use *extended algebraic decision diagrams*, which have been successfully applied to MDPs (Zamani et al., 2012). It is not obvious whether XADD methods provide us with a computationally feasible solution for $D > 3$, but their use is worthy of future study.

8. Conclusion

We have presented a general and explicit development of finite-horizon fitted-Q iteration with an arbitrary number of reward signals and linear value function approximation using an arbitrary number of state features. This included a detailed treatment of the 3-reward function case using triangulation primitives from computational geometry and a method for identifying globally dominated actions under linear function approximation. We also presented an example of how our methods can be used to construct real-world decision aid by considering symptom reduction, weight gain, and quality of life in sequential treatments for schizophrenia. Finally, we have discussed future directions in which to take this work that will further enable our methods to make a positive impact on the field of evidence-based clinical decision support.

Acknowledgments

We extend our sincere thanks to our reviewers, whose detailed comments have enabled us to substantially improve our work. We acknowledge support from Natural Sciences and Engineering Research Council of Canada (NSERC) and the National Institutes of Health (NIH) grants R01 MH080015 and P50 DA10075. Data used in the preparation of this article were obtained from the limited access data sets distributed from the NIH-supported "Clinical Antipsychotic Trials of Intervention Effectiveness in Schizophrenia" (CATIE-Sz). The study was supported by NIMH Contract N01MH90001 to the University of North Carolina at Chapel Hill. The ClinicalTrials.gov identifier is NCT00014001. This manuscript reflects the views of the authors and may not reflect the opinions or views of the CATIE-Sz Study Investigators or the NIH.

Appendix A. Proofs

Note that Lemmas 3 and 4 are known (or deemed "obvious") in the computational geometry literature, but are proved here for completeness.

A.1 Proof of Lemma 3

Over each $\mathcal{R}_i$, $f_{\max} = f_i$ which is linear. Each $\mathcal{R}_i$ is an intersection of the convex polytope $\mathcal{R}$ with an intersection of half-spaces of the form $\{\delta : f_i(\delta) \geq f_j(\delta)\}$, which are also convex polytopes. Thus each $\mathcal{R}_i$ is a convex polytope.

A.2 Proof of Lemma 4

For any point δ in a set $\mathcal{U} \cap \mathcal{V}$ as above, we have $g_1(\delta) = \delta^\mathsf{T} \mathbf{w}_\mathcal{U}$ and $g_2(\delta) = \delta^\mathsf{T} \mathbf{w}_\mathcal{V}$. Therefore, for such δ, we have

$$\begin{aligned}\alpha_1 \cdot g_1(\delta) + \alpha_2 \cdot g_2(\delta) &= \alpha_1 \cdot (\delta^\mathsf{T} \mathbf{w}_\mathcal{U}) + \alpha_2 \cdot (\delta^\mathsf{T} \mathbf{w}_\mathcal{V}), \\ &= \delta^\mathsf{T} (\alpha_1 \cdot \mathbf{w}_\mathcal{U} + \alpha_2 \cdot \mathbf{w}_\mathcal{V}), \\ &= \delta^\mathsf{T} \mathbf{w}_{\mathcal{U} \cap \mathcal{V}}.\end{aligned}$$

Therefore within each set given by the intersections above, both g_1 and g_2 are linear.

A.3 Proof of Lemma 5

Suppose a is optimal for some (ψ_{s_T}, δ) in the domain but is not optimal for any (ψ_{s_T}, δ) on the boundary of the domain. Further suppose that a is not optimal at any triple-point. Then the region where a is optimal must be completely enclosed by the region where a *single* other action a' is optimal. However, by Equation (8), the boundary between the regions where a is superior to a' and vice-versa is a hyperbola composed of two sets (sheets) that are each continuous and have infinite extent in both ψ_{s_T} and δ. The set must therefore intersect the boundary of the domain of (ψ_{s_T}, δ) and thus there must exist a certificate for a on the boundary. Thus we have a contradiction.

A.4 Proof of Proposition 7

Assume there is a region inside the domain where a is optimal. Assume a is not optimal at any $(p+D+1)$-tuple point. Since a is not optimal at a point where $p+D+1$ actions are optimal, the region where a is optimal must have on its boundary points where k actions are simultaneously optimal for some $k < p+D+1$. Choose the maximum k for which this is true. This boundary is defined by a system of $k-1$ polynomial equations on $p+D$ variables of the form (10); call the variables $\zeta_1, \zeta_2, ..., \zeta_{p+D}$. Since we assume the region where a is optimal is in the interior of the domain, there exists an interior point ζ^* that is a solution to the system of equations. Create a new system of $k-1$ equations and $k-1$ unknowns by fixing the last $(p+D)-(k-1)$ variables to $\zeta_k = \zeta_k^*$, $\zeta_{k+1} = \zeta_{k+1}^*$ ..., $\zeta_{p+D} = \zeta_{p+D}^*$. The point $(\zeta_1^*, ..., \zeta_{k-1}^*)$ is a solution to this reduced system. Suppose the solution of the reduced system is a continuous function of ζ_k^*, which holds if Conjecture 6 is true. Then if we move ζ_k^* toward a boundary from its original value, either we will find a point satisfying the original system with ζ_k^* on the boundary and the remaining variables in the interior of the domain, or another variable or variables will reach its boundary first, and the remainder of the variables will be in the interior of the domain. In either case, there exists a point on the boundary of the domain of interest where action a is optimal.

References

D. B. Allison, J. L. Mentore, M. Heo, L. P. Chandler, J. C. Cappelleri, M. C. Infante, and P. J. Weiden. Antipsychotic-induced weight gain: A comprehensive research synthesis. *American Journal of Psychiatry*, 156:1686–1696, November 1999.

L. Barrett and S. Narayanan. Learning all optimal policies with multiple criteria. In *Proceedings of the 25th International Conference on Machine Learning*, pages 41–47, 2008.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*, chapter 2.1, page 12. Athena Scientific, 1996.

O. Bonnichsen. Elicitation of ostomy pouch preferences: a discrete-choice experiment. *Patient*, 4 (3):163–175, 2011.

C. Boutilier. A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 239–246, 2002.

P. Brass. On the size of higher-dimensional triangulations. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*, volume 52, pages 147–152. MSRI Publications, 2005.

A. Castelletti, S. Galelli, M. Restelli, and R. Soncini-Sessa. Tree-based reinforcement learning for optimal water reservoir operation. *Water Resources Research*, 46(W06502), 2010.

CGAL. CGAL, Computational Geometry Algorithms Library, 2011. URL `http://www.cgal.org`.

L. P. Chew. Constrained delaunay triangulations. In *Proceedings of the Third Annual Symposium on Computational geometry*, SCG '87, pages 215–222, New York, NY, USA, 1987.

R. D. Cook and S. Weisberg. *Applied Regression Including Computing and Graphics*. Wiley, August 1999.

D. A. Cox, D. O'Shea, and J. B. Little. *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 1997.

J. A. Cramer, R. Rosenheck, W. Xu, J. Thomas, W. Henderson, and D. S. Charney. Quality of life in schizophrenia: A comparison of instruments. *Schizophrenia Bulletin*, 26(3):659–666, 2000.

C. C. Davis, M. Claudius, L. A. Palinkas, J. B. Wong, and L. K. Leslie. Putting families in the center: Family perspectives on decision making and ADHD and implications for ADHD care. *Journal of Attention Disorders*, Oct 2011. E-pub ahead of print.

M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry Algorithms and Applications*. Springer, 3 edition, 2008.

M. Ehrgott. *Multicriteria Optimization*, chapter 3. Springer, second edition, 2005.

Z. Gábor, Z. Kalmár, and C. Szepesvári. Multi-criteria reinforcement learning. In *Proceedings of the 15th International Conference on Machine Learning*, pages 197–205, 1998.

G. H. Golub and C. F. Van Loan. *Matrix Computation*. John Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996. ISBN 0471935271.

B. Grünbaum. *Convex Polytopes*, volume 221 of *Graduate Texts in Mathematics*. Springer-Verlag, 1967. ISBN 0387004246;.

D. W. Heinrichs, T. E. Hanlon, and W. T. C. Jr. The Quality of Life Scale: An instrument for rating the schizophrenic deficit syndrome. *Schizophrenia Bulletin*, 10(3):388–398, 1984.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(12):99–134, 1998.

S. R. Kay, A. Fiszbein, and L. A. Opfer. The Positive and Negative Syndrome Scale (PANSS) for schizophrenia. *Schizophrenia Bulletin*, 13(2):261–276, 1987.

P. W. Keller, S. Mannor, and D. Precup. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *Proceedings of the 23rd International Conference on Machine learning*, ICML '06, pages 449–456, 2006.

A. Y. Kuk, J. Li, and A. J. Rush. Recursive subsetting to identify patients in the STAR*D: a method to enhance the accuracy of early prediction of treatment outcome and to inform personalized care. *Journal of Clinical Psychiatry*, 71(11):1502–8, November 2010.

E. B. Laber, M. Qian, D. J. Lizotte, and S. A. Murphy. Statistical inference in dynamic treatment regimes. Technical Report 50, Univ. of Michigan Statistics Department, 2009.

E. B. Laber, D. J. Lizotte, and B. Ferguson. Set-valued dynamic treatment regimes for competing outcomes. `arXiv:1207.3100v2 [stat.ME]`, 2012.

D. J. Lizotte, M. Bowling, and S. A. Murphy. Efficient reinforcement learning with multiple reward functions for randomized controlled trial analysis. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning*, pages 695–702, Haifa, Israel, June 2010. Omnipress.

S. Mannor and N. Shimkin. A geometric approach to multi-criterion reinforcement learning. *Journal of Machine Learning Research*, 5:325–260, 2004.

S. Mannor and J. N. Tsitsiklis. Mean-variance optimization in Markov decision processes. In *Proceedings of the 28th International Conference on Machine Learning*, pages 177–184, 2011.

M. S. McDonagh, K. Peterson, S. Carson, R. Fu, and S. Thakurta. Drug class review: Atypical antipsychotic drugs. Technical report, Oregon Health & Science University, July 2010. Drug Effectiveness Review Project, Update 3.

J. R. McKay. *Treating Substance Use Disorders with Adaptive Continuing Aftercare*. American Psychological Association, 2009.

A. L. Miller, J. A. Chiles, and J. K. C. et al. The Texas Medication Algorithm Project (TMAP) schizophrenia algorithms. *Journal of Clinical Psychiatry*, 60:649–657, 1999.

S. A. Murphy. An experimental design for the development of adaptive treatment strategies. *Statistics in Medicine*, 24:1455–1481, 2005.

S. A. Murphy, S. W. Oslin, A. J. Rush, and J. Zhu. Methodological challenges in constructing effective treatment sequences for chronic psychiatric disorders. *Neuropsychopharmacology*, 32: 257–262, 2007.

S. Natarajan and P. Tadepalli. Dynamic preferences in multi-criteria reinforcement learning. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 601–608, 2005.

National Institutes of Health. *Clinical Guidelines on the Identification, and Treatment of Overweight and Obesity in Adults: The Evidence Report*. National Heart, Lung, and Blood Institute, Sept 1998. NIH Publication no. 98-4083.

A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670, 2000.

J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, pages 1025–1032, 2003.

J. Pineau, G. Gordon, and S. Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006.

J. Pineau, M. G. Bellemare, A. J. Rush, A. Ghizaru, and S. A. Murphy. Constructing evidence-based treatment strategies using methods from computer science. *Drug and Alcohol Dependence*, 88 (Suppl 2):S52–S60, May 2007.

P. Poupart and C. Boutilier. Value-directed compression of POMDPs. *Advances in Neural Information Processing Systems*, 15:1547–1554, 2002.

D. L. Sackett. Evidence-based medicine: What it is and what it isn't. *British Medical Journal*, 312 (7023):71–72, 1996.

S. Shortreed, E. B. Laber, D. J. Lizotte, T. S. Stroup, J. Pineau, and S. A. Murphy. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine Learning*, pages 1–28, 2010. ISSN 0885-6125.

T. S. Stroup, J. P. McEvoy, M. S. Swartz, M. J. Byerly, I. D. Glick, J. M. Canive, M. McGee, G. M. Simpson, M. D. Stevens, and J. A. Lieberman. The national institute of mental health clinical antipschotic trials of intervention effectiveness (CATIE) project: Schizophrenia trial design and protocol deveplopment. *Schizophrenia Bulletin*, 29(1):15–31, 2003.

B. Sturmfels. *Solving Systems of Polynomial Equations*. Regional conference series in mathematics. Published for the Conference Board of the Mathematical Sciences by the American Mathematical Society, 2002. ISBN 9780821832516.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

M. S. Swartz, D. O. Perkins, T. S. Stroup, J. P. McEvoy, J. M. Nieri, and D. D. Haal. Assessing clinical and functional outcomes in the clinical antipsychotic of intervention effectiveness (CATIE) schizophrenia trial. *Schizophrenia Bulletin*, 29(1):33–43, 2003.

C. Szepesvári. *Algorithms for Reinforcement Learning*. Morgan & Claypool, July 2010.

P. F. Thall. Some geometric methods for constructing decision criteria based on two-dimensional parameters. *Journal of Statistical Planning and Inference*, 138(2):516–527, 2008.

D. J. Uherka and A. M. Sergott. On the continuous dependence of the roots of a polynomial on its coefficients. *The American Mathematical Monthly*, 84(5):368–370, 1977. ISSN 0002-9890.

P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, 84:51–80, 2011. ISSN 0885-6125. 10.1007/s10994-010-5232-5.

T. Wang, P. Poupart, M. Bowling, and D. Schuurmans. Compact, convex upper bound iteration for approximate pomdp planning. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 1245–1251, 2006.

J. R. Weisz, B. C. Chu, and A. J. Polo. Treatment dissemination and evidence-based practice: Strengthening intervention through clinician-researcher collaboration. *Clinical Psychology: Science and Practice*, 11(3):300–307, 2004. ISSN 1468-2850.

Working Group for the Canadian Psychiatric Association and the Canadian Alliance for Research on Schizophrenia. Canadian clinical practice guidelines for the treatment of schizophrenia. *Canadian Journal of Psychiatry*, 43(suppl. 2):25–40S, 1998.

Z. Zamani, S. Sanner, and C. Fang. Symbolic dynamic programming for continuous state and action MDPs. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012. To appear.

Y. Zhao, M. R. Kosorok, and D. Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in Medicine*, 28:3294–3315, 2009.

Journal of Machine Learning Research 13 (2012) 3297-3321 Submitted 5/12; Revised 10/12; Published 11/12

Human Gesture Recognition on Product Manifolds

Yui Man Lui LUI@CS.COLOSTATE.EDU
Department of Computer Science
Colorado State University
Fort Collins, CO 80523-1873, USA

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

Action videos are multidimensional data and can be naturally represented as data tensors. While tensor computing is widely used in computer vision, the geometry of tensor space is often ignored. The aim of this paper is to demonstrate the importance of the intrinsic geometry of tensor space which yields a very discriminating structure for action recognition. We characterize data tensors as points on a product manifold and model it statistically using least squares regression. To this aim, we factorize a data tensor relating to each order of the tensor using Higher Order Singular Value Decomposition (HOSVD) and then impose each factorized element on a Grassmann manifold. Furthermore, we account for underlying geometry on manifolds and formulate least squares regression as a composite function. This gives a natural extension from Euclidean space to manifolds. Consequently, classification is performed using geodesic distance on a product manifold where each factor manifold is Grassmannian. Our method exploits appearance and motion without explicitly modeling the shapes and dynamics. We assess the proposed method using three gesture databases, namely the Cambridge hand-gesture, the UMD Keck body-gesture, and the CHALEARN gesture challenge data sets. Experimental results reveal that not only does the proposed method perform well on the standard benchmark data sets, but also it generalizes well on the one-shot-learning gesture challenge. Furthermore, it is based on a simple statistical model and the intrinsic geometry of tensor space.

Keywords: gesture recognition, action recognition, Grassmann manifolds, product manifolds, one-shot-learning, kinect data

1. Introduction

Human gestures/actions are the natural way for expressing intentions and can be instantly recognized by people. We use gestures to depict sign language to deaf people, convey messages in noisy environments, and interface with computer games. Having automated gesture-based communication would broaden the horizon of human-computer interaction and enrich our daily lives. In recent years, many gesture recognition algorithms have been proposed (Mitra and Acharya, 2007; Wang et al., 2009; Bilinski and Bremond, 2011). However, reliable gesture recognition remains a challenging area due in part to the complexity of human movements. To champion the recognition performance, models are often complicated, causing difficulty for generalization. Consequently, heavy-duty models may not have substantial gains in overall gesture recognition problems.

In this paper, we propose a new representation to gesture recognition based upon tensors and the geometry of product manifolds. Since human actions are expressed as a sequence of video frames, an action video may be characterized as a third order data tensor. The mathematical framework

for working with high order tensors is multilinear algebra which is a useful tool for characterizing multiple factor interactions. Tensor computing has been successfully applied to many computer vision applications such as face recognition (Vasilescu and Terzopoulos, 2002), visual tracking (Li et al., 2007), and action classification (Vasilescu, 2002; Kim and Cipolla, 2009). However, the geometrical aspect of data tensors remains unexamined. The goal of this paper is to demonstrate the importance of the intrinsic geometry of tensor space where it provides a very discriminating structure for action recognition.

Notably, several recent efforts (Lui, 2012a) have been inspired by the characteristics of space and the associated construction of classifiers based upon the intrinsic geometry inherent in particular manifolds. Veeraraghavan et al. (2005) modeled human shapes from a shape manifold and expressed the dynamics of human silhouettes using an autoregressive (AR) model on the tangent space. Turaga and Chellappa (2009) extended this framework and represented the trajectories on a Grassmann manifold for activity classification. The use of tangent bundles on special manifolds was investigated by Lui (2012b) where a set of tangent spaces was exploited for action recognition. Age estimation was also studied using Grassmann manifolds (Turaga et al., 2010). The geodesic velocity from an average face to the given face was employed for age estimation where the space of landmarks was interpreted as a Grassmann manifold. Lui and Beveridge (2008) characterized tangent spaces of a registration manifold as elements on a Grassmann manifold for face recognition. The importance of the ordering on Stiefel manifolds was demonstrated by Lui et al. (2009) and an illumination model was applied to synthesize such elements for face recognition. These successes motivate the exploration of the underlying geometry of tensor space.

The method proposed in this paper characterizes action videos as data tensors and demonstrates their association with a product manifold. We focus attention on the intrinsic geometry of tensor space, and draw upon the fact that the geodesic on a product manifold is equivalent to the Cartesian product of geodesics from multiple factor manifolds. In other words, elements of a product manifold are the set of all elements inherited from factor manifolds. Thus, in our approach, action videos are factorized to three factor elements using Higher Order Singular Value Decomposition (HOSVD) in which the factor elements give rise to three factor manifolds. We further extend the product manifold representation to least squares regression. In doing so, we consider the underlying geometry and formulate least squares regression as a composite function. As such, we ensure that both the domain values and the range values reside on a manifold through the regression process. This yields a natural extension from Euclidean space to manifolds. The least squares fitted elements from a training set can then be exploited for gesture recognition where the similarity is expressed in terms of the geodesic distance on a product manifold associated with fitted elements from factor manifolds.

We demonstrate the merits of our method on three gesture recognition problems including hand gestures, body gestures, and gestures collected from the Microsoft Kinect™ camera for the one-shot-learning CHALEARN gesture challenge. Our experimental results reveal that our method is competitive to the state-of-the-art methods and generalizes well to the one-shot-learning scheme, yet is based on a simple statistical model. The key contributions of the proposed work are summarized as follows:

- A new way of relating tensors on a product manifold to action recognition.
- A novel formulation for least squares regression on manifolds.
- The use of appearance and motion without explicitly modeling shapes or dynamics.

- A simple pixel-based representation (no silhouette or skeleton extraction).
- No extensive training and parameter tuning.
- No explicit assumption on action data.
- Competitive performance on gesture recognition.
- Applicable to other visual applications.

The rest of this paper is organized as follows: Related work is summarized in Section 2. Tensor algebra, orthogonal groups, and Grassmann manifolds are reviewed in Section 3. The formulation of the proposed product manifold is presented in Section 4 and is further elaborated with examples in Section 5. The statistical modeling on manifolds is introduced in Section 6. Section 7 reports our experimental results. Section 8 discusses the effect of using raw pixels for action recognition. Finally, we conclude this paper in Section 9.

2. Related Work

Many researchers have proposed a variety of techniques for action recognition in recent years. We highlight some of this work here, including bag-of-features models, autoregressive models, 3D Fourier transforms, tensor frameworks, and product spaces.

In the context of action recognition, bag-of-features models (Dollar et al., 2005; Wang et al., 2009; Bilinski and Bremond, 2011) may be among the most popular methods wherein visual vocabularies are learned from feature descriptors and spatiotemporal features are typically represented by a normalized histogram. While encouraging results have been achieved, bag-of-features methods have heavy training loads prior to classification. In particular, feature detection and codebook generation can consume tremendous amounts of time if the number of training samples is large. Recently, Wang et al. (2009) have evaluated a number of feature descriptors and bag-of-features models for action recognition. This study concluded that different sampling strategies and feature descriptors were needed to achieve the best results on alternative action data sets. Similar conclusions were also found by Bilinski and Bremond (2011) where various sizes of codebooks are needed for different data sets in order to obtain peak performances.

Another school of thought for action classification is using an autoregressive (AR) model. Some of the earliest works involved dynamic texture recognition (Saisan et al., 2001) and human gait recognition (Bissacco et al., 2001). These works represented actions using AR models. The authors found that the most effective way to compare dynamics was by computing the Martin distance between AR models. Veeraraghavan et al. (2005) modeled human silhouettes based on Kendall's theory of shape (Kendall, 1984) where shapes were expressed on a shape manifold. This method modeled the dynamics of human silhouettes using an AR model on the tangent space of the shape manifold. The sequences of human shapes were compared by computing the distance between the AR models. Turaga and Chellappa (2009) investigated statistical modeling with AR models for human activity analysis. In their work, trajectories were considered a sequence of subspaces represented by AR models on a Grassmann manifold. As such, the dynamics were learned and kernel density functions with Procrustes representation were applied to density estimation.

Three-dimensional Fourier transform has been demonstrated as a valuable tool in action classification. Weinland et al. (2006) employed Fourier magnitudes and cylindrical coordinates to represent motion templates. Consequently, the action matching was invariant to translations and rotations

around the z-axis. Although this method was view invariant, the training videos needed to be acquired from multiple cameras. Rodriguez et al. (2008) synthesized a filter respond using the Clifford Fourier transform for action recognition. The feature representation was computed using spatiotemporal regularity flow from the xy-parallel component. The advantage of using Clifford algebra is the direct use of vector fields to Fourier transform.

Data tensors are the multidimensional generalizations to matrices. Vasilescu (2002) modeled the joint angle trajectories on human motion as a set of factorized matrices from a data tensor. Signatures corresponding to motion and identity were then extracted using PCA for person identification. Kim and Cipolla (2009) extended canonical correlation analysis to the tensor framework by developing a Tensor Canonical Correlation Algorithm (TCCA). This method factorized data tensors to a set of matrices and learned a set of projection matrices maximizing the canonical correlations. The inner product was employed to compute the similarity between two data tensors. The use of SIFT features with CCA was also considered for gesture recognition by Kim and Cipolla (2007). Recently, nonnegative tensor factorization has been exploited for action recognition by Krausz and Bauckhage (2010) where action videos were factorized using a gradient descent method and represented as the sum of rank-1 tensors associated with a weighting factor. As a result, the appearance was captured by the basis images and the dynamics was encoded with the weighting factor.

Product spaces have received attention in applications related to spatiotemporal interactions. Datta et al. (2009) modeled the motion manifold as a collection of local linear models. This method learned a selection of mappings to encode the motion manifold from a product space. Lin et al. (2009) proposed a probabilistic framework for action recognition using prototype trees. Shape and motion were explicitly learned and characterized via hierarchical K-means clustering. The joint likelihood framework was employed to model the joint shape-motion space. Li and Chellappa (2010) investigated the product space of spatial and temporal submanifolds for action alignment. Sequential importance sampling was then used to find the optimal alignment. Despite these efforts, the geometry of the product space has not been directly considered and the geodesic nature on the product manifold remains unexamined.

3. Mathematical Background

In this section, we briefly review the background mathematics used in this paper. Particularly, we focus on the elements of tensor algebra, orthogonal groups, Stiefel manifolds, and Grassmann manifolds.

3.1 Tensor Representation

Tensors provide a natural representation for high dimensional data. We consider a video as a third order data tensor $\in \mathbb{R}^{X \times Y \times T}$ where X, Y, and T are the image width, image height, and video length, respectively. High order data tensors can be regarded as a multilinear mapping over a set of vector spaces. Generally, useful information can be extracted using tensor decompositions. In particular, a Higher Order Singular Value Decomposition (HOSVD) (De Lathauwer et al., 2000) is considered in this paper because the data tensor can be factorized in a closed-form. A recent review paper on tensor decompositions can be found in Kolda and Bader (2009). Before we describe HOSVD, we illustrate a building block operation called matrix unfolding.

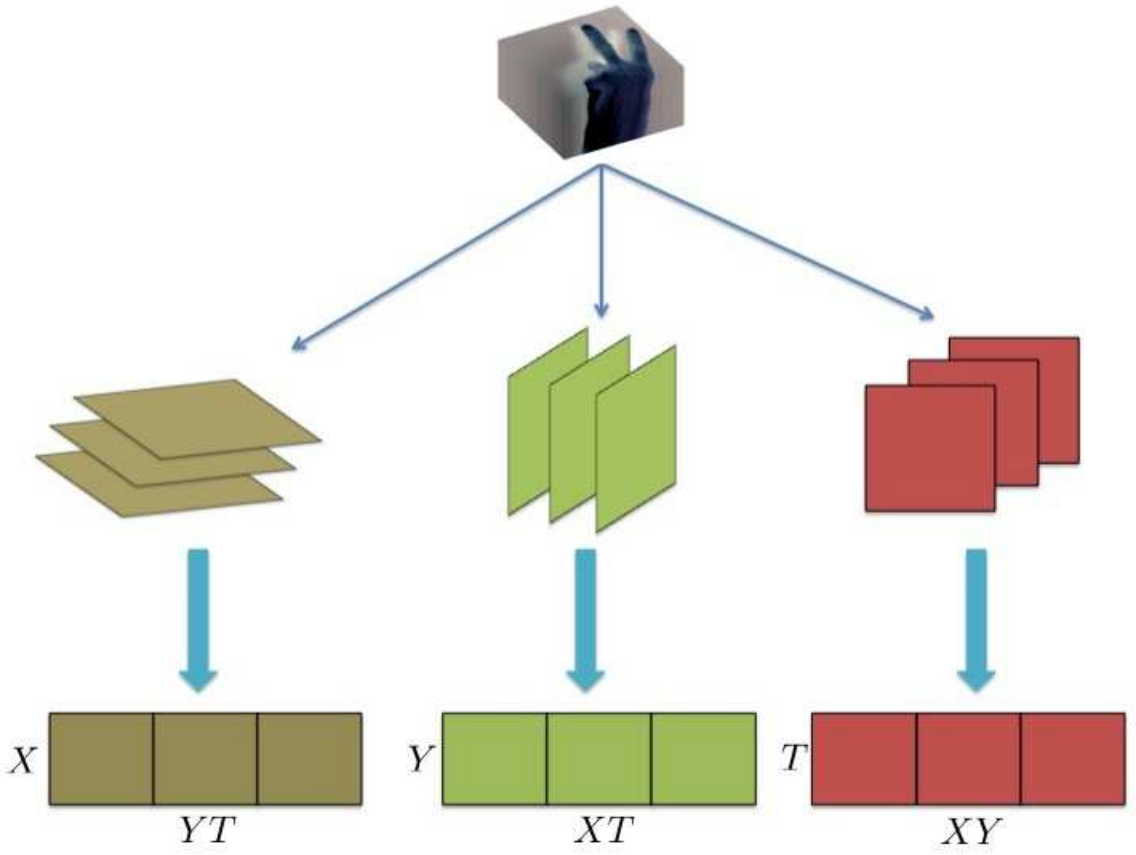


Figure 1: An example of matrix unfolding for a third order tensor. The illustration is for a video action sequence with two spatial dimensions X and Y and a temporal dimension T.

3.1.1 MATRIX UNFOLDING

Let $\mathcal{A}$ be an order N data tensor $\in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The data tensor $\mathcal{A}$ can be converted to a set of matrices via a matrix unfolding operation. Matrix unfolding maps a tensor $\mathcal{A}$ to a set of matrices $A_{(1)}, A_{(2)}, \ldots, A_{(N)}$, where $A_{(k)} \in \mathbb{R}^{I_k \times (I_1 \times \cdots \times I_{k-1} \times I_{k+1} \cdots \times I_N)}$ is a mode-k matrix of $\mathcal{A}$. An example of matrix unfolding of a third order, that is, $N = 3$, tensor is given in Figure 1. As Figure 1 shows, we can slice a third order tensor in three different ways along each axis and concatenate these slices into three different matrices $A_{(1)}$, $A_{(2)}$, and $A_{(3)}$ where the rows of an unfolded matrix are represented by a single variation of the tensor and the columns are composed by two variations of the tensor.

3.1.2 HIGHER ORDER SINGULAR VALUE DECOMPOSITION

Just as a data matrix can be factorized using a Singular Value Decomposition (SVD), a data tensor can also be factorized using Higher Order Singular Value Decomposition (HOSVD), also known as multilinear SVD. HOSVD operates on the unfolded matrices $A_{(k)}$, and each unfolded matrix may

be factored using SVD as follows:

$$A_{(k)} = U^{(k)}\Sigma^{(k)}V^{(k)^T} \tag{1}$$

where $\Sigma^{(k)}$ is a diagonal matrix, $U^{(k)}$ is an orthogonal matrix spanning the column space of $A_{(k)}$ associated with nonzero singular values, and $V^{(k)}$ is an orthogonal matrix spanning the row space of $A_{(k)}$ associated with nonzero singular values. Then, an N order tensor can be decomposed using HOSVD as follows:

$$\mathcal{A} = \mathcal{S} \times_1 U^{(1)} \times_2 U^{(2)} \ldots \times_n U^{(N)}$$

where $\mathcal{S} \in \mathbb{R}^{(I_1 \times I_2 \times \cdots \times I_N)}$ is a core tensor, $U^{(1)}, U^{(2)}, \ldots, U^{(N)}$ are orthogonal matrices spanning the column space described in (1), and $\times_k$ denotes mode-k multiplication. The core tensor signifies the interaction of mode matrices and is generally not diagonal when the tensor order is greater than two.

3.2 Orthogonal Groups

Matrix Lie groups arise in various kinds of non-Euclidean geometry (Belinfante and Kolman, 1972). The General Linear Group[1] $\mathcal{GL}(n)$ is a set of nonsingular $n \times n$ matrices defined as:

$$\mathcal{GL}(n) = \{Y \in \mathbb{R}^{n \times n} : \det(Y) \neq 0\}.$$

The $\mathcal{GL}(n)$ is closed under a group operation, that is, matrix multiplication. This is because the product of two nonsingular matrices is a nonsingular matrix. Of practical importance here is the fact that elements of $\mathcal{GL}(n)$ are full rank and thus their row and column spaces span $\mathbb{R}^n$. A further subgroup of $\mathcal{GL}(n)$ is the orthogonal group denoted as:

$$\mathcal{O}(n) = \{Y \in \mathbb{R}^{n \times n} : Y^T Y = I\}.$$

It is known that the determinants of orthogonal matrices can be either $+1$ or -1 where the matrices with the determinant of 1 are rotation matrices and the matrices with the determinant of -1 are reflection matrices.

3.3 Stiefel Manifolds

The Stiefel manifold $\mathcal{V}_{n,p}$ is a set of $n \times p$ orthonormal matrices defined as:

$$\mathcal{V}_{n,p} = \{Y \in \mathbb{R}^{n \times p} : Y^T Y = I\}.$$

The Stiefel manifold $\mathcal{V}_{n,p}$ can be considered a quotient space of $\mathcal{O}(n)$ so we can identify an isotropy subgroup H of $\mathcal{O}(n)$ expressed as $\left\{ \begin{bmatrix} I_p & 0 \\ 0 & Q_{n-p} \end{bmatrix} : Q_{n-p} \in \mathcal{O}(n-p) \right\}$ where the isotropy subgroup leaves the element unchanged. Thus, the Stiefel manifold can be expressed as $\mathcal{V}_{n,p} = \mathcal{O}(n) \,/\, \mathcal{O}(n-p)$. From a group theory point of view, $\mathcal{O}(n)$ is a Lie group and $\mathcal{O}(n-p)$ is its subgroup so that $\mathcal{O}(n) \,/\, \mathcal{O}(n-p)$ represents the orbit space. In other words, $\mathcal{V}_{n,p}$ is the quotient group of $\mathcal{O}(n)$ by $\mathcal{O}(n-p)$.

1. In this paper, we are only interested in the field of real number $\mathbb{R}$. Unitary groups may be considered in other contexts.

3.4 Grassmann Manifolds

When we impose a group action of $O(n)$ onto the Stiefel manifold, this gives rise to the equivalence relation between orthogonal matrices so that the elements of Stiefel manifolds are rotation and reflection invariant. In other words, elements are considered being equivalent if there exists a $p \times p$ orthogonal matrix Q_p which maps one point into the other. This equivalence relation can be written as:

$$\lfloor Y \rfloor = \{YQ_p : Q_p \in O(n)\} \tag{2}$$

where $\lfloor Y \rfloor$ is an element on the Grassmann manifold. Therefore, the Grassmann manifold $\mathcal{G}_{n,p}$ is a set of p-dimensional linear subspaces of $\mathbb{R}^n$ and its isotropy subgroup composes all elements of $\left\{ \begin{bmatrix} Q_p & 0 \\ 0 & Q_{n-p} \end{bmatrix} : Q_p \in O(p), Q_{n-p} \in O(n-p) \right\}$. The quotient representation of Grassmann manifolds is expressed as $\mathcal{G}_{n,p} = O(n) \,/\, (O(p) \times O(n-p)) = \mathcal{V}_{n,p} \,/\, O(p)$. As such, the element of the Grassmann manifold represents the orbit of a Stiefel manifold under the group action of orthogonal groups. More details on the treatment of Grassmann manifolds can be found in Edelman et al. (1998) and Absil et al. (2008).

4. Elements of Product Manifolds

This section discusses the elements of product manifolds in the context of gesture recognition. We illustrate the essence of product manifolds and the factorization of action videos. Further, we describe the realization of geodesic distance on the product manifold and its use for action classification.

4.1 Product Manifolds

A product manifold can be recognized as a complex compound object in a high dimensional space composed by a set of lower dimensional objects. For example, the product of a line with elements y in $\mathbb{R}^1$ and a solid circle with elements x in $\mathbb{R}^2$ becomes a cylinder with elements (x, y) in $\mathbb{R}^3$ as shown in Figure 2. Formally, this product topology can be expressed as:

$$\begin{aligned} I &= \{y \in \mathbb{R} : |y| < 1\}, \\ D^2 &= \{x \in \mathbb{R}^2 : |x| < 1\}, \\ D^2 \times I &= \{(x,y) \in \mathbb{R}^2 \times \mathbb{R} : |x| < 1 \text{ and } |y| < 1\} \end{aligned}$$

where D^2 and I are viewed as topological spaces.

The cylinder may be equally well interpreted as either a circle of intervals or an interval of circles. In general, a product manifold may be viewed as the cross section of lower dimensional objects. Formally, let $\mathcal{M}_1$, $\mathcal{M}_2$, ... , $\mathcal{M}_q$ be a set of manifolds. The set $\mathcal{M}_1 \times \mathcal{M}_2 \times \ldots \times \mathcal{M}_q$ is called the product of the manifolds where the manifold topology is equivalent to the product topology. Hence, a product manifold is defined as:

$$\begin{aligned} \mathcal{M} &= \mathcal{M}_1 \times \mathcal{M}_2 \times \cdots \times \mathcal{M}_q \\ &= \{(x_1, x_2, \ldots, x_q) : x_1 \in \mathcal{M}_1, x_2 \in \mathcal{M}_2, \ldots, x_q \in \mathcal{M}_q\} \end{aligned}$$

where $\times$ denotes the Cartesian product, $\mathcal{M}_k$ represents a factor manifold (a topological space), and x_k is an element in $\mathcal{M}_k$. Note that the dimension of a product manifold is the sum of all factor manifolds (Lee, 2003).

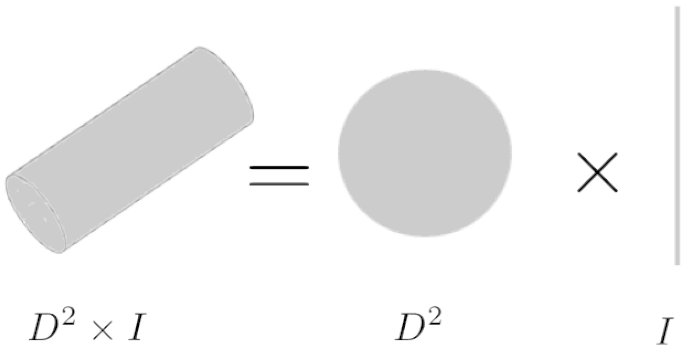


Figure 2: An example of a product manifold: A cylinder is a cross product of a circle and an interval.

The product manifold naturally expresses a compound topological space associated with a number of factor manifolds. For action video classification, third order data tensors are manifested as elements on three factor manifolds. As such, video data can be abstracted as points and classified on a product manifold.

4.2 Factorization in Product Spaces

As discussed in Section 3, HOSVD operates on the unfolded matrices (modes) via matrix unfolding in which the variation of each mode is captured by HOSVD. However, the traditional definition of HOSVD does not lead to a well-defined product manifold in the context of action recognition.

We observe that the column of every unfolded matrix $A_{(k)}$ is composed by multiple orders from the original data tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. This fact can also be observed in Figure 1. Let m be the dimension of the columns, $I_1 \times I_2 \times \cdots \times I_{k-1} \times I_{k+1} \cdots \times I_N$, and p be the dimension of the rows, I_k, for an unfolded matrix $A_{(k)}$. We can then assume that the dimension of the columns is greater than the dimension of the rows due to the nature of matrix unfolding for action videos, that is, $m > p$. This implies that the unfolded matrix $A_{(k)}$ only spans p dimensions.

Alternatively, one can factorize the data tensor using the right orthogonal matrices (Lui et al., 2010). From the context of action videos, the HOSVD can be expressed as:

$$\mathcal{A} = \hat{\mathcal{S}} \times_1 V^{(1)}_{\text{horizontal-motion}} \times_2 V^{(2)}_{\text{vertical-motion}} \times_3 V^{(3)}_{\text{appearance}}$$

where $\hat{\mathcal{S}}$ is a core tensor, $V^{(k)}$ are the orthogonal matrices spanning the row space with the first p rows associated with non-zero singular values from the unfolded matrices, respectively. Because we are performing action recognition on videos, the orthogonal matrices, $V^{(1)}_{\text{horizontal-motion}}$, $V^{(2)}_{\text{vertical-motion}}$, and $V^{(3)}_{\text{appearance}}$, correspond to horizontal motion, vertical motion, and appearance. Figure 3 shows some examples from the action decomposition.

From the factorization of HOSVD, each $V^{(k)}$ is a tall orthogonal matrix, thus it is an element on a Stiefel manifold. When we impose a group action of the orthogonal group, elements on the Stiefel manifold become rotation and reflection invariant. In other words, they are elements on the Grassmann manifold described in (2). As such, the action data are represented as the orbit of

elements on the Stiefel manifold under the rotation and reflection actions with respect to appearance and dynamics. Section 5 will discuss how we benefit from imposing such a group action on the Stiefel manifold.

4.3 Geodesic Distance on Product Manifolds

The geodesic in a product manifold $\mathcal{M}$ is the product of geodesics in $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_q$ (Ma et al., 1998; Begelfor and Werman, 2006). Hence, for any differentiable curve γ parametrized by t, we have $\gamma(t) = (\gamma_i(t), \gamma_j(t))$ where γ is the geodesic on the product manifold $\mathcal{M}$, and γ_i and γ_j are the geodesics on the factor manifold $\mathcal{M}_i$ and $\mathcal{M}_j$ respectively. From this observation, the geodesic distance on a product manifold may be expressed as a Cartesian product of canonical angles computed by factor manifolds.

Just as there are alternatives to induce a metric on a Grassmann manifold (Edelman et al., 1998) using canonical angles, the geodesic distance on a product manifold could also be defined in different ways. One possible choice is the chordal distance that approximates the geodesic via a projection embedding (Conway et al., 1996). Consequently, we define the geodesic distance on a product manifold as:

$$d_{\mathcal{M}}(\mathcal{A}, \mathcal{B}) = \| \sin\Theta \|_2 \tag{3}$$

where $\mathcal{A}$ and $\mathcal{B}$ are the N order data tensors, $\Theta = (\theta_1, \theta_2, \ldots, \theta_N)$, and $\theta_k \in \mathcal{G}_k$ is a set of canonical angles (Björck and Golub, 1973) computed independently from each factor (Grassmann) manifold.

This development of geodesic distance on the product manifold can be related back to our cylinder example where a circle in $\mathbb{R}^2$ and a line in $\mathbb{R}^1$ form a cylinder in $\mathbb{R}^3$ where $\mathbb{R}^3$ is the product space. Recall that a Grassmann manifold is a set of p-dimensional linear subspaces. In analogous fashion, the product of a set of $p_1, p_2, \ldots, p_N$ linear subspaces forms a set of product subspaces whose dimension is $(p_1 + p_2 + \ldots + p_N)$. The product subspaces are the elements on a product manifold. This observation is consistent with the Θ in (3) where the number of canonical angles agrees with the dimension of product subspaces on the product manifold.

Note that canonical angles θ_k are measured between $V_{\mathcal{A}}^{(k)}$ and $V_{\mathcal{B}}^{(k)}$ where each is an orthogonal matrix spanning the row space associated with nonzero singular values from a mode-k unfolded matrix. As such, an N order tensor in $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ would span N row spaces in $I_1, I_2, \ldots, I_N$, respectively, and the dimension of a product manifold is the sum of each order of a data tensor, that is, $(\sum_{i=1}^{N} = I_1 + I_2 + \ldots + I_N)$.

5. The Product Manifold Representation

The tensor representation on a product manifold models the variations in both space and time for action videos. Specifically, the product manifold captures the individual characteristics of spatial and temporal evolution through three factor manifolds. As such, one factor manifold is acquiring the change in time, resulting in the appearance (XY) component, while the other two capture the variations in horizontal and vertical directions, demonstrating the horizontal motion (YT) and vertical motion (XT). Putting all these representations together, geodesic distance on the product manifold measures the changes in both appearance and dynamics.

The aim of this section is to illustrate how the product manifold characterizes appearance and dynamics from action videos. To visualize the product manifold representation, let us consider

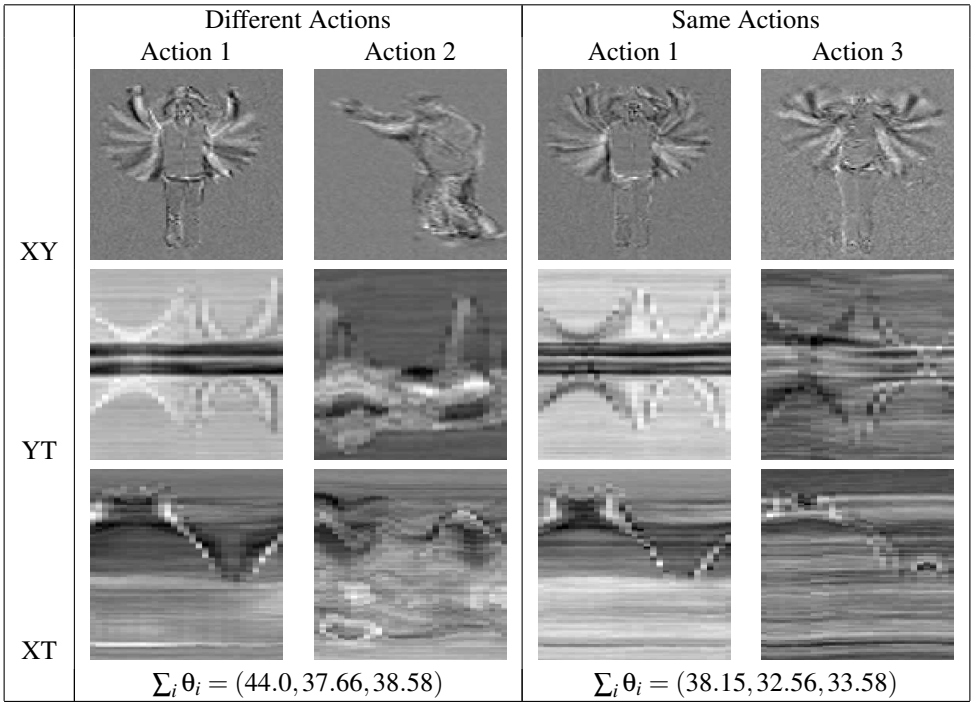


Figure 3: Examples of appearance and motion changes where the first row is the overlay appearances, the second and third rows are the overlay horizontal motion and vertical motion, and the bottom row gives the sum of canonical angles computed from each factorization of the pairs of canonical variates.

the example given in Figure 3 where the first row expresses the pairs of overlay appearance (XY) canonical variates, the second and third rows reveal the pairs of overlay horizontal motion (YT) and vertical motion (XT) canonical variates, and the bottom row gives the sum of canonical angles computed from the pairs of canonical variates. Note that the canonical variates are elements on Stiefel manifolds. In the first column, two distinct actions are factorized to canonical variates. We can see that all canonical variates exhibit very different characteristics in both appearance and motions. On the contrary, the second column shows the same action performed by different actors and the canonical variates are much more similar than the first column, resulting in smaller canonical angles overall.

One of the advantages of the product manifold representation is that actions do not need to be aligned in temporal space. To demonstrate this merit, we permute the frame order from action 3 denoted as action 4 and match it to action 1. Figure 4 shows the pairs of canonical variates between actions (1, 3) and actions (1, 4). We should first note that the appearance (XY) of action 3 and action 4 span the same space despite the visual differences resulting in the identical sum of canonical angles 38.15. This is because elements on the Grassmann manifold are rotation and reflection invariant from elements of the Stiefel manifold. This important concept is illustrated in Figure 5 where the exchange matrix $O(p)$ maps the appearance of action 4 to the appearance of action 3.

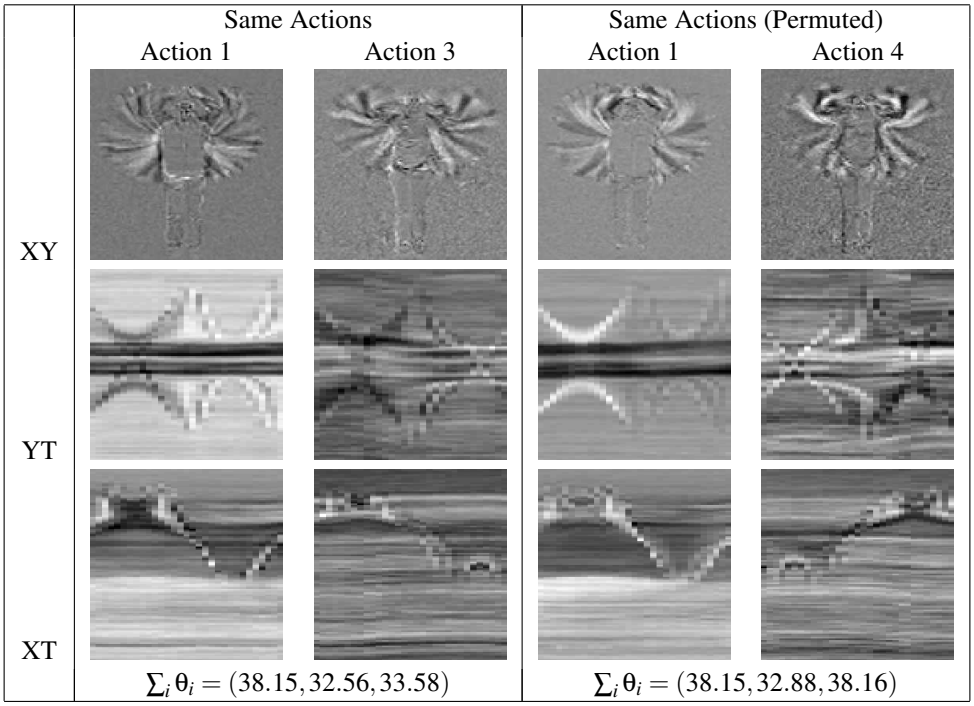


Figure 4: Examples of appearance and motion changes where Action 4 is a permuted version of Action 3. The canonical angles for the appearance indicates that the action is not affected by the frame order.

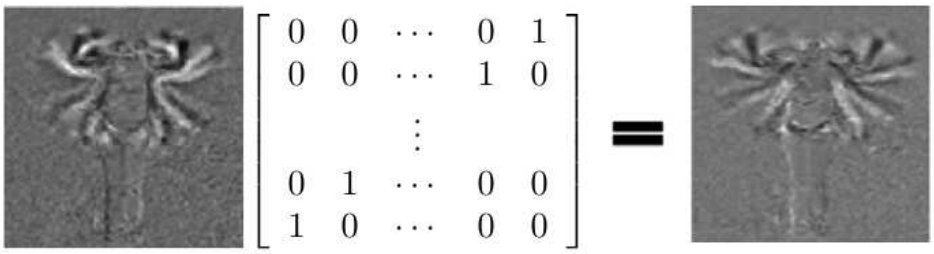


Figure 5: The characterization of the Grassmann manifold where a point is mapped to another point on the Stiefel manifold via an exchanged matrix. The group action is $(X, Q) \longmapsto XQ$ where $X \in \mathcal{V}_{n,p}$ and $Q \in O(p)$ so that elements on the Grassmann manifold are closed under the orthogonal matrix multiplication.

In the example given in Figure 4, the most prominent change is related to the motion in vertical directions (XT) between action 3 and action 4. This arises from the fact that the change of motion mostly occurs in the vertical direction when we permute the order of the video frames from action 3. Consequently, the sum of canonical angles in XT varies from 33.58 to 38.16 which is less similar to action 1. When we identify a waving hand moving from top to bottom and from bottom to

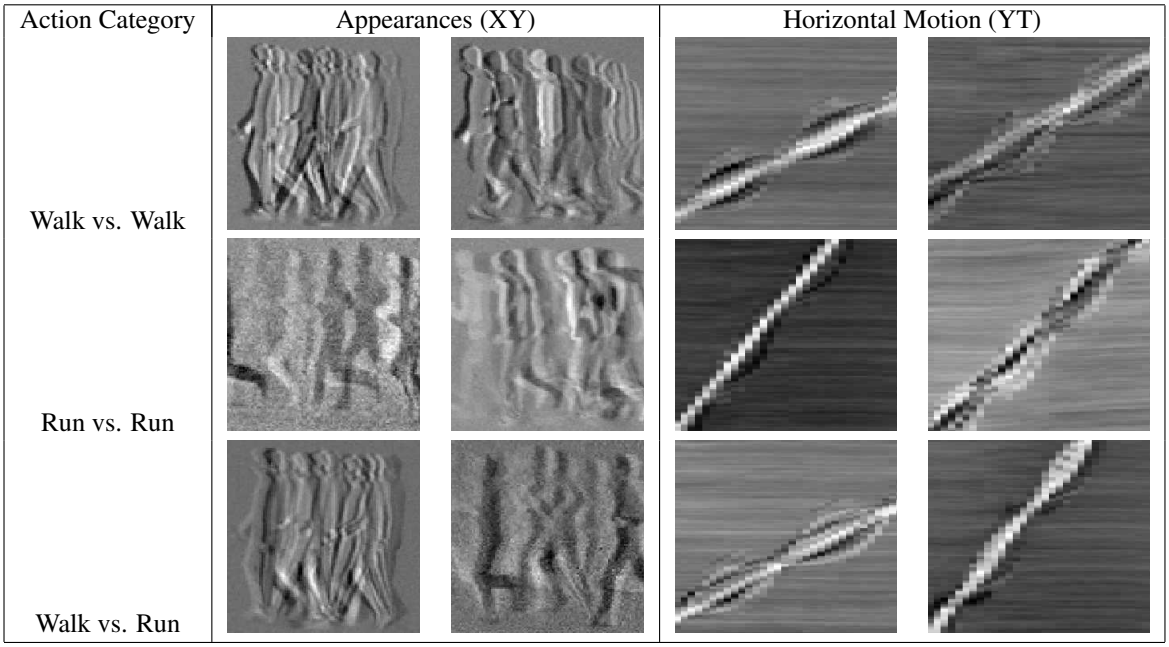


Figure 6: Illustration of capturing the rate of actions. The first column shows the change of appearance while the second column reveals the change of horizontal motion where the slopes exhibit the rate of motion.

top, the vertical motion is the key feature. Otherwise, a simple cyclical search can compensate such variation. As a result, the product manifold representation is resilient to misregistration in the temporal space for appearance while keeping the dynamics intact.

Another intriguing attribute of the product manifold representation is its ability to capture the rate of motion, which is useful in identifying some particular actions. Figure 6 shows the pairs of canonical variates of two similar actions - walking and running. First, we note that there is little information from the vertical motion since the movements of walking and running occur horizontally. The appearance differences between walking and running are not substantial, which is shown in the first column of Figure 6. The key information between walking and running is embedded in the horizontal motion (YT). While the structure of horizontal motion between walking and running is similar exhibiting a line-like pattern, they have very distinct slopes shown in the horizontal motion column of Figure 6. These slopes characterize the rate of motion and are the key factors in recognizing these types of actions. In particular, when walking and running are compared depicted in the third row of Figure 6, the idiosyncratic aspect is captured by the rate of horizontal motion. In general, it is possible to see the rate of motion through both motion representations depending on the type of actions.

6. Statistical Modeling

Least squares regression is one of the fundamental techniques in statistical analysis. It is simple and often outperforms complicated models when the number of training samples is small (Hastie

et al., 2001). Since video data do not reside in Euclidean space, we pay attention to the manifold structure. Here, we introduce a nonlinear regression framework in non-Euclidean space for gesture recognition. We formulate least squares regression as a composite function; as such, both domain and range values are constrained on a manifold through the regression process. The least squares fitted elements from a training set can then be exploited for gesture recognition.

6.1 Linear Least Squares Regression

Before we discuss the geometric extension, we will first review the standard form of least squares fitting. We consider a regression problem $y = A\beta$ where $y \in \mathbb{R}^n$ is the regression value, $A([a_1|a_2|\cdots|a_k]) \in \mathbb{R}^{n\times k}$ is the training set, and $\beta \in \mathbb{R}^k$ is the fitting parameter. The residual sum-of-squares can be written as:

$$R(\beta) = \| y - A\beta \|^2 \tag{4}$$

and the fitting parameter β can be obtained by minimizing the residual sum-of-squares error from (4). Then, we have

$$\hat{\beta} = (A^T A)^{-1} A^T y.$$

The regressed pattern from the training set has the following form

$$\hat{y} = A\hat{\beta} = A(A^T A)^{-1} A^T y. \tag{5}$$

The key advantage of least squares fitting is its simplicity and it intuitively measures the best fit of the data.

6.2 Least Squares Regression on Manifolds

Non-Euclidean geometry often arises in computer vision applications. We consider the nonlinear nature of space and introduce a geometric framework for least squares regression. First, we extend the linear least squares regression from (5) to a nonlinear form by incorporating a kernel function shown in the following

$$A(A \star A)^{-1}(A \star y)$$

where $\star$ is a nonlinear similarity operator. Obviously, $\star$ is equal to $x^T y$ in the linear case. In this paper, we employ the RBF kernel given as:

$$x \star y \quad = \quad \exp(-\frac{\sum_k \theta_k}{\sigma}) \tag{6}$$

where x and y are the elements on a factor manifold, θ_k is the canonical angle computed from the factor manifold, and σ is set to 2 in all our experiments. While other kernel functions can be considered, our goal is to demonstrate our geometric framework and choose a commonly used RBF kernel operator.

Considering the similarity measure given in (6), the regression model becomes three sub-regression estimators given by

$$\psi^{(k)}(y) = A^{(k)}(A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)}) \tag{7}$$

Algorithm 1 Weighted Karcher Mean Computation

1: Initialize a base point μ on a manifold
2: **while** not converged **do**
3: Apply the logarithmic map to the training samples Y_i to the base point μ
4: Compute the weighted average on the tangent space at the base point μ
5: Update the base point μ by applying the exponential map on the weighted average
6: **end while**

where k denotes the mode of unfolding, $A^{(k)}$ is a set of orthogonal matrices factorized from HOSVD, and $y^{(k)}$ is an orthogonal matrix from the unfolded matrix.

To gain a better insight on the regression model, we explore the geometrical interpretation from (7). Given p training instances, the first element, $A^{(k)}$, is a set of factorized training samples residing on a manifold. Furthermore, $(A^{(k)} \star A^{(k)})^{-1}$ produces a $p \times p$ matrix from the training set and $(A^{(k)} \star y^{(k)})$ would create a $p \times 1$ vector. Therefore, the rest of the regression provides a weighting vector characterizing the training data on a factor manifold as:

$$\mathrm{w} = (A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)})$$

where the weighting vector is in a vector space, that is, $\mathrm{w} \in \mathcal{V}$.

Now, we have a set of factorized training samples, $A^{(k)}$, on a manifold and a weighting vector, w, in a vector space. To incorporate these two elements with the least squares fitting given in (7), we make a simple modification and reformulate the regression as follows

$$\Psi^{(k)}(y) = A^{(k)} \bullet (A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)}) \tag{8}$$

where $\bullet$ is an operator mapping points from a vector space back to a factor manifold. By introducing an additional operator, we ensure that both the domain values $y^{(k)}$ and the range values $\Psi^k(y)$ reside on a manifold. From a function composition point of view, the proposed regression technique can be viewed as a composition map $\mathcal{G} \circ \mathcal{H}$ where $\mathcal{H} : \mathcal{M} \longrightarrow \mathcal{V}$ and $\mathcal{G} : \mathcal{V} \longrightarrow \mathcal{M}$ where $\mathcal{M}$ is a manifold and $\mathcal{V}$ is a vector space.

One possible way to realize the composition map, $\mathcal{G} \circ \mathcal{H}$, is to employ the tangent space and modify the Karcher mean (Karcher, 1977). The computation of Karcher mean considers the intrinsic geometry and iteratively minimizes the distance between the updated mean and all data samples via the tangent space. Since w is the weighting vector, it naturally produces the weight between training samples. All we need is to apply the weighting vector to weight the training samples on a factor manifold. This is equivalent to computing the weighted Karcher mean, which is an element of a manifold.

So far, our geometric formulation on least squares regression is very general. To make it specific for gesture recognition, we impose rotation and reflection invariance to the factorized element $V^{(k)}$ such that they are elements on a Grassmann manifold and the computation of the weighted Karcher mean can be realized. Here, we sketch the pseudo-code in Algorithm 1. As Algorithm 1 illustrates, the first step is to initialize a base point on a manifold. To do so, we compute the weighted average from the training samples in Euclidean space and project it back to the Grassmann manifold using QR factorization. Then, we iteratively update the base point on the Grassmann manifold. The update procedure involves the standard logarithmic map and the exponential map on Grassmann

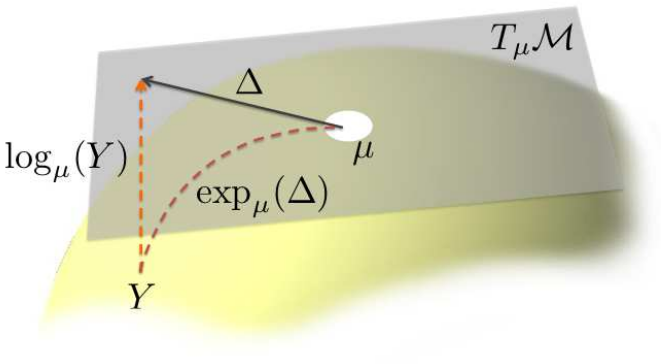


Figure 7: An illustration of logarithmic and exponential maps where Y and μ are points on a manifold, Δ is the tangent vector, and $T_\mu\mathcal{M}$ is the tangent space at μ.

manifolds (Edelman et al., 1998) described as follows

$$\log_\mu(Y_i) \quad = \quad U_1\Theta_1V_1^T$$

where μ is the base point for the tangent space, Y_i is a training instance factorized from the Grassmann manifold, $\mu_\perp\mu_\perp^TY_i(\mu^TY_i)^{-1} = U_1\Sigma_1V_1^T$, $\Theta_1 = \arctan(\Sigma_1)$, and $\mu_\perp$ is the orthogonal complement to μ.

$$\exp_\mu(\Delta) \quad = \quad \mu V_2\cos(\Sigma_2) + U_2\sin(\Sigma_2)$$

where Δ is the weighted tangent vector at μ and $\Delta = U_2\Sigma_2V_2^T$. From a geometric point of view, the logarithmic operator maps a point on a manifold to a tangent space whereas the exponential map projects a point in the tangent space back to the manifold. A pictorial illustration is given in Figure 7. In addition, the Karcher mean calculation exhibits fast convergence (Absil et al., 2004). Typically, convergence can be reached within 10 iterations in our experiments. A sample run is depicted in Figure 8 where expeditious reduction of residuals occurs in the first few iterations.

To perform gesture recognition, a set of training videos is collected. All videos are normalized to a standard size. During the test phase, the category of a query video is determined by

$$j^* = \operatorname*{argmin}_j \mathcal{D}(Y, \Psi_j(Y))$$

where Y is a query video, Ψ_j is the regression instance for the class j given in (8), and $\mathcal{D}$ is a geodesic distance measure. Because the query gesture Y and the regression instance are realized as elements on a product manifold, we employ the chordal distance given in (3) for gesture classification.

In summary, the least squares regression model applies HOSVD on a query gesture Y and factorizes it to three sub-regression models ($\Psi_j^{(1)}$, $\Psi_j^{(2)}$, $\Psi_j^{(3)}$) on three Grassmann manifolds where regressions are performed. The distance between the regression output and query is then characterized on a product manifold; gesture recognition is achieved using the chordal distance. We note that

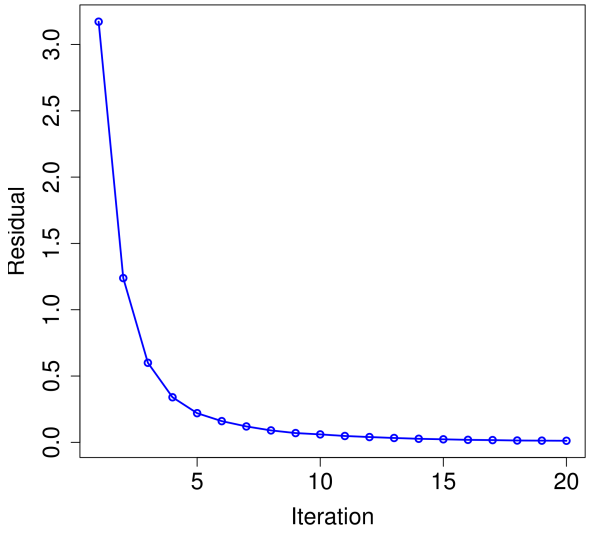


Figure 8: The residual values of tangent vectors.

our least squares framework is applicable to many matrix manifolds as long as the logarithmic and exponential maps are well-defined. Furthermore, when the kernel operator is $\star = x^T y$, $\log_x(y) = y$, and $\exp_x(\Delta) = x + \Delta$, the regression model in (8) becomes the canonical least squares regression in Euclidean space.

When statistical models exhibit high variance, shrinkage techniques are often applied (Hastie et al., 2001). We see that a simple regularization parameter turns least squares regression into ridge regression. This observation can also be applied to our non-Euclidean least squares regression framework.

7. Experimental Results

This section summarizes our empirical results and demonstrates the proficiency of our framework on gesture recognition. To facilitate comparison, we first evaluate our method using two publicly available gesture data sets namely Cambridge hand-gesture (Kim and Cipolla, 2009) and UMD Keck body-gesture (Lin et al., 2009). We further extend our method to the one-shot-learning gesture challenge (CHALEARN, 2011). Our experiments reveal that not only does our method perform well on the standard benchmark data sets, but also it generalizes well on the one-shot-learning gesture challenge.

7.1 Cambridge Hand-Gesture Data Set

Our first experiment is conducted using the Cambridge hand-gesture data set which has 900 video sequences with nine different hand gestures (100 video sequences per gesture class). The gesture data are collected from five different illumination sets labeled as Set1, Set2, Set3, Set4, and Set5. Example gestures are shown in Figure 9.

Figure 9: Hand gesture samples. Flat-Leftward, Flat-Rightward, Flat-Contract, Spread-Leftward, Spread-Rightward, Spread-Contract, V-Shape-Leftward, V-Shape-Rightward, and V-Shape-Contract.

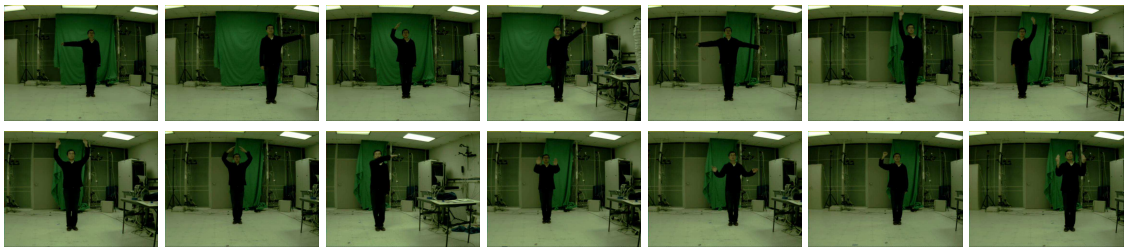


Figure 10: Body gesture samples. First row: Turn Left, Turn Right, Attention Left, Attention Right, Attention Both, Stop Left, and Stop Right. Second row: Stop Both, Flap, Start, Go Back, Close Distance, Speed Up, and Come Near.

We follow the experimental protocol employed by Kim and Cipolla (2009) where Set5 is the target set, and Set1, Set2, Set3, and Set4 are the test sets. The target Set5 is further partitioned into a training set and validation set (90 video sequences in the training set and 90 video sequences in the validation set). We employ five random trials in selecting the training and validation videos in Set5. The recognition results are summarized in Table 1 where the classification rates are the average accuracy obtained from five trial runs followed by the standard deviation. As Table 1 shows, our method performs very well across all illumination sets obtaining 91.7% average classification rate.

7.2 UMD Keck Body-Gesture Data Set

The UMD Keck body-gesture data set consists of 14 naval body gestures acquired from both static and dynamic backgrounds. In the static background, the subjects and the camera remain stationary whereas the subjects and the camera are moving in the dynamic environment during the performance of the gesture. There are 126 videos collected from the static scene and 168 videos taken from the dynamic environment. Example gestures are given in Figure 10.

We follow the experimental protocol proposed by Lin et al. (2009) for both static and dynamic settings. The region of interest is tracked by a simple correlation filter. In the static background, the protocol is leave-one-subject-out (LOSO) cross-validation. As for the dynamic environment, the gestures acquired from the static scene are used for training while the gestures collected from the dynamic environment are the test videos. The recognition results for both static and dynamic backgrounds are reported in Table 2. We can see that our method is competitive to the current state-of-the-art methods in both protocols. One of the key advantages of our method is its direct use of raw pixels while the prototype-tree (Lin et al., 2009), MMI-2+SIFT (Qiu et al., 2011), and CC K-

Method	Set1	Set2	Set3	Set4	Total
Graph Embedding (Yuan et al., 2010)	-	-	-	-	82%
TCCA (Kim and Cipolla, 2009)	81%	81%	78%	86%	82±3.5%
DCCA+SIFT (Kim and Cipolla, 2007)	-	-	-	-	85±2.8%
RLPP (Harandi et al., 2012)	86%	86%	85%	88%	86.3±1.3%
TB$\{\mathcal{V}_{n,p}\}$ (Lui, 2012b)	88%	84%	85%	87%	86±3.0%
PM 1-NN (Lui et al., 2010)	89%	86%	89%	87%	88±2.1%
Our Method	93%	89%	91%	94%	91.7±2.3%

Table 1: Recognition results on the Cambridge Hand-Gesture data set (Five trial runs).

Method	Static Setting	Dynamic Setting
HOG3D (Bilinski and Bremond, 2011)	-	53.6%
Shape Manifold (Abdelkadera et al., 2011)	82%	-
MMI-2+SIFT (Qiu et al., 2011)	95%	-
CC K-Means (Jiang et al., 2012))	-	92.9%
Prototype-Tree (Lin et al., 2009)	95.2%	91.1%
TB$\{\mathcal{V}_{n,p}\}$ (Lui, 2012b)	92.1%	91.1%
PM 1-NN (Lui et al., 2010)	92.9%	92.3%
Our Method	94.4%	92.3%

Table 2: Recognition results on the UMD Keck Body-Gesture data set.

means (Jiang et al., 2012) methods operate on silhouette images which require image segmentation prior to classification. This makes our representation more generic.

7.3 One-Shot-Learning Gesture Challenge

Microsoft Kinect™ has recently revolutionized gesture recognition by providing both RGB and depth images. To facilitate the adaptation to new gestures, CHALEARN (Guyon et al., 2012) has organized a one-shot-learning challenge for gesture recognition.

The key aspect of one-shot-learning is to perform machine learning on a single training example. As such, intra-class variability needs to be modeled from a single example or learned from different domains. While traditional machine learning techniques require a large amount of training data to model the statistical distribution, least squares regression appears to be more robust when the size of training samples is limited (Hastie et al., 2001). We employ our least squares regression framework and model the intra-class variability by synthesizing training examples from the original training instance. Consequently, we apply the same regression framework on the product manifold to the one-shot-learning gesture challenge.

One of the gesture variations is performing gesture positions. Our initial studies for frame alignment did not yield positive results due in part to the incidental features of the upper body. Since gesture positions are the key source of variations, we synthesize training examples for translational instances on both RGB and depth images. The synthesized examples are generated by shifting the entire action video horizontally and vertically. Specifically, we synthesize two vertically (up/down) and four horizontally (left/right) translated instances along with the original training example. As such, we have seven training instances for RGB and depth images, respectively. We stress that we

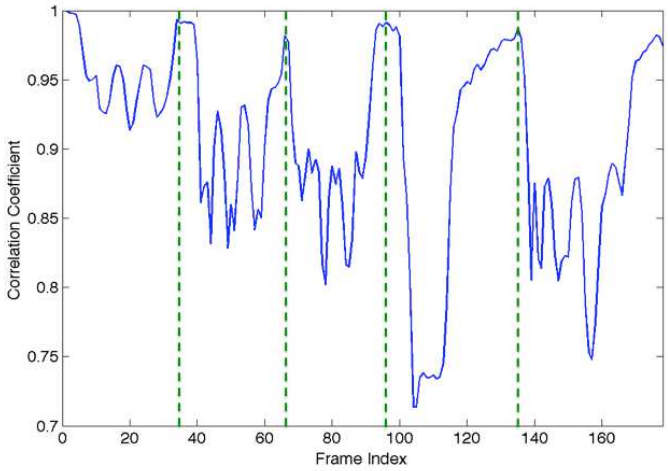


Figure 11: An illustration of temporal segmentation where the dash lines indicate the peak locations and the resting frames from the action sequence.

do not apply any spatial segmentation or intensity normalization to video data; alignment is the only variation that we synthesize for one-shot-learning. Our experiments on the training batches indicate that there is about 2% gain by introducing the translational variations.

We assess the effectiveness of the proposed framework on the development data set for the one-shot-learning gesture challenge. The development data set consists of 20 batches of gestures. Each batch is made of 47 gesture videos and split into a training set and a test set. The training set includes a small set of vocabulary spanning from 8 to 15 gestures. Every test video contains 1 to 5 gestures. Detailed descriptions of the gesture data can be found in Guyon et al. (2012).

Since the number of gestures varies for test videos, we perform temporal segmentation to localize each gesture segment. It is supposed that the actor will return to the resting position before performing a new gesture. Thus, we employ the first frame as a template and compute the correlation coefficient with subsequent frames. We can then localize the gesture segments by identifying the peak locations from the correlations; the number of gestures is the number of peaks + 1. An illustration of temporal segmentation is given in Figure 11 where the peak locations provide a good indication for the resting frames. Furthermore, we fix the spatial dimension to 32×32 and dynamically determine the number of frames by selecting 90% of the PCA energy from each training batch. Linear interpolation is then applied to normalize the video length.

The recognition performance is evaluated using the Levenshtein distance (Levenshtein, 1966), also known as edit distance. Table 3 shows the average errors over 20 batches. As Table 3 reveals, our method significantly outperforms the baseline algorithm (CHALEARN, 2011) and achieves 28.73% average Levenshtein distance per gesture on the development data set. Our method also ranks among the top algorithms in the gesture challenge (Guyon et al., 2012). This illustrates that our method can be effectively adopted for one-shot-learning from the traditional supervised learning paradigm.

	Baseline		Our Method	
Batch	TeLev%	TeLen%	TeLev%	TeLen%
devel01	53.33	12.22	13.33	4.44
devel02	68.89	16.67	35.56	14.44
devel03	77.17	5.43	71.74	20.65
devel04	52.22	30.00	10.00	2.22
devel05	43.48	10.87	9.78	7.61
devel06	66.67	17.78	37.78	14.44
devel07	81.32	19.78	18.68	3.30
devel08	58.43	12.36	8.99	5.62
devel09	38.46	9.89	13.19	1.10
devel10	75.82	21.98	50.55	1.10
devel11	67.39	18.48	35.87	2.17
devel12	52.81	5.62	22.47	4.49
devel13	50.00	17.05	9.09	2.27
devel14	73.91	22.83	28.26	3.26
devel15	50.00	8.70	21.74	0.00
devel16	57.47	17.24	31.03	6.90
devel17	66.30	32.61	30.43	4.35
devel18	70.00	28.89	40.00	11.11
devel19	71.43	15.38	49.45	3.30
devel20	70.33	36.26	35.16	12.09
Average	62.32	18.01	28.73	6.24

Table 3: Recognition results on the development data for the one-shot-learning challenge where TeLev is the sum of the Levenshtein distance divided by the true number of gestures and TeLen is the average error made on the number of gestures.

While our method performs well on the one-shot-learning gesture challenge, it is not a complete system yet. There are three particular batches that cause difficulties for our algorithm. These batches are devel03, devel10, and devel19 where the example frames are shown in Figure 12. These three batches share a common characteristic that the gesture is only distinguishable by identifying the hand positions. Since we do not have a hand detector, the gross motion dominates the whole action causing it to be confused with other similar gestures.

Another source of errors is made by the temporal segmentation. While the actor is supposed to return to the resting position before performing a new gesture, this rule has not always been observed. As a result, such variation introduces a mismatch between the template and subsequent frames resulting errors in partitioning the video sequence. The large error in devel03 is caused by the need for hand positions and temporal segmentation. Future work will focus on combining both appearance and motion for temporal segmentation.

Nevertheless, the experimental results from the Cambridge hand-gesture and the UMD Keck body-gesture data sets are encouraging. These findings illustrate that our method is effective in both hand gestures and body gestures. Once we have a reliable hand detector, we expect to further improve gesture recognition from a single training example. Currently, the processing time on 20 batches (2,000 gestures) including both training and testing is about 2 hours with a non-optimized MATLAB implementation on a 2.5GHz Intel Core i5 iMac.

Figure 12: Gesture samples on the one-shot-learning gesture challenge (devel03, devel10, and devel19).

8. Discussion

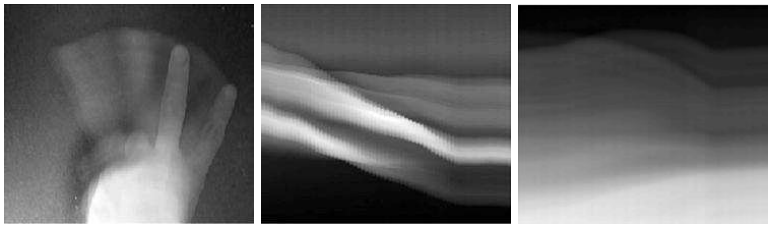
The proposed method is geometrically motivated. It decomposes a video tensor to three Stiefel manifolds via HOSVD where the orthogonal elements are imposed to Grassmannian spaces. As mentioned before, one of the key advantages of our method is its direct use of raw pixels. This gives rise to a practical and important question. *How robust can the raw pixel representation be against background clutter?*

To address this concern, we synthesize an illustrative example given in Figure 13. The first, second, and third columns depict the appearance, horizontal motion, and vertical motion of the gesture, respectively. A V-shape rightward gesture and a flat leftward gesture are shown in the first row and second row. We superpose a cluttered background on every frame of the flat leftward gesture exhibited in the third row. While the appearances between the uniform flat gesture and the cluttered flat gesture emerge differently, the deterioration on the dynamics is quite minimal. As a result, the gesture performed with the background clutter can still be discriminated against other gestures. Numerically, the sum of the canonical angles between the uniform (second row) and the cluttered background (third row) gestures is (56.09, 7.99, 9.17) resulting in a geodesic distance of 5.91 on the product manifold. In contrast, the sum of the canonical angles between the V-shape (first row) and the flat (second row) gestures is (76.35, 23.66, 18.42) yielding a geodesic distance of 8.29. In addition, when the V-shape gesture (first row) matches against the cluttered flat gesture (third row), the sum of the canonical angles is (76.09, 23.75, 18.84) and the geodesic distance is 8.31. This finding reveals that the geodesic distance between the uniform and cluttered background gestures are quite similar against inter-class gestures, while the geodesic distance is significantly smaller for the intra-class gestures. Hence, raw pixels can be directly exploited in our representation.

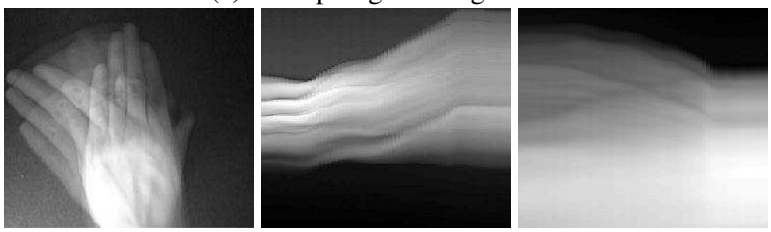
As technology advances, we can now separate the foreground and background more easily using a Kinect™ camera. We hypothesize that better recognition results may be obtained when the foreground gestures are extracted. On the other hand, our method can still perform gracefully when a cluttered background is present.

9. Conclusions

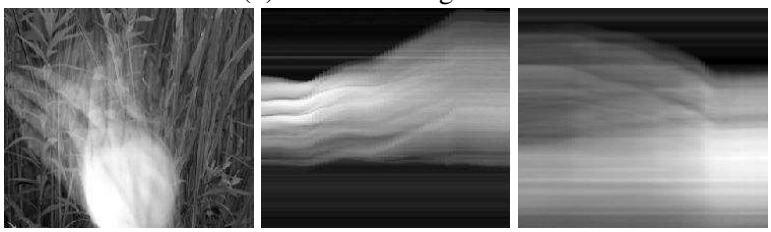
This paper promotes the importance of the underlying geometry of data tensors. We have presented a geometric framework for least squares regression and applied it to gesture recognition. We view action videos as third order tensors and impose them on a product manifold where each factor is



(a) V-shape rightward gesture.



(b) Flat leftward gesture.



(c) Superposed cluttered background on the flat leftward gesture.

Figure 13: The effect of background clutter. Appearance, horizontal motion, and vertical motion are depicted in the first, second, and third columns, respectively.

Grassmannian. The realization of points on these Grassmannians is achieved by applying HOSVD to a tensor representation of the action video. A natural metric is inherited from the factor manifolds since the geodesic on the product manifold is given by the product of the geodesic on the Grassmann manifolds.

The proposed approach provides a useful metric and a regression model based on latent geometry for action recognition. To account for the underlying geometry, we formulate least squares regression as a composite function. This formulation provides a natural extension from Euclidean space to manifolds. Experimental results demonstrate that our method is effective and generalizes well to the one-shot-learning scheme.

For longer video sequences, micro-action detection is needed which may be modeled effectively using HMM. Future work will focus on developing more sophisticated models for gesture recognition and other regression techniques on matrix manifolds for visual applications.

References

M. F. Abdelkadera, W. Abd-Almageeda, A. Srivastavab, and R. Chellappa. Gesture and action recognition via modeling trajectories on riemannian manifolds. *Computer Vision and Image Understanding*, 115(3):439–455, 2011.

P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematicae*, 80(2):199–220, 2004.

P.-A Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

E. Begelfor and M. Werman. Affine invariance revisited. In *IEEE Conference on Computer Vision and Pattern Recognition, New York*, 2006.

J.G.F. Belinfante and B. Kolman. *A Survey of Lie Groups and Lie Algebras with Applications and Computational Methods*. SIAM, 1972.

P. Bilinski and F. Bremond. Evaluation of local descriptors for action recognition in videos. In *ICVS*, 2011.

A. Bissacco, A. Chiuso, Y. Ma, and S. Soatto. Recognition of human gaits. In *IEEE Conference on Computer Vision and Pattern Recognition, Hawaii*, pages 270–277, 2001.

Å. Björck and G.H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, pages 579–594, 1973.

CHALEARN. Chalearn gesture dataset (cgd 2011), chalearn, california, 2011.

J.H. Conway, R.H. Hardin, and N.J.A. Sloane. Packing lines, planes, etc.: Packings in grassmannian spaces. *Experimental Mathematics*, 5(2):139–159, 1996.

A. Datta, Y. Sheikh, and T. Kanade. Modeling the product manifold of posture and motion. In *Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (in conjunction with ICCV)*, 2009.

L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (in conjunction with ICCV)*, 2005.

A. Edelman, R. Arias, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1998.

I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hammer, and H. J. E. Balderas. Chalearn gesture challenge: Design and first results. In *CVPR Workshop on Gesture Recognition*, 2012.

M. T. Harandi, C. Sanderson, A. Wiliem, and B. C. Lovell. Kernel analysis over riemannian manifolds for visual recognition of actions, pedestrians and textures. In *WACV*, 2012.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

Z. Jiang, Z. Lin, and L. Davis. Class consistent k-means: Application to face and action recognition. *Computer Vision and Image Understanding*, 116(6):730–741, 2012.

H. Karcher. Riemannian center of mass and mollifier smoothing. *Comm. Pure Appl. Math.*, 30(5): 509–541, 1977.

D. Kendall. Shape manifolds, procrustean metrics and complex projective spaces. *Bull. London Math. Soc.*, 16:81–121, 1984.

T-K. Kim and R. Cipolla. Gesture recognition under small sample size. In *Asian Conference on Computer Vision*, 2007.

T-K. Kim and R. Cipolla. Canonical correlation analysis of video volume tensors for action categorization and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8): 1415–1428, 2009.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3), September 2009.

B. Krausz and C. Bauckhage. Action recognition in videos using nonnegative tensor factorization. In *International Conference on Pattern Recognition*, 2010.

J. Lee. *Introduction to Smooth Manifolds*. Springer, 2003.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

R. Li and R. Chellappa. Group motion segmentation using a spatio-temporal driving force model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo. Robust visual tracking based on incremental tensor subspace learning. In *IEEE International Conference on Computer Vision*, 2007.

Z. Lin, Z. Jiang, and L. Davis. Recognizing actions by shape-motion prototype trees. In *IEEE International Conference on Computer Vision*, 2009.

Y. M. Lui. Advances in matrix manifolds for computer vision. *Image and Vision Computing*, 30 (6-7):380–388, 2012a.

Y. M. Lui. Tangent bundles on special manifolds for action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(6):930–942, 2012b.

Y. M. Lui and J. R. Beveridge. Grassmann registration manifolds for face recognition. In *European Conference on Computer Vision, Marseille, France*, 2008.

Y. M. Lui, J. R. Beveridge, and M. Kirby. Canonical stiefel quotient and its application to generic face recognition in illumination spaces. In *IEEE International Conference on Biometrics : Theory, Applications and Systems, Washington, D.C.*, 2009.

Y. M. Lui, J. R. Beveridge, and M. Kirby. Action classification on product manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition, San Francisco*, 2010.

Y. Ma, J. Košecká, and S. Sastry. Optimal motion from image sequences: A riemannian viewpoint, 1998. Technical Report No. UCB/ERL M98/37, EECS Department, University of California, Berkeley.

S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, Cybernetics - Part C: Applications and Reviews*, 37:311–324, 2007.

Q. Qiu, Z. Jiang, and R. Chellappa. Sparse dictionary-based representation and recognition of action attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

M. Rodriguez, J. Ahmed, and M. Shah. Action mach: A spatio-temporal maximum average correlation height filter for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

P. Saisan, G. Doretto, Y-N. Wu, and S. Soatto. Dynamic texture recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

P. Turaga and R. Chellappa. Locally time-invariant models of human activities using trajectories on the grassmannian. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

P. Turaga, S. Biswas, and R. Chellappa. The role of geometry for age estimation. In *IEEE International conference Acoustics, Speech and Signal Processing*, 2010.

M. A. O. Vasilescu. Human motion signatures: Analysis, synthesis, recognition. In *International Conference on Pattern Recognition, Quebec City, Canada*, pages 456–460, 2002.

M. A. O. Vasilescu and D. Terzopoulos. Multilinear image analysis for facial recognition. In *International Conference on Pattern Recognition, Quebec City, Canada*, pages 511–514, 2002.

A. Veeraraghavan, A. K. Roy-Chowdhury, and R. Chellappa. Matching shape sequences in video with applications in human movement analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (12):1896–1909, 2005.

H. Wang, M. Ullah, A Klaser, I. Laptev, and C. Schmid. Evaulation of local spatio-temporal features for action recognition. In *British Machine Vision Conference*, 2009.

D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104:249–257, 2006.

Y. Yuan, H. Zheng, Z. Li, and D. Zhang. Video action recognition with spatio-temporal graph embedding and spline modeling. In *ICASSP*, 2010.

Journal of Machine Learning Research 13 (2012) 3323-3348 Submitted 4/12; Revised 10/12; Published 11/12

Large-scale Linear Support Vector Regression

Chia-Hua Ho B95082@CSIE.NTU.EDU.TW
Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: Sathiya Keerthi

Abstract

Support vector regression (SVR) and support vector classification (SVC) are popular learning techniques, but their use with kernels is often time consuming. Recently, linear SVC without kernels has been shown to give competitive accuracy for some applications, but enjoys much faster training/testing. However, few studies have focused on linear SVR. In this paper, we extend state-of-the-art training methods for linear SVC to linear SVR. We show that the extension is straightforward for some methods, but is not trivial for some others. Our experiments demonstrate that for some problems, the proposed linear-SVR training methods can very efficiently produce models that are as good as kernel SVR.

Keywords: support vector regression, Newton methods, coordinate descent methods

1. Introduction

Support vector regression (SVR) is a widely used regression technique (Vapnik, 1995). It is extended from support vector classification (SVC) by Boser et al. (1992). Both SVR and SVC are often used with the kernel trick (Cortes and Vapnik, 1995), which maps data to a higher dimensional space and employs a kernel function. We refer to such settings as nonlinear SVR and SVC. Although effective training methods have been proposed (e.g., Joachims, 1998; Platt, 1998; Chang and Lin, 2011), it is well known that training/testing large-scale nonlinear SVC and SVR is time consuming.

Recently, for some applications such as document classification, linear SVC without using kernels has been shown to give competitive performances, but training and testing are much faster. A series of studies (e.g., Keerthi and DeCoste, 2005; Joachims, 2006; Shalev-Shwartz et al., 2007; Hsieh et al., 2008) have made linear classifiers (SVC and logistic regression) an effective and efficient tool. On the basis of this success, we are interested in whether linear SVR can be useful for some large-scale applications. Some available document data come with real-valued labels, so for them SVR rather than SVC must be considered. In this paper, we develop efficient training methods to demonstrate that, similar to SVC, linear SVR can sometimes achieve comparable performance to nonlinear SVR, but enjoys much faster training/testing.

We focus on methods in the popular package LIBLINEAR (Fan et al., 2008), which currently provides two types of methods for large-scale linear SVC.[1] The first is a Newton-type method to solve

1. We mean standard SVC using L2 regularization. For L1-regularized problems, the solvers are different.

the primal-form of SVC (Lin et al., 2008), while the second is a coordinate descent approach for the dual form (Hsieh et al., 2008). We show that it is straightforward to extend the Newton method for linear SVR, but some careful redesign is essential for applying coordinate descent methods.

LIBLINEAR offers two types of training methods for linear SVC because they complement each other. A coordinate descent method quickly gives an approximate solution, but may converge slowly in the end. In contrast, Newton methods have the opposite behavior. We demonstrate that similar properties still hold when these training methods are applied to linear SVR.

This paper is organized as follows. In Section 2, we introduce the formulation of linear SVR. In Section 3, we investigate two types of optimization methods for training large-scale linear SVR. In particular, we propose a condensed implementation of coordinate descent methods. We conduct experiments in Section 4 on some large regression problems. A comparison between linear and nonlinear SVR is given, followed by detailed experiments of optimization methods for linear SVR. Section 5 concludes this work.

2. Linear Support Vector Regression

Given a set of training instance-target pairs $\{(\mathbf{x}_i, y_i)\}$, $\mathbf{x}_i \in \mathbf{R}^n$, $y_i \in \mathbf{R}$, $i = 1, \ldots, l$, linear SVR finds a model $\mathbf{w}$ such that $\mathbf{w}^T\mathbf{x}_i$ is close to the target value y_i. It solves the following regularized optimization problem.

$$\min_{\mathbf{w}} \quad f(\mathbf{w}), \quad \text{where} \quad f(\mathbf{w}) \equiv \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \xi_\varepsilon(\mathbf{w}; \mathbf{x}_i, y_i). \tag{1}$$

In Equation (1), $C > 0$ is the regularization parameter, and

$$\xi_\varepsilon(\mathbf{w}; \mathbf{x}_i, y_i) = \begin{cases} \max(|\mathbf{w}^T\mathbf{x}_i - y_i| - \varepsilon, 0) & \text{or} \quad (2) \\ \max(|\mathbf{w}^T\mathbf{x}_i - y_i| - \varepsilon, 0)^2 & (3) \end{cases}$$

is the ε-insensitive loss function associated with $(\mathbf{x}_i, y_i)$. The parameter ε is given so that the loss is zero if $|\mathbf{w}^T\mathbf{x}_i - y_i| \le \varepsilon$. We refer to SVR using (2) and (3) as L1-loss and L2-loss SVR, respectively. It is known that L1 loss is not differentiable, while L2 loss is differentiable but not twice differentiable. An illustration of the two loss functions is in Figure 1. Once problem (1) is minimized, the prediction function is $\mathbf{w}^T\mathbf{x}$.

Standard SVC and SVR involve a bias term b so that the prediction function is $\mathbf{w}^T\mathbf{x} + b$. Recent works on large-scale linear classification often omit the bias term because it hardly affects the performance on most data. We omit a bias term b in problem (1) as well, although in Section 4.5 we briefly investigate the performance with/without it.

It is well known (Vapnik, 1995) that the dual problem of L1-/L2-loss SVR is

$$\min_{\alpha^+, \alpha^-} f_A(\alpha^+, \alpha^-) \qquad \text{subject to} \quad 0 \le \alpha_i^+, \alpha_i^- \le U, \forall i = 1, \ldots, l, \tag{4}$$

where

$$\begin{aligned} & f_A(\alpha^+, \alpha^-) \\ &= \frac{1}{2}(\alpha^+ - \alpha^-)^T Q(\alpha^+ - \alpha^-) + \sum_{i=1}^{l} \left(\varepsilon(\alpha_i^+ + \alpha_i^-) - y_i(\alpha_i^+ - \alpha_i^-) + \frac{\lambda}{2}((\alpha_i^+)^2 + (\alpha_i^-)^2)\right). \end{aligned} \tag{5}$$

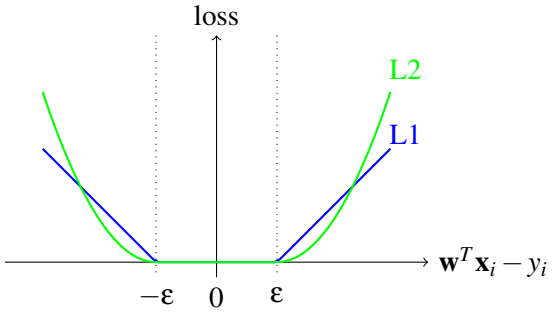


Figure 1: L1-loss and L2-loss functions.

In Equation (5), $Q \in \mathbf{R}^{l\times l}$ is a matrix with $Q_{ij} \equiv \mathbf{x}_i^T\mathbf{x}_j$, and

$$(\lambda, U) = \begin{cases} (0, C) & \text{if L1-loss SVR,} \\ (\frac{1}{2C}, \infty) & \text{if L2-loss SVR.} \end{cases}$$

We can combine α^+ and α^- so that

$$\alpha = \begin{bmatrix} \alpha^+ \\ \alpha^- \end{bmatrix} \quad \text{and} \quad f_A(\alpha) = \frac{1}{2}\alpha^T \begin{bmatrix} \bar{Q} & -Q \\ -Q & \bar{Q} \end{bmatrix} \alpha + \begin{bmatrix} \varepsilon\mathbf{e} - \mathbf{y} \\ \varepsilon\mathbf{e} + \mathbf{y} \end{bmatrix}^T \alpha,$$

where $\bar{Q} = Q + \lambda I$, I is the identity matrix, and $\mathbf{e}$ is the vector of ones. In this paper, we refer to (1) as the primal SVR problem, while (4) as the dual SVR problem. The primal-dual relationship indicates that primal optimal solution $\mathbf{w}^*$ and dual optimal solution $(\alpha^+)^*$ and $(\alpha^-)^*$ satisfy

$$\mathbf{w}^* = \sum_{i=1}^{l} ((\alpha_i^+)^* - (\alpha_i^-)^*)\mathbf{x}_i.$$

An important property of the dual problem (4) is that at optimum,

$$(\alpha_i^+)^*(\alpha_i^-)^* = 0, \forall i.^2$$

The dual problem of SVR has $2l$ variables, while SVC has only l. If a dual-based solver is applied without a careful design, the cost may be significantly higher than that for SVC.

3. Optimization Methods for Training Linear SVR

In this section, we extend two linear-SVC methods in LIBLINEAR for linear SVR. The first is a Newton method for L2-loss SVR, while the second is a coordinate descent method for L1-/L2-loss SVR.

2. This result can be easily proved. From (5), if $\alpha_i^+\alpha_i^- \neq 0$, then for any $0 < \eta \leq \min(\alpha_i^+, \alpha_i^-)$, replacing α_i^+ and α_i^- with $\alpha_i^+ - \eta$ and $\alpha_i^- - \eta$ gives a smaller function value: $f_A(\alpha^+, \alpha^-) - 2\eta\varepsilon - \lambda((\alpha_i^+ + \alpha_i^-)\eta - \eta^2)$. Therefore, $(\alpha_i^+)^*(\alpha_i^-)^*$ must be zero.

Algorithm 1 A trust region Newton method for L2-loss SVR

1. Given $\mathbf{w}^0$.
2. For $k = 0, 1, 2, \ldots$
   2.1. If (7) is satisfied,
        return $\mathbf{w}^k$.
   2.2. Solve subproblem (6).
   2.3. Update $\mathbf{w}^k$ and Δ_k to $\mathbf{w}^{k+1}$ and Δ_{k+1}.

3.1 A Trust Region Newton Method (TRON) for L2-loss SVR

TRON (Lin and Moré, 1999) is a general optimization method for differentiable unconstrained and bound-constrained problems, where the primal problem of L2-loss SVR is a case. Lin et al. (2008) investigate the use of TRON for L2-loss SVC and logistic regression. In this section, we discuss how TRON can be applied to solve large linear L2-loss SVR.

The optimization procedure of TRON involves two layers of iterations. At the k-th outer-layer iteration, given the current position $\mathbf{w}^k$, TRON sets a trust-region size Δ_k and constructs a quadratic model

$$q_k(\mathbf{s}) \equiv \nabla f(\mathbf{w}^k)^T \mathbf{s} + \frac{1}{2}\mathbf{s}^T \nabla^2 f(\mathbf{w}^k)\mathbf{s}$$

as the approximation to $f(\mathbf{w}^k + \mathbf{s}) - f(\mathbf{w}^k)$. Then, in the inner layer, TRON solves the following problem to find a Newton direction under a step-size constraint.

$$\min_s \quad q_k(\mathbf{s}) \qquad \text{subject to} \quad \|\mathbf{s}\| \le \Delta_k. \tag{6}$$

TRON adjusts the trust region Δ_k according to the approximate function reduction $q_k(\mathbf{s})$ and the real function decrease; see details in Lin et al. (2008).

To compute a truncated Newton direction by solving (6), TRON needs the gradient $\nabla f(\mathbf{w})$ and Hessian $\nabla^2 f(\mathbf{w})$. The gradient of L2-loss SVR is

$$\nabla f(\mathbf{w}) = \mathbf{w} + 2C(X_{I_1,:})^T(X_{I_1,:}\mathbf{w} - \mathbf{y}_{I_1} - \varepsilon\mathbf{e}_{I_1}) - 2C(X_{I_2,:})^T(-X_{I_2,:}\mathbf{w} + \mathbf{y}_{I_2} - \varepsilon\mathbf{e}_{I_2}),$$

where

$$X \equiv [\mathbf{x}_1, \ldots, \mathbf{x}_l]^T, I_1 \equiv \{i \mid \mathbf{w}^T\mathbf{x}_i - y_i > \varepsilon\}, \text{ and } I_2 \equiv \{i \mid \mathbf{w}^T\mathbf{x}_i - y_i < -\varepsilon\}.$$

However, $\nabla^2 f(\mathbf{w})$ does not exist because L2-loss SVR is not twice differentiable. Following Mangasarian (2002) and Lin et al. (2008), we use the generalized Hessian matrix. Let

$$I \equiv I_1 \cup I_2.$$

The generalized Hessian can be defined as

$$\nabla^2 f(\mathbf{w}) = I + 2C(X_{I,:})^T D_{I,I} X_{I,:},$$

where I is the identity matrix, and D is an l-by-l diagonal matrix with

$$D_{ii} \equiv \begin{cases} 1 & \text{if } i \in I, \\ 0 & \text{if } i \notin I. \end{cases}$$

From Theorem 2.1 of Lin and Moré (1999), the sequence $\{\mathbf{w}^k\}$ globally converges to the unique minimum of (1).[3] However, because generalized Hessian is used, it is unclear if $\{\mathbf{w}^k\}$ has local quadratic convergence enjoyed by TRON for twice differentiable functions.

For large-scale problems, we cannot store an n-by-n Hessian matrix in the memory. The same problem has occurred in classification, so Lin et al. (2008) applied an iterative method to solve (6). In each inner iteration, only some Hessian-vector products are required and they can be performed without storing Hessian. We consider the same setting so that for any vector $\mathbf{v} \in \mathbf{R}^n$,

$$\nabla^2 f(\mathbf{w})\mathbf{v} = \mathbf{v} + 2C(X_{I,:})^T(D_{I,I}(X_{I,:}\mathbf{v})).$$

For the stopping condition, we follow the setting of TRON in LIBLINEAR for classification. It checks if the gradient is small enough compared with the initial gradient.

$$\|\nabla f(\mathbf{w}^k)\|_2 \le \varepsilon_s \|\nabla f(\mathbf{w}^0)\|_2, \tag{7}$$

where $\mathbf{w}^0$ is the initial iterate and ε_s is stopping tolerance given by users. Algorithm 1 gives the basic framework of TRON.

Similar to the situation in classification, the most expensive operation is the Hessian-vector product. It costs $O(|I|n)$ to evaluate $\nabla^2 f(\mathbf{w})\mathbf{v}$.

3.2 Dual Coordinate Descent Methods (DCD)

In this section, we introduce DCD, a coordinate descent method for the dual form of SVC/SVR. It is used in LIBLINEAR for both L1- and L2-loss SVC. We first extend the setting of Hsieh et al. (2008) to SVR and then propose a better algorithm using properties of SVR. We also explain why the preferred setting for linear SVR may be different from that for nonlinear SVR.

3.2.1 A DIRECT EXTENSION FROM CLASSIFICATION TO REGRESSION

A coordinate descent method sequentially updates one variable by solving the following subproblem.

$$\begin{aligned} \min_z \quad & f_A(\alpha + z\mathbf{e}_i) - f_A(\alpha) \\ \text{subject to} \quad & 0 \le \alpha_i + z \le U. \end{aligned}$$

where

$$f_A(\alpha + z\mathbf{e}_i) - f_A(\alpha) = \nabla_i f_A(\alpha) z + \frac{1}{2}\nabla^2_{ii} f_A(\alpha) z^2$$

and $\mathbf{e}_i \in \mathbf{R}^{2l\times 1}$ is a vector with i-th element one and others zero. The optimal value z can be solved in a closed form, so α_i is updated by

$$\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\nabla_i f_A(\alpha)}{\nabla^2_{ii} f_A(\alpha)}, 0\right), U\right), \tag{8}$$

where

$$\nabla_i f_A(\alpha) = \begin{cases} (Q(\alpha^+ - \alpha^-))_i + \varepsilon - y_i + \lambda\alpha_i^+, & \text{if } 1 \le i \le l, \\ -(Q(\alpha^+ - \alpha^-))_{i-l} + \varepsilon + y_{i-l} + \lambda\alpha_{i-l}^-, & \text{if } l+1 \le i \le 2l, \end{cases} \tag{9}$$

3. Note that the objective function of (1) is strictly convex, so (1) has a unique global minimum.

Algorithm 2 A DCD method for linear L1-/L2-loss SVR

1. Given α^+ and α^-. Let $\alpha = \begin{bmatrix}\alpha^+ \\ \alpha^-\end{bmatrix}$ and the corresponding $\mathbf{u} = \sum_{i=1}^{l}(\alpha_i - \alpha_{i+l})\mathbf{x}_i$.
2. Compute the Hessian diagonal $\bar{Q}_{ii}$, $\forall i = 1,\ldots,l$.
3. For $k = 0,1,2,\ldots$
   - For $i \in \{1,\ldots,2l\}$ // select an index to update
     3.1. If $|\nabla_i^P f_A(\alpha)| \neq 0$
       3.1.1. Update α_i by (8), where $(Q(\alpha^+ - \alpha^-))_i$ or $(Q(\alpha^+ - \alpha^-))_{i-l}$ is evaluated by $\mathbf{u}^T\mathbf{x}_i$ or $\mathbf{u}^T\mathbf{x}_{i-l}$. See Equation (9).
       3.1.2. Update $\mathbf{u}$ by (10).

and

$$\nabla_{ii}^2 f_A(\alpha) = \begin{cases} \bar{Q}_{ii} & \text{if } 1 \le i \le l, \\ \bar{Q}_{i-l,i-l} & \text{if } l+1 \le i \le 2l. \end{cases}$$

To efficiently implement (8), techniques that have been employed for SVC can be applied. First, we precalculate $\bar{Q}_{ii} = \mathbf{x}_i^T\mathbf{x}_i + \lambda, \forall i$ in the beginning. Second, $(Q(\alpha^+ - \alpha^-))_i$ is obtained using a vector $\mathbf{u}$.

$$(Q(\alpha^+ - \alpha^-))_i = \mathbf{u}^T\mathbf{x}_i, \text{ where } \mathbf{u} \equiv \sum_{i=1}^{l}(\alpha_i^+ - \alpha_i^-)\mathbf{x}_i.$$

If the current iterate α_i is updated to $\bar{\alpha}_i$ by (8), then vector $\mathbf{u}$ can be maintained by

$$\mathbf{u} \leftarrow \begin{cases} \mathbf{u} + (\bar{\alpha}_i - \alpha_i)\mathbf{x}_i, & \text{if } 1 \le i \le l, \\ \mathbf{u} - (\bar{\alpha}_{i-l} - \alpha_{i-l})\mathbf{x}_{i-l}, & \text{if } l+1 \le i \le 2l. \end{cases} \tag{10}$$

Both (8) and (10) cost $O(n)$, which is the same as the cost in classification.

Hsieh et al. (2008) check the projected gradient $\nabla^P f_A(\alpha)$ for the stopping condition because α is optimal if and only if $\nabla^P f_A(\alpha)$ is zero. The projected gradient is defined as

$$\nabla_i^P f_A(\alpha) \equiv \begin{cases} \min(\nabla_i f_A(\alpha),\ 0) & \text{if } \alpha_i = 0, \\ \max(\nabla_i f_A(\alpha),\ 0) & \text{if } \alpha_i = U, \\ \nabla_i f_A(\alpha) & \text{if } 0 < \alpha_i < U. \end{cases} \tag{11}$$

If $\nabla_i^P f_A(\alpha) = 0$, then (8) and (11) imply that α_i needs not be updated. We show the overall procedure in Algorithm 2.

Hsieh et al. (2008) apply two techniques to make a coordinate descent method faster. The first one is to permute all variables at each iteration to decide the order for update. We find that this setting is also useful for SVR. The second implementation technique is shrinking. By gradually removing some variables, smaller optimization problems are solved to save the training time. In Hsieh et al. (2008), they remove those which are likely to be bounded (i.e., 0 or U) at optimum. Their shrinking strategy can be directly applied here, so we omit details.

While we have directly applied a coordinate descent method to solve (4), the procedure does not take SVR's special structure into account. Note that α^+ and α^- in (5) are very related. We can see that in the following situations some operations in Algorithm 2 are redundant.

1. We pointed out in Section 2 that an optimal α of (4) satisfies

$$\alpha_i^+ \alpha_i^- = 0, \forall i. \tag{12}$$

If one of α^+ or α^- is positive at optimum, it is very possible that the other is zero throughout all final iterations. Because we sequentially select variables for update, these zero variables, even if not updated in steps 3.1.1–3.1.2 of Algorithm 2, still need to be checked in the beginning of step 3.1. Therefore, some operations are wasted. Shrinking can partially solve this problem, but alternatively we may explicitly use the property (12) in designing the coordinate descent algorithm.

2. We show that some operations in calculating the projected gradient in (11) are wasted if all we need is the largest component of the projected gradient. Assume $\alpha_i^+ > 0$ and $\alpha_i^- = 0$. If the optimality condition at α_i^- is not satisfied yet, then

$$\nabla^P_{i+l} f_A(\alpha) = \nabla_{i+l} f_A(\alpha) = -(Q(\alpha^+ - \alpha^-))_i + \varepsilon + y_i + \lambda \alpha_i^- < 0.$$

We then have

$$\begin{aligned} 0 < -\nabla_{i+l} f_A(\alpha) &= (Q(\alpha^+ - \alpha^-))_i - \varepsilon - y_i - \lambda \alpha_i^- \\ &< (Q(\alpha^+ - \alpha^-))_i + \varepsilon - y_i + \lambda \alpha_i^+ = \nabla_i f_A(\alpha), \end{aligned} \tag{13}$$

so a larger violation of the optimality condition occurs at α_i^+. Thus, when $\alpha_i^+ > 0$ and $\alpha_i^- = 0$, checking $\nabla_{i+l} f_A(\alpha)$ is not necessary if we aim to find the largest element of the projected gradient.

In Section 3.2.2, we propose a method to address these issues. However, the straightforward coordinate descent implementation discussed in this section still possesses some advantages. See the discussion in Section 3.2.3.

3.2.2 A NEW COORDINATE DESCENT METHOD BY SOLVING α^+ AND α^- TOGETHER

Using the property (12), the following problem replaces $(\alpha_i^+)^2 + (\alpha_i^-)^2$ in (5) with $(\alpha_i^+ - \alpha_i^-)^2$ and gives the same optimal solutions as the dual problem (4).

$$\min_{\alpha^+, \alpha^-} \frac{1}{2}(\alpha^+ - \alpha^-)^T Q(\alpha^+ - \alpha^-) + \sum_{i=1}^{l} \left(\varepsilon(\alpha_i^+ + \alpha_i^-) - y_i(\alpha_i^+ - \alpha_i^-) + \frac{1}{2}(\alpha_i^+ - \alpha_i^-)^2\right). \tag{14}$$

Further, Equation (12) and $\alpha_i^+ \geq 0, \alpha_i^- \geq 0$ imply that at optimum,

$$\alpha_i^+ + \alpha_i^- = |\alpha_i^+ - \alpha_i^-|.$$

With $\bar{Q} = Q + \lambda I$ and defining

$$\beta = \alpha^+ - \alpha^-,$$

problem (14) can be transformed as

$$\min_{\beta} \quad f_B(\beta) \quad \text{subject to} \quad -U \leq \beta_i \leq U, \forall i, \tag{15}$$

where

$$f_B(\beta) \equiv \frac{1}{2}\beta^T \bar{Q}\beta - \mathbf{y}^T\beta + \varepsilon\|\beta\|_1.$$

If β^* is an optimum of (15), then

$$(\alpha_i^+)^* \equiv \max(\beta_i^*, 0) \quad \text{and} \quad (\alpha_i^-)^* \equiv \max(-\beta_i^*, 0)$$

are optimal for (4).

We design a coordinate descent method to solve (15). Interestingly, (15) is in a form similar to the primal optimization problem of L1-regularized regression and classification. In LIBLINEAR, a coordinate descent solver is provided for L1-regularized L2-loss SVC (Yuan et al., 2010). We will adapt some of its implementation techniques here. A difference between L1-regularized classification and the problem (15) is that (15) has additional bounded constraints.

Assume β is the current iterate and its i-th component, denoted as a scalar variable s, is being updated. Then the following one-variable subproblem is solved.

$$\min_s \quad g(s) \quad \text{subject to } -U \le s \le U, \tag{16}$$

where β is considered as a constant vector and

$$\begin{aligned} g(s) &= f_B(\beta + (s-\beta_i)\mathbf{e}_i) - f_B(\beta) \\ &= \varepsilon|s| + (\bar{Q}\beta - \mathbf{y})_i(s-\beta_i) + \frac{1}{2}\bar{Q}_{ii}(s-\beta_i)^2 + \text{ constant}. \end{aligned} \tag{17}$$

It is well known that (17) can be reduced to "soft-thresholding" in signal processing and has a closed-form minimum. However, here we decide to give detailed derivations of solving (16) because of several reasons. First, s is now bounded in $[-U, U]$. Second, the discussion will help to explain our stopping condition and shrinking procedure.

To solve (16), we start with checking the derivative of $g(s)$. Although $g(s)$ is not differentiable at $s = 0$, its derivatives at $s \ge 0$ and $s \le 0$ are respectively

$$\begin{aligned} g_p'(s) &= \varepsilon + (\bar{Q}\beta - \mathbf{y})_i + \bar{Q}_{ii}(s-\beta_i) \quad \text{if } s \ge 0, \text{ and} \\ g_n'(s) &= -\varepsilon + (\bar{Q}\beta - \mathbf{y})_i + \bar{Q}_{ii}(s-\beta_i) \quad \text{if } s \le 0. \end{aligned}$$

Both $g_p'(s)$ and $g_n'(s)$ are linear functions of s. Further,

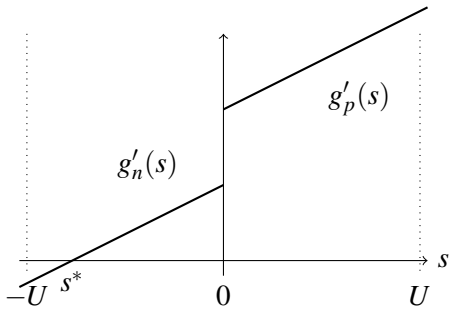
$$g_n'(s) \le g_p'(s), \forall s \in \mathcal{R}.$$

For any strictly convex quadratic function, the unique minimum occurs when the first derivative is zero. Because $g(s)$ is only piece-wise quadratic, we consider three cases in Figure 2 according to the values of $g_p'(s)$ and $g_n'(s)$. In Figure 2(a), $0 < g_n'(0) < g_p'(0)$, so $g(0)$ is the smallest on the positive side:

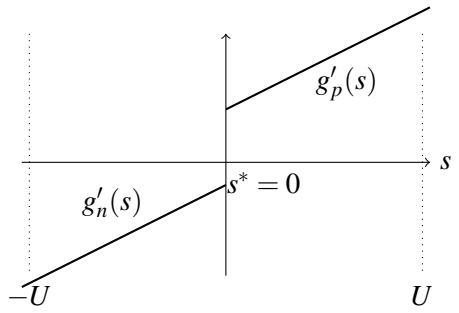
$$g(0) \le g(s), \forall s \ge 0. \tag{18}$$

For $s \le 0$, $g_n'(s) = 0$ has a root because the line of $g_n'(s)$ intersects the x-axis. With (18), this root is the minimum for both $s \le 0$ and $s \ge 0$. By solving $g_n'(s) = 0$ and taking the condition $0 < g_n'(0)$, the solution of (16) is

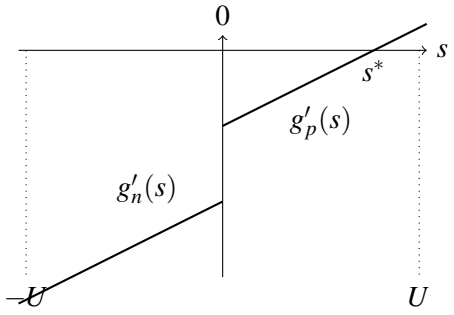
$$\beta_i - \frac{-\varepsilon + (\bar{Q}\beta - \mathbf{y})_i}{\bar{Q}_{ii}} \quad \text{if } -\varepsilon + (\bar{Q}\beta - \mathbf{y})_i > \bar{Q}_{ii}\beta_i. \tag{19}$$



(a) $0 < g'_n(0) < g'_p(0)$.



(b) $g'_n(0) \le 0 \le g'_p(0)$.



(c) $g'_n(0) < g'_p(0) < 0$.

Figure 2: We discuss the minimization of (16) using three cases. The y-axis indicates the value of $g'_p(s)$ and $g'_n(s)$. The point s^* denotes the optimal solution.

We also need to take the constraint $s \in [-U, U]$ in Equation (16) into account. If the value obtained in (19) is smaller than $-U$, then $g'_n(s) > 0, \forall s \ge -U$. That is, $g(s)$ is an increasing function and the minimum is at $s = -U$.

The situation is similar in Figure 2(c), where the minimum occurs at $g'_p(s) = 0$. For the remaining case in Figure 2(b),

$$g'_n(0) \le 0 \le g'_p(0). \tag{20}$$

Inequalities in (20) imply that $g(s)$ is a decreasing function at $s \le 0$, but is an increasing function at $s \ge 0$. Thus, an optimal solution occurs at $s = 0$. A summary of the three cases shows that the subproblem (16) has the following closed form solution.

$$s \leftarrow \max(-U, \min(U, \beta_i + d)), \tag{21}$$

where

$$d \equiv \begin{cases} -\frac{g'_p(\beta_i)}{\bar{Q}_{ii}} & \text{if } g'_p(\beta_i) < \bar{Q}_{ii}\beta_i, \\ -\frac{g'_n(\beta_i)}{\bar{Q}_{ii}} & \text{if } g'_n(\beta_i) > \bar{Q}_{ii}\beta_i, \\ -\beta_i & \text{otherwise.} \end{cases} \tag{22}$$

In (22), we simplify the solution form in (19) by using the property

$$g'_p(\beta_i) = \varepsilon + (\bar{Q}\beta - \mathbf{y})_i, \text{ and } g'_n(\beta_i) = -\varepsilon + (\bar{Q}\beta - \mathbf{y})_i. \tag{23}$$

Algorithm 3 A new DCD method which solves (15) for linear L1-/L2-loss SVR

1. Given β and the corresponding $\mathbf{u} = \sum_{i=1}^{l} \beta_i \mathbf{x}_i$.
2. Compute the Hessian diagonal $\bar{Q}_{ii}$, $\forall i = 1, \ldots, l$.
3. For $k = 0, 1, 2, \ldots$
   - For $i \in \{1, \ldots, l\}$ // select an index to update
     3.1. Find s by (21), where $(Q\beta)_i$ is evaluated by $\mathbf{u}^T \mathbf{x}_i$.
     3.2. $\mathbf{u} \leftarrow \mathbf{u} + (s - \beta_i)\mathbf{x}_i$.
     3.3. $\beta_i \leftarrow s$.

Following the same technique in Section 3.2.1, we maintain a vector $\mathbf{u}$ and calculate $(\bar{Q}\beta)$ by

$$(\bar{Q}\beta)_i = \mathbf{u}^T \mathbf{x}_i + \lambda \beta_i, \text{ where } \mathbf{u} = \sum_{i=1}^{l} \beta_i \mathbf{x}_i.$$

The new DCD method to solve (15) is sketched in Algorithm 3.

For the convergence, we show in Appendix A that Algorithm 3 is a special case of the general framework in Tseng and Yun (2009) for non-smooth separable minimization. Their Theorem 2(b) implies that Algorithm 3 converges in an at least linear rate.

Theorem 1 *For L1-loss and L2-loss SVR, if β^k is the k-th iterate generated by Algorithm 3, then $\{\beta^k\}$ globally converges to an optimal solution β^*. The convergence rate is at least linear: there are $0 < \mu < 1$ and an iteration k_0 such that*

$$f_B(\beta^{k+1}) - f_B(\beta^*) \leq \mu(f_B(\beta^k) - f_B(\beta^*)), \forall k \geq k_0.$$

Besides Algorithm 3, other types of coordinate descent methods may be applicable here. For example, at step 3 of Algorithm 3, we may randomly select a variable for update. Studies of such random coordinate descent methods with run time analysis include, for example, Shalev-Shwartz and Tewari (2011), Nesterov (2010), and Richtárik and Takáč (2011).

For the stopping condition and the shrinking procedure, we will mainly follow the setting in LIBLINEAR for L1-regularized classification. To begin, we study how to measure the violation of the optimality condition of (16) during the optimization procedure. From Figure 2(c), we see that

$$\text{if } 0 < \beta_i^* < U \text{ is optimal for (16), then } g_p'(\beta_i^*) = 0.$$

Thus, if $0 < \beta_i < U$, $|g_p'(\beta_i)|$ can be considered as the violation of the optimality. From Figure 2(b), we have that

$$\text{if } \beta_i^* = 0 \text{ is optimal for (16), then } g_n'(\beta_i^*) \leq 0 \leq g_p'(\beta_i^*).$$

Thus,

$$\begin{cases} g_n'(\beta_i) & \text{if } \beta_i = 0 \text{ and } g_n'(\beta_i) > 0, \\ -g_p'(\beta_i) & \text{if } \beta_i = 0 \text{ and } g_p'(\beta_i) < 0 \end{cases}$$

gives the violation of the optimality. After considering all situations, we know that

$$\beta_i \text{ is optimal for (16) if and only if } v_i = 0,$$

where

$$v_i \equiv \begin{cases} |g_n'(\beta_i)| & \text{if } \beta_i \in (-U,0), \text{ or } \beta_i = -U \text{ and } g_n'(\beta_i) \le 0, \\ |g_p'(\beta_i)| & \text{if } \beta_i \in (0,U), \text{ or } \beta_i = U \text{ and } g_p'(\beta_i) \ge 0, \\ g_n'(\beta_i) & \text{if } \beta_i = 0 \text{ and } g_n'(\beta_i) \ge 0, \\ -g_p'(\beta_i) & \text{if } \beta_i = 0 \text{ and } g_p'(\beta_i) \le 0, \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

If β is unconstrained (i.e., $U = \infty$), then (24) reduces to the minimum-norm subgradient used in L1-regularized problems. Based on it, Yuan et al. (2010) derive their stopping condition and shrinking scheme. We follow them to use a similar stopping condition.

$$\|\mathbf{v}^k\|_1 < \varepsilon_s \|\mathbf{v}^0\|_1, \tag{25}$$

where $\mathbf{v}^0$ and $\mathbf{v}^k$ are the initial violation and the violation in the k-th iteration, respectively. Note that $\mathbf{v}^k$'s components are sequentially obtained via (24) in l coordinate descent steps of the k-th iteration.

For shrinking, we remove bounded variables (i.e., $\beta_i = 0$, U, or $-U$) if they may not be changed at the final iterations. Following Yuan et al. (2010), we use a "tighter" form of the optimality condition to conjecture that a variable may have stuck at a bound. We shrink β_i if it satisfies one of the following conditions.

$$\beta_i = 0 \text{ and } g_n'(\beta_i) < -M < 0 < M < g_p'(\beta_i), \tag{26}$$

$$\beta_i = U \text{ and } g_p'(\beta_i) < -M, \text{ or} \tag{27}$$

$$\beta_i = -U \text{ and } g_n'(\beta_i) > M, \tag{28}$$

where

$$M \equiv \max_i v_i^{k-1} \tag{29}$$

is the maximal violation of the previous iteration. The condition (26) is equivalent to

$$\beta_i = 0 \text{ and } -\varepsilon + M < (\bar{Q}\beta)_i - y_i < \varepsilon - M. \tag{30}$$

This is almost the same as the one used in Yuan et al. (2010); see Equation (32) in that paper. However, there are some differences. First, because they solve L1-regularized SVC, ε in (30) becomes the constant one. Second, they scale M to a smaller value. Note that M used in conditions (26), (27), and (28) controls how aggressive our shrinking scheme is. In Section 4.6, we will investigate the effect of using different M values.

For L2-loss SVR, α_i is not upper-bounded in the dual problem, so (26) becomes the only condition to shrink variables. This makes L2-loss SVR have less opportunity to shrink variables than L1-loss SVR. The same situation has been known for L2-loss SVC.

In Section 3.2.1, we pointed out some redundant operations in calculating the projected gradient of $f_A(\alpha^+, \alpha^-)$. If $0 < \beta_i < U$, we have $\alpha_i^+ = \beta_i$ and $\alpha_i^- = 0$. In this situation, Equation (13) indicates that for finding the maximal violation of the optimality condition, we only need to check $\nabla_i^P f_A(\alpha)$ rather than $\nabla_{i+l}^P f_A(\alpha)$. From (11) and (23),

$$\nabla_i^P f_A(\alpha) = (\bar{Q}\beta - \mathbf{y})_i + \varepsilon = g_p'(\beta).$$

This is what we checked in (24) when $0 < \beta < U$. Therefore, no operations are wasted.

Algorithm 4 Details of Algorithm 3 with a stopping condition and a shrinking implementation.

1. Given β and corresponding $\mathbf{u} = \sum_{i=1}^{l} \beta_i \mathbf{x}_i$.
2. Set $\lambda = 0$ and $U = C$ if L1-loss SVR; $\lambda = 1/(2C)$ and $U = \infty$ if L2-loss SVR.
3. Compute the Hessian diagonal $\bar{Q}_{ii}$, $\forall i = 1, \ldots, l$.
4. $M \leftarrow \infty$, and compute $\|\mathbf{v}^0\|_1$ by (24).
5. $T \leftarrow \{1, \ldots, l\}$.
6. For $k = 0, 1, 2, \ldots$
   - 6.1. Randomly permute T.
   - 6.2. For $i \in T$ // select an index to update
     - 6.2.1. $g_p' \leftarrow -y_i + \mathbf{u}^T \mathbf{x}_i + \lambda\beta_i + \varepsilon$, $g_n' \leftarrow -y_i + \mathbf{u}^T \mathbf{x}_i + \lambda\beta_i - \varepsilon$.
     - 6.2.2. Find v_i^k by (24).
     - 6.2.3. If any condition in (26), (27), and (28) is satisfied
       $T \leftarrow T \backslash \{i\}$.
       continue
     - 6.2.4. Find s by (21).
     - 6.2.5. $\mathbf{u} \leftarrow \mathbf{u} + (s - \beta_i)\mathbf{x}_i$.
     - 6.2.6. $\beta_i \leftarrow s$.
   - 6.3. If $\|\mathbf{v}^k\|_1 / \|\mathbf{v}^0\|_1 < \varepsilon_s$
     If $T = \{1, \ldots, l\}$
       break
     else
       $T \leftarrow \{1, \ldots, l\}$, and $M \leftarrow \infty$.
     else
       $M \leftarrow \|\mathbf{v}^k\|_\infty$.

Algorithm 4 is the overall procedure to solve (15). In the beginning, we set $M = \infty$, so no variables are shrunk at the first iteration. The set T in Algorithm 4 includes variables which have not been shrunk. During the iterations, the stopping condition of a smaller problem of T is checked. If it is satisfied but T is not the full set of variables, we reset T to be $\{1, \ldots, l\}$; see the if-else statement in step 6.3 of Algorithm 4. This setting ensures that the algorithm stops only after the stopping condition for problem (15) is satisfied. Similar approaches have been used in LIBSVM (Chang and Lin, 2011) and some solvers in LIBLINEAR.

3.2.3 DIFFERENCE BETWEEN DUAL COORDINATE DESCENT METHODS FOR LINEAR AND NONLINEAR SVR

The discussion in Sections 3.2.1–3.2.2 concludes that α_i^+ and α_i^- should be solved together rather than separately. Interestingly, for nonlinear (kernel) SVR, Liao et al. (2002) argue that the opposite is better. They consider SVR with a bias term, so the dual problem contains an additional linear constraint.

$$\sum_{i=1}^{l} (\alpha_i^+ - \alpha_i^-) = 0.$$

Because of this constraint, their coordinate descent implementation (called decomposition methods in the SVM community) must select at least two variables at a time. They discuss the following two settings.

1. Considering $f_A(\alpha)$ and selecting $i, j \in \{1, \ldots, 2l\}$ at a time.

2. Selecting $i, j \in \{1, \ldots, l\}$ and then updating α_i^+, α_i^-, α_j^+, and α_j^- together. That is, a four-variable subproblem is solved.

The first setting corresponds to ours in Section 3.2.1, while the second is related to that in Section 3.2.2. We think Liao et al. (2002) prefer the first because of the following reasons, from which we can see some interesting differences between linear and nonlinear SVM.

1. For nonlinear SVM, we can afford to use gradient information for selecting the working variables; see reasons explained in Section 4.1 of Hsieh et al. (2008). This is in contrast to the sequential selection for linear SVM. Following the gradient-based variable selection, Liao et al. (2002, Theorem 3.4) show that if an optimal $(\alpha_i^+)^* > 0$, then α_i^- remains zero in the final iterations without being selected for update. The situation for $(\alpha_i^-)^* > 0$ is similar. Therefore, their coordinate descent algorithm implicitly has a shrinking implementation, so the first concern discussed in Section 3.2.1 is alleviated.

2. Solving a four-variable subproblem is complicated. In contrast, for the two-variable subproblem of α_i^+ and α_i^-, we demonstrate in Section 3.2.2 that a simple closed-form solution is available.

3. The implementation of coordinate descent methods for nonlinear SVM is more complicated than that for linear because of steps such as gradient-based variable selection and kernel-cache maintenance, etc. Thus, the first setting of minimizing $f_A(\alpha)$ possesses the advantage of being able to reuse the code of SVC. This is the approach taken by the nonlinear SVM package LIBSVM (Chang and Lin, 2011), in which SVC and SVR share the same optimization solver. In contrast, for linear SVC/SVR, the implementation is simple, so we can have a dedicated code for SVR. In this situation, minimizing $f_B(\beta)$ is more preferable than $f_A(\alpha)$.

4. Experiments

In this section, we compare nonlinear/linear SVR and evaluate the methods described in Sections 3. Two evaluation criteria are used. The first one is mean squared error (MSE).

$$\text{mean squared error } = \frac{1}{l}\sum_{i=1}^{l}(y_i - \mathbf{w}^T\mathbf{x}_i)^2.$$

The other is squared correlation coefficient (R^2). Given the target values $\mathbf{y}$ and the predicted values $\mathbf{y}'$, R^2 is defined as

$$\frac{\left(\sum_i (y_i' - E[y_i'])(y_i - E[y_i])\right)^2}{\sigma_{\mathbf{y}}^2 \sigma_{\mathbf{y}'}^2} = \frac{\left(l\sum_i y_i' y_i - (\sum_i y_i')(\sum_i y_i)\right)^2}{\left(l\sum_i y_i^2 - (\sum_i y_i)^2\right)\left(l\sum_i y_i'^2 - (\sum_i y_i')^2\right)}.$$

4.1 Experimental Settings

We consider the following data sets in our experiments. All except CTR are publicly available at LIBSVM data set.[4]

- MSD: We consider this data because it is the largest regression set in the UCI Machine Learning Repository (Frank and Asuncion, 2010). It is originally from Bertin-Mahieux et al. (2011). Each instance contains the audio features of a song, and the target value is the year the song was released. The original target value is between 1922 and 2011, but we follow Bertin-Mahieux et al. (2011) to linearly scale it to $[0,1]$.
- TFIDF-2006, LOG1P-2006: This data set comes from some context-based analysis and discussion of the financial condition of a corporation (Kogan et al., 2009).[5] The target values are the log transformed volatilities of the corporation. We use records in the last year (2006) as the testing data, while the previous five years (2001–2005) for training.
  There are two different feature representations. TFIDF-2006 contains TF-IDF (term frequency and inverse document frequency) of unigrams, but LOG1P-2006 contains

$$\log(1+\mathrm{TF}),$$

  where TF is the term frequency of unigrams and bigrams. Both representations also include the volatility in the past 12 months as an additional feature.
- CTR: The data set is from an Internet company. Each feature vector is a binary representation of a web page and an advertisement block. The target value is the click-through-rate (CTR) defined as (#clicks)/(#page views).
- KDD2010b: This is a classification problem from KDD Cup 2010. The class label indicates whether a student answered a problem correctly or not on a online tutoring system. We consider this classification problem because of several reasons. First, we have not found other large and sparse regression problems. Second, we are interested in the performance of SVR algorithms when a classification problem is treated as a regression one.

The numbers of instances, features, nonzero elements in training data, and the range of target values are listed in Table 1. Except MSD, all others are large sparse data.

We use the zero vector as the initial solution of all algorithms. All implementations are in C++ and experiments are conducted on a 64-bit machine with Intel Xeon 2.0GHz CPU (E5504), 4MB cache, and 32GB main memory. Programs used for our experiment can be found at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/exp.html`.

4.2 A Comparison Between Two DCD Algorithms

Our first experiment is to compare two DCD implementations (Algorithms 2 and 4) so that only the better one is used for subsequence analysis. For this comparison, we normalize each instance to a unit vector and consider L1-loss SVR with $C = 1$ and $\varepsilon = 0.1$.

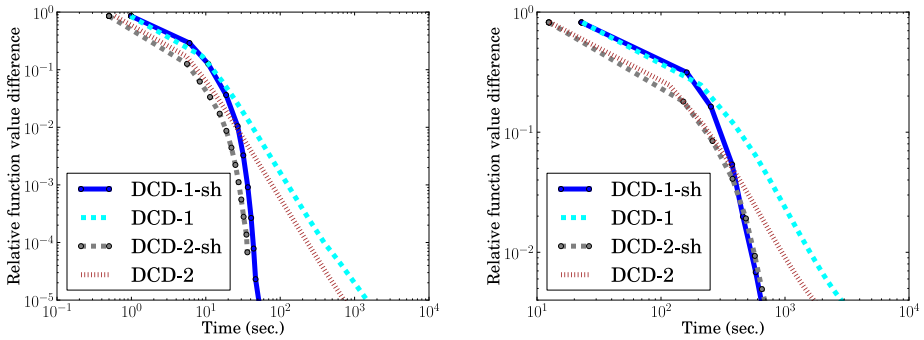
Because the results for all data sets are similar, we only present the results of MSD and CTR in Figure 3. The x-axis is the training time, and the y-axis is the relative difference to the dual optimal function value.

$$\frac{f_A(\alpha) - f_A(\alpha^*)}{|f_A(\alpha^*)|}, \tag{31}$$

4. Data sets can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.
5. The raw data are available at `http://www.ark.cs.cmu.edu/10K/`.

Data	#instances		#features	#non-zeros	range of $\mathbf{y}$
	training	testing		in training	
MSD	463,715	51,630	90	41,734,346	$[0,1]$
TFIDF-2006	16,087	3,308	150,360	19,971,015	$[-7.90,-0.52]$
LOG1P-2006	16,087	3,308	4,272,227	96,731,839	$[-7.90,-0.52]$
CTR	11,382,195	208,988	22,510,600	257,526,282	$[0,1]$
KDD2010b	19,264,097	748,401	29,890,095	566,345,888	$\{0,1\}$

Table 1: Data set statistics: #non-zeros means the number of non-zero elements in all training instances. Note that data sets are sorted according to the number of features.



(a) MSD (b) CTR

Figure 3: A comparison between two DCD algorithms. We present training time and relative difference to the dual optimal function values. L1-loss SVR with $C = 1$ and $\varepsilon = 0.1$ is used. Data instances are normalized to unit vectors. DCD-1-sh and DCD-2-sh are DCD-1 and DCD-2 with shrinking, respectively. Both x-axis and y-axis are in log scale.

where α^* is the optimum solution. We run optimization algorithms long enough to get an approximate $f_A(\alpha^*)$. In Figure 3, DCD-1 and DCD-1-sh are Algorithm 2 without/with shrinking, respectively. DCD-2, and DCD-2-sh are the proposed Algorithm 4. If shrinking is not applied, we simply plot the value (31) once every eight iterations. With shrinking, the setting is more complicated because the stopping tolerance ε_s affects the shrinking implementation; see step 6.3 in Algorithm 4. Therefore, we run Algorithms 2 and 4 several times under various ε_s values to obtain pairs of (training time, function value).

Results show that DCD-2 is significantly faster than DCD-1; note that the training time in Figure 3 is log-scaled. This observation is consistent with our discussion in Section 3.2.1 that Algorithm 2 suffers from some redundant operations. We mentioned that shrinking can reduce the overhead and this is supported by the result that DCD-1-sh becomes closer to DCD-2-sh. Based on this experiment, we only use Algorithm 4 in subsequent analysis.

This experiment also reveals how useful the shrinking technique is. For both Algorithms 2 and 4, shrinking very effectively reduces the training time.

Data (percentage for training)	Linear (DCD)				RBF (LIBSVM)				
	ε	C	test MSE	training time (s)	ε	C	γ	test MSE	training time (s)
MSD (1%)	2^{-4}	2^{5}	0.0155	2.25	2^{-4}	2^{5}	2^{-3}	0.0129	4.66
TFIDF-2006	2^{-10}	2^{6}	0.2031	32.06	2^{-6}	2^{6}	2^{0}	0.1965	3921.61
LOG1P-2006	2^{-4}	2^{1}	0.1422	16.95	2^{-10}	2^{1}	2^{0}	0.1381	16385.7
CTR (0.1%)	2^{-6}	2^{-3}	0.0296	0.05	2^{-8}	2^{-2}	2^{0}	0.0294	15.36
KDD2010b (0.1%)	2^{-4}	2^{-1}	0.0979	0.07	2^{-6}	2^{0}	2^{0}	0.0941	97.23

Table 2: Testing MSE and training time using DCD for linear SVR and LIBSVM for nonlinear SVR with RBF kernel. L1-loss SVR is used. Parameter selection is conducted by five-fold CV. Because LIBSVM's running time is long, for some data, we use only subsets for training.

4.3 A Comparison Between Linear and Nonlinear SVR

We wrote in Section 1 that the motivation of this research work is to check if for some applications linear SVR can give competitive MSE/R^2 with nonlinear SVR, but enjoy faster training. In this section, we compare DCD for linear SVR with the package LIBSVM (Chang and Lin, 2011) for nonlinear SVR. We consider L1-loss SVR because LIBSVM does not support L2 loss.

For LIBSVM, we consider RBF kernel, so Q_{ij} in Equation (5) becomes

$$Q_{ij} \equiv e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|^2},$$

where γ is a user-specified parameter. Because LIBSVM's training time is very long, we only use 1% training data for MSD, and 0.1% training data for CTR and KDD2010b. We conduct five-fold cross validation (CV) to find the best $C \in \{2^{-4}, 2^{-3}, \ldots, 2^{6}\}$, $\varepsilon \in \{2^{-10}, 2^{-8}, \ldots, 2^{-2}\}$, and $\gamma \in \{2^{-8}, 2^{-7}, \ldots, 2^{0}\}$. For LIBSVM, we assign 16GB memory space for storing recently used kernel elements (called kernel cache). We use stopping tolerance 0.1 for both methods although their stopping conditions are slightly different. Each instance is normalized to a unit vector.

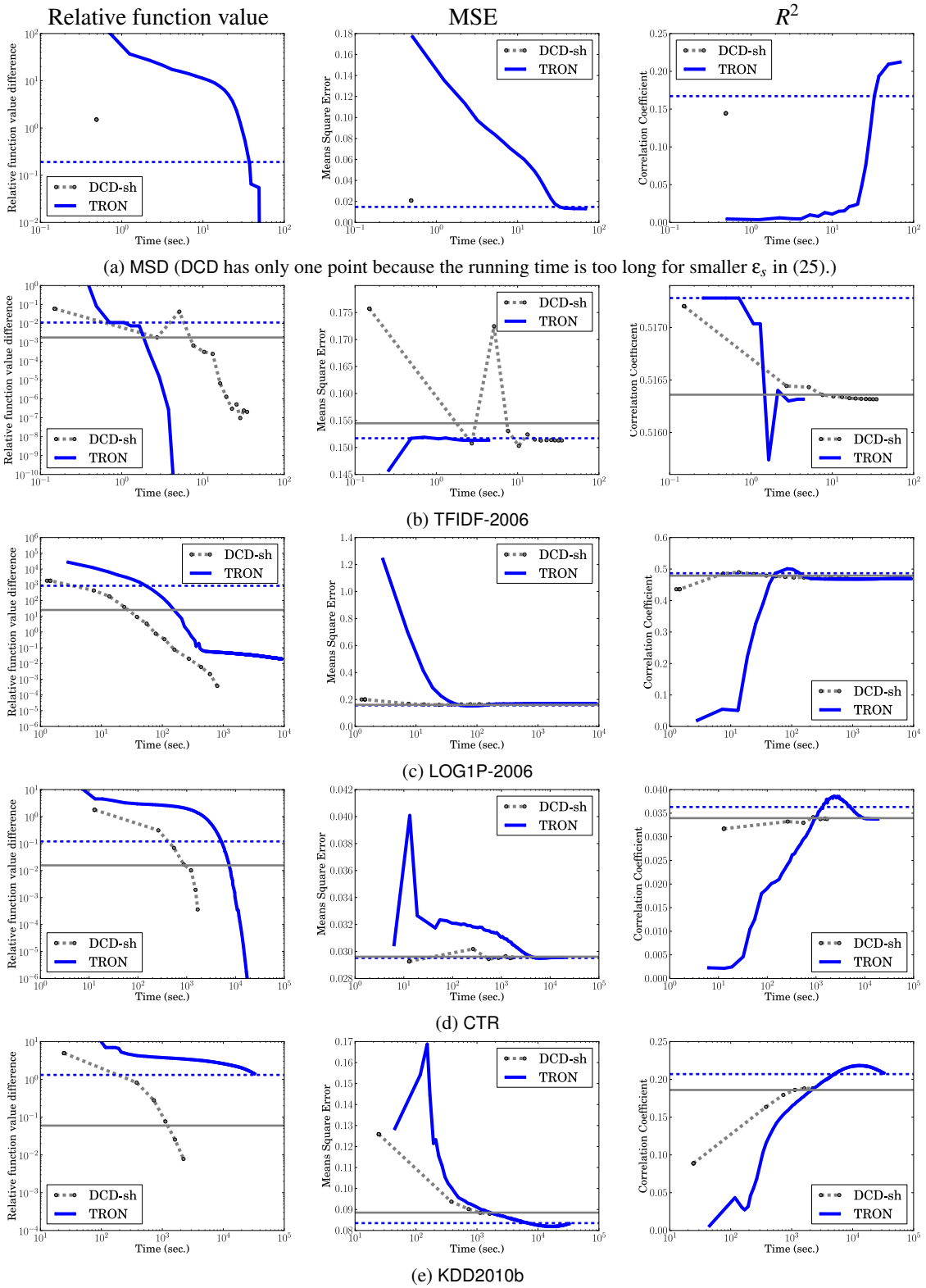
In Table 2, we observe that for all data sets except MSD, nonlinear SVR gives only marginally better MSE than linear SVR, but the training time is prohibitively long. Therefore, for these data sets, linear SVR is more appealing than nonlinear SVR.

4.4 A Comparison Between TRON and DCD on Data with/without Normalization

In this section, we compare the two methods TRON and DCD discussed in Section 3 for training linear SVR. We also check if their behavior is similar to when they are applied to linear SVC. Because TRON is not applicable to L1-loss SVR, L2-loss SVR is considered.

A common practice in document classification is to normalize each feature vector to have unit length. Because the resulting optimization problem may have a better numerical condition, this normalization procedure often helps to shorten the training time. We will investigate its effectiveness for regression data.

We begin with comparing TRON and DCD on the original data without normalization. Figure 4 shows the following three values along the training time.



(a) MSD (DCD has only one point because the running time is too long for smaller ε_s in (25).)

(b) TFIDF-2006

(c) LOG1P-2006

(d) CTR

(e) KDD2010b

Figure 4: A comparison between TRON and DCD-sh (DCD with shrinking) on function values, MSE, and R^2. L2-loss SVR with $C = 1$ and $\varepsilon = 0.1$ is applied to the original data without normalization. The dotted and solid horizontal lines respectively indicate the function values of TRON using stopping tolerance $\varepsilon_s = 0.001$ in (7) and DCD using $\varepsilon_s = 0.1$ in (25).

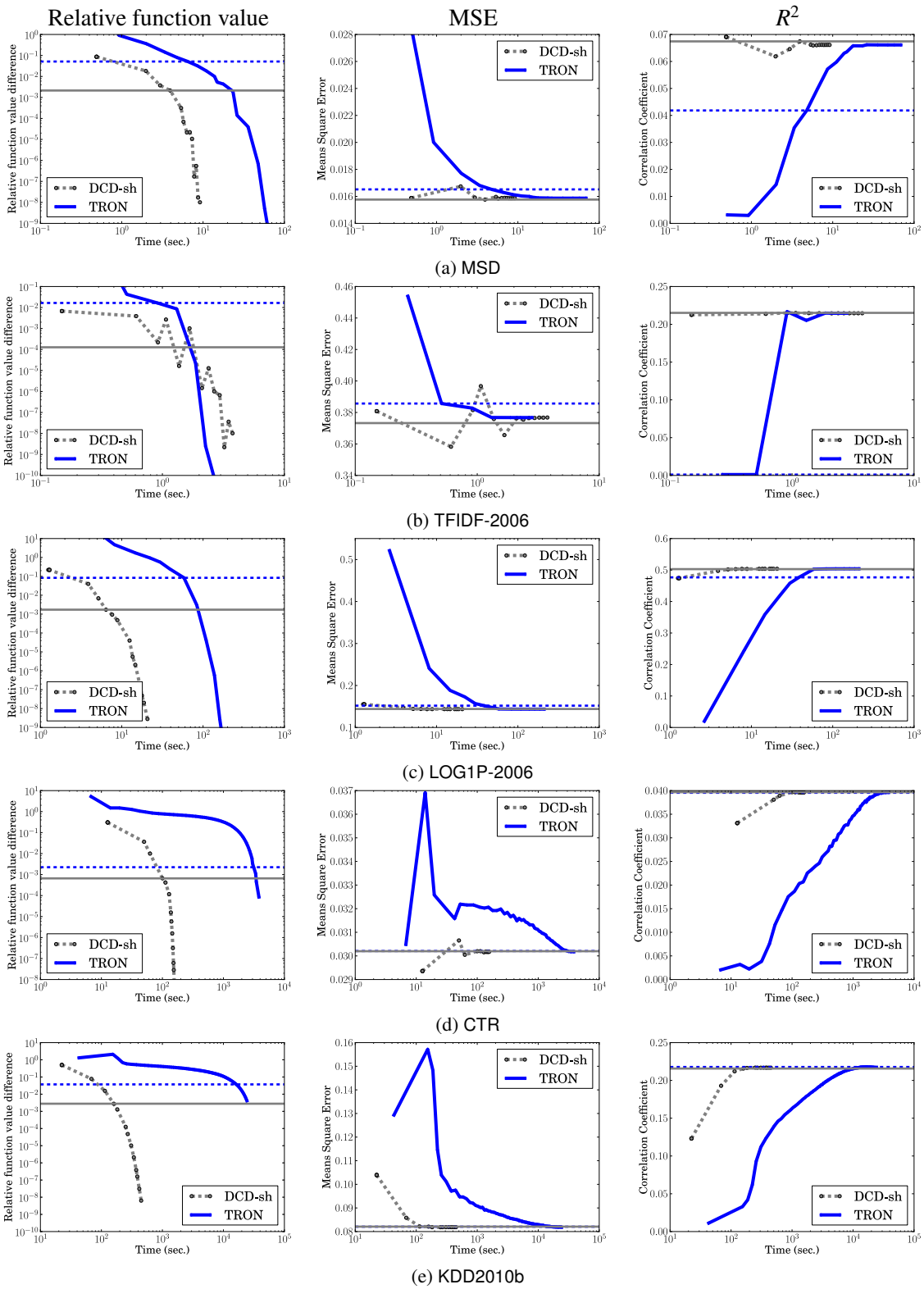


Figure 5: A comparison between TRON and DCD-sh (DCD with shrinking) on function values, MSE, and R^2. All settings are the same as Figure 4 except that data instances are normalized to unit vectors.

1. the relative difference to the optimal primal function value

$$\frac{f(\mathbf{w}) - f(\mathbf{w}^*)}{|f(\mathbf{w}^*)|}, \tag{32}$$

2. MSE,
3. R^2.

Although DCD solves the dual problem, for calculating (32), we can obtain a corresponding primal value using $\mathbf{w} = X^T\beta$. Primal values obtained in this way may not be decreasing, so DCD's curves in the first column of Figure 4 may fluctuate.[6] Because practically users apply TRON or DCD under a fixed stopping tolerance, we draw two horizontal lines in Figure 4 to indicate the result using a typical tolerance value. We use $\varepsilon_s = 0.001$ in (7) and $\varepsilon_s = 0.1$ in (25).

We observe that DCD is worse than TRON for data with few features, but becomes better for data with more features. For MSD, which has only 90 features, DCD's primal function value is so unstable that it does not reach the stopping condition for drawing the horizontal line. A primal method like TRON is more suitable for this data set because of the smaller number of variables. In contrast, KDD2010b has 29 million features, and DCD is much more efficient than TRON. This result is consistent with the situation in classification (Hsieh et al., 2008).

Next, we compare TRON and DCD on data normalized to have unit length. Results of function values, testing MSE, and testing R^2 are shown in Figures 5. By comparing Figures 4 and 5, we observe that both methods have shorter training time for normalized data. For example, for CTR, DCD is 10 times faster, while TRON is 1.6 times faster. DCD becomes very fast for all problems including MSD. Therefore, like the classification case, if data have been properly normalized, DCD is generally faster than TRON.

To compare the testing performance without/with data normalization, we show MSE in Table 3. We use DCD so we can not get MSD's result. An issue of the comparison between Figures 4 and 5 is that we use $C = 1$ and $\varepsilon = 0.1$ without parameter selection. We tried to conduct parameter selection but can only report results of the normalized data. The running time is too long for the original data. From Table 3, except TFIDF-2006, normalization does not cause inferior MSE values. Therefore, for the practical use of linear SVR, data normalization is a useful preprocessing procedure.

4.5 With and Without the Bias Term in the SVR Prediction Function

We omit the bias term in the discussion so far because we suspect that it has little effect on the performance. LIBLINEAR supports a common way to include a bias term by appending one more feature to each data instance.

$$\mathbf{x}_i^T \leftarrow [\mathbf{x}_i^T, 1] \quad \mathbf{w}^T \leftarrow [\mathbf{w}^T, b].$$

We apply L1-loss SVR on normalized data sets to compare MSE values with and without the bias term. With the stopping tolerance $\varepsilon_s = 0.001$, the results in Table 4 show that MSE values obtained with/without the bias term are similar for almost all data sets. Results in Table 2 also support this finding because LIBSVM solves SVR with a bias term. Therefore, in general the bias term may not be needed for linear SVR if data are large and sparse.

6. This phenomenon has been known in earlier studies on primal- and dual-based methods. See, for example, the discussion in Chapelle (2007) and the comparison in Hsieh et al. (2008).

Data	Original $C = 1, \varepsilon = 0.1$ MSE	Normalized $C = 1, \varepsilon = 0.1$ MSE	Normalized + parameter selection C	ε	MSE
MSD	N/A	0.0151	2^1	2^{-4}	0.0153
TFIDF-2006	0.1473	0.3828	2^6	2^{-10}	0.2030
LOG1P-2006	0.1605	0.1418	2^1	2^{-4}	0.1421
CTR	0.0299	0.0294	2^{-1}	2^{-6}	0.0287
KDD2010b	0.0904	0.0809	2^1	2^{-4}	0.0826

Table 3: Test MSE without and with data normalization. L1-loss SVR is used. Parameter selection is only applied to the normalized data because the running time is too long for the original data. Notice that for some problems, test MSE obtained after parameter selection is slightly worse than that of using $C = 1$ and $\varepsilon = 0.1$. This situation does not happen if MAE is used. Therefore, the reason might be that L1 loss is more related to MAE than MSE. See the discussion in Section 4.7.

Data	without bias	with bias
MSD	0.0151	0.0126
TFIDF-2006	0.3828	0.3264
LOG1P-2006	0.1418	0.1419
CTR	0.0294	0.0295
KDD2010b	0.0809	0.0807

Table 4: MSE of L1-loss SVR with and without the bias term.

4.6 Aggressiveness of DCD's Shrinking Scheme

In Section 3.2.2, we introduced DCD's shrinking scheme with a parameter M defined as the maximal violation of the optimality condition. We pointed out that the smaller M is, the more aggressive the shrinking method is. To check if choosing M by the way in (29) is appropriate, we compare the following settings.

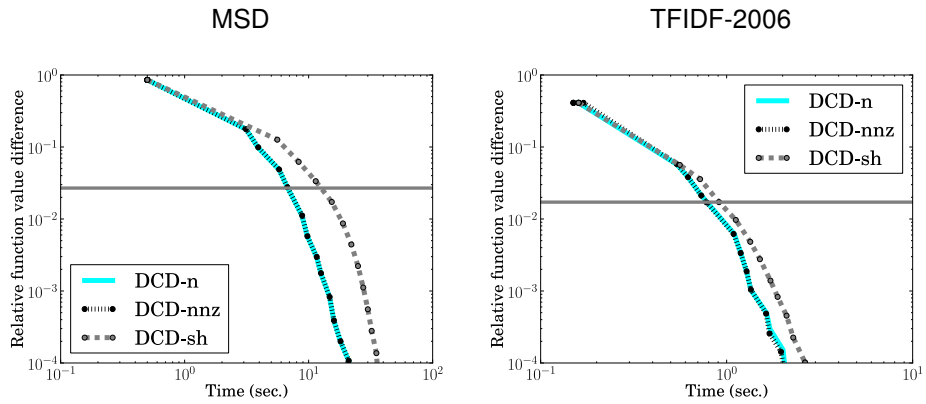
1. DCD-sh: The method in Section 3.2.2 using M defined in (29).
2. DCD-nnz: M is replaced by $M/\bar{n}$, where $\bar{n}$ is the average number of non-zero feature values per instance.
3. DCD-n: M is replaced by M/n, where n is the number of features.

Because

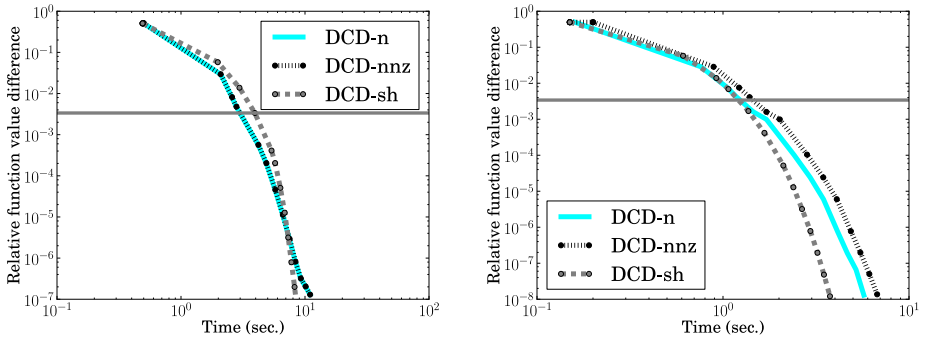
$$\frac{M}{n} < \frac{M}{\bar{n}} < M,$$

DCD-n is the most aggressive setting, while DCD-sh is the most conservative.

Using L1-loss SVR, Figure 6(a) shows the relationship between the relative difference to the optimal dual function value and the training time of MSD and TFIDF-2006. Results indicate that these data sets need a more aggressive shrinking strategy. However, if L2-loss SVR is applied instead, Figure 6(b) shows different results. Aggressive shrinking strategies make the results worse.



(a) L1-loss SVR



(b) L2-loss SVR

Figure 6: A comparison of three shrinking settings. Linear SVR with $C = 1$ and $\varepsilon = 0.1$ is applied on normalized data. We show the relative difference to the dual optimal function values and training time (in seconds).

A possible reason is that less variables are shrunk for L2-loss SVR (see explanation in Sections 3.2.2), so an aggressive strategy may wrongly shrink some variables.

4.7 L1-/L2-loss SVR, Least-square Regression, and the Need of the Parameter ε

If $\varepsilon = 0$, L1- and L2-loss SVR are respectively reduced to

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\|\mathbf{y} - X\mathbf{w}\|_1 \tag{33}$$

and

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\|\mathbf{y} - X\mathbf{w}\|_2^2. \tag{34}$$

Problem (34) is the regularized least-square regression (also called ridge regression by Hoerl and Kennard, 1970). We are interested in the need of using ε-insensitive loss function. Both our TRON

Data	L1-loss SVR							L2-loss SVR						
	with ε				ε = 0			with ε				ε = 0		
	ε	C	MSE	time	C	MSE	time	ε	C	MSE	time	C	MSE	time
MSD	2^{-4}	2^{1}	0.015	24	2^{6}	0.015	560	2^{-10}	2^{6}	**0.014**	157	2^{1}	**0.014**	7
TFIDF-2006	2^{-10}	2^{6}	**0.203**	32	2^{6}	0.334	37	2^{-10}	2^{6}	0.225	36	2^{6}	0.225	41
LOG1P-2006	2^{-4}	2^{1}	0.142	17	2^{0}	0.143	12	2^{-10}	2^{2}	**0.138**	16	2^{1}	**0.138**	10
CTR	2^{-6}	2^{-1}	**0.028**	347	2^{-1}	0.029	342	2^{-8}	2^{4}	**0.028**	960	2^{0}	**0.028**	121
KDD2010b	2^{-4}	2^{1}	0.082	291	2^{3}	0.086	795	2^{-10}	2^{-1}	**0.077**	122	2^{-1}	**0.077**	122
	ε	C	MAE	time	C	MAE	time	ε	C	MAE	time	C	MAE	time
MSD	2^{-6}	2^{-4}	**0.087**	3	2^{-3}	**0.087**	3	2^{-10}	2^{0}	0.09	4	2^{6}	0.088	159
TFIDF-2006	2^{-10}	2^{6}	**0.307**	36	2^{6}	0.441	32	2^{-10}	2^{6}	0.33	36	2^{6}	0.33	36
LOG1P-2006	2^{-4}	2^{0}	**0.238**	11	2^{0}	0.24	12	2^{-10}	2^{1}	0.244	11	2^{1}	0.244	10
CTR	2^{-8}	2^{-4}	**0.062**	93	2^{-4}	0.063	73	2^{-10}	2^{0}	0.074	107	2^{0}	0.074	90
KDD2010b	2^{-10}	2^{-2}	**0.117**	94	2^{-2}	**0.117**	106	2^{-10}	2^{-1}	0.171	124	2^{-1}	0.17	129

Table 5: A comparison between L1-/L2-loss SVR with/without using ε. Note that L2-loss SVR with ε = 0 is the same as regularized least-square regression. We present both test MSE and MAE, and boldface the best setting. Training time is in second.

and DCD implementations can be applied to the situation of ε = 0, so we conduct a comparison in Table 5 using DCD.

We also would like to compare L1 and L2 losses, so in Table 5, we present both MSE and MAE. MAE (mean absolute error) is defined as

$$\text{mean absolute error } = \frac{1}{l}\sum_{i=1}^{l} |y_i - \mathbf{w}^T \mathbf{x}_i|.$$

The reason of considering both is that L1 loss is directly related to MAE of training data, while L2 loss is related to MSE.

Results in Table 5 show that for both L1- and L2-loss SVR, MAE/MSE is similar with and without using ε. The only exception is TFIDF-2006. Under the same C, L1-loss SVR gives 0.307 MAE with $\varepsilon = 2^{-6}$, while 0.441 MAE with ε = 0. An investigation shows that the stopping condition of DCD used to generate Table 5 is too loose for this problem. If a strict stopping condition is used, the two MAE values become close to each other. Therefore, for these data sets, we may not need to use ε-insensitive loss functions. Without ε, time for parameter selection can be reduced. We suspect that ε-insensitive losses are still useful in some occasions, though more future experiments are needed for drawing conclusions.

For the comparison between L1 and L2 losses, Table 5 indicates that regardless of ε = 0 or not, L1-loss SVR gives better MAE while L2-loss SVR is better for MSE. This result is reasonable because we have mentioned that L1 and L2 losses directly model training MAE and MSE, respectively. Therefore, it is important to apply a suitable loss function according to the performance measure used for the application.

Next, we briefly discuss approaches for training L1- and L2-loss SVR with ε = 0. For regularized least-square regression, the solution of problem (34) can be obtained by solving a linear

system:

$$\mathbf{w}^* = \left(X^T X + \frac{I}{C}\right)^{-1} X^T \mathbf{y}. \tag{35}$$

Then approaches such as conjugate gradient methods can be applied instead of TRON or DCD. For the situation of using L1 loss and $\varepsilon = 0$, problem (33) is not differentiable. Nor does it have a simple solution like that in (35). However, the dual problem becomes simpler because the non-differentiable $\varepsilon\|\beta\|_1$ in (15) is removed. Then a simplified DCD can be used to minimize (15).

4.8 Summary of the Experiments

We summarize conclusions made by experiments in this section.

1. DCD is faster by solving (14) than solving (4). Further, the shrinking strategies make both DCD methods faster.
2. Linear SVR can have as good MSE values as nonlinear SVR if the data set has many features.
3. TRON is less sensitive than DCD to data with/without normalization.
4. With data normalization, DCD is generally much faster than TRON.
5. The bias term does not affect the MSE of large and sparse data sets.
6. To achieve good MSE, L2 loss should be used. In contrast, for MAE, we should consider L1 loss.

5. Discussions and Conclusions

In this paper, we extend LIBLINEAR's SVC solvers TRON and DCD to solve large-scale linear SVR problems. The extension for TRON is straightforward, but is not trivial for DCD. We propose an efficient DCD method to solve a reformulation of the dual problem. Experiments show that many properties of TRON and DCD for SVC still hold for SVR.

An interesting future research direction is to apply coordinate descent methods for L1-regularized least-square regression, which has been shown to be related to problem (15). However, we expect some differences because the former is a primal problem, while the latter is a dual problem.

For this research work, we had difficulties to obtain large and sparse regression data. We hope this work can motivate more studies and more public data in the near future.

In summary, we have successfully demonstrated that for some document data, the proposed methods can efficiently train linear SVR, while achieve comparable testing errors to nonlinear SVR. Based on this study, we have expanded the package LIBLINEAR (after version 1.9) to support large-scale linear SVR.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan via the grant 98-2221-E-002-136-MY3. The authors thank Kai-Wei Chang, Cho-Jui Hsieh, and Kai-Min Chuang for the early development of the regression code in LIBLINEAR. They also thank associate editor and anonymous reviewers for valuable comments.

Appendix A. Linear Convergence of Algorithm 3

To apply results in Tseng and Yun (2009), we first check if problem (15) is covered in their study. Tseng and Yun (2009) consider

$$\min_{\beta} F(\beta) + \varepsilon P(\beta), \tag{36}$$

where $F(\beta)$ is a smooth function and $P(\beta)$ is proper, convex, and lower semicontinuous. We can write (15) in the form of (36) by defining

$$F(\beta) \equiv \frac{1}{2}\beta^T \bar{Q}\beta - \mathbf{y}^T\beta, \quad \text{and} \quad P(\beta) \equiv \begin{cases} \|\beta\|_1 & \text{if } -U \le \beta_i \le U, \forall i, \\ \infty & \text{otherwise.} \end{cases}$$

Both $F(\beta)$ and $P(\beta)$ satisfy the required conditions. Tseng and Yun (2009) propose a general coordinate descent method. At each step certain rules are applied to select a subset of variables for update. Our rule of going through all l indices in one iteration is a special case of "Gauss-Seidel" rules discussed in their paper. If each time only one variable is updated, the subproblem of their coordinate descent method is

$$\min_{s} \begin{cases} \nabla_i F(\beta)(s-\beta_i) + \frac{1}{2}H(s-\beta_i)^2 + \varepsilon|s| & \text{if } -U \le s \le U, \\ \infty & \text{otherwise,} \end{cases} \tag{37}$$

where H is any positive value. Because we use $H = \bar{Q}_{ii}$, if $\bar{Q}_{ii} > 0$, then (16) is a special case of (37). We will explain later that $\bar{Q}_{ii} = 0$ is not a concern.

Next, we check conditions and assumptions required by Theorem 2(b) of Tseng and Yun (2009). The first one is

$$\|\nabla F(\beta_1) - \nabla F(\beta_2)\| \le L\|\beta_1 - \beta_2\|, \forall \beta_1, \beta_2 \in \{\beta \mid F(\beta) < \infty\}.$$

Because $F(\beta)$ is a quadratic function, this condition easily holds by setting the largest eigenvalue of $\bar{Q}$ as L. We then check Assumption 1 of Tseng and Yun (2009), which requires that

$$\underline{\lambda} \le \bar{Q}_{ii} \le \bar{\lambda}, \forall i, \tag{38}$$

where $\underline{\lambda} > 0$. The only situation that (38) fails is when $\mathbf{x}_i = 0$ and L1-loss SVR is applied. In this situation, $\beta^T\bar{Q}\beta$ is not related to β_i and the minimization of $-y_i\beta_i + \varepsilon|\beta_i|$ shows that the optimal β_i^* is

$$\beta_i^* = \begin{cases} U & \text{if } -y_i + \varepsilon < 0, \\ -U & \text{if } -y_i - \varepsilon > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{39}$$

We can remove these variables before applying DCD, so (38) is satisfied.[7]

For Assumption 2 in Tseng and Yun (2009), we need to show that the solution set of (36) is not empty. For L1-loss SVR, following Weierstrass' Theorem, the compact feasible domain

7. Actually, we do not remove these variables. Under the IEEE floating-point arithmetic, at the first iteration, the first two cases in (22) are ∞ and $-\infty$, respectively. Then, (21) projects the value back to U and $-U$, which are the optimal value shown in (39). Therefore, variables corresponding to $\mathbf{x}_i = \mathbf{0}$ have reached their optimal solution at the first iteration. Because they will never be updated, it is like that they have been removed.

($\beta \in [-U, U]^l$) implies the existence of optimal solutions. For L2-loss SVR, the strictly quadratic convex $F(\beta)$ implies that $\{\beta \mid F(\beta) + \varepsilon P(\beta) \leq F(\beta^0) + \varepsilon P(\beta^0)\}$ is compact, where β^0 is any vector. Therefore, the solution set is also nonempty. Then Lemma 7 in Tseng and Yun (2009) implies that a quadratic $L(\beta)$ and a polyhedral $P(\beta)$ make their Assumption 2 hold.

Finally, by Theorem 2(b) of Tseng and Yun (2009), $\{\beta^k\}$ generated by Algorithm 3 globally converges and $\{f_B(\beta^k)\}$ converges at least linearly.

References

Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *In Proceedings of the Twelfth International Society for Music Information Retrieval Conference (ISMIR 2011)*, 2011.

Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Olivier Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5): 1155–1178, 2007.

Corina Cortes and Vladimir Vapnik. Support-vector network. *Machine Learning*, 20:273–297, 1995.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf`.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the Twenty Fifth International Conference on Machine Learning (ICML)*, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/cddual.pdf`.

Thorsten Joachims. Making large-scale SVM learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–184, Cambridge, MA, 1998. MIT Press.

Thorsten Joachims. Training linear SVMs in linear time. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

S. Sathiya Keerthi and Dennis DeCoste. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

Shimon Kogan, Dimitry Levin, Bryan R. Routledge, Jacob S. Sagi, and Noah A. Smith. Predicting risk from financial reports with regression. In *In Proceedings of the North American Association for Computational Linguistics Human Language Technologies Conference*, pages 272–280, 2009.

Shuo-Peng Liao, Hsuan-Tien Lin, and Chih-Jen Lin. A note on the decomposition methods for support vector regression. *Neural Computation*, 14:1267–1281, 2002.

Chih-Jen Lin and Jorge J. Moré. Newton's method for large-scale bound constrained problems. *SIAM Journal on Optimization*, 9:1100–1127, 1999.

Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/logistic.pdf`.

Olvi L. Mangasarian. A finite Newton method for classification. *Optimization Methods and Software*, 17(5):913–929, 2002.

Yurii E. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. Technical report, CORE Discussion Paper, Université Catholique de Louvain, Louvain-la-Neuve, Louvain, Belgium, 2010. URL `http://www.ucl.be/cps/ucl/doc/core/documents/coredp2010_2web.pdf`.

John C. Platt. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA, 1998. MIT Press.

Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. Technical report, School of Mathematics, University of Edinburgh, 2011.

Shai Shalev-Shwartz and Ambuj Tewari. Stochastic methods for l1-regularized loss minimization. *Journal of Machine Learning Research*, 12:1865–1892, 2011.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In *Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML)*, 2007.

Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117:387–423, 2009.

Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, NY, 1995.

Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A comparison of optimization methods and software for large-scale l1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/l1.pdf`.

Journal of Machine Learning Research 13 (2012) 3349-3386 Submitted 4/12; Revised 10/12; Published 11/12

Sparse and Unique Nonnegative Matrix Factorization Through Data Preprocessing

Nicolas Gillis* NICOLAS.GILLIS@UCLOUVAIN.BE
ICTEAM Institute
Université catholique de Louvain
B-1348 Louvain-la-Neuve, Belgium

Editor: Inderjit Dhillon

Abstract

Nonnegative matrix factorization (NMF) has become a very popular technique in machine learning because it automatically extracts meaningful features through a sparse and part-based representation. However, NMF has the drawback of being highly ill-posed, that is, there typically exist many different but equivalent factorizations. In this paper, we introduce a completely new way to obtaining more well-posed NMF problems whose solutions are sparser. Our technique is based on the preprocessing of the nonnegative input data matrix, and relies on the theory of M-matrices and the geometric interpretation of NMF. This approach provably leads to optimal and sparse solutions under the separability assumption of Donoho and Stodden (2003), and, for rank-three matrices, makes the number of exact factorizations finite. We illustrate the effectiveness of our technique on several image data sets.

Keywords: nonnegative matrix factorization, data preprocessing, uniqueness, sparsity, inverse-positive matrices

1. Introduction

Given an m-by-n nonnegative matrix $M \geq 0$ and a factorization rank r, nonnegative matrix factorization (NMF) looks for two nonnegative matrices U and V of dimension m-by-r and r-by-n respectively such that $M \approx UV$. To assess the quality of an approximation, a popular choice is the Frobenius norm of the residual $||M-UV||_F$ and NMF can for example be formulated as the following optimization problem

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{r \times n}} ||M-UV||_F^2 \quad \text{such that } U \geq 0 \text{ and } V \geq 0. \tag{1}$$

Assuming that M is a matrix where each column represents an element of a data set (for example, a vectorized image of pixel intensities), NMF can be interpreted in the following way. Since $M_{:j} \approx \sum_{k=1}^{r} U_{:k} V_{kj} \ \forall j$, each column $M_{:j}$ of M is reconstructed using an additive linear combination of nonnegative basis elements (the columns of U). These basis elements can be interpreted in the same way as the columns of M (for example, as images). Moreover, they can only be summed up (since V is nonnegative) in order to approximate the original data matrix M which leads to a part-based representation: NMF will automatically extract localized and meaningful features from the data set.

*. The author is a postdoctoral researcher with the fonds de la recherche scientifique-FNRS (F.R.S.-FNRS). This text presents research results of the Belgian Program on Interuniversity Poles of Attraction initiated by the Belgian State, Prime Minister's Office, Science Policy Programming. The scientific responsibility is assumed by the author.

The most famous illustration of such a decomposition is when the columns of M represent facial images for which NMF is able to extract common features such as eyes, noses and lips (Lee and Seung, 1999); see Figure 8 in Section 6.

NMF has become a very popular data analysis technique and has been successfully used in many different areas such as hyperspectral imaging (Pauca et al., 2006), text mining (Xu et al., 2003), clustering (Ding et al., 2005), air emission control (Paatero and Tapper, 1994), blind source separation (Cichocki et al., 2009), and music analysis (Févotte et al., 2009).

1.1 Geometric Interpretation of NMF

A very useful tool for understanding NMF better is its geometric interpretation. In fact, NMF is closely related to a problem in computational geometry consisting in finding a polytope nested between two given polytopes. In this section, we briefly recall this connection, which will be extensively used throughout the paper.

Let (U,V) be an exact NMF of M (that is, $M = UV$, $U \geq 0$ and $V \geq 0$), and let us assume that no column of U or M is all zeros; otherwise they can be removed without loss of generality.

Definition 1 (Pullback map) *Given an m-by-n nonnegative matrix X without all-zero column, $D(X)$ is the n-by-n diagonal matrix whose diagonal elements are the inverse of the ℓ_1-norms of the columns of X:*

$$D(X)_{ii} = ||X_{:i}||_1^{-1} = \left(\sum_{k=1}^{m} |X_{ki}| \right)^{-1} \quad \forall i, \quad D(X)_{ij} = 0 \; \forall i \neq j, \tag{2}$$

and $\theta(X) = XD(X)$ is the pullback map of X so that $\theta(X)$ is column stochastic, that is, $\theta(X)$ is nonnegative and its columns sum to one.

We have that (see Chu and Lin, 2008)

$$M = UV \iff \theta(M) = MD(M) = \underbrace{UD(U)}_{\theta(U)} \underbrace{D(U)^{-1}VD(M)}_{V'} \iff \theta(M) = \theta(U)V',$$

where V' must be column stochastic since $\theta(M)$ and $\theta(U)$ are both column stochastic and $\theta(M) = \theta(U)V'$. Therefore, the columns of $\theta(M)$ are convex combinations (linear combinations with nonnegative weights summing to one) of the columns of $\theta(U)$. This implies that

$$\text{conv}(\theta(M)) \quad \subseteq \quad \text{conv}(\theta(U)) \quad \subseteq \quad \Delta^m, \tag{3}$$

where $\text{conv}(X)$ denotes the convex hull of the columns of matrix X, and $\Delta^m = \{x \in \mathbb{R}^m \mid \sum_i^m x_i = 1, x_i \geq 0 \; 1 \leq i \leq m\}$ is the unit simplex (of dimension $m-1$). An exact NMF $M = UV$ can then be geometrically interpreted as a polytope $T = \text{conv}(\theta(U))$ nested between an inner polytope $\text{conv}(\theta(M))$ and an outer polytope Δ^m.

> Hence finding the minimal number of nonnegative rank-one factors to reconstruct M exactly is equivalent to finding a polytope T with minimum number of vertices nested between two given polytopes: the inner polytope $\text{conv}(\theta(M))$ and the outer polytope Δ^m.

This problem is referred to as the nested polytopes problem (NPP), and is then equivalent to computing an exact nonnegative matrix factorization (Hazewinkel, 1984); see also Gillis and Glineur (2012a) and the references therein. In the remaining of the paper, we will denote NPP(M) the NPP instance corresponding to M with inner polytope $\text{conv}(\theta(M))$ and outer polytope Δ^m.

Remark 2 *The geometric interpretation can also be equivalently characterized in terms of cones, see Donoho and Stodden (2003), for which we have*

$$\text{cone}(M) \quad \subseteq \quad \text{cone}(U) \quad \subseteq \quad \mathbb{R}^m_+,$$

where $\text{cone}(X) = \{x | x = Xa, a \geq 0\}$. *The geometric interpretation based on convex hulls from Equation* (3) *amounts to the intersection of the cones with the hyperplane* $\{x | \sum x_i = 1\}$ *(this is the reason why zero columns of M and U need to be discarded in that case).*

1.2 Uniqueness of NMF

There are several difficulties in using NMF in practice. In particular, the optimization problem (1) is NP-hard (Vavasis, 2009), and typically only convergence to stationary points is guaranteed by standard algorithms. There does not seem to be an easy way to go around this (except if the factorization rank is very small, see Arora et al., 2012) since NMF problems typically have many local minima.

Another difficulty is the non-uniqueness: even if one is given an optimal (or good) NMF (U,V) of M, there might exist many equivalent solutions $(UQ, Q^{-1}V)$ for non-monomial[1] matrices Q with $UQ \geq 0$ and $Q^{-1}V \geq 0$, see Laurberg et al. (2008). Such transformations lead to different interpretations, especially when the supports of U and V change. For example, in document classification, each entry M_{ij} of matrix M indicates the 'importance' of word i in document j (for example, the number of appearances of word i in text j). The factors (U,V) of NMF are interpreted as follows: the columns of U represent the topics (that is, bags of words) while the columns of V link the documents to these topics. The sparsity patterns of U and V are then a crucial characteristic since they indicate which words belong to which topics and which topics is discussed by which documents.

Different approaches exist to obtain (more) well-posed NMF problems and most of them are based on the incorporation of additional constraints into the NMF model, for example,

- **Sparsity**. Require the factors in NMF to be sparse. Under some appropriate assumptions, this leads to a unique solution (Theis et al., 2005). Geometrically, requiring the matrix U to be sparse is equivalent to requiring the vertices of the nested polytope $\text{conv}(\theta(U))$ to be located on the low-dimensional faces of the outer polytope Δ^m, hence making the problem more well posed. In practice, the most popular technique to obtain sparser solutions is to add sparsity inducing penalty terms, such as a ℓ_1-norm penalty (Kim and Park, 2007) (see also Section 6). Another possibility is to use a projection onto the set of sparse matrices (Hoyer, 2004).

- **Minimum Volume**. Require the polytope $\text{conv}(\theta(U))$ to have minimum volume (Miao and Qi, 2007; Huck et al., 2010; Zhou et al., 2011) which has a long history in hyperspectral imaging (Craig, 1994). Again, this constraint is typically enforced using a proper penalty term in the objective function. Volume maximization of $\text{conv}(\theta(U))$ is also possible, leading to a sparser factor U (since the columns of U will be encouraged to be on the faces of Δ^m), see Wang et al. (2010), which is essentially equivalent to performing volume minimization for the matrix transpose. In fact, taking the polar of the three polytopes in Equation (3) interchanges the role of the inner and outer polytopes, while the polar of $\text{conv}(\theta(M))$ is given by $\text{conv}(\theta(M^T))$, see Gillis (2011, Section 3.6).

1. A monomial matrix is a permutation of a diagonal matrix with positive diagonal elements.

- **Orthogonality**. Require the columns of matrix U to be orthogonal (Ding et al., 2006). Geometrically, it amounts to position the vertices of $\text{conv}(\theta(U))$ on the low-dimensional faces of Δ^m so that if one of the columns of $\theta(U)$ is not on a facet of Δ^m (that is, $U_{ik} > 0$ for some i,k), then all the other columns of U must be on that facet (that is, $U_{ip} = 0\ \forall p \neq k$). This condition is rather restrictive, but proved successful in some situations, for example in clustering; see Ding et al. (2005) and Pompili et al. (2011).

1.3 Outline of the Paper

In this paper, we address the problem of uniqueness and introduce a completely new approach to make NMF problems more well posed, and obtain sparser solutions. Our technique is based on a preprocessing of the input matrix M to make it sparser while preserving its nonnegativity and its column space. The motivation is based on the geometric interpretation of NMF which shows that *sparser matrices will correspond to more well-posed NMF problems whose solutions are sparser*.

In Section 2, we recall how sparsity of M makes the corresponding NMF problem more well posed. In particular, we give a new result linking the support of M and the uniqueness of the corresponding NMF problem. In Section 3, we introduce a preprocessing $\mathcal{P}(M) = MQ$ of M where Q is an inverse-positive matrix, that is, Q has full rank and its inverse Q^{-1} is nonnegative. Hence, if (U,V') is an NMF of $\mathcal{P}(M)$ with $\mathcal{P}(M) \approx UV'$, then $(U,V'Q^{-1})$ is an NMF of M since $M = \mathcal{P}(M)Q^{-1} \approx UV'Q^{-1}$ and $V'Q^{-1} \geq 0$. In Section 4, we prove some important properties of the preprocessing; in particular that it is well-defined, invariant to permutation and scaling, and optimal under the separability assumption of Donoho and Stodden (2003). Moreover, in the exact case for rank-three matrices (that is, $M = UV$ and $\text{rank}(M) = 3$), we show how the preprocessing can be used to obtain an equivalent NMF problem with a finite number of solutions. In Section 5, we address some practical issues of using the preprocessing: the computational cost, the rescaling of the columns $\mathcal{P}(M)$ and the ability to dealing with sparse and noisy matrices. In Section 6, we present some very promising numerical experiments on facial and hyperspectral image data sets.

2. Non-Uniqueness, Geometry and Sparsity

Let $M \in \mathbb{R}^{m\times n}_+$ and $(U,V) \in \mathbb{R}^{m\times r}_+ \times \mathbb{R}^{r\times n}_+$ be an exact nonnegative matrix factorization of $M = UV$. The minimum r such that such a decomposition exists is the nonnegative rank of M and will be denoted $\text{rank}_+(M)$. If U is not full rank (that is, $\text{rank}(U) < r$), then the decomposition is typically not unique. In fact, the convex combinations (given by $V \geq 0$) cannot in general be uniquely determined: the polytope $T = \text{conv}(\theta(U))$ has r vertices while its dimension is strictly smaller than $r-1$ implying that any point in the interior of T can be reconstructed with infinitely many convex combinations of the r vertices of T. However, if all columns of $\text{conv}(\theta(M))$ are located on k-dimensional faces of T having exactly $k+1$ vertices, then the convex combinations given by V are unique (Sun and Xin, 2011).

In practice, it is therefore often implicitly assumed that $\text{rank}_+(M) = \text{rank}(M) = r$ hence $\text{rank}(U) = r$ (since U has r columns and spans the column space of M of dimension r); see the discussion by Arora et al. (2012) and the references therein. In this situation, the uniqueness can be characterized as follows:

Theorem 3 (Laurberg et al., 2008) *Let $(U,V) \in \mathbb{R}^{m\times r}_+ \times \mathbb{R}^{r\times n}_+$ and $M = UV$ with $\text{rank}(M) = \text{rank}(U) = r$. Then the following statments are equivalent:*

(i) The exact NMF (U,V) of M is unique (up to permutation and scaling).

(ii) There does not exist a non-monomial invertible matrix Q such that $U' = UQ \geq 0$ and $V' = Q^{-1}V \geq 0$.

(iii) The polytope $\text{conv}(\theta(U))$ *is the unique solution of NPP(M) with r vertices.*

It is interesting to notice that the columns of M containing zero entries are located on the boundary of the outer polytope Δ^m, and these points must be on the boundary of any solution T of NPP(M). Therefore, if M contains many zero entries, it is more likely that the set of exact NMF of M will be smaller, since there is less degree of freedom to fill in the space between the inner and outer polytopes. In particular, Donoho and Stodden (2003) showed that "requiring that some of the data are spread across the faces of the nonnegative orthant, there is unique simplicial cone", that is, there is a unique $\text{conv}(\theta(U))$.

In the following, based on the assumption that $\text{rank}(M) = \text{rank}_+(M)$, we provide a new uniqueness result using the geometric interpretation of NMF and the sparsity pattern of M.

Lemma 4 *Let $M \in \mathbb{R}^{m\times n}$ with $r = \text{rank}(M) = \text{rank}_+(M)$, and M have no all-zero columns. If r columns of $\theta(M)$ coincide with r different vertices of $\Delta^m \cap \text{col}(\theta(M))$, then the exact NMF of M is unique.*

Proof Let $(U,V) \in \mathbb{R}_+^{m\times r} \times \mathbb{R}_+^{r\times n}$ be such that $M = UV$. Since $r = \text{rank}(M) = \text{rank}_+(M)$, we must have $\text{rank}(U) = r$ and $\text{col}(U) = \text{col}(M)$ (where $\text{col}(X)$ denotes the column space of matrix X), hence

$$\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(U)) \subseteq \Delta^m \cap \text{col}(\theta(M)).$$

Since r columns of $\theta(M)$ coincide with r vertices of $\Delta^m \cap \text{col}(\theta(M))$, we have that $\text{conv}(\theta(U)) = \text{conv}(\theta(M))$ is the unique solution of NPP(M), and Theorem 3 allows to conclude. ■

In order to identify such matrices, it would be nice to characterize the vertices of $\Delta^m \cap \text{col}(\theta(M))$ based solely on the sparsity pattern of M. By definition, the vertices of $\Delta^m \cap \text{col}(\theta(M))$ are the intersection of $r-1$ of its facets, and the facets of $\Delta^m \cap \text{col}(\theta(M))$ are given by

$$F_i = \{x \in \Delta^m \cap \text{col}(\theta(M)) \mid x_i = 0\}.$$

Therefore, a vertex of $\Delta^m \cap \text{col}(\theta(M))$ must contain at least $r-1$ zero entries. However, this is not a sufficient condition because some facets might be redundant, for example, if the ith row of M is identically equal to zero (for which $F_i = \Delta^m \cap \text{col}(\theta(M))$) or if the ith and jth row of M are equal to each other (for which $F_i = F_j$).

Lemma 5 *A column of M containing $r-1$ zeros whose corresponding rows have different sparsity patterns corresponds to a vertex of* $\text{conv}(\theta(M)) \cap \Delta^m$.

Proof Let c be one of the columns of M with at least $r-1$ zeros corresponding to rows with different sparsity patterns, that is, there exists $\mathcal{J} \subseteq \{i \mid c_i = 0\}$ with $|\mathcal{J}| = r-1$ such that the rows of $M(\mathcal{J},:)$ have different sparsity patterns. Let also $F_k = \{x \mid x_{\mathcal{J}(k)} = 0\}$ for $1 \leq k \leq r-1$ denote the $r-1$ facets with $\theta(c) \in F_k \ \forall k$. To show that $\theta(c)$ is a vertex of $\text{conv}(\theta(M)) \cap \Delta^m$, it suffices to show that the

$r-1$ facets are not redundant: for all $1 \leq k < p \leq r-1$, there exist x_k and x_p in $\text{conv}(\theta(M)) \cap \Delta^m$ such that $x_k \in F_k, x_k \notin F_p$ and $x_p \in F_p, x_p \notin F_k$. Because the rows of $M(\mathcal{J},:)$ have different sparsity patterns, for all $1 \leq k < p \leq r-1$, there must exist two indices h and l such that $M(\mathcal{J}(k),h) = 0$ and $M(\mathcal{J}(p),h) > 0$ while $M(\mathcal{J}(k),l) > 0$ and $M(\mathcal{J}(p),l) = 0$. Therefore, $\theta(M_{:h}) \in F_k, \theta(M_{:h}) \notin F_p$ and $\theta(M_{:l}) \in F_p, \theta(M_{:l}) \notin F_k$ and the proof is complete. ■

Theorem 6 *Let $M \in \mathbb{R}^{m \times n}$ with $r = \text{rank}(M) = \text{rank}_+(M)$. If M has r non-zero columns each having $r-1$ zero entries whose corresponding rows have different sparsity patterns, then the NMF of M is unique.*

Proof This follows directly from Lemma 4 and 5. ■

Here is an example,

$$M = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix},$$

with $\text{rank}(M) = \text{rank}_+(M) = 3$ whose unique NMF is $M = MI$, where I is the identity matrix of appropriate dimension. Other examples include matrices containing an r-by-r monomial submatrix; see also Kalofolias and Gallopoulos (2012) and the references therein. It is interesting to notice that this result implies that the only 3-by-3 rank-three nonnegative matrices having a unique exact NMF are the monomial matrices (permutation and scaling of the identity matrix) since all other matrices have at least two distinct exact NMF: $M = MI = IM$.

Finally, although sparsity is neither a necessary (see Remark 7 below) nor a sufficient condition for uniqueness (except in some cases, see for example Theorem 6 or Donoho and Stodden, 2003), the geometric interpretation of NMF shows that sparser matrices M lead to more well-posed NMF problems. In fact, many points of the inner polytope in NPP(M) are located on the boundary of the outer polytope Δ^m. Moreover, because the solution T must contain these points, it will have zero entries as well. In particular, assuming M does not contain a zero column, it is easy to check that for $M = UV$ we have

$$M_{ij} = 0 \quad \Rightarrow \quad \exists k \text{ such that } U_{ik} = 0.$$

Remark 7 *Having many zero entries in M is not a necessary condition for having an unique NMF. In fact, Laurberg et al. (2008) showed that there exist positive matrices with unique NMF. However, for an NMF (U,V) to be unique, the support of each column of U (resp. row of V) cannot be contained in the support of any another column (resp. row) so that each column of U (resp. row of V) must have at least one zero entry. In fact, assume the support of the kth column of U is contained in the support of lth column. Then noting $\bar{p} = \text{argmin}_{\{p|U(p,k)\neq 0\}} \frac{U(p,l)}{U(p,k)}$, $\varepsilon = \frac{U(\bar{p},l)}{U(\bar{p},k)}$, and*

$$D_{kl} = -\varepsilon, \quad D_{ii} = 1 \; \forall i, \quad D_{ij} = 0 \text{ otherwise},$$

one can check that D^{-1} is as follows

$$D^{-1}_{kl} = \varepsilon, \quad D^{-1}_{ii} = 1 \; \forall i, \quad D^{-1}_{ij} = 0 \text{ otherwise},$$

that is $D^{-1} \geq 0$. Therefore $(UD, D^{-1}V)$ is an equivalent NMF with a different sparsity pattern since $(UD)_{:l} = UD_{:l} = U_{:l} - \varepsilon U_{:k} \geq 0$, and $U_{\bar{p}l} > 0$ while $(UD)_{\bar{p}l} = 0$.

3. Preprocessing for More Well-Posed and Sparser NMF

In this section, we introduce a completely new approach to obtain more well-posed NMF problems whose solutions are sparser. As it was shown in the previous paragraph, this can be achieved by working with sparser nonnegative matrices. Hence, we look for an n-by-n matrix Q such that $MQ = M'$ is nonnegative, sparse and Q is inverse-positive. In other words, we would like to solve the following problem:

$$\min_{Q \in \mathbb{R}^{n \times n}} ||MQ||_0 \quad \text{such that} \quad MQ \geq 0 \text{ and } Q^{-1} \geq 0, \tag{4}$$

where $||X||_0$ is the ℓ_0-'norm' which counts the number of non-zero entries in X. Assuming we can solve (4) and obtain a matrix $M' = MQ$, then any NMF (U, V') of M' with $M' \approx UV'$ gives a NMF for M. In fact,

$$M \;=\; M'Q^{-1} \;\approx\; UV'Q^{-1} \;=\; UV, \;\text{ where } V = V'Q^{-1} \geq 0,$$

for which we have

$$||M - UV||_F = ||M'Q^{-1} - UV'Q^{-1}||_F = ||(M' - UV')Q^{-1}||_F \leq ||M' - UV'||_F \; ||Q^{-1}||_2.$$

In particular, if the NMF of M' is exact, then we also have an exact NMF for $M = M'Q^{-1} = UV'Q^{-1} = UV$. The converse direction, however, is not always true. We return to this point in Section 4.3.

In the remaining of this section, we propose a way to finding approximate solutions to problem (4). First, we briefly review some properties of inverse-positive matrices (Section 3.1) in order to deal with the constraint $Q^{-1} \geq 0$. Then, we replace the ℓ_0-'norm' with the ℓ_2-norm and solve the corresponding optimization problem using constrained linear least squares (Section 3.2).

3.1 Inverse-Positive Matrices

In this section, we recall the definition of three types of matrices: Z-matrices, M-matrices and inverse-positive matrices, briefly recall how they are related and provide some useful properties. We refer the reader to the book of Berman and Plemmons (1994) and the references therein for more details on the subject.

Definition 8 *An n-by-n Z-matrix is a real matrix with non-positive off-diagonal entries.*

Definition 9 *An n-by-n M-matrix is a real matrix of the following form:*

$$A = sI - B, \quad s > 0, \quad B \geq 0,$$

where the spectral radius[2] $\rho(B)$ *of B satisfies* $s \geq \rho(B)$.

It is easy to see that an M-matrix is also a Z-matrix.

Definition 10 *An n-by-n matrix Q is inverse positive if and only if* Q^{-1} *exists and* Q^{-1} *is nonnegative. We will denote this set* $I\mathcal{P}^n$*:*

$$I\mathcal{P}^n = \{Q \in \mathbb{R}^{n \times n} \mid Q \text{ is full rank and } Q^{-1} \geq 0\}.$$

2. The spectral radius $\rho(B)$ of a n-by-n matrix B is the supremum among all the absolute values of the eigenvalues of B: $\rho(B) = \max_i |\lambda_i(B)|$.

It can be shown that inverse-positive Z-matrices are M-matrices:

Theorem 11 (Berman and Plemmons 1994, Theorem 2.3) *Let A be a Z-matrix. Then the following conditions are equivalent :*

- *A is an invertible M-matrix.*
- $A = sI - B$ *with* $B \geq 0$, $s > \rho(B)$.
- $A \in I\mathcal{P}^n$.

Here is another well-known theorem in matrix theory which will be useful, see Taussky (1949) and the references therein.

Definition 12 *An n-by-n matrix A is irreducible if and only if there does not exist an n-by-n permutation matrix P such that*

$$P^T A P = \begin{pmatrix} B & C \\ 0 & D \end{pmatrix},$$

where B and D are square matrices.

Definition 13 *An n-by-n matrix A is irreducibly diagonally dominant if A is irreducible,*

$$|A_{ii}| \geq \sum_{k \neq i} |A_{ki}|, \quad \text{for } i = 1, 2, \ldots, n,$$

and the inequality is strict for at least one i.

Theorem 14 *If A is irreducibly diagonally dominant, then A is nonsingular.*

3.2 Constrained Linear Least Squares Formulation for (4)

The ℓ_0-'norm' is of combinatorial nature and typically leads to intractable optimization problems. The standard approach is to use the ℓ_1-norm instead but we propose here to use the ℓ_2-norm. The reason is twofold:

- When looking at the structure of problem (4), we observe that any (reasonable) norm will induce solutions with zero entries. In fact, some of the constraints $MQ \geq 0$ will always be active at optimality because of the objective function $||MQ||$.
- The ℓ_2-norm is smooth hence its optimization can be performed more efficiently.[3]

We then would like to solve

$$\min_{Q \in I\mathcal{P}^n} ||MQ||_F^2 \quad \text{such that} \quad MQ \geq 0. \tag{5}$$

Optimizing over the set of inverse-positive matrices $I\mathcal{P}^n$ seems to be very difficult. At least, describing $I\mathcal{P}^n$ explicitly as a semi-algebraic set requires about n^2 polynomial inequalities of degree

3. Because of the constraint $MQ \geq 0$, the ℓ_1-norm problem can actually be decoupled into n linear programs (LP) in n variables and $m+n$ constraints, and can be solved effectively. However, in the noisy case (cf. Section 5.3), we would need to introduce mn auxiliary variables (one for each term of the objective function) which turns out to be impractical.

up to n, each with up to $n!$ terms. However, we are not aware of a rigorous analysis of the complexity of this type of problems; this is a topic for further research.

For this reason, we will restrict the search space to the subset of Z-matrices, that is, inverse-positive matrices of the form $Q = sI - B$, where s is a nonnegative scalar, I is the identity matrix of appropriate dimension and B is a nonnegative matrix such that $\rho(B) < s$, see Section 3.1. It is important to notice that

- The scalar s cannot be chosen arbitrarily. In fact, making s go to zero and $B = 0$, the objective function value goes to zero, which is optimal for (5). The same degree of freedom is in fact present in the original problem (4) since Q and αQ for any $\alpha > 0$ are equivalent solutions. Therefore, without loss of generality, we fix s to one .

- The diagonal entries of B cannot be chosen arbitrarily. In fact, taking B arbitrarily close (but smaller) to the identity matrix, the infimum of (5) will be equal to zero. We then have to set an upper bound (smaller than one) for the diagonal entries of B. It can be checked that this upper bound will always be attained (because of the minimization), and that the optimal solutions corresponding to different upper bounds will be multiples of each other. We therefore fix the bound to zero implying $B_{ii} = 0$ for all i so that $Q_{ii} = 1$ for all i.

Finally, we would like to solve

$$\min_{Q \in \mathcal{Q}^n} \quad ||MQ||_F^2 \quad \text{such that} \quad MQ \geq 0,$$

where

$$\mathcal{Q}^n = \{Q \in \mathbb{R}^{n \times n} \mid Q = I - B, B \geq 0, B_{ii} = 0 \; \forall i, \rho(B) < 1\} \subset I\mathcal{P}^n.$$

Since $MQ = M(I - B) \geq 0$, this problem is equivalent to

$$\begin{aligned} \min_{B \in \mathbb{R}^{n \times n}} \quad & \sum_{i=1}^{n} \left\| M_{:i} - \sum_{k \neq i} M_{:k} B_{ki} \right\|_2^2 \\ \text{such that} \quad & M \geq MB, \\ & \rho(B) < 1, \\ & B_{ii} = 0 \; \forall i, \; B \geq 0. \end{aligned} \tag{6}$$

Without the constraint on the spectral radius of B, this is a constrained linear least squares problem (CLLS) in $O(n^2)$ variables and $O(n^2 + mn)$ constraints. The ith column of $M' = MQ$, which is the preprocessed version of M, will then be given by the following linear combination

$$M'_{:i} = MQ_{:i} = M_{:i} - \sum_{k=1}^{n} M_{:k} B_{ki} \geq 0, \quad \text{where } B_{ki} \geq 0 \; \forall i, k \text{ and } B_{ii} = 0. \tag{7}$$

This means that we will subtract from each column of M a nonnegative linear combination of the other columns of M in order to maximize its sparsity while keeping its nonnegativity. Intuitively, this amounts to keeping only the non-redundant information from each column of M (see Section 6 for some visual examples).

3.2.1 RELAXING THE CONSTRAINT ON THE SPECTRAL RADIUS

In general, there is no easy way to deal with the non-convex constraint $\rho(B) < 1$. In particular, this constraint may lead to difficult optimization problems, for example, finding the nearest stable matrix to an unstable one:

$$\min_X ||X - A|| \quad \text{such that} \quad \rho(X) \leq 1,$$

see Polyak and Shcherbakov (2005) and the references therein. This means that even the projection on the feasible set is non-trivial.

However, we will prove in Section 4 that if the columns of M are not multiples of each other, then any optimal solution of problem (6) without the constraint on the spectral radius of B, that is, any optimal solution B^* of

$$\min_{B \in \mathbb{R}^{n \times n}_+} \quad \sum_{i=1}^{n} \left\| M_{:i} - \sum_{k \neq i} M_{:k} B_{ki} \right\|_2^2 \quad \text{such that} \quad M \geq MB,\ B_{ii} = 0\ \forall i, \tag{8}$$

automatically satisfies $\rho(B^*) < 1$ (Theorem 21). Hence, the approach may only fail when there are repetitions in the data set. The reason is that when a column is multiple of another one, say $M_{:i} = \alpha M_{:j}$ for $i \neq j$ and $\alpha > 0$, then taking $B_{ij} = \alpha$ (0 otherwise for that column) gives $MQ_{:i} = M_{:i} - \alpha M_{:j} = 0$ and similarly for $M_{:j}$. Hence we have lost a component in our data set and potentially produce a lower rank matrix MQ. In practice, it will be important to make sure that the columns of M are not multiples of each other (even though it is usually not the case for well-constructed data sets).

4. Properties of the Preprocessing

In the remainder of the paper, we denote $\mathcal{B}^*(M)$ the set of optimal solutions of problem (8) for the data matrix M, and $\mathcal{P}$ the preprocessing operator defined as

$$\mathcal{P} : \mathbb{R}^{m \times n}_+ \to \mathbb{R}^{m \times n}_+ : M \mapsto \mathcal{P}(M) = M(I - B^*), \text{ where } B^* \in \mathcal{B}^*(M).$$

In this section, we prove some important properties of $\mathcal{P}$ and $\mathcal{B}^*(M)$:

- The preprocessing operator $\mathcal{P}$ is well-defined (Theorem 15).
- The preprocessing operator $\mathcal{P}$ is invariant to permutation and scaling of the columns of M (Lemma 16).
- If the columns of $\theta(M)$ are distinct, then $\rho(B^*) < 1$ for any $B^* \in \mathcal{B}^*(M)$ (Theorem 21).
- If the vertices of $\text{conv}(\theta(M))$ are distinct then
  - There exists $B^* \in \mathcal{B}^*(M)$ such that $\rho(B^*) < 1$ (Corollary 22).
  - $\text{rank}(\mathcal{P}(M)) = \text{rank}(M)$ and $\text{rank}_+(\mathcal{P}(M)) \geq \text{rank}_+(M)$ (Corollary 19).
- If the matrix M is separable, then the preprocessing allows to recover a sparse and optimal solution of the corresponding NMF problem (Theorem 24). In particular, it is always optimal for rank-two matrices (Corollary 25).
- If the matrix has rank-three, then the preprocessing yields an instance in which the number of solutions of the exact NMF problem is finite (Theorem 29).

4.1 General Properties

A crucial property of our preprocessing is that it is well-defined.

Theorem 15 *The preprocessing $\mathcal{P}(M)$ is well-defined: for any $B_1^* \in \mathcal{B}^*(M), B_2^* \in \mathcal{B}^*(M)$, we have $M(I-B_1^*) = M(I-B_2^*) = \mathcal{P}(M)$.*

Proof Problem (8) can be decoupled into n independent CLLS (one for each column of M) of the form:

$$\min_{b \in \mathbb{R}_+^{n-1}} \|d - Cb\|^2 \text{ such that } Cb \leq d, \tag{9}$$

which is equivalent to

$$\min_{b \in \mathbb{R}_+^{n-1}, y \in \mathbb{R}^m} \quad \|d-y\|^2 \text{ such that } y \leq d, y = Cb.$$

The result follows from the fact that the ℓ_2 projection onto a polyhedral set (actually any convex set) yields a unique point. ■

Another important property of the preprocessing is its invariance to permutation and scaling of the columns of M.

Lemma 16 *Let M be a nonnegative matrix and P be a monomial matrix. Then, $\mathcal{P}(MP) = \mathcal{P}(M)P$.*

Proof We are going to show something slightly stronger; namely that B^* is an optimal solution of (8) for matrix M if and only if $P^{-1}B^*P$ is an optimal solution of (8) for matrix MP:

$$B^* \in \mathcal{B}^*(M) \quad \iff \quad P^{-1}B^*P \in \mathcal{B}^*(MP).$$

First, note that B is a feasible solution of (8) for M if and only if $P^{-1}BP$ is a feasible solution of (8) for MP. In fact, nonnegativity of B and its diagonal zero entries are clearly preserved under permutation and scaling while

$$M \geq MB \iff MP \geq MBP \iff MP \geq (MP)(P^{-1}BP).$$

Hence there is one-to-one correspondence between feasible solutions of (8) for M and (8) for MP.

Then, let B^* be an optimal solution of (8). Because (8) can be decoupled into n independent CLLS's, one for each column of B (cf. Equation (9)), we have

$$||M_{:i} - MB^*_{:i}||_2^2 \leq ||M_{:i} - MB_{:i}||_2^2, \quad \forall i,$$

for any feasible solution B of (8). Letting $p \in \mathbb{R}_+^n$ be such that p_i is equal to the non-zero entry of the ith row of P, we have

$$\begin{aligned}
\sum_i p_i^2 ||M_{:i} - MB^*_{:i}||_2^2 &= \sum_i ||M_{:i}p_i - MPP^{-1}B^*_{:i}p_i||_2^2 \\
&= ||MP - MPP^{-1}B^*P||_F^2 \\
&\leq \sum_i p_i^2 ||M_{:i} - MB_{:i}||_2^2 = ||MP - MPP^{-1}BP||_F^2,
\end{aligned}$$

for any feasible solution $B' = P^{-1}BP$ of (8) for MP. This proves $B^* \in \mathcal{B}^*(M) \Rightarrow P^{-1}B^*P \in \mathcal{B}^*(MP)$. The other direction follows directly by using the permutation P^{-1} on the matrix MP. ■

It is interesting to observe that if a column of M belongs to the convex cone generated by the other columns, then the corresponding column of $\mathcal{P}(M)$ is equal to zero.

Lemma 17 *Let $\mathcal{J} = \{1,2,\ldots,n\}\backslash\{i\}$. Then $\mathcal{P}(M)_{:i} = 0$ if and only if $M_{:i} \in \text{cone}(M(:,\mathcal{J}))$.*

Proof We have that

$$\mathcal{P}(M)_{:i} = M_{:i} - \sum_{k\neq i} B^*_{ki} M_{:k} = 0, \quad B^*_{ki} \geq 0 \iff M_{:i} = \sum_{k\neq i} B^*_{ki} M_{:k}, \quad B^*_{ki} \geq 0.$$

■

The preprocessed matrix $\mathcal{P}(M)$ may contain all-zero columns, for which the function $\theta(.)$ is not defined (cf. Definition 1). We extend the definition to matrices with zero columns as follows: $\theta(X)$ is the matrix whose columns are the normalized non-zero columns of X, that is, letting Y be the matrix X where the non-zero columns have been removed, we define $\theta(X) = \theta(Y)$. Hence $\text{conv}(\theta(X))$ denotes the convex hull of the normalized non-zero columns of X.

Another straightforward property is that the preprocessing can only inflate the convex hull defined by the columns of $\theta(M)$.

Lemma 18 *Let $M \in \mathbb{R}^{m\times n}_+$. If the vertices of* $\text{conv}(\theta(M))$ *are non-repeated, then*

$$\text{conv}(\theta(M)) \quad \subseteq \quad \text{conv}(\theta(\mathcal{P}(M))) \quad \subseteq \quad \Delta^m \cap \text{col}(\theta(M)).$$

Proof By construction, since $\mathcal{P}(M) = MQ$, $\text{col}(\theta(\mathcal{P}(M))) \subseteq \text{col}(\theta(M))$ and $\text{conv}(\theta(\mathcal{P}(M))) \subseteq \Delta^m \cap \text{col}(\theta(M))$. Let i be the index corresponding to a vertex of $\theta(M)$ and $\mathcal{J} = \{1,2,\ldots,n\}\backslash\{i\}$. Because vertices of $\theta(M)$ are non-repeated, we have $M_{:i} \notin \text{conv}(\theta(M(:,\mathcal{J})))$, while

$$\mathcal{P}(M)_{:i} = M_{:i} - \sum_{k\neq i} b_{ki} M_{:k} \quad \iff \quad M_{:i} = \mathcal{P}(M)_{:i} + \sum_{k\neq i} b_{ki} M_{:k}.$$

Hence $M_{:i} \in \text{conv}(\theta([\mathcal{P}(M)_{:i} M(:,\mathcal{J})]))$, which implies that

$$\text{conv}(\theta(M)) \subseteq \text{conv}(\theta([\mathcal{P}(M)_{:i} M(:,\mathcal{J})])),$$

so that replacing $M_{:i}$ by $\mathcal{P}(M)_{:i}$ extends $\text{conv}(\theta(M))$. Since this holds for all vertices, the proof is complete. ■

Corollary 19 *Let $M \in \mathbb{R}^{m\times n}_+$. If no column of M is multiple of another column, then*

$$\text{rank}(\mathcal{P}(M)) = \text{rank}(M) \quad \textit{and} \quad \text{rank}_+(\mathcal{P}(M)) \geq \text{rank}_+(M).$$

Proof Without loss of generality, we can assume that M does not have a zero column. In fact, a preprocessed zero column remains zero while it cannot influence the preprocessing of the other columns (see Equation (7)). Then, by Lemma 18, we have

$$\text{conv}(\theta(M)) \quad \subseteq \quad \text{conv}(\theta(\mathcal{P}(M))) \quad \subseteq \quad \Delta^m \cap \text{col}(\theta(M)),$$

implying $\text{rank}_+(\mathcal{P}(M)) \geq \text{rank}_+(M)$ and $\text{rank}(\mathcal{P}(M)) = \text{rank}(M)$.

Another way to prove this result is to use Corollary 22 (see below) guaranteeing the existence of an inverse-positive matrix Q such that $\mathcal{P}(M) = MQ$ which implies $\text{rank}(\mathcal{P}(M)) = \text{rank}(M)$. Moreover, any exact NMF $(U,V) \in \mathbb{R}^{m\times r} \times \mathbb{R}^{r\times n}$ of $\mathcal{P}(M)$ gives $M = UVQ^{-1}$ hence $\text{rank}_+(M) \leq \text{rank}_+(\mathcal{P}(M))$. ■

We now prove that if no column of M is multiple of another column (that is, the columns of $\theta(M)$ are distinct) then $\rho(B^*) < 1$ for any $B^* \in \mathcal{B}^*(M)$ whence $Q = I - B^*$ is an inverse positive matrix.

Lemma 20 *Let A be a column stochastic matrix and $Q = I - B$ where $B \geq 0$ and $B_{ii} = 0$ for all i be such that $AQ \geq 0$. Then,*

$$\sum_k B_{ki} \leq 1, \quad \forall i,$$

so that Q is diagonally dominant. Moreover, if $A_{:i} \notin \text{conv}(A(:,\mathcal{J}))$ where $\mathcal{J} = \{1,2,\ldots,n\}\backslash\{i\}$, then

$$\sum_k B_{ki} < 1.$$

Proof By assumption, we have for all i

$$A_{:i} \geq AB_{:i} = \sum_k A_{:k}B_{ki},$$

which implies

$$1 = ||A_{:i}||_1 \geq ||AB_{:i}||_1 = ||\sum_k A_{:k}B_{ki}||_1 = ||B_{:i}||_1 = \sum_k B_{ki},$$

because A and B are nonnegative. Moreover, if $A_{:i} \notin \text{conv}(A(:,\mathcal{J}))$, then there exists at least one index j such that $A_{ji} > A_{j:}B_{:i}$ (Lemma 17) so that the above inequality is strict. ■

Theorem 21 *If no column of M is multiple of another column, then any optimal solution B^* of* (8) *satisfies $\rho(B^*) < 1$ whence $Q = I - B^*$ is inverse positive.*

Proof By Theorem 11, $\rho(B^*) < 1$ if and only if $Q = I - B^*$ is inverse positive if and only if Q is a nonsingular M-matrix. Let us then show that Q is a nonsingular M-matrix. First, we can assume without loss of generality that

- Matrix M does not contain a column equal to zero. In fact, if M does, say the first column is equal to zero, then we must have $B_{:1} = 0$ (since $M_{:1} \geq MB_{:1}$ and there is not other zero column in M). The matrix Q is then a nonsingular M-matrix if and only if $Q(2{:}n, 2{:}n)$ is.

- The columns of M sum to one. In fact, letting $P = D(M)$ be defined as in Equation (2), by Lemma 16, B^* is an optimal solution for M if and only if $P^{-1}B^*P$ is an optimal solution for MP. Since B^* and $P^{-1}B^*P$ share the same eigenvalues, $\rho(B^*) < 1 \iff \rho(P^{-1}B^*P) < 1$.

- Let $B \in \mathcal{B}^*(M)$, $Q = I - B^*$, and P be a permutation matrix such that

$$P^T QP = \begin{pmatrix} Q^{(1)} & Q^{(12)} & Q^{(13)} & \dots & Q^{(1k)} \\ 0 & Q^{(2)} & Q^{(23)} & \dots & Q^{(2k)} \\ 0 & 0 & Q^{(3)} & \dots & Q^{(3k)} \\ \vdots & \dots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & Q^{(k)} \end{pmatrix}$$
$$= I - \begin{pmatrix} B^{(1)} & B^{(12)} & B^{(13)} & \dots & B^{(1k)} \\ 0 & B^{(2)} & B^{(23)} & \dots & B^{(2k)} \\ 0 & 0 & B^{(3)} & \dots & B^{(3k)} \\ \vdots & \dots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & B^{(k)} \end{pmatrix},$$

  where $Q^{(i)}$ are irreducible for all i. Without loss of generality, by Lemma 16, we can then assume that Q has this form.

In the following we show that $Q^{(p)}$ is nonsingular for each $1 \leq p \leq k$ hence Q is. By Theorem 14, if $Q^{(p)}$ is irreducibly diagonally dominant, then $Q^{(p)}$ is nonsingular and the proof is complete. We already have that $Q^{(p)}$ is irreducible for $1 \leq p \leq k$. Let I_p denote the index set such that $Q^{(p)} = Q(I_p, I_p)$. We have $M(I_p,:)$ is column stochastic, and

$$\mathcal{P}(M)(I_p,:) = M(I_p,:) - \sum_{l=1}^{p-1} M(I_l,:)B^{(lp)} - M(I_p,:)B^{(p)} \geq 0,$$

implying that $M(I_p,:) \geq M(I_p,:)B^{(p)}$. Moreover the columns of $M(I_p,:)$ are distinct so that there is at least one which does not belong to the convex hull of the others. Hence, by Lemma 20, $Q^{(p)}$ is irreducibly diagonally dominant. ■

Corollary 22 *Let $M \in \mathbb{R}_+^{m\times n}$. If the vertices of* $\mathrm{conv}(\theta(M))$ *are non-repeated, then there exists an optimal solution $B^* \in \mathcal{B}^*(M)$ such that $\rho(B^*) < 1$, that is, such that $Q = I - B^*$ is an inverse-positive matrix.*

Proof Let us show that there exists an optimal solution such that Q is a nonsingular M-matrix. First, by Lemma 20, Q is diagonally dominant implying $\rho(B) \leq 1$ so that Q is an M-matrix (cf. Theorem 21). We can assume without loss of generality that the r first columns of M correspond to the vertices of $\mathrm{conv}(\theta(M))$. This implies that there exists an optimal solution $B^* \in \mathcal{B}^*(M)$ such that

$$Q = \begin{pmatrix} Q_1 & Q_{12} \\ 0 & I \end{pmatrix} = I - \begin{pmatrix} B_1^* & B_{12}^* \\ 0 & 0 \end{pmatrix}, \text{ where } Q_1, B_1^* \in \mathbb{R}^{r\times r} \text{ and } Q_{12}, B_{12}^* \in \mathbb{R}^{r\times(n-r)}.$$

In fact, by assumption, the last columns of M belong to the convex cone of the r first ones and can then be set to zero (which is optimal) using only the first r columns (cf. Lemma 17). Lemma 20 applies on matrix Q_1 and $M(:,1{:}r)$ since

$$MQ(:,1{:}r) = M(:,1{:}r) - M(:,1{:}r)B_1^* \geq 0,$$

while by assumption no column of $M(:,1{:}r)$ belong to the convex hull of the other columns, so that Q_1 is strictly diagonally dominant hence is a nonsingular M-matrix. ■

Finally, what really matters is that the vertices of $\text{conv}(\theta(M))$ are non-repeated. In that case, the preprocessing is unique and the preprocessed matrix has the same rank as the original one. The fact that Q could be singular is not too dramatic. In fact, given an NMF (U,V') of the preprocessed matrix $\mathcal{P}(M) = MQ \approx UV'$, we can obtain the optimal factor V for matrix M by solving the nonnegative least squares problem $V = \text{argmin}_{X\geq 0} ||M - UX||_F^2$ (instead of taking $V = V'Q^{-1}$) and obtain $M \approx UV$.

4.2 Recovery Under Separability

A nonnegative matrix M is called *separable* if it can be written as $M = UV$ where $U \in \mathbb{R}_+^{m\times r}$, $V \in \mathbb{R}_+^{r\times n}$, and for each $i = 1,\ldots,r$ there is some column $f(i)$ of V that has a single nonzero entry and this entry is in the ith row, that is, V contains a monomial submatrix. In other words, each column of U appears (up to a scaling factor) as a column of M. Arora et al. (2012) showed that the NMF problem corresponding to a separable nonnegative matrix can be solved in polynomial time (while NMF is NP-hard in general; see Introduction). In this section, we show that the preprocessing is able to solve this problem while generating a sparser solution than the one obtained with the algorithm of Arora et al. (2012). We refer the reader to Gillis and Vavasis (2012) and the references therein for more details about NMF algorithms for separable matrices.

It is worth noting that the separability assumption is equivalent to the pure-pixel assumption in hyperspectral imaging (for each constitutive material present in the image, there is at least one pixel containing only that material), see Craig (1994), or, in document classification, to the assumption that, for each topic, there is at least one document corresponding only to that topic (or, considering the matrix transpose, that there is at least one word corresponding only to that topic, see Arora et al., 2012). Geometrically, separabilty means that the vertices of $\text{conv}(\theta(M))$ are given by the columns of $\theta(U)$. We have the following straightforward lemma:

Lemma 23 *$M = UV$ is separable (that is, $U \geq 0$, $V \geq 0$ and V contains a monomial submatrix) if and only if* $\text{conv}(\theta(M)) = \text{conv}(\theta(U))$.

Proof $M = UV$ where $U \geq 0, V \geq 0$ and V contains a monomial submatrix if and only if the vertices of $\theta(U)$ and $\theta(M)$ coincide if and only if $\text{conv}(\theta(M)) = \text{conv}(\theta(U))$. ■

Theorem 24 *If M is separable and the r vertices of $\theta(M)$ are non-repeated, then $\mathcal{P}(M)$ has r non-zero columns, say $S_{:1}, S_{:2}, \ldots S_{:r}$, such that* $\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(S))$, *that is, there exists $R \geq 0$ such that $M = SR$.*

Proof This is a consequence of Lemmas 17, 18 and 23. ■

Theorem 24 shows that the preprocessing is able to identify the r columns of $M = UV$ corresponding to the vertices of $\theta(M)$. Moreover, it returns a sparser matrix S, namely $\mathcal{P}(U)$, whose cone contains the columns of M. Remark also that Theorem 24 does not require M to be full rank: the dimension of $\text{conv}(\theta(M))$ can be smaller than $r-1$.

Corollary 25 *For any rank-two nonnegative matrix M whose columns are not multiples of each other, $\mathcal{P}(M)$ has only two non-zero columns, say $S_{:1}$ and $S_{:2}$ such that* $\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(S))$, *that is, there exists $R \geq 0$ such that $M = SR$. In other words, the preprocessing technique is optimal as it is able to identify an optimal nonnegative basis for the NMF problem corresponding to the matrix M.*

Proof A rank-two nonnegative matrix is always separable. In fact, a two-dimensional pointed cone is always spanned by two extreme vectors. In particular, $\text{rank}(M) = 2 \iff \text{rank}_+(M) = 2$ (Thomas, 1974). ■

Example 1 *Here is an example with a rank-three separable matrix*

$$M = \begin{pmatrix} 5 & 5 & 5 & 5 & 9 & 1 & 4 & 1 & 7 & 7 \\ 10 & 6 & 5 & 3 & 7 & 8 & 4 & 1 & 5 & 8 \\ 8 & 9 & 9 & 4 & 7 & 8 & 3 & 9 & 6 & 7 \end{pmatrix}^T \begin{pmatrix} 1 & 0 & 0 & 2 & 3 & 6 & 4 & 4 \\ 0 & 1 & 0 & 5 & 7 & 7 & 7 & 4 \\ 0 & 0 & 1 & 9 & 4 & 4 & 8 & 6 \end{pmatrix}. \tag{10}$$

Its (rounded) preprocessed version is given by

$$\mathcal{P}(M) = \begin{pmatrix} 3.6 & 3.9 & 3.9 & 4.3 & 7.6 & 0 & 3.3 & 0.5 & 5.9 & 5.7 \\ 6.3 & 2.5 & 1.6 & 0 & 1.5 & 6.5 & 1.5 & 0 & 0.7 & 3.4 \\ 0.8 & 2.4 & 2.7 & 0.7 & 0.7 & 1.8 & 0 & 4.2 & 0.9 & 0.6 \end{pmatrix}^T \begin{pmatrix} I_3 & 0_{3\times 5} \end{pmatrix},$$

where I_3 is the 3-by-3 identity matrix and $0_{3\times 5}$ is the 3-by-5 all-zero matrix. Figure 1 shows the geometric interpretation of the preprocessing.

4.3 Uniqueness and Robustness Through Preprocessing

A potential drawback of the preprocessing is that it might increase the nonnegative rank of M. In this section, we show how to modulate the preprocessing to prevent this behavior.

Let us define

$$\mathcal{P}^\alpha(M) = M(I - \alpha B^*) = M - \alpha MB^*,$$

where $0 \leq \alpha \leq 1$ and $B^* \in \mathcal{B}^*(M)$. Notice that $\mathcal{P}^\alpha(M)$ is well-defined because for any $B_1^*, B_2^* \in \mathcal{B}^*(M)$ we have $MB_1^* = MB_2^*$; see Theorem 15.

Lemma 26 *Let M be a nonnegative matrix such that the vertices of* $\text{conv}(\theta(M))$ *are non-repeated. Then, for any $0 \leq \alpha \leq \beta \leq 1$,*

$$\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(\mathcal{P}^\alpha(M))) \subseteq \text{conv}(\theta(\mathcal{P}^\beta(M))) \subseteq \text{col}(\theta(M)) \cap \Delta^m.$$

Therefore,

$$\text{rank}_+(M) \leq \text{rank}_+(\mathcal{P}^\alpha(M)) \leq \text{rank}_+(\mathcal{P}^\beta(M)).$$

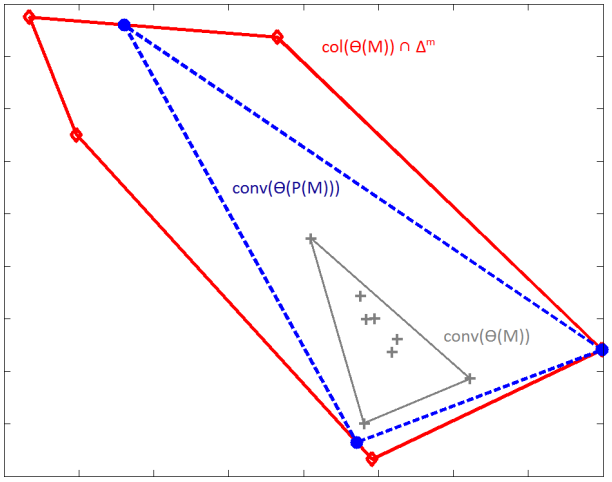


Figure 1: Geometric interpretation of the preprocessing of matrix M from Equation (10).

Proof The proof can be obtained by following exactly the same steps as the proof of Lemma 18. ■

Lemma 27 *Let M be a nonnegative matrix such that the vertices of* $\operatorname{conv}(\theta(M))$ *are non-repeated, then the supremum*

$$\bar{\alpha} = \sup_{0 \le \alpha \le 1} \alpha \quad \textit{such that} \quad \operatorname{rank}_+(\mathcal{P}^{\alpha}(M)) = \operatorname{rank}_+(M) \tag{11}$$

is attained.

Proof We can assume without loss of generality that M does not have all-zero columns. In fact, if $M_{:i} = 0$ for some i then $\mathcal{P}^{\alpha}(M)_{:i} = 0$ for all $\alpha \in [0,1]$ so that the nonnegative rank of $\mathcal{P}^{\alpha}(M)$ is not affected by the zero columns of M.

Then, if $\bar{\alpha} = 1$, the proof is complete. Otherwise, one can easily check that, for any $0 \le \bar{\alpha} < 1$, we have $\mathcal{P}^{\bar{\alpha}}(M)_{:i} \neq 0 \ \forall i$ (using a similar argument as in Lemma 17).

Finally, the result follows from the upper-semicontinuity of the nonnegative rank (Bocci et al., 2011, Theorem 3.1): 'If P is a nonnegative matrix, without zero columns and with $\operatorname{rank}_+(P) = k$, then there exists a ball $\mathcal{B}(P,\varepsilon)$ centered at P and of radius $\varepsilon > 0$ such that $\operatorname{rank}_+(N) \ge k$ for all $N \in \mathcal{B}(P,\varepsilon)$'. Therefore, if the supremum of (11) was not attained, the matrix $\mathcal{P}_{\bar{\alpha}}(M)$ would satisfy $\operatorname{rank}_+(\mathcal{P}_{\bar{\alpha}}(M)) > \operatorname{rank}_+(M)$ while for any $\alpha < \bar{\alpha}$ we would have $\operatorname{rank}_+(\mathcal{P}_{\alpha}(M)) = \operatorname{rank}_+(M)$, a contradiction. ■

Hence working with matrix $\mathcal{P}^{\bar{\alpha}}(M)$ instead of M will reduce the number of solutions of the NMF problem while preserving the nonnegative rank:

Theorem 28 *Let M be a nonnegative matrix for which the vertices of* $\operatorname{conv}(\theta(M))$ *are non-repeated, let also $\bar{\alpha}$ be defined as in Equation* (11)*. Then any exact NMF (U,V) of $\mathcal{P}^{\bar{\alpha}}(M)$ corresponds to an*

exact NMF (U, VQ^{-1}) *of* M*, while the converse is not true. In fact,*

$$\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(\mathcal{P}^{\bar{\alpha}}(M))).$$

Therefore, the NMF problem for $\mathcal{P}^{\bar{\alpha}}(M)$ *is more well posed.*

Proof This follows directly from the definition of $\bar{\alpha}$, and Lemmas 26 and 27. ■

We now illustrate Corollary 28 on a simple example, which will lead to three other important results.

Example 2 (Nested Squares) *Let*

$$M = \begin{pmatrix} 5 & 3 & 3 & 5 \\ 3 & 5 & 5 & 3 \\ 5 & 5 & 3 & 3 \\ 3 & 3 & 5 & 5 \end{pmatrix}.$$

The problem NPP(M) restricted to the column space of M is made up of two nested squares, $\text{conv}(\theta(M))$ *and* $\text{col}(\theta(M)) \cap \Delta^m$*, centered at* $(0,0)$ *with side length 2 and 8 respectively, see Figure 2. The polygon corresponding to* $\mathcal{P}^{\alpha}(M)$ *is a square centered at* $(0,0)$ *with side length depending on* α*, between 2 (for* $\alpha = 0$*) and 8 (for* $\alpha = 1$*). We can show that the largest such square still included in a triangle corresponds to*

$$\mathcal{P}^{\bar{\alpha}}(M) = \mathcal{P}^{\bar{\alpha}} \begin{pmatrix} 5 & 3 & 3 & 5 \\ 3 & 5 & 5 & 3 \\ 5 & 5 & 3 & 3 \\ 3 & 3 & 5 & 5 \end{pmatrix} = \frac{1}{a} \begin{pmatrix} 1+a & 1-a & 1-a & 1+a \\ 1-a & 1+a & 1+a & 1-a \\ 1+a & 1+a & 1-a & 1-a \\ 1-a & 1-a & 1+a & 1+a \end{pmatrix}, \tag{12}$$

where $a = \sqrt{2} - 1$ *and* $\bar{\alpha} = \frac{4a-1}{3a}$ *(this follows from the proof of Theorem 29; see below). Hence, the polygon* $\text{conv}(\theta(\mathcal{P}^{\bar{\alpha}}(M)))$ *is a square centered at* $(0,0)$ *with side length* $8a$ *in between* $\text{conv}(\theta(M))$ *and* $\text{col}(\theta(M)) \cap \Delta^m$*, see Figure 2. Unfortunately, the exact NMF of* $\mathcal{P}^{\bar{\alpha}}(M)$ *is non-unique. In fact, we will see later that it has 8 solutions (the ones drawn on Figure 2 and their rotations).*

Example 2 illustrates the following three important facts:
Fact 1. Defining a well-posed NMF problem is not always possible. In other words, *there does not exist any 'reasonable' NMF formulation having always a unique solution (up to permutation and scaling).* In fact, Example 2 shows that, because of the symmetry of the problem, any solution of NPP(M) can be rotated by 90, 180 or 270 degrees to obtain a different solution with exactly the same characteristics (the rotated solutions cannot be distinguished in any reasonable way). For example, there are 4 solutions which are the sparsest, each containing one vertex of $\text{col}(\theta(M)) \cap \Delta^m$, see $\text{conv}(\theta(U_2))$ on Figure 2, including

$$U_2 = \begin{pmatrix} 1 & a & 0 \\ 0 & 1-a & 1 \\ a & 1 & 0 \\ 1-a & 0 & 1 \end{pmatrix}, \text{ and } U_2^{(180)} = \begin{pmatrix} 0 & 1-a & 1 \\ 1 & a & 0 \\ 1-a & 1 & 1 \\ a & 0 & 0 \end{pmatrix},$$

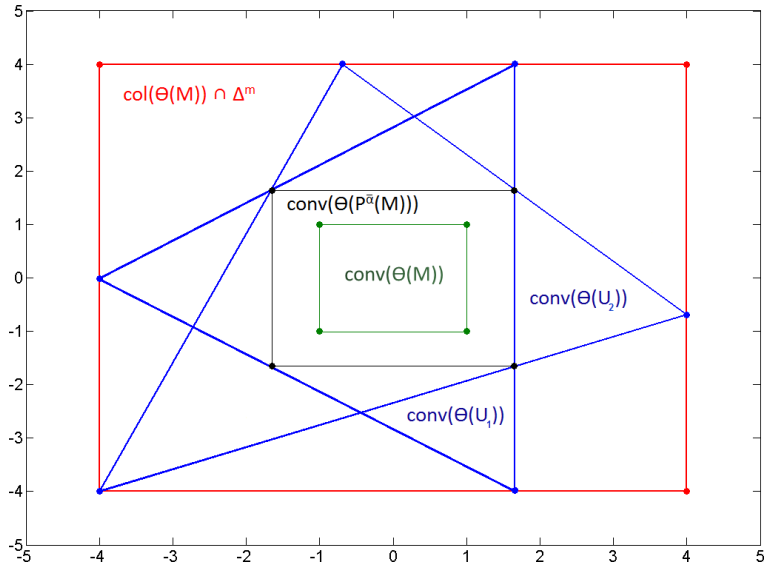


Figure 2: Geometric interpretation of the preprocessing of matrix M from Equation (12).

where $U_2^{(180)}$ is the rotation of 180 degrees of U_2.

Fact 2. The preprocessing makes NMF more robust. For any m-by-n matrix E such that $\text{col}(E) \subseteq \text{col}(M)$, $M+E \geq 0$, and

$$\text{conv}(\theta(M)) \subseteq \text{conv}(\theta(M+E)) \subseteq \text{conv}(\mathcal{P}^{\bar{\alpha}}(M)),$$

the exact NMF (U,V) of $\mathcal{P}^{\bar{\alpha}}(M)$ will still provide an optimal factor U for the perturbed matrix $M+E$. In particular, if the matrix M is positive, then one can show that[4] $\text{conv}(\theta(M))$ is strictly contained in $\text{conv}(\mathcal{P}^{\bar{\alpha}}(M))$ (given that $\bar{\alpha} > 0$) so that any sufficiently small perturbation E with $\text{col}(E) \subseteq \text{col}(M)$ will satisfy the conditions above.

In Example 2, the vertices of M can be perturbed and, as long as they remain inside the square defined by $\text{conv}(\mathcal{P}^{\bar{\alpha}}(M))$ (see Figure 2), the exact NMF of $\text{conv}(\mathcal{P}^{\bar{\alpha}}(M))$ will provide an exact NMF for the perturbed matrix M. (More precisely, any matrix E such that $\text{col}(E) \subseteq \text{col}(M)$ and $\max_{i,j}|E_{ij}| \leq \sqrt{2}-1$ will satisfy $\text{conv}(\theta(M+E)) \subseteq \text{conv}(\mathcal{P}^{\bar{\alpha}}(M))$.)

Fact 3. The preprocessing makes the NMF problem more well-posed. In Example 2, even though the NMF of $\mathcal{P}^{\bar{\alpha}}(M)$ is non-unique, the set of solutions has been drastically reduced: from a two-dimensional space to a zero-dimensional one containing eight points: $\text{conv}(\theta(U_1))$, $\text{conv}(\theta(U_2))$ and the corresponding rotated solutions, see Figure 2.

Theorem 29 *Let $M \in \mathbb{R}_+^{m\times n}$ be such that $\text{rank}(M) = \text{rank}_+(M) = 3$ and let $\bar{\alpha}$ be defined as in Equation (11). Assume also that $\text{conv}(\theta(\mathcal{P}(M)))$ has at least four vertices. Then the number of solutions of NPP($\mathcal{P}^{\bar{\alpha}}(M)$) with three vertices is smaller than $m+n$.*

Proof Let P and Q denote the outer and inner polygons of NPP($\mathcal{P}^{\bar{\alpha}}(M)$), respectively. Let us also parametrize the boundary of the outer polygon P with the parameter $t \in [0,1]$ and the function

$$x : \mathbb{R}_+ \to \mathbb{R}^2 : t \mapsto x(t) \in P,$$

4. Using the same ideas as in Lemma 18 and the fact that any preprocessed column must contain at least one zero entry.

where x is a continuous function with $x(0) = x(1)$ and $\{x(t) \mid t \in [0,1]\}$ is equal to the boundary of P. We also define the function x for values of t larger than one using $x(t) = x(t - \lfloor t \rfloor)$ where $\lfloor t \rfloor$ is the largest integer not exceeding t. Using the construction of Aggarwal et al. (1989), we define the function $f_k : \mathbb{R}_+ \to \mathbb{R}_+ : t \mapsto f_k(t)$ as follows. Let $t_1 \in [0,1)$ and $x(t_1)$ be the corresponding point on the boundary of P. From $x(t_1)$, we can trace the tangent to Q (that is, Q is on one side of the tangent, and the tangent touches Q), say in the clock-wise direction, intersect it with P and hence obtain a new point $x(t_2)$ on the boundary P (see Figure 3 for an illustration on the nested squares problem). We assume without loss of generality that $t_2 \geq t_1$ (if t_2 happens to be larger than one, we

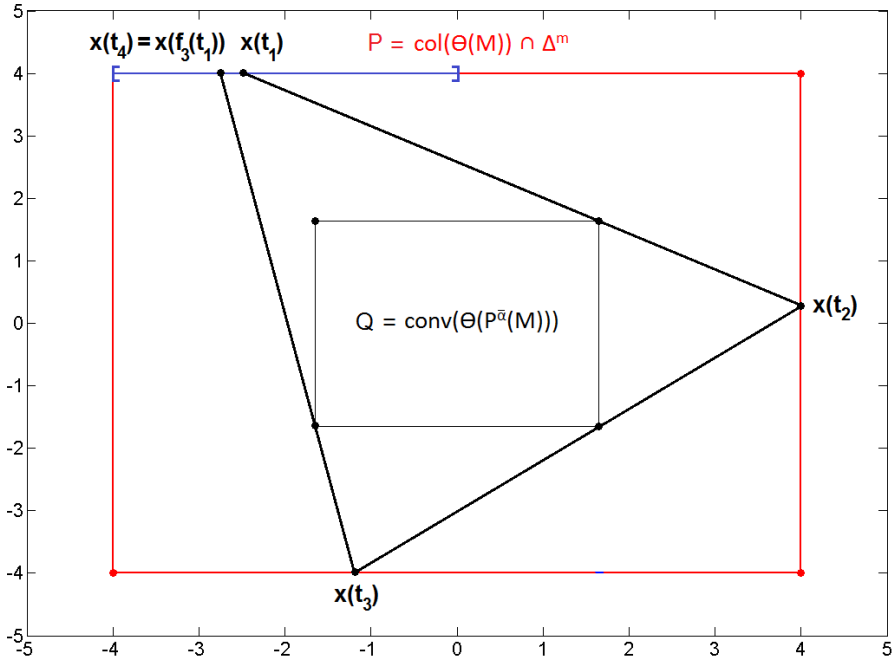


Figure 3: Mapping of the point $x(t_1)$ to $x(t_4)$ using the construction of Aggarwal et al. (1989).

do not round it down with the equivalent value $t_2 - \lfloor t_2 \rfloor$). Starting from $x(t_2)$, we can use the same procedure to obtain $x(t_3)$ and we apply this procedure k times to obtain the point $x(t_{k+1})$, where $t_{k+1} \geq \cdots \geq t_2 \geq t_1$. Finally, we define $f_k(t_1) = t_{k+1}$.

Aggarwal et al. (1989) showed that $x(t_1)$ can be taken as a vertex of a feasible solution of NPP($\mathcal{P}^{\bar{\alpha}}(M)$) with k vertices if and only if $f_k(t_1) = t_{k+1} \geq t_1 + 1$, that is, we were able to turn around Q inside P in $k+1$ steps (in fact, $x(t_1)$, $x(t_2)$, ..., and $x(t_k)$ are the vertices of a feasible solution).

Aggarwal et al. (1989) also showed that the function f_k is continuous, non-decreasing, and depends continuously on the vertices of Q (see also Appendix A). Figure 4 displays the function f_4 for the nested squares (Example 2).

If $\mathrm{col}(\theta(M)) \cap \Delta^m$ has three vertices, then $\bar{\alpha} = 1$. In fact, we have that

$$\theta(\mathcal{P}^{\alpha}(M)) \subseteq \mathrm{col}(\theta(M)) \cap \Delta^m \quad \text{for any } 0 \leq \alpha \leq 1,$$

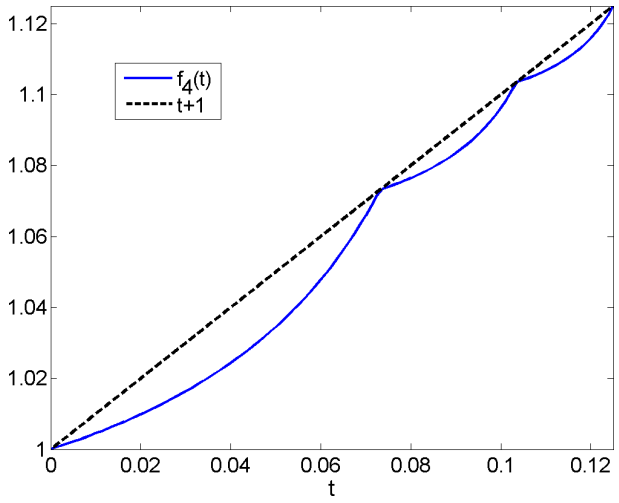


Figure 4: Function $f_4(t)$ for Example 2 using the construction of Aggarwal et al. (1989) (see also Figure 4 and Appendix A). We only plot the function f_4 in the interval $[0, \frac{1}{8}]$ because, by symmetry, $f_4(x+\frac{1}{8}) = f_4(x)+\frac{1}{8}$.

implying $\text{rank}_+(\mathcal{P}^\alpha(M)) = 3$ for all $0 \leq \alpha \leq 1$. Moreover, because $\theta(\mathcal{P}^\alpha(M))$ has at least four vertices, $\text{col}(\theta(M)) \cap \Delta^m$ is the unique solution of the corresponding NPP problem: the outer polygon is a triangle while the inner polygon has at least four vertices which are located on the edges of the outer triangle (since $\bar{\alpha} = 1$ and each column of $\mathcal{P}(M)$ contains at least one zero entry).

Let us then assume that $\text{col}(\theta(M)) \cap \Delta^m$ has at least four vertices. We show that this implies $\bar{\alpha} < 1$. Assume $\bar{\alpha} = 1$. The polygons $P = \text{col}(\theta(M)) \cap \Delta^m$ and $Q = \theta(\mathcal{P}(M))$ have at least 4 vertices. Moreover, the vertices of Q are located on the boundary of P (because $\bar{\alpha} = 1$) on at least two different sides of P (three vertices cannot be on the same side). It can be shown by inspection that the optimal solution of this NPP instance must have at least four vertices, hence $\text{rank}_+(\mathcal{P}(M)) > 3$, a contradiction.

Next, we show that $f_4(t) \leq t+1$. Assume there exists t such that $f_4(t) > t+1$. By continuity of f_4 with respect to the vertices of $Q = \text{conv}(\theta(\mathcal{P}^{\bar{\alpha}}(M)))$, there exists $\varepsilon > 0$ sufficiently small such that $\bar{\alpha}+\varepsilon < 1$ and such that the function f_4' for the NPP instance with inner polygon $Q' = \text{conv}(\theta(\mathcal{P}^{\bar{\alpha}+\varepsilon}(M)))$ and the same outer polygon P satisfies $f_4(t) > t+1$ hence $\text{rank}_+(\mathcal{P}^{\bar{\alpha}+\varepsilon}(M)) \leq 3$, a contradiction.

In Appendix A, we prove that f_k is made up of pieces which are either constant or strictly convex, with at most $m+n$ break points corresponding to different solutions to the NPP. Therefore, because f_4 is continuous and smaller than $t+1$, it can intersect the line $t+1$ only at the break points. Since there are at most $m+n$ such points corresponding to different NPP solutions, the number of solutions of NPP($\mathcal{P}^{\bar{\alpha}}(M)$) with three vertices is smaller than $m+n$. (Notice that the bound is tight for the nested squares example with 8 solutions.) ■

Remark 30 *If* $\mathrm{conv}(\theta(\mathcal{P}(M)))$ *has three vertices, they define a feasible solution for the corresponding NPP problem (that is,* $\mathcal{P}(M)$ *is separable, see Theorem 23). However, the number of solutions might be not be finite in that case. Here is an example*

$$M = \begin{pmatrix} 0 & 0.5 & 0.25 & 0 \\ 1 & 0.5 & 0.75 & 1 \\ 1 & 0 & 0.1 & 0.5 \\ 0 & 1 & 0.9 & 0.5 \end{pmatrix} \quad \textit{and} \quad \mathcal{P}(M) = \begin{pmatrix} 0 & 0.5 & 0 & 0 \\ 1 & 0.5 & 0.3 & 0.5 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0.3 & 0.5 \end{pmatrix},$$

whose corresponding NPP problems are represented on Figure 5: the NPP of $\mathcal{P}(M)$ *does not have a finite number of solutions.*

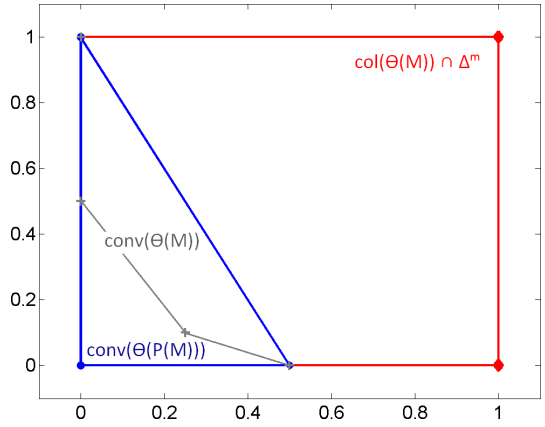


Figure 5: Counter-example for Theorem 29 when $\mathcal{P}(M)$ has three vertices.

The fact that the NPP of the matrix $\mathcal{P}^{\bar{\alpha}}(M)$ can have several different solutions is untypical and, we believe, could be due to the symmetry of the problem (as in Example 2). We conjecture that, in general, the solution to NPP($\mathcal{P}^{\bar{\alpha}}(M)$) is unique. In particular, we observed on randomly generated matrices that it was, see Example 1. In fact, as the function $f_k(.)$ defined in Theorem 29 depends continuously on the inner and outer polytopes Q and P, if these polytopes are generated randomly, there is no reason for the values of the function $f_k(.)$ at the break points to be located on the same line as on Figure 4.

We also conjecture that Theorem 29 holds true for any rank:

Conjecture 31 *Let* M *be such that* $\mathrm{rank}(M) = \mathrm{rank}_+(M) = k$ *and* $\mathrm{conv}(\theta(\mathcal{P}(M)))$ *has at least* $(k+1)$ *vertices, and* $\bar{\alpha}$ *be defined as in Equation* (11)*, then the number of solutions of NPP(*$\mathcal{P}^{\bar{\alpha}}(M)$*) is finite.*

Unfortunately, the geometric construction of Aggarwal et al. (1989) cannot be generalized to three dimensions (or higher). To prove the conjecture, we would need to show that

- Any solution of NPP($\mathcal{P}^{\bar{\alpha}}(M)$) is isolated. Intuitively, the preprocessing $\mathcal{P}^{\bar{\alpha}}(M)$ of M grows the inner polytope Q as long as the corresponding NPP instance has a solution with $\mathrm{rank}_+(M)$ vertices. If a solution was not isolated, it could be moved around while remaining feasible, which indicates that we could grow the inner polytope Q hence increase $\bar{\alpha}$.

- The number of isolated solutions is finite. We conjecture that the solutions can be characterized in terms of the faces of P and Q, which are finite (depending on m and n).

Remark 32 *Of course computing $\bar{\alpha}$ is non-trivial. However, for matrices of small rank, this could be done effectively. In fact, checking whether the nonnegative rank of an m-by-n is equal to* $\operatorname{rank}(M)$ *can be done in polynomial time in m and n provided that the rank is fixed (Arora et al., 2012). In particular, the algorithm of Aggarwal et al. (1989) does it in $O((m+n)\log(\min(m,n)))$ operations for rank-three matrices (Gillis and Glineur, 2012a). Hence, one could for example use a bisection method to find a good lower bound $\beta \lesssim \bar{\alpha}$ and use the corresponding matrix NPP($\mathcal{P}^{\beta}(M)$) to have a more well-posed NMF problem whose solutions will be solutions of the original one.*

5. Preprocessing in Practice

In this section, we address three important practical considerations of the preprocessing.

5.1 Computational Complexity of Solving (8)

It is rather straightforward to check that problem (8) can be decoupled into n independent CLLS's, each corresponding to a different column of M; for example, for the ith column of M, we have

$$\min_{b \in \mathbb{R}^n_+} \quad ||M_{:i} - Mb||_2^2 \quad \text{such that} \quad M_{:i} \geq Mb,\ b_i = 0. \tag{13}$$

We then have n CLLS's with n variables (actually $n-1$ since variable $b_i = 0$ can be removed) and $m+n$ constraints. Using interior point methods, the computational complexity for solving (13) is of the order of $O(n^{3.5})$; hence the total computational cost is of the order $O(n^{4.5})$.

Figure 6 shows the computational time needed for solving (8) with respect to m for n fixed and vice versa, for randomly generated matrices (using the *rand(.)* function of MATLAB®) on a laptop 3GHz Intel® CORE i7-2630QM CPU @2GHz 8Go RAM running MATLAB® R2011b using the function *lsqlin(.)* of MATLAB®. The computational time is linear in m while being of the order of n^3 in n, smaller than the expected $O(n^{4.5})$. Therefore, in practice, the dimension m can be rather large while, on a standard machine, n cannot be much larger than 1000. Using parallel architecture would allow to solve larger scale problems (see also Section 7).

5.2 Normalization of the Columns of $\mathcal{P}(M)$

Since the aim eventually is to provide a good approximate NMF to the original data matrix M, we observed that normalizing the columns of the preprocessed matrix $\mathcal{P}(M)$ to match the norm of the corresponding columns of M gives better results. That is, we replace $\mathcal{P}(M)$ with $D\mathcal{P}(M)$ where

$$D_{ii} = \frac{||M_{:i}||_2}{||\mathcal{P}(M)_{:i}||_2} \text{ for all } i, \quad \text{and} \quad D_{ij} = 0 \text{ for all } i \neq j.$$

This scaling does not change the nice properties of the preprocessing since D is a monomial matrix, hence QD still is an inverse-positive matrix. This scaling degree of freedom is related to the fact that we fixed the diagonal entries of Q to one, see Section 3.2.

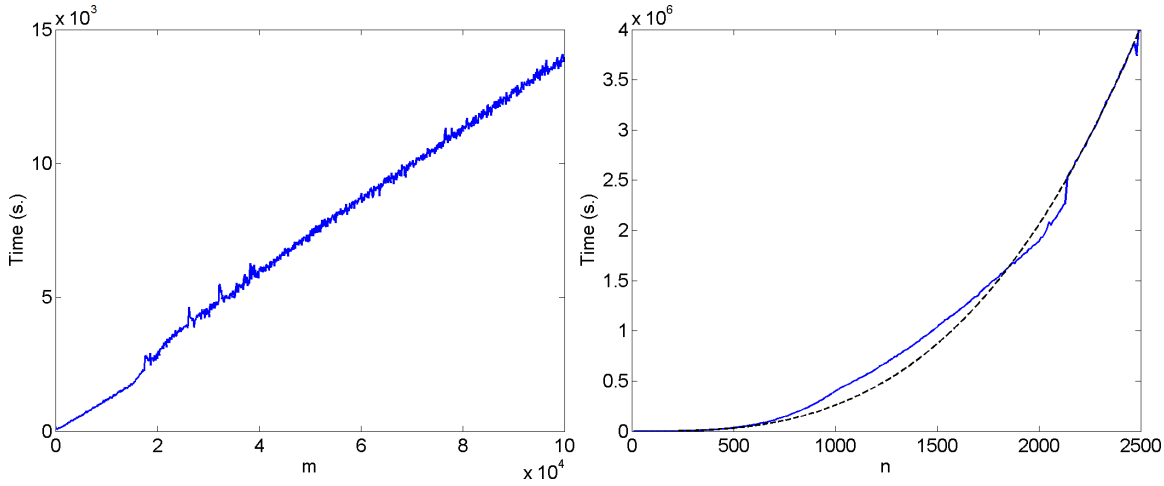


Figure 6: Computational time for solving (8). On the left, m-by-100 randomly generated matrices; on the right, 1000-by-n randomly generated matrices (plain) and the polynomial $2.6^*10^{-4}n^3$ (dashed).

The reason for this choice is that NMF algorithms are sensitive to the norm of the columns of M. In fact, when using the Frobenius norm, we have that the following two problems are equivalent

$$\min_{U\geq 0, V\geq 0} ||M-UV||_F^2 \quad \equiv \quad \min_{X\geq 0, Y\geq 0} \sum_{i=1}^{n} ||M_{:i}||_2^2 \left\| \frac{M_{:i}}{||M_{:i}||_2} - XY_{:i} \right\|_2^2.$$

Therefore, to give each column of $\mathcal{P}(M)$ the same importance in the objective function as in the original NMF problem, it makes sense to use the scaling above. This is particularly critical if there are outliers in the data set: the outliers do not look similar to the other columns of M hence their preprocessing will not reduce much their ℓ_2-norm (because they are further away from the convex cone generated by the other columns of M). Therefore, their relative importance in the objective function will increase in the NMF problem corresponding to $\mathcal{P}(M)$, which is not desirable.

5.3 Dealing with Noisy Input Matrices and/or Obtaining Sparser Preprocessing

Our technique will typically be useless when the input matrix is noisy and sparse. For example, we have

$$M = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}, \mathcal{P}(M) = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{while} \quad M_\delta = \begin{pmatrix} 0 & \delta \\ 1 & 0 \\ 1 & 1 \end{pmatrix} = \mathcal{P}(M_\delta),$$

for any $\delta > 0$. This shows that the preprocessing is very sensitive to small positive entries of M. In order to deal with such noisy and sparse matrices, we propose to relax the nonnegativity constraint $MQ \geq 0$ in (8), and solve instead

$$\min_{B\in\mathbb{R}_+^{n\times n}} \quad \sum_{i=1}^{n} \left\| M_{:i} - \sum_{k\neq i} M_{:k}B_{ki} \right\|_2^2 \quad \text{such that} \quad M_{:i} + \varepsilon ||M_{:i}||_\infty e \geq \sum_{k\neq i} M_{:k}B_{ki}, \ \forall i, \qquad (14)$$

where $0 < \varepsilon \ll 1$ and e is the vector of all ones of appropriate dimension. We will denote the corresponding preprocessing $\mathcal{P}_\varepsilon(M) = M(I - B_\varepsilon^*)$ where B_ε^* is an optimal solution of (14). For the example above with $\delta = \varepsilon = 10^{-2}$, we obtain

$$\mathcal{P}_\varepsilon(M_\delta) = \begin{pmatrix} -10^{-2} & 10^{-2} \\ 1 & -10^{-2} \\ 10^{-4} & 0.99 \end{pmatrix}.$$

In practice, this technique also allows to obtain preprocessed matrices with more entries equal or smaller than zero. When choosing the parameter ε, it is very important to check whether $\rho(B_\varepsilon^*) < 1$ so that the rank of $\mathcal{P}_\varepsilon(M)$ is equal to the rank of M and no information is lost (we can recover the original matrix $M = \mathcal{P}_\varepsilon(M)(I - B_\varepsilon^*)^{-1}$ given $\mathcal{P}_\varepsilon(M)$ and B_ε^*).

6. Application to Image Processing

In this section, we apply the preprocessing technique to several image data sets. By construction, the preprocessing procedure will remove from each image a linear combination of the other images. As we will see, this will highlight certain localized parts of these images, essentially because the preprocessed matrices are sparser than the original ones. We will then show that combining the preprocessing with standard NMF algorithms naturally leads to better part-based decompositions, because sparser matrices lead to sparser NMF solutions, see Section 2.

A direct comparison between NMF applied on the original matrix and NMF applied on the preprocessed matrix is not very informative in itself: while the former will feature a lower approximation error, the latter will provide a sparser part-based representation. This does not really tell us whether the improvements in the part-based representation and sparsity are worth the increase in approximation error. For that reason, we choose to compare them with a standard sparse NMF technique, described below, in order to better assess whether the increase in sparsity achieved is worth the loss in reconstruction accuracy. Hence, we compare the following three different approaches:

- **Nonnegative matrix factorization (NMF)**. It solves the original NMF problem from Equation (1) using the accelerated HALS algorithm (A-HALS) of Gillis and Glineur (2012b) (with parameters $\alpha = 0.5$ and $\varepsilon = 0.1$ as suggested by the authors), which is a block coordinate descent method.

- **Preprocessed NMF for different values of** ε. It first computes the preprocessed matrix $\mathcal{P}_\varepsilon(M)$ (cf. Section 5.3), then solves the NMF problem for the rescaled preprocessed matrix $\mathcal{P}_\varepsilon(M)D \approx UV'$ (cf. Section 5.2) using A-HALS and finally returns (U,V) where $V = \operatorname{argmin}_{X \geq 0} ||M - UX||_F^2$. This approach will be denoted Pre-NMF(ε). (We will also indicate in brackets the error obtained when using $V = V'Q^{-1}$, which will be, by construction, always higher.) Notice that the preprocessed matrix may contain negative entries (when $\varepsilon > 0$) which is handled by A-HALS. We do not set these entries to zero for two important reasons: (i) we want to preserve the column space of M, (ii) the negative entries of M lead to sparser NMF solutions: Geometrically, a negative entry in M means that a vertex of $\text{conv}(M)$ (the inner polytope) is not contained in Δ^m (the outer polytope) making NPP(M) infeasible (as a negative entry cannot be obtained with nonnegative ones). However, the approximate solution T of NPP(M) will have to be close to the boundary of Δ^m to approximate well that vertex. In particular, Gillis and Glineur (2008) showed that if an entry of M, say at position (i, j), is

smaller than $-||\max(0,M)||_F$ then $(UV)_{ij} = 0$ for any optimal solution of NMF (1). Therefore, when indicating the sparsity of the preprocessed matrix, negative entries will be counted as zeros as they lead to even sparser NMF decompositions.

- **Sparse NMF**. The most standard technique to obtain sparse solutions for NMF problems is to use a sparsity-inducing penalty term in the objective function. In particular, it is well-known that adding an l_1-norm penalty term induces sparser solutions (Kim and Park, 2007), and we therefore solve the following problem:

$$\min_{U,V\geq 0} ||M-UV||_F^2 + \sum_{i=1}^{r} \mu_i ||U_{:i}||_1, \quad ||U_{:i}||_\infty = 1 \; \forall i,$$

where $||x||_1 = \sum_i |x_i|$, $||x||_\infty = \max_i |x_i|$ and μ_i are positive parameters controlling the sparsity of the columns of U. In order to solve sNMF, we also use A-HALS which can easily be adapted to handle this situation. The ℓ_∞-norm constraints is not restrictive because of the degree of freedom in the scaling of the columns of U and the corresponding rows of V, while it prevents matrix U to converge to zero. The theoretical motivation is that the l_1-norm is the convex envelope of the l_0-norm (that is, the largest convex function smaller than the l_0-norm) in the ℓ_∞-ball, see Recht et al. (2010) and the references therein.

In order to compare sparse NMF with Pre-NMF(ε), the parameters μ_i $1 \leq i \leq r$ are tuned in order to match the sparsity obtained by Pre-NMF(ε). The corresponding approach will be denoted sNMF(ε).

For each approach, we will keep the best solution obtained among the same ten random initializations (using the *rand(.)* function of MATLAB®) and each run was allowed 1000 (outer) iterations of the A-HALS algorithm. We will use the relative error

$$100\,\frac{||M-UV||_F}{||M||_F}$$

to asses the quality of an approximation. We will also display the error obtain by the truncated singular value decomposition (SVD) for the same factorization rank to serve as a comparison. For the sparsity, we use the percentage of non-zero entries[5]

$$s(U) = 100\,\frac{\#\text{zeros}(U)}{mr} \;\in\; [0,100], \text{ for } U \in \mathbb{R}^{m\times r}.$$

Because the solution computed with Pre-NMF does not directly aim at minimizing the error $||M-UV||_F^2$, it is not completely fair to use this measure for comparison. In fact, it would be better to compare the quality of the sparsity patterns obtained by the different techniques. For this reason, we use the same post-processing procedure as described by Gillis and Glineur (2010) which benefits all algorithms: once a solution is computed by one of the algorithms, the zero entries of U are fixed and we minimize $\min_{U\geq 0, V\geq 0} ||M-UV||_F^2$ on the remaining (nonzero) entries (again, A-HALS can easily be adapted to handle this situation and we perform 100 additional steps on each solution), and report the new relative approximation error as "Improved", while the original relative error before this postprocessing will be reported as "Plain". The code is available at `https://sites.google.com/site/nicolasgillis/code`.

5. The negative entries of the preprocessed matrix $\mathcal{P}_\varepsilon(M)$ for $\varepsilon > 0$ will be counted as zeros.

6.1 CBCL Data Set

The CBCL face data set[6] is made of 2429 gray-level images of faces with 19×19 pixels (black is one and white is zero). We look for an approximation of rank $r = 49$ as in Lee and Seung (1999). Because of the large number of images in the data set, the preprocessing is rather slow. In fact, we have seen in Section 5.1 that it is in $O(n^{4.5})$ where n is the number of images in the data set (it would take about one week on a laptop). Therefore, we only keep every third image for a total of 810 images, which takes less than three hours for the preprocessing; about 10-15 seconds per image.[7]

Table 1 reports the sparsity and the value of $\rho(B^*_\varepsilon)$ for the preprocessed matrices with different values of the parameter ε. As explained in Section 5.3, the sparsity of $\mathcal{P}_\varepsilon(M)$ increases with ε, and ε was chosen to make sure that $\rho(B^*_\varepsilon) < 1$ implying $\text{rank}(\mathcal{P}_\varepsilon(M)) = \text{rank}(M)$.

	M	$\mathcal{P}(M)$	$\mathcal{P}_{0.05}(M)$	$\mathcal{P}_{0.1}(M)$
s(.)	0	0.001	20.92	38.03
$\rho(B^*_\varepsilon)$	0	0.71	0.83	0.90

Table 1: CBCL data set: sparsity of the preprocessed matrices $\mathcal{P}_\varepsilon(M) = MQ$ and corresponding spectral radius of $B^*_\varepsilon = I - Q$.

Figure 7 displays a sample of images of the CBCL data set along with the corresponding preprocessed images for different values of ε.

Figure 7: From top to bottom: CBCL sample images, corresponding preprocessed images for $\varepsilon = 0$, $\varepsilon = 0.05$, and $\varepsilon = 0.1$.

We observe that the preprocessing is able to highlight some parts of the images: the eyes (faces 5 and 9), the eyebrows (faces 3, 4, 8, 10, 11, 13 and 16), the mustache (faces 14 and 15), the glasses (faces 6, 7 and 12), the nose (faces 1 to 4) or the mouth (faces 1 to 5). Recall that the preprocessing removes from each image of the original data set a linear combination of other images. Therefore,

6. Available at `http://cbcl.mit.edu/software-datasets/FaceData2.html`.
7. The MATLAB® function *lsqlin* for solving CLLS problems is much slower than *quadprog* with interior point (which is much faster than *quadprog* with active set).

the parts of the images which are significantly different from the other images are better preserved, hence highlighted.

We now compare the three approaches described in the introduction of this section. Table 2 gives the numerical results and shows that Pre-NMF performs competitively with sNMF in all cases (similar relative error for similar sparsity levels).

	Plain	Improved	$s(U)$	$s(V)$
SVD	7.28	7.28	0	0
NMF	7.97	7.96	53.27	11.36
Pre-NMF(0)	9.28 (9.76)	8.37	76.78	4.42
sNMF(0)	8.34	8.20	77.62	5.19
Pre-NMF(0.05)	11.12 (12.66)	9.15	90.14	2.16
sNMF(0.05)	9.24	8.90	91.12	2.22
Pre-NMF(0.1)	13.12 (23.47)	9.88	94.58	1.17
sNMF(0.1)	10.30	9.89	94.77	1.14

Table 2: Comparison of the relative approximation error and sparsity for the CBCL image data set. (In brackets, it is the error obtained when using $V = V'Q^{-1}$, instead of $V = \text{argmin}_{X \geq 0} ||M - UX||_F^2$.).

Figure 8 displays the basis elements obtained for NMF, Pre-NMF(0), Pre-NMF(0.1) and sNMF(0.1). The decomposition by parts obtained by Pre-NMF(0.1) is comparable to sNMF(0.1), reinforcing the observation above (cf. Table 2) that Pre-NMF performs competitively with sNMF.

Our technique has the advantage that only one parameter has to be chosen (namely ε) and that sparse solutions are naturally obtained. In fact, the user does not need to know in advance the desired sparsity level: one just has to try different values of $\varepsilon \in [0,1]$ (making sure $\rho(B_\varepsilon^*) < 1$) and a sparse factor U will automatically be generated (no parameters have to be tuned in the course of the optimization process). Moreover, Pre-NMF proves to be less sensitive to initialization than sNMF: we rerun both algorithms for $\varepsilon = 0.1$ with 100 different initializations (using exactly the same settings as above) and observe the following:

- Among the hundred solutions generated by sNMF(0.01), three did not achieve the required sparsity (being lower than 0.85, while all others were around 0.95 as imposed). In particular, the variance of the sparsity of the factor U for PreNMF(0.01) is $8.69\,10^{-7}$ while it is much higher $1.31\,10^{-3}$ for sNMF(0.01). (Note that after removing the three outliers, the variance of sNMF(0.01) is still higher being $3.23\,10^{-6}$.)

- The average of the relative error of Pre-NMF(0.01) is 9.94, slightly lower than sNMF(0.01) with 9.96.

- The variance of the relative error of Pre-NMF(0.01) is $5.97\,10^{-3}$, lower than sNMF(0.01) with $2.36\,10^{-2}$.

Remark 33 *We have also tested other sparse NMF techniques and they could not match the results obtained by sNMF, especially for high sparsity requirement. In particular, we tested the following*

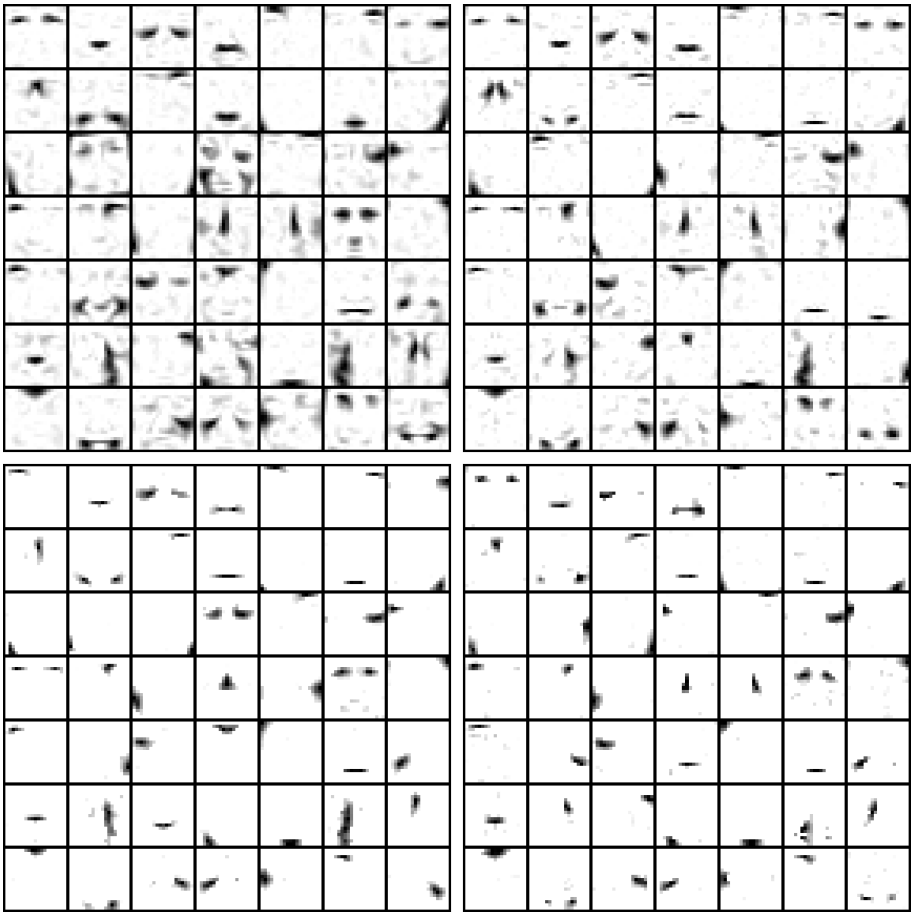


Figure 8: From left to right, top to bottom: basis elements for the CBCL data set obtained with NMF, Pre-NMF(0), Pre-NMF(0.1) and sNMF(0.1).

standard formulation using only one penalty parameter (Kim and Park, 2007)

$$\min_{U,V\geq 0} ||M-UV||_F^2 + \mu \sum_i ||U_{:i}||_1,$$

and the algorithm of Hoyer (2004).

6.2 Hubble Telescope Hyperspectral Image

The Hubble data set consists of 100 spectral images of the Hubble telescope, 128×128 pixels each (Pauca et al., 2006). It is composed of eight materials;[8] see the fourth row on Figure 10. The preprocessing took about one minute (about 0.5 second per image).[9] Figure 9 displays a sample of images of the simulated Hubble database along with the corresponding preprocessed images. The

8. These are true Hubble satellite material spectral signatures provided by the NASA Johnson Space Center.
9. The MATLAB® function *lsqlin* for solving CLLS problems was again much slower (about ten times) than *quadprog* with active set or with interior point which were comparable in this case.

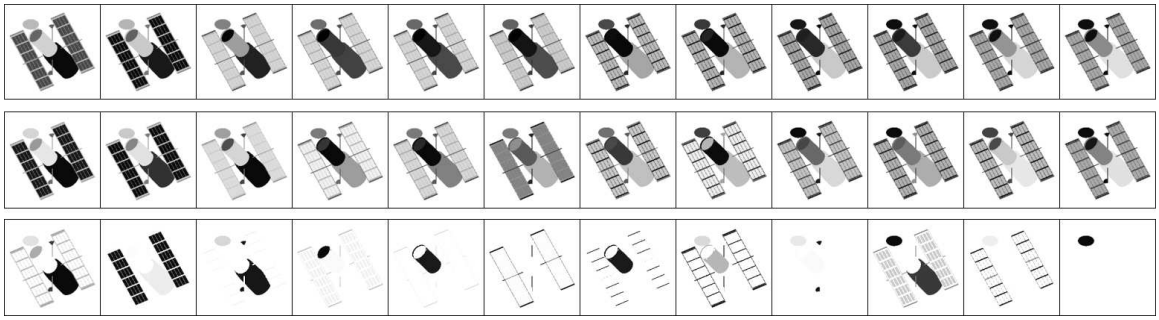


Figure 9: From top to bottom: Sample of Hubble images, corresponding preprocessed images for $\varepsilon = 0$ and $\varepsilon = 0.01$.

preprocessing for $\varepsilon = 0.01$ highlights extremely well the constitutive parts of the Hubble telescope: it is in fact able to extract some materials individually. Table 3 gives the sparsity and the value of ρ for the different preprocessed matrices.

	M	$\mathcal{P}(M)$	$\mathcal{P}_{0.01}(M)$
s(.)	57	57	80
$\rho(B_\varepsilon^*)$	0	0.9808	0.9979

Table 3: Hubble data set: sparsity of the preprocessed matrices $\mathcal{P}_\varepsilon(M) = MD$ and corresponding spectral radius of $B_\varepsilon^* = I - D$.

Table 4 reports the numerical results. Although sNMF(0.01) identifies a solution with slightly lower reconstruction error than Pre-NMF(0.01) (2.90 vs. 2.93), it is not able to identify the constitutive materials properly while Pre-NMF(0.01) *perfectly separates all eight constitutive materials*. It is also important to point out that the solutions generated by Pre-NMF(0.01) with different initializations correspond in most cases[10] to this optimal decomposition while the solutions generated by sNMF are typically very different (and with very different objective function values). This indicates that the NMF problem corresponding to the preprocessed matrix is more well posed.[11]

The comparison between sNMF(0) and Pre-NMF(0) is also interesting: the basis elements generated by Pre-NMF(0) (see second row of Figure 10) identify the constitutive materials much more effectively as six of them are almost perfectly extracted, while sNMF(0) only identifies one (while another is extracted as two separate basis elements).

10. We used 100 random initializations and obtained 61 times the optimal decomposition (in the other cases, it is always able to detect at least six of the eight materials).
11. Of course, in general, even if an NMF formulation has a unique global minimum (up to permutation and scaling), it will still have many local minima. Therefore, even in that situation, solutions generated with standard nonlinear optimization algorithms might still be rather different for different initializations.

	Plain	Improved	$s(U)$	$s(V)$
SVD	0.01	0.01	58	0
NMF	0.06	0.05	58.02	2.25
Pre-NMF(0)	0.08 (0.08)	0.07	59.16	0.13
sNMF(0)	0.37	0.36	64.14	0.63
Pre-NMF(0.01)	14.08 (75.09)	2.93	93.71	0
sNMF(0.01)	3.39	2.90	93.94	0

Table 4: Comparison of the relative approximation error and sparsity for the Hubble data set. (Notice that for $\varepsilon = 0.01$, the solution obtained using $V = V'Q^{-1}$ has a very high reconstruction error; the reason being that $Q = (I - B_\varepsilon^*)$ is close to being singular since $\rho(B_\varepsilon^*) = 0.9979$.)

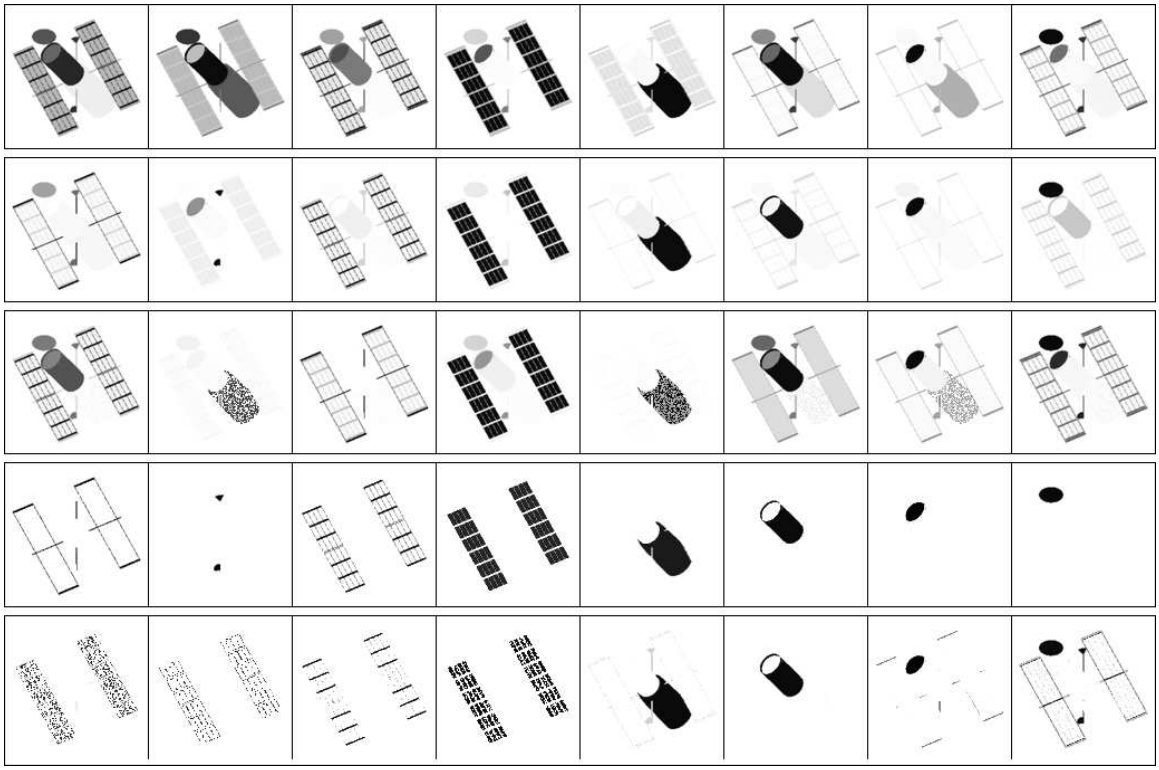


Figure 10: From top to bottom: basis elements for the Hubble data set obtained by NMF, Pre-NMF(0), sNMF(0), Pre-NMF(0.01) and sNMF(0.01).

7. Conclusion and Further Research

In this paper, we introduced a completely new approach to make NMF problems more well posed and have sparser solutions. It is based on the preprocessing of the nonnegative data matrix M:

given M, we compute an inverse positive matrix Q such that the preprocessed matrix $\mathcal{P}(M) = MQ$ remains nonnegative and is sparse. The computation of Q relies on the resolution of constrained linear least squares problems (CLLS). We proved that the preprocessing is well-defined, invariant to permutation and scaling of the columns of matrix M, and preserves the rank of M (as long as the vertices of $\text{conv}(\theta(M))$ are non repeated).

Because $\mathcal{P}(M)$ is sparser than M, the corresponding NMF problem will be more well posed and have sparser solutions. In particular, we were able to show that

- Under the separability assumption of Donoho and Stodden (2003), the preprocessing is optimal as it identifies the vertices of the convex hull of the columns of M.

- Since any rank-two matrix satisfies the separability assumption, the preprocessing is optimal for any nonnegative rank-two matrix.

- In the exact rank-three case (that is, $M = UV$, $\text{rank}(M) = \text{rank}_+(M) = 3$), the preprocessing can be used to make the set of optimal solutions of the NMF problem finite. We conjecture that, generically, it makes it unique and that this result holds for higher rank matrices.

We also proposed a more general preprocessing that relaxes the constraint that $\mathcal{P}(M)$ has to be nonnegative, which is able to deal better with noisy and sparse matrices. Moreover, it generates sparser preprocessed matrices hence sparser NMF solutions. We experimentally showed the effectiveness of this strategy on facial and hyperspectral image data sets. In particular, it performed competitively with a state-of-the-art sparse NMF technique based on ℓ_1-norm penalty functions. It is robust to high sparsity requirement and no parameters have to be tuned in the course of the optimization process. Only one parameter has to be chosen which will allow the user to generate more or less sparse preprocessed matrices.

The main drawback of the technique seems to be its computational cost: n CLLS problems in n variables and $m+n$ constraints have to be solved (where n in the number of columns of M) for a total computational cost of the order of $O(n^{4.5})$ (using MATLAB® on a standard laptop, it limits n to be smaller than 1000 for a few hours of computation). It would then be particularly interesting to investigate strategies to speed up the preprocessing. Using faster solvers is one possible approach (probably in detriment of the accuracy), for example, based on first-order methods.[12] Another possibility would be to use the following heuristic: since the preprocessing removes from each column of M a linear combinations of the other columns, one could use only a subset of k columns of M to be subtracted from the other columns of M. This amounts to fixing variables to zero in the CLLS problems and would reduce the computational complexity to $O(nk^{3.5})$. This subset of columns could for example be selected such that its convex hull has a large volume, see, for example, Klingenberg et al. (2009) for a possible heuristic; or such that they form the best possible basis for the remaining columns (that is, use a column subset selection algorithm); see Boutsidis et al. (2009) and the references therein.

Finally, a particularly challenging direction for research would be to design other data preprocessing techniques for NMF. One approach would be to characterizing the set of inverse positive matrices better: in this paper, we only worked with the subset of invertible M-matrices. For exam-

12. We have developed an alternating direction method (ADM), along with Ting Kei Pong, which allowed us to preprocess the CBCL data set in about 10 hours with 10^{-3} relative accuracy; the code is available upon request.

ple, the matrix[13]

$$M = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

would not be modified by our preprocessing (because each column contains a zero entry corresponding to positive ones in all other columns) although its NMF is not unique (cf. Section 2). In fact, we have

$$MQ = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix} = 2 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

where Q is inverse positive with $Q^{-1} = \frac{1}{2}M$, and the NMF of MQ is unique. This example shows that working with a larger set of inverse positive matrices would allow to obtain sparser preprocessed data matrices, hence more well-posed NMF problems with sparser solutions.

Acknowledgments

The author acknowledges a discussion with Mariya Ishteva about uniqueness issues of NMF which motivated the study of inverse-positive matrices in this context. The author would like to thank K.C. Sivakumar and F.-X. Orban de Xivry for helpful discussions on inverse positive matrices and on the problem of finding the closest stable matrix to a given one, respectively, and Stephen Vavasis for carefully reading and commenting a first draft of this manuscript. The authors also thanks the anonymous reviewers for their insightful comments, which helped to improve the paper.

Appendix A. Proof for Theorem 29

In this section, we prove that the function f_k defined in Theorem 29 is continuous and made up of pieces which are either constant or strictly convex (which we refer to as piecewise constant/strictly convex). The construction described below is the same as the one proposed by Aggarwal et al. (1989) and we refer the reader to that paper for more details. The novelty of our proof is to use that construction to show that f_k is piecewise constant/strictly convex (it was already shown to be continuous and nondecreasing by Aggarwal et al., 1989).

Proof Let $x(t_1)$ be on the boundary of P and define the sequence $x(t_2), \ldots, x(t_{k+1})$ as in Theorem 29 (clock-wise). As shown by Aggarwal et al. (1989), the function $f_k(t_1) = t_{k+1}$ only depends on

1. The sides of P on which the points $x(t_i)$ $1 \leq i \leq k+1$ lie ;
2. The intersections of the segments $[x(t_i), x(t_{i+1})]$ $1 \leq i \leq k$ with Q ;

and, given that these sides and intersections do not change, f_k is continuously differentiable and can be characterized in closed form (see below). These sides and intersections will change when either

- One of the points $x(t_i)$ switches from one side of the boundary of P to another. These points correspond to the vertices of P (P has at most m vertices since it is a polygon defined with m inequalities); or,

13. We thank Mariya Ishteva for providing us with this example.

- One of the intersections of the segments $[x(t_i), x(t_{i+1})]$ $1 \leq i \leq k$ with Q changes. There is a one-to-one correspondence between these points and the sides of Q (Q has at most n vertices hence at most n sides).

These points where the description of f_k changes (and where f_k is not continuously differentiable) are called the contact change points. Turning around the boundary of P, we might encounter more than $m+n$ such points. However, two contact change points corresponding to the same change are associated with the same sequence $x(t_i)$ $1 \leq i \leq k+1$ hence the same solution to the NPP. In fact, both sequences must share at least one point (either a vertex of P or the intersections of a line containing a side of Q with the boundary of P) which implies, by construction, that they are the same. Therefore, there are at most $m+n$ contact change points corresponding to different sequences $x(t_i)$ $1 \leq i \leq k+1$ on the boundary of P (Aggarwal et al., 1989).

It remains to show that the pieces of f_k between two contact change points are either constant or strictly convex.

Let us then construct the function f_k between two contact change points. Without loss of generality, we may assume that the perimeter of the outer polygon P is equal to one (otherwise scale the polygons P and Q accordingly), and that the parametrization x of the boundary of P has the following property: the distance traveled when following the boundary between $x(s)$ and $x(t)$ is equal to $|(s-\lfloor s \rfloor)-(t-\lfloor t \rfloor)|$. In particular, if $0 \leq s \leq t \leq 1$, then the distance traveled between $x(t)$ and $x(s)$ along the boundary of P is $t-s$. We may also assume without loss of generality that $x(0)=(0,0)$ is the vertex on P preceding $x(t_1)$ and that $x(t_1)=(0,t_1)$: this amounts to translating and rotating P and Q. We also define (see Figure 11 for an illustration)

- $q=(q_1,q_2)$, the tangent point on Q between $x(t_1)$ and $x(t_2)$.
- θ, the angle between the sides of P on which $x(t_1)$ and $x(t_2)$ are.
- p, the intersection between the sides on which $x(t_1)$ and $x(t_2)$ are (note that p is on the boundary of P if and only if there is one and only one vertex of P between $x(t_1)$ and $x(t_2)$).
- d, the distance between $x(0)$ and p.
- s, the distance between p and $x(t_2)$.
- a, the projection of q on the line $[x(0),p]$.
- b, the projection of $x(t_2)$ on the line $[x(0),p]$.

Case 1: The point q is on the same side as $x(t_1)$. This implies that $x(t_2)=p$ for any $t_1 < q_1$ and no other points of the sequence is changed since $x(t_2)$ remains the same. Therefore, the function $t_{k+1}=f_k(t_1)$ is constant. (Notice that $x(q_1)$ is a contact change point since $x(t_2)$ will switch side when $t_1=q_1$.)
Case 2: The point q is on the same side as $x(t_2)$. This implies that $x(t_2)=q$ for any $t_1 < d$. Therefore, the function $t_{k+1}=f_k(t_1)$ is constant. (Notice that the next contact change point will be the first vertex of P that $x(t_1)$ crosses.)
Case 3: The point q is not on the same side as $x(t_1)$ or $x(t_2)$ (it is in the interior of P). Using the similarity between the triangles $\Delta x(t_1)aq$ and $\Delta x(t_1)bx(t_2)$, we have that (Aggarwal et al., 1989, Equation (1))

$$\frac{q_2}{q_1-t_1}=\frac{s\sin(\theta)}{d-t_1+s\cos(\theta)},$$

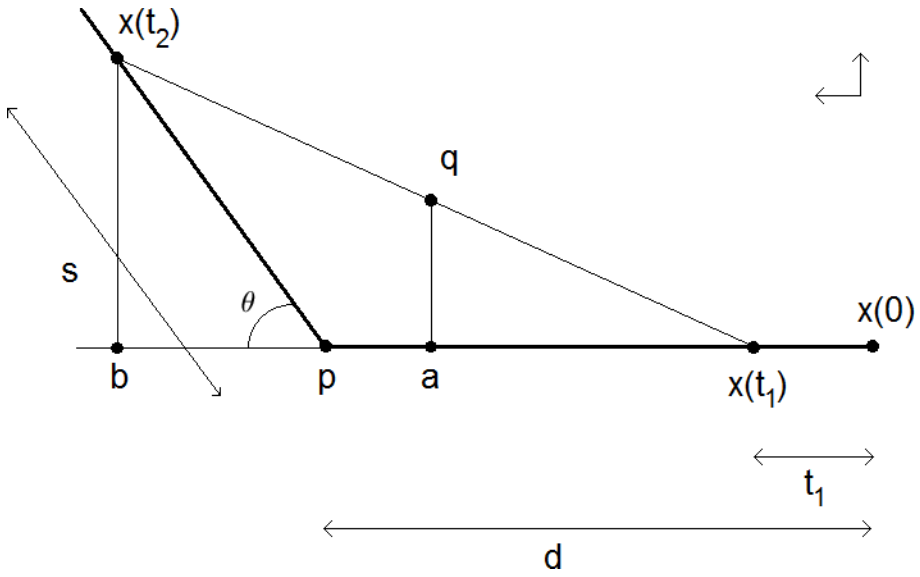


Figure 11: Construction of the function f_1 between two contact change points (see Aggarwal et al. 1989, Figure 3, for a similar illustration).

implying

$$s = \frac{q_2}{\sin(\theta)} \frac{d - t_1}{q_1 - q_2 \cot(\theta) - t_1} = g_1(t_1).$$

Let us show that $g_1(t_1)$ is strictly convex, that is, $g_1''(t_1) > 0$. Since q is not on the same side as $x(t_1)$ or $x(t_2)$, we have $q_2 > 0$ and $0 < \theta < \pi$ implying $\frac{q_2}{\sin(\theta)} > 0$. Hence it suffices to show that $h(t_1) = \frac{d-t_1}{l-t_1}$ is strictly convex, where $l = q_1 - q_2 \cot(\theta)$. Since $s > 0$ and $d > t_1$, we must have $l - t_1 > 0$. (Notice that $x(l)$ is a contact change point. In fact, for $t_1 = l$, the segments $[x(t_1), q]$ and $[p, x(t_2)]$ become parallel implying that the intersection of Q with the segment $[x(t_1), x(t_2)]$ will change.)

We then have

$$h'(t_1) = \frac{d - l}{(l - t_1)^2}.$$

Since h is a strictly increasing function of t_1 (Aggarwal et al., 1989), $h'(t_1) > 0$ hence $d > l$ and

$$h''(t_1) = 2\frac{d - l}{(l - t_1)^3} > 0,$$

so that $g_1(t)$ is strictly convex. Finally, we have

$$f_1(t_1) = t_2 = c_1 + s = c_1 + g_1(t_1),$$

where either

- $c_1 = 0$ and g_1 is a constant (cases 1. and 2.).

- c_1 is an appropriate constant and g_1 is an increasing and strictly convex function (case 3.).

By construction, the same relationship will apply between t_2 and t_3 with

$$f_2(t_1) = t_3 = c_2 + g_2(s) = c_2 + g_2(g_1(t_1)),$$

where c_2 is an appropriate constant and g_2 is either constant, or strictly convex and increasing. After $k+1$ steps, we have

$$f_k(t_1) = t_{k+1} = c_k + g_k(s) = c_k + (g_k \circ g_{k-1} \circ \cdots \circ g_1)(t_1),$$

where c_k is an appropriate constant and the functions g_i are either constant, or strictly convex and increasing. If one of the functions g_i $1 \leq i \leq k$ is constant, then f_k is constant. Otherwise the function $f_k(t_1) = c_k + (g_{k-1} \circ \cdots \circ g_1)(t_1)$ is strictly convex since it is a constant plus the composition of strictly convex and *increasing* functions. (In fact, the composition of one-dimensional increasing and strictly convex functions is increasing and strictly convex.) ■

References

A. Aggarwal, H. Booth, J. O'Rourke, and S. Suri. Finding minimal convex nested polygons. *Information and Computation*, 83(1):98–110, 1989.

S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization – provably. In *Proceedings of the 44th Symposium on Theory of Computing*, STOC '12, pages 145–162, New York, NY, USA, 2012. ACM.

A. Berman and R.J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. SIAM, 1994.

C. Bocci, E. Carlini, and F. Rapallo. Perturbation of matrices and non-negative rank with a view toward statistical models. *SIAM. J. Matrix Anal. & Appl.*, 32(4):1500–1512, 2011.

C. Boutsidis, M.W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 968–977, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

M.T. Chu and M.M. Lin. Low-dimensional polytope approximation and its applications to nonnegative matrix factorization. *SIAM J. Sci. Comput.*, 30(3):1131–1155, 2008.

A. Cichocki, S. Amari, R. Zdunek, and A.H. Phan. *Non-negative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley-Blackwell, 2009.

M.D. Craig. Minimum-volume tranforms for remotely sensed data. *IEEE Trans. on Geoscience and Remote Sensing*, 32(3):542–552, 1994.

C. Ding, X. He, and H.D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *SIAM Int. Conf. Data Mining (SDM'05)*, pages 606–610, 2005.

C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *In Proc. of the 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 126–135, 2006.

D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems 16*, 2003.

C. Févotte, N. Bertin, and J.L. Durrieu. Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural Computation*, 21(3):793–830, 2009.

N. Gillis. *Nonnegative Matrix Factorization: Complexity, Algorithms and Applications*. PhD thesis, Université catholique de Louvain, 2011.

N. Gillis and F. Glineur. Nonnegative factorization and the maximum edge biclique problem. CORE discussion paper 2008/64, 2008.

N. Gillis and F. Glineur. Using underapproximations for sparse nonnegative matrix factorization. *Pattern Recognition*, 43(4):1676–1687, 2010.

N. Gillis and F. Glineur. On the geometric interpretation of the nonnegative rank. *Linear Algebra and its Applications*, 437(11):2685–2712, 2012a.

N. Gillis and F. Glineur. Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization. *Neural Computation*, 24(4):1085–1105, 2012b.

N. Gillis and S.A. Vavasis. Fast and robust recursive algorithms for separable nonnegative matrix factorization. arXiv:1208.1237, 2012.

H. Hazewinkel. On positive vectors, positive matrices and the specialization ordering. Technical report, CWI Report PM-R8407, 1984.

P.O. Hoyer. Nonnegative matrix factorization with sparseness constraints. *J. Machine Learning Research*, 5:1457–1469, 2004.

A. Huck, M. Guillaume, and J. Blanc-Talon. Minimum dispersion constrained nonnegative matrix factorization to unmix hyperspectral data. *IEEE Trans. on Geoscience and Remote Sensing*, 48 (6):2590–2602, 2010.

V. Kalofolias and E. Gallopoulos. Computing symmetric nonnegative rank factorizations. *Linear Algebra and its Applications*, 436(2):421–435, 2012.

H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12):1495–1502, 2007.

B. Klingenberg, J. Curry, and A. Dougherty. Non-negative matrix factorization: Ill-posedness and a geometric algorithm. *Pattern Recognition*, 42(5):918–928, 2009.

H. Laurberg, M.G. Christensen, M.D. Plumbley, L.K. Hansen, and S.H. Jensen. Theorems on positive data: On the uniqueness of NMF. *Computational Intelligence and Neuroscience*, 2008. Article ID 764206.

D.D. Lee and H.S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791, 1999.

L. Miao and H. Qi. Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization. *IEEE Trans. on Geoscience and Remote Sensing*, 45 (3):765–777, 2007.

P. Paatero and U. Tapper. Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

V.P. Pauca, J. Piper, and R.J. Plemmons. Nonnegative matrix factorization for spectral data analysis. *Linear Algebra and its Applications*, 406(1):29–47, 2006.

B.T. Polyak and P.S. Shcherbakov. Hard problems in linear control theory: Possible approaches to solution. *Automation and Remote Control*, 66:681–718, 2005.

F. Pompili, N. Gillis, P.-A. Absil, and F. Glineur. Two algorithms for orthogonal nonnegative matrix factorization with application to clustering. arXiv:1201.090, 2011.

G.B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

Y. Sun and J. Xin. Underdetermined sparse blind source separation of nonnegative and partially overlapped data. *SIAM Journal on Scientific Computing*, 33(4):2063–2094, 2011.

O. Taussky. A recurring theorem on determinants. *The American Mathematical Monthly*, 56(10): 672–676, 1949.

F.J. Theis, K. Stadlthanner, and T. Tanaka. First results on uniqueness of sparse non-negative matrix factorization. In *13th European Signal Processing Conference, EUSIPCO, Antalya, Turkey*, 2005.

L.B. Thomas. Rank factorization of nonnegative matrices. *SIAM Review*, 16(3):393–394, 1974.

S.A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM J. on Optimization*, 20 (3):1364–1377, 2009.

F.-Y. Wang, C.-Y. Chi, T.-H. Chan, and Y. Wang. Nonnegative least-correlated component analysis for separation of dependent sources by volume maximization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(5):875–888, 2010.

W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proc. of the 26th Annual Int. ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '03, pages 267–273, New York, NY, USA, 2003. ACM.

G. Zhou, S. Xie, Z. Yang, J.-M. Yang, and Z. He. Minimum-volume-constrained nonnegative matrix factorization: Enhanced ability of learning parts. *IEEE Trans. on Neural Networks*, 22(10):1626–1637, 2011.

Journal of Machine Learning Research 13 (2012) 3387-3439 Submitted 10/11; Revised 5/12; Published 11/12

Learning Linear Cyclic Causal Models with Latent Variables

Antti Hyttinen ANTTI.HYTTINEN@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Frederick Eberhardt FDE@CMU.EDU
Department of Philosophy
Baker Hall 135
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA

Patrik O. Hoyer PATRIK.HOYER@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Editor: Christopher Meek

Abstract

Identifying cause-effect relationships between variables of interest is a central problem in science. Given a set of experiments we describe a procedure that identifies linear models that may contain cycles and latent variables. We provide a detailed description of the model family, full proofs of the necessary and sufficient conditions for identifiability, a search algorithm that is complete, and a discussion of what can be done when the identifiability conditions are not satisfied. The algorithm is comprehensively tested in simulations, comparing it to competing algorithms in the literature. Furthermore, we adapt the procedure to the problem of cellular network inference, applying it to the biologically realistic data of the DREAM challenges. The paper provides a full theoretical foundation for the causal discovery procedure first presented by Eberhardt et al. (2010) and Hyttinen et al. (2010).

Keywords: causality, graphical models, randomized experiments, structural equation models, latent variables, latent confounders, cycles

1. Introduction

Inferring causal relationships from data is of fundamental importance in many areas of science. One cannot claim to have fully grasped a complex system unless one has a detailed understanding of how the different components of the system affect each other, and one cannot predict how the system will respond to some targeted intervention without such an understanding. It is well known that a statistical dependence between two measured quantities leaves the causal relation underdetermined—in

addition to a causal effect from one variable to another (in either or both directions), the dependence might be due to a common cause (a confounder) of the two.

In light of this underdetermination, *randomized experiments* have become the gold standard of causal discovery. In a randomized experiment, the values of some variable x_i are assigned at random by the experimenter and, consequently, in such an experiment any correlation between x_i and another measured variable x_j can uniquely be attributed to a causal effect of x_i on x_j, since any incoming causal effect on x_i (from x_j, a common cause, or otherwise) would be 'broken' by the randomization. Since their introduction by Fisher (1935), randomized experiments now constitute an important cornerstone of experimental design.

Since the 1980s causal graphical models based on directed graphs have been developed to systematically represent causal systems (Glymour et al., 1987; Verma and Pearl, 1988). In this approach, causal relations among a set of variables $\mathcal{V}$ are represented by a set of directed edges $\mathcal{D} \subseteq (\mathcal{V} \times \mathcal{V})$ connecting nodes in a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{D})$, where a directed edge from node x_i to node x_j in the graph represents the *direct* causal effect of x_i on x_j (relative to the set of variables $\mathcal{V}$). The causal relationships in such a model are defined in terms of stochastic functional relationships (or alternatively conditional probability distributions) that specify how the value of each variable is influenced by the values of its direct causes in the graph. In such a model, randomizing a variable x_i is tantamount to removing all arrows pointing *into* that variable, and replacing the functional relationship (or conditional probability distribution) with the distribution specified in the experiment. The resulting truncated model captures the fact that the value of the variable in question is no longer influenced by its normal causes but instead is determined explicitly by the experimenter. Together, the graph structure and the parameters defining the stochastic functional relationships thus determine the joint probability distribution over the full variable set under any experimental conditions.

The question that interests us here is how, and under what conditions, we can learn (i.e., infer from data) the structure and parameters of such causal models. The answer to this question depends largely on what assumptions we are willing to make about the underlying models and what tools of investigation we consider. For instance, some causal discovery methods require assuming that the causal structure is *acyclic* (has no directed cycles), while others require *causal sufficiency*, that is, that there are no unmeasured common causes affecting the measured variables. Many algorithms provide provably consistent estimates only under the assumption of *faithfulness*, which requires that the structure of the graph uniquely determines the set of (conditional) independencies that hold between the variables. For some methods the functional form of the relationships has to take a certain predetermined form (e.g., linearity). Under various combinations of the above assumptions, it is possible to consistently infer (at least partial information concerning) the causal relationships underlying the observed data from non-experimental ('passive observational') data (Richardson, 1996; Spirtes et al., 2000; Pearl, 2000; Chickering, 2002a,b; Shimizu et al., 2006).

In many cases, researchers may not be willing to make some of the assumptions mentioned above, or they may want to guarantee that the full structure of the model is inferred (as opposed to only inferring an equivalence class of possible models, a common result of many discovery methods). A natural step is thus to use the power of randomized experiments. The question then becomes: Under what assumptions on the model and for what sets of experiments can one guarantee consistent learning of the underlying causal structure. Here, almost all of the existing literature has focused on the acyclic case (Cooper and Yoo, 1999; Tong and Koller, 2001; Murphy, 2001;

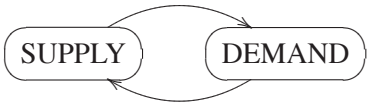


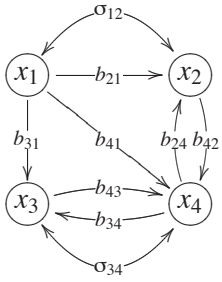
Figure 1: Classic supply-demand model.

Eberhardt et al., 2005; Meganck et al., 2005; Nyberg and Korb, 2006; Eberhardt and Scheines, 2007; Eaton and Murphy, 2007).

The acyclicity assumption, common to most discovery algorithms, permits a straightforward interpretation of the causal model and is appropriate in some circumstances. But in many cases the assumption is clearly ill-suited. For example, in the classic demand-supply model (Figure 1) demand has an effect on supply and vice versa. Intuitively, the true causal structure is acyclic over time since a cause always precedes its effect: Demand of the previous time step affects supply of the next time step. However, while the causally relevant time steps occur at the order of days or weeks, the measures of demand and supply are typically cumulative averages over much longer intervals, obscuring the faster interactions. A similar situation occurs in many biological systems, where the interactions occur on a much faster time-scale than the measurements. In these cases a cyclic model provides the natural representation, and one needs to make use of causal discovery procedures that do not rely on acyclicity (Richardson, 1996; Schmidt and Murphy, 2009; Itani et al., 2008).

In this contribution we consider the problem of learning the structure and parameters of linear cyclic causal models from equilibrium data. We derive a necessary and sufficient condition for identifiability based on second-order statistics, and present a consistent learning algorithm. Our results and learning method *do not* rely on causal sufficiency (the absence of hidden confounding), *nor* do they require faithfulness, that is, that the independencies in the data are fully determined by the graph structure. To our knowledge these results are the first under assumptions that are this weak. Given that the model space is very general (essentially only requiring linearity), randomized experiments are needed to obtain identification. While for certain kinds of experimental data it is easy to identify the full causal structure, we show that significant savings either in the number of experiments or in the number of randomized variables per experiment can be achieved. All-in-all, the present paper provides the full theoretical backbone and thorough empirical investigation of the inference method that we presented in preliminary and abbreviated form in Eberhardt et al. (2010) and Hyttinen et al. (2010). It establishes a concise theory for learning linear cyclic models with latent variables.

We start in Section 2 by introducing the model and its assumptions, how the model is to be interpreted, and how experimental interventions are represented. In Section 3 we derive conditions (on the set of randomized experiments to be performed) that are necessary and sufficient for model identification. These results provide the foundation for the correct and complete learning method presented in Section 4. This section also discusses the underdetermination which results if the identifiability conditions are not met. Section 5 presents empirical results based on thorough simulations, comparing the performance of our procedure to existing methods. Finally, we adapt the procedure to the problem of cellular network inference, and apply it to the biologically realistic *in silico* data of the DREAM challenges in Section 6. Some extensions and conclusions are given in Sections 7 and 8.



$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ b_{21} & 0 & 0 & b_{24} \\ b_{31} & 0 & 0 & b_{34} \\ b_{41} & b_{42} & b_{43} & 0 \end{bmatrix} \quad \mathbf{\Sigma_e} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & 0 & 0 \\ \sigma_{12} & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & \sigma_{34} \\ 0 & 0 & \sigma_{34} & \sigma_4^2 \end{bmatrix}$$

Figure 2: An example of a linear cyclic model with latent variables. A non-zero coefficient b_{21} is represented in the graph by the arc $x_1 \to x_2$. Similarly, the non-zero covariance between disturbances e_1 and e_2 is represented by the arc $x_1 \leftrightarrow x_2$. In the graph the disturbance term for each individual variable has been omitted for clarity. Note that a pair of opposing directed edges, such as $x_3 \to x_4$ and $x_3 \leftarrow x_4$, represents reciprocal causation (feedback relationship) between the variables, whereas a double-headed arrow, such as $x_3 \leftrightarrow x_4$, represents confounding.

2. Model

We start by presenting the basic interpretation of the cyclic model in the passive observational (Section 2.1) and experimental settings (Section 2.2). We establish canonical forms for both the model and the experiments to simplify the presentation of the subsequent theory. We then discuss different stability assumptions to ensure the presence of model equilibria, and show how they relate to the model interpretation and model marginalization (Section 2.3).

2.1 Linear Cyclic Model with Latent Variables

Following the framework presented in Bollen (1989), we consider a general linear structural equation model (SEM) with correlated errors as our underlying data generating model. In such a model the value of each observed variable $x_j \in \mathcal{V}$ $(j = 1, ..., n)$ is determined by a linear combination of the values of its causal parents $x_i \in \text{pa}(x_j)$ and an additive disturbance ('noise') term e_j:

$$x_j \quad := \quad \sum_{x_i \in \text{pa}(x_j)} b_{ji} x_i + e_j.$$

Representing all the observed variables as a vector $\mathbf{x}$ and the corresponding disturbances as a vector $\mathbf{e}$, these structural equations can be represented by a single matrix equation

$$\mathbf{x} \quad := \quad \mathbf{B}\mathbf{x} + \mathbf{e}, \tag{1}$$

where $\mathbf{B}$ is the $(n \times n)$-matrix of coefficients b_{ji}. A graphical representation of such a causal model is given by representing any non-zero causal effect b_{ji} by an edge $x_i \to x_j$ in the corresponding graph. An example graph and matrix $\mathbf{B}$ are shown in Figure 2.

The set of equations is said to be *recursive* or *acyclic* if the graph describing the causal relations has no directed cycles, or (equivalently) if there exists a causal order of the variables for which

the corresponding matrix $\mathbf{B}$ is lower triangular. When the graph contains directed cycles (feedback loops), such as for the model of Figure 2, then the model is said to be *non-recursive* or *cyclic*. In this paper we do *not* assume a priori that the underlying model is acyclic. In other words, our model family allows for *both* cyclic and acyclic cases.

While in a 'fully observed' SEM the disturbance terms e_i would be assumed to be independent of each other, we allow for unobserved confounding by modeling arbitrary correlations among the disturbances $e_1, ..., e_n$. Specifically, denote by $\boldsymbol{\mu}_\mathbf{e}$ and $\boldsymbol{\Sigma}_\mathbf{e}$ the mean vector and the variance-covariance matrix (respectively) of the disturbance vector $\mathbf{e}$. The diagonal elements of $\boldsymbol{\Sigma}_\mathbf{e}$ represent the variances of the disturbances, while the off-diagonal entries represent the covariances. In the corresponding graph a non-zero covariance between e_i and e_j is represented by the double-headed arc $x_i \leftrightarrow x_j$. Notice that in this implicit representation, a latent variable that confounds three observed variables is represented by three (pairwise) covariances. To keep the notation as simple as possible, we will adopt the assumption standard in the literature that the disturbances have zero mean, that is, $\boldsymbol{\mu}_\mathbf{e} = \mathbf{0}$. In Appendix A we show that it is usually possible to transform the observed data to a form consistent with this assumption. We are thus ready to define the underlying data-generating model:

Definition 1 (Linear Cyclic Model with Latent Variables) *A linear cyclic model with latent variables $\mathcal{M} = (\mathbf{B}, \boldsymbol{\Sigma}_\mathbf{e})$, is a structural equation model over a set of observed variables $x_1, \cdots, x_n \in \mathcal{V}$ of the form of Equation 1, where the disturbance vector $\mathbf{e}$ has mean $\boldsymbol{\mu}_\mathbf{e} = \mathbf{0}$ and an arbitrary symmetric positive-definite variance-covariance matrix $\boldsymbol{\Sigma}_\mathbf{e}$.*

In order to give a fully generative explanation of the relationship between the model parameters and the data, additional constraints on $\mathbf{B}$ are needed. Typically, a cyclic model is used to represent a causal process that is collapsed over the time dimension and where it is assumed that the data sample is taken after the causal process has 'settled down'. The traditional interpretation of non-recursive SEMs assumes that the disturbances represent background conditions that do not change until the system has reached equilibrium and measurements are taken. So for a given set of initial values for the variables $\mathbf{x}(0)$, a data vector is generated by drawing one vector of disturbances $\mathbf{e}$ from the error distribution and iterating the system

$$\mathbf{x}(t) \quad := \quad \mathbf{B}\mathbf{x}(t-1) + \mathbf{e} \tag{2}$$

by adding in the constant (with respect to time) $\mathbf{e}$ at every time step until convergence. At time t the vector $\mathbf{x}$ thus has the value

$$\mathbf{x}(t) \quad := \quad (\mathbf{B})^t \mathbf{x}(0) + \sum_{i=0}^{t-1} (\mathbf{B})^i \mathbf{e}.$$

For $\mathbf{x}(t)$ to converge to an equilibrium, the geometric sequence $(\mathbf{B}^i)_{i=0\ldots t}$ and the geometric series $\sum_{i=0}^{t-1} \mathbf{B}^i$ must converge as $t \to \infty$. For arbitrary $\mathbf{x}(0)$ and arbitrary $\mathbf{e}$, a necessary and sufficient condition for this is that the eigenvalues λ_k of $\mathbf{B}$ satisfy $\forall k : |\lambda_k| < 1$ (Fisher, 1970). In that case $(\mathbf{B})^t \to 0$ and $\sum_{i=0}^{t-1} \mathbf{B}^i \to (\mathbf{I} - \mathbf{B})^{-1}$ as $t \to \infty$, so $\mathbf{x}(t)$ converges to

$$\mathbf{x} \quad = \quad (\mathbf{I} - \mathbf{B})^{-1} \mathbf{e},$$

where $(\mathbf{I} - \mathbf{B})$ is guaranteed to be invertible given the above restriction on the eigenvalues. Notice that the observed value $\mathbf{x}$ at equilibrium is independent of the starting point $\mathbf{x}(0)$, and completely

determined by $\mathbf{B}$ and $\mathbf{e}$. Multiple samples of $\mathbf{x}$ are obtained by repeating this equilibrating process for different samples of $\mathbf{e}$. Hence, for $\mathcal{M} = (\mathbf{B}, \mathbf{\Sigma_e})$ the variance-covariance matrix over the observed variables is

$$\mathbf{C_x} = E\{\mathbf{x}\mathbf{x}^T\} = (\mathbf{I}-\mathbf{B})^{-1}E\{\mathbf{e}\mathbf{e}^T\}(\mathbf{I}-\mathbf{B})^{-T} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma_e}(\mathbf{I}-\mathbf{B})^{-T}. \quad (3)$$

The equilibrium we describe here corresponds to what Lauritzen and Richardson (2002) called a *deterministic* equilibrium, since the equilibrium value of $\mathbf{x}(t)$ is fully determined given a sample of the disturbances $\mathbf{e}$. Such an equilibrium stands in contrast to a *stochastic* equilibrium, resulting from a model in which the disturbance term is sampled anew at each time step in the equilibrating process. We briefly return to consider such models in Section 7. We note that if the model happens to be acyclic (i.e., has no feedback loops), the interpretation in terms of a deterministic equilibrium coincides with the standard recursive SEM interpretation, with no adjustments needed.

It is to be expected that in many systems the value of a given variable x_i at time t has a non-zero effect on the value of the same variable at time $t+1$. (For instance, such systems are obtained when approximating a linear differential equation with a difference equation.) In such a case the coefficient b_{ii} (a diagonal element of $\mathbf{B}$) is by definition non-zero, and the model is said to exhibit a 'self-loop' (a directed edge from a node to itself in the graph corresponding to the model). As will be discussed in Section 2.3, such self-loops are inherently unidentifiable from equilibrium data, so there is a need to define a standardized model which abstracts away non-identifiable parameters. For this purpose we introduce the following definition.

Definition 2 (Canonical Model) *A linear cyclic model with latent variables* $(\mathbf{B}, \mathbf{\Sigma_e})$ *is said to be a* canonical model *if it does not contain self-loops (i.e., the diagonal of* $\mathbf{B}$ *is zero).*

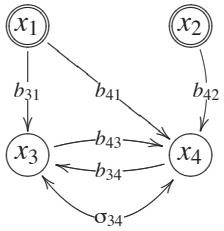
We will show in Section 2.3 how one can obtain the canonical model that yields in all experiments the same observations at equilibrium as an arbitrary (i.e., including self-loops) linear cyclic model with latent variables.

2.2 Experiments

As noted in the introduction, one of the aims of inferring causal models is the ability to predict how a system will react when it is subject to intervention. One key feature of linear cyclic models with latent variables is that they naturally integrate the representation of experimental manipulations, as discussed in this subsection.

We characterize an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ as a partition of the observed variables $\mathcal{V}$ (i.e., $\mathcal{J}_k \cup \mathcal{U}_k = \mathcal{V}$ and $\mathcal{J}_k \cap \mathcal{U}_k = \emptyset$) into a set $\mathcal{J}_k$ of intervened variables and a set $\mathcal{U}_k$ of passively observed variables. Note that in this representation, a passive observational data set is a 'null-experiment' in which $\mathcal{J}_k = \emptyset$ and $\mathcal{U}_k = \mathcal{V}$. Following the standard view (Spirtes et al., 2000; Pearl, 2000), we consider in this paper randomized "surgical" interventions that break all incoming causal influences to the intervened variables by setting the intervened variables to values determined by an exogenous intervention distribution with mean $\boldsymbol{\mu}_\mathbf{c}^k$ and covariance $\mathrm{cov}(\mathbf{c}) = \mathbf{\Sigma}_\mathbf{c}^k$. In the graph of the underlying model, this corresponds to cutting all edges into the intervened nodes; see Figure 3 for an example.

To simplify notation, we denote by $\mathbf{J}_k$ and $\mathbf{U}_k$ two $(n \times n)$ diagonal 'indicator matrices', where $(\mathbf{J}_k)_{ii} = 1$ if and only if $x_i \in \mathcal{J}_k$, all other entries of $\mathbf{J}_k$ are zero, and $\mathbf{U}_k = \mathbf{I} - \mathbf{J}_k$. The vector $\mathbf{c}$ represents the values of the intervened variables determined by the intervention distribution, and



$$\mathbf{U}_k\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b_{31} & 0 & 0 & b_{34} \\ b_{41} & b_{42} & b_{43} & 0 \end{bmatrix} \quad \mathbf{U}_k\mathbf{\Sigma_e}\mathbf{U}_k + \mathbf{J}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & \sigma_{34} \\ 0 & 0 & \sigma_{34} & \sigma_4^2 \end{bmatrix}$$

Figure 3: Manipulated model corresponding to an intervention on variables x_1 and x_2 in the model of Figure 2, that is, the result of an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ with $\mathcal{J}_k = \{x_1, x_2\}$ and $\mathcal{U}_k = \{x_3, x_4\}$.

is zero otherwise. The behavior of the model in an experiment $\mathcal{E}_k$ is then given by the structural equations

$$\mathbf{x} := \mathbf{U}_k\mathbf{B}\mathbf{x} + \mathbf{U}_k\mathbf{e} + \mathbf{c}. \tag{4}$$

For an intervened variable $x_j \in \mathcal{J}_k$, the manipulated model in Equation 4 replaces the original equation $x_j := \sum_{i \in \mathrm{pa}(j)} b_{ji} x_i + e_j$ with the equation $x_j := c_j$, while the equations for passively observed variables $x_u \in \mathcal{U}_k$ remain unchanged.

Here the intervention vector $\mathbf{c}$ is constant throughout the equilibrating process, holding the intervened variables fixed at the values sampled from the intervention distribution. A different approach could consider interventions that only "shock" the system initially, and then allow the intervened variables to fluctuate. This would require a different representation and analysis from the one we provide here.

As in the passive observational setting discussed in Section 2.1, we have to ensure that the time series representation of the *experimental* setting

$$\mathbf{x}(t) := \mathbf{U}_k\mathbf{B}\mathbf{x}(t-1) + \mathbf{U}_k\mathbf{e} + \mathbf{c}$$

is guaranteed to converge to an equilibrium as $t \to \infty$, where both $\mathbf{c}$ and $\mathbf{e}$ are time-invariant. We do so by extending the assumption that guarantees convergence in the passive observational setting to *all* experimental settings.

Definition 3 (Asymptotic Stability) *A linear cyclic model with latent variables* $(\mathbf{B}, \mathbf{\Sigma_e})$ *is* asymptotically stable *if and only if for every possible experiment* $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$*, the eigenvalues* λ_i *of the matrix* $\mathbf{U}_k\mathbf{B}$ *satisfy* $\forall i : |\lambda_i| < 1$.

Asymptotic stability implies that in an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ the samples we obtain at equilibrium are given by $\mathbf{x} = (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\mathbf{U}_k\mathbf{e} + \mathbf{c})$. Note that the passive observational case is included in terms of the null-experiment where $\mathcal{J}_k$ is empty. In practice, the assumption of asymptotic stability

implies that the system under investigation will not break down or explode under *any* intervention, so the equilibrium distributions are well defined for all circumstances. Obviously, this will not be true for many real feedback systems, and in fact the assumption can be weakened for our purposes. However, as we discuss in more detail in Section 2.3, the assumption of an underlying generating model that satisfies asymptotic stability simplifies the interpretation of our results. For an acyclic model $(\mathbf{B}, \mathbf{\Sigma_e})$ all eigenvalues of all matrices $\mathbf{U}_k\mathbf{B}$ are zero, so the stability condition is in this case trivially fulfilled.

In general, experiments can take many forms: Apart from varying several rather than just one variable at the same time, the interventions on the variables can be independent from one another, or correlated, with different means and variances for each intervened variable. To simplify notation for the remainder of this paper, we will adopt a standardized notion of an experiment:

Definition 4 (Canonical Experiment) *An experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ is said to be a* canonical experiment *if the intervened variables in $\mathcal{J}_k$ are randomized surgically and uncorrelated with the disturbances and with each other, with zero mean and unit variance.*

This notational simplification makes the partition into intervened and passively observed variables the only parameter specifying an experiment, and allows us to derive the theory purely in terms of the covariance matrices $\mathbf{C}^k_\mathbf{x}$ of an experiment. The following lemma shows that we can make the assumption of uncorrelated components of $\mathbf{c}$ without loss of generality. First, however, we need one additional piece of notation: For any $(n \times n)$-matrix $\mathbf{A}$, we denote by $\mathbf{A}_{\mathcal{S}_r\mathcal{S}_c}$ the block of $\mathbf{A}$ that remains after deleting the rows corresponding to variables in $\mathcal{V} \setminus \mathcal{S}_r$ and columns corresponding to variables in $\mathcal{V} \setminus \mathcal{S}_c$, keeping the order of the remaining rows and columns unchanged.

Lemma 5 (Correlated Experiment) *If in an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$, where intervention variables $\mathbf{c}$ are randomized*[1] *independently of the disturbances $\mathbf{e}$ such that $E(\mathbf{c}) = \boldsymbol{\mu}^k_\mathbf{c}$ and $cov(\mathbf{c}) = \mathbf{\Sigma}^k_\mathbf{c}$, a linear cyclic model with latent variables $(\mathbf{B}, \mathbf{\Sigma_e})$ produces mean $\tilde{\boldsymbol{\mu}}^k_\mathbf{x}$ and covariance matrix $\tilde{\mathbf{C}}^k_\mathbf{x}$, then in a canonical experiment where intervention variables $\mathbf{c}$ are randomized independently of $\mathbf{e}$ with $E(\mathbf{c}) = \mathbf{0}$ and $cov(\mathbf{c}) = \mathbf{J}_k$, the model produces observations with mean and covariance given by*

$$\boldsymbol{\mu}^k_\mathbf{x} = \mathbf{0}, \tag{5}$$

$$\mathbf{C}^k_\mathbf{x} = \tilde{\mathbf{C}}^k_\mathbf{x} - \tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k}(\tilde{\mathbf{T}}^k_\mathbf{x})^T + \tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{T}}^k_\mathbf{x})^T, \tag{6}$$

where $\tilde{\mathbf{T}}^k_\mathbf{x} = (\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{V}\mathcal{J}_k}((\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k})^{-1}$.

Proof To improve readability, proofs for all lemmas and theorems in this paper are deferred to the appendix.

The lemma shows that whenever in an actual experiment the values given to the intervened variables are not mutually uncorrelated, we can easily convert the estimated mean and covariance matrix to a standardized form that would have been found, had the interventions been uncorrelated with zero mean and unit variance.[2] The substantive assumption is that the values of the intervened

1. Randomization implies here that the covariance matrix of the intervention variables $\text{cov}(\mathbf{c}_{\mathcal{J}_k}) = (\mathbf{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}$ is symmetric positive-definite.
2. The lemma should come as no surprise to readers familiar with multiple linear regression: The $[\bullet, j]$-entries of the matrix $\mathbf{T}^k_\mathbf{x}$ are the regression coefficients when x_j is regressed over the intervened variables. The regressors do not have to be uncorrelated to obtain unbiased estimates of the coefficients.

variables (the components of $\mathbf{c}$) are uncorrelated with the disturbances (the components of $\mathbf{e}$). This excludes so-called 'conditional interventions' where the values of the intervened variables depend on particular observations of other (passively observed) variables in the system. We take this to be an acceptably weak restriction.

Mirroring the derivation in Section 2.1, in a *canonical* experiment $\mathcal{E}_k$ the mean and covariance are given by:

$$\boldsymbol{\mu}_{\mathbf{x}}^k = \mathbf{0}, \tag{7}$$

$$\mathbf{C}_{\mathbf{x}}^k = (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\boldsymbol{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}. \tag{8}$$

We can now focus on analyzing the covariance matrix obtained from a *canonical* experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ on a *canonical* model $(\mathbf{B}, \boldsymbol{\Sigma}_{\mathbf{e}})$. For notational simplicity we assume without loss of generality that variables $x_1, \cdots, x_j \in \mathcal{J}_k$ are intervened on and variables $x_{j+1}, \cdots, x_n \in \mathcal{U}_k$ are passively observed. The covariance matrix for this experiment then has the block form

$$\mathbf{C}_{\mathbf{x}}^k = \begin{bmatrix} \mathbf{I} & (\mathbf{T}_{\mathbf{x}}^k)^T \\ \mathbf{T}_{\mathbf{x}}^k & (\mathbf{C}_{\mathbf{x}}^k)_{\mathcal{U}_k\mathcal{U}_k} \end{bmatrix}, \tag{9}$$

where

$$\begin{aligned} \mathbf{T}_{\mathbf{x}}^k &= (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k}, \\ (\mathbf{C}_{\mathbf{x}}^k)_{\mathcal{U}_k\mathcal{U}_k} &= (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}(\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k}(\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k})^T + (\boldsymbol{\Sigma}_{\mathbf{e}})_{\mathcal{U}_k\mathcal{U}_k})\,(\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-T}. \end{aligned}$$

The upper left hand block is the identity matrix $\mathbf{I}$, since in a canonical experiment the intervened variables are randomized independently with unit variance. We will consider the more complicated lower right hand block of covariances between the passively observed variables in Section 3.2. The lower left hand block $\mathbf{T}_x^k$ consists of covariances that represent the so-called *experimental effects* of the intervened $x_i \in \mathcal{J}_k$ on the passively observed $x_u \in \mathcal{U}_k$. An experimental effect $t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ is the overall causal effect of a variable x_i on a variable x_u in the experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$; it corresponds to the coefficient of x_i when x_u is regressed on the set of intervened variables in this experiment. If only variable x_i is intervened on in the experiment, then the experimental effect $t(x_i \rightsquigarrow x_u || \{x_i\})$ is standardly called the *total effect* and denoted simply as $t(x_i \rightsquigarrow x_u)$. If all observed variables except for x_u are intervened on, then an experimental effect is called a *direct effect*: $t(x_i \rightsquigarrow x_u || \mathcal{V} \setminus \{x_u\}) = b(x_i \to x_u) = (\mathbf{B})_{ui} = b_{ui}$.

The covariance between two variables can be computed by so called 'trek-rules'. Some form of these rules dates back to the method of path analysis in Wright (1934). In our case, these trek-rules imply that the experimental effect $t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ can be expressed as the sum of contributions by all directed paths starting at x_i and ending in x_u in the manipulated graph, denoted by the set $\mathcal{P}(x_i \rightsquigarrow x_u || \mathcal{J}_k)$. The contribution of each path $p \in \mathcal{P}(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ is determined by the product of the coefficients b_{ml} associated with the edges $x_l \to x_m$ on the path, as formalized by the following formula

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k) = \sum_{p \in \mathcal{P}(x_i \rightsquigarrow x_u || \mathcal{J}_k)} \prod_{(x_l \to x_m) \in p} b_{ml},$$

where the product is taken over all edges $x_l \to x_m$ on the path p. The full derivation of this formula is presented in Appendix C (see also Equation 12a in Mason, 1956).

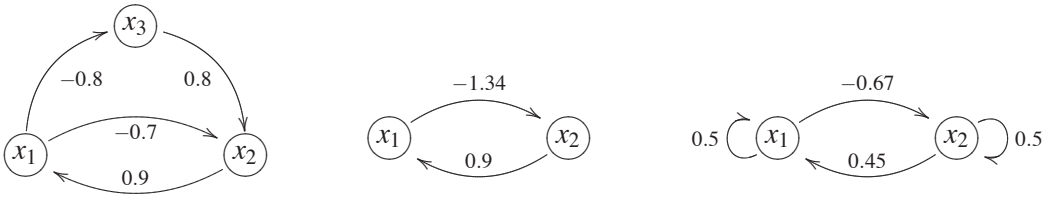


Figure 4: Left: The original asymptotically stable model. Center: The marginalized model that is only weakly stable. Right: A marginalized model with self cycles that is asymptotically stable.

If the model includes cycles, there will be an infinite number of directed paths from one variable to the other. In the example model of Figure 3, the experimental effects can be calculated using the trek-rules as follows:

$$t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) = (b_{31} + b_{41}b_{34})(1 + b_{43}b_{34} + (b_{43}b_{34})^2 + \cdots) = \frac{b_{31} + b_{41}b_{34}}{1 - b_{43}b_{34}}, \quad (10)$$

$$t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) = (b_{41} + b_{31}b_{43})(1 + b_{43}b_{34} + (b_{43}b_{34})^2 + \cdots) = \frac{b_{41} + b_{31}b_{43}}{1 - b_{43}b_{34}}. \quad (11)$$

The convergence of the geometric series is guaranteed by the assumption of asymptotic stability for the experiment $\mathcal{J}_k = \{x_1, x_2\}$, which ensures that the (only) non-zero eigenvalue $\lambda = b_{43}b_{34}$ satisfies $|\lambda| < 1$.

Note that the experimental effects are unaffected by the latent confounding. Since the interventions break any incoming arrows on the intervened variables, this independence also follows directly from the graphical d-separation criterion extended to cyclic graphs (Spirtes, 1995): In Figure 3, variables x_1 and x_3 are not d-connected by any of the undirected paths through the double headed arrows.

2.3 Marginalization

One of the key features of linear structural equation models with correlated errors is that the model family is closed under marginalization. That is, if instead of the original variable set $\mathcal{V}$ we only have access to a subset $\tilde{\mathcal{V}} \subset \mathcal{V}$ of variables, then if the original model $(\mathbf{B}, \mathbf{\Sigma_e})$ is in the model family, then the marginalized model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_\mathbf{e})$ over $\tilde{\mathcal{V}}$ is in the family, too. Any directed paths through marginalized variables are transformed into directed edges in $\tilde{\mathbf{B}}$, and any confounding effect of the marginalized variables is integrated into the covariance matrix $\tilde{\mathbf{\Sigma}}_\mathbf{e}$ of the disturbances.

For example, in Figure 4 on the left we show the graph structure and the edge coefficients of an asymptotically stable model $(\mathbf{B}, \mathbf{\Sigma_e})$ over the variables $\mathcal{V} = \{x_1, x_2, x_3\}$. For the purpose of argument, assume that variable x_3 is not observed. We thus want to describe a marginalized model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_\mathbf{e})$ over just the variables $\tilde{\mathcal{V}} = \{x_1, x_2\}$. Critically, the two models should produce the same observations with respect to the variables x_1 and x_2 in both the passive observational setting and in any experiment intervening on $\{x_1\}$, $\{x_2\}$, or $\{x_1, x_2\}$. In other words, the marginalized model should be such that any observations on $\tilde{\mathcal{V}}$ coincides with those obtained from the original model in all experiments that can be performed in both. Thus, in the experiment intervening on x_1, the experimental

effect $t(x_1 \rightsquigarrow x_2 || \{x_1\}) = -0.7 - 0.8 \cdot 0.8 = -1.34$ of the original model should equal the corresponding experimental effect of the marginalized model. If we do not want to add any additional self-cycles, the only possibility is to set $\tilde{b}_{21} = -1.34$. Similarly, we set $\tilde{b}_{12} = t(x_2 \rightsquigarrow x_1 || \{x_2\}) = 0.9$. This gives the model of Figure 4 (center).

Note, however, that while the original model was asymptotically stable (as can easily be seen by computing the eigenvalues of $\mathbf{B}$), the marginalized canonical model is *not* asymptotically stable, as $\tilde{\mathbf{B}}$ has an eigenvalue that is larger than 1 in absolute value. We thus see that when relevant variables are not included in the analysis, asymptotic stability may not hold under marginalization. Fortunately, it turns out that for our purposes of identification a much weaker assumption is sufficient. We term this assumption *weak stability*:

Definition 6 (Weak Stability) *A linear cyclic causal model with latent variables* $(\mathbf{B}, \mathbf{\Sigma_e})$ *is* weakly stable *if and only if for every experiment* $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$*, the matrix* $\mathbf{I} - \mathbf{U}_k\mathbf{B}$ *is invertible*.

Note that the invertibility of matrix $\mathbf{I} - \mathbf{U}_k\mathbf{B}$ is equivalent to matrix $\mathbf{U}_k\mathbf{B}$ not having any eigenvalues equal to exactly 1. (Complex-valued eigenvalues with modulus 1 are allowed as long as the eigenvalue in question is not exactly $1 + 0i$.) Any asymptotically stable model is therefore by definition also weakly stable.

We noted earlier that asymptotic stability is an unnecessarily strong assumption for our context. In fact, weak stability is all that is *mathematically* required for all the theory presented in this article. However, while mathematically expedient, weak stability alone can lead to interpretational ambiguities: Under the time series interpretation of a cyclic model that we presented in Equation 2, a weakly stable model that is not asymptotically stable will fail to have an equilibrium distribution for one or more experiments. While Figure 4 illustrates that asymptotic stability may be lost when marginalizing hidden variables, one cannot in general know whether a learned model that is not asymptotically stable for some experiments corresponds to such an unproblematic case, or whether the underlying system truly is unstable under those experiments.

For the remainder of this article, to ensure a consistent interpretation of any learned model, we assume that there is a true *underlying* asymptotically stable data generating model, possibly including hidden variables—thereby guaranteeing well-defined equilibrium distributions for all experiments. The interpretation of any learned weakly stable model $(\mathbf{B}, \mathbf{\Sigma_e})$ is then only that the distribution *over the observed variables* produced at equilibrium by the true underlying asymptotically stable model has mean and covariance as described by Equations 7 and 8.[3] All equations derived for asymptotically stable models carry over to weakly stable models.[4] In the following two Lemmas, we give the details of how the canonical model over the observed variables is related to the original linear cyclic model in the case of hidden variables and self-cycles (respectively).

The marginalized model of any given linear structural equation model with latent variables can be obtained with the help of the following Lemma.

Lemma 7 (Marginalization) *Let* $(\mathbf{B}, \mathbf{\Sigma_e})$ *be a weakly stable linear cyclic model over the variables* $\mathcal{V}$*, with latent variables. Let* $\mathcal{M} \subset \mathcal{V}$ *denote the set of marginalized variables. Then the marginal-*

3. Alternatively, one could avoid making this assumption of asymptotic stability of the underlying model, but in that case the predictions of the outcomes of experiments must be conditional on the experiments in question resulting in equilibrium distributions.

4. The sums of divergent geometric series can be evaluated by essentially extending the summing formula $\sum_{i=0}^{\infty} b^i = \frac{1}{1-b}$ to apply also when $b > 1$ (Hardy, 1949).

ized model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ *over variables* $\tilde{\mathcal{V}} = \mathcal{V} \setminus \mathcal{M}$ *defined by*

$$\begin{aligned} \tilde{\mathbf{B}} &= \mathbf{B}_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}} + \mathbf{B}_{\tilde{\mathcal{V}}\mathcal{M}}(\mathbf{I} - \mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}, \\ \tilde{\mathbf{\Sigma}}_{\mathbf{e}} &= (\mathbf{I} - \tilde{\mathbf{B}}) \left[(\mathbf{I} - \mathbf{B})^{-1} \mathbf{\Sigma}_{\mathbf{e}} (\mathbf{I} - \mathbf{B})^{-T} \right]_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}} (\mathbf{I} - \tilde{\mathbf{B}})^{T} \end{aligned}$$

is also a weakly stable linear cyclic causal model with latent variables. The marginalized covariance matrix of the original model and the covariance matrix of the marginalized model are equal in any experiments where any subset of the variables in $\tilde{\mathcal{V}}$ *are intervened on.*

The expressions for $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{\Sigma}}_{\mathbf{e}}$ have simple intuitive explanations. First, the coefficient matrix $\tilde{\mathbf{B}}$ of the marginalized model is given by the existing coefficients between the variables in $\tilde{\mathcal{V}}$ in the original model plus any paths in the original model from variables in $\tilde{\mathcal{V}}$ through variables in $\mathcal{M}$ and back to variables in $\tilde{\mathcal{V}}$. Second, the disturbance covariance matrix $\tilde{\mathbf{\Sigma}}_{\mathbf{e}}$ for the marginalized model is obtained by taking the observed covariances over the variables in $\tilde{\mathcal{V}}$ and accounting for the causal effects among the variables in $\tilde{\mathcal{V}}$, so as to ensure that the resulting covariances in the marginal model equal those of the original model in any experiment.

In addition to marginalizing unobserved variables, we may be interested in deriving the canonical model (i.e., without self-loops) from an arbitrary linear cyclic model with self-loops. This is possible with the following lemma.

Lemma 8 (Self Cycles) *Let* $\mathbf{U}_i$ *be an* $(n \times n)$*-matrix that is all zero except for the element* $(\mathbf{U}_i)_{ii} = 1$*. For a weakly stable model* $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$ *containing a self-loop for variable* x_i *with coefficient* b_{ii}*, we can define a model without that self-loop given by*

$$\begin{aligned} \tilde{\mathbf{B}} &= \mathbf{B} - \frac{b_{ii}}{1 - b_{ii}} \mathbf{U}_i (\mathbf{I} - \mathbf{B}), \\ \tilde{\mathbf{\Sigma}}_{\mathbf{e}} &= (\mathbf{I} + \frac{b_{ii}}{1 - b_{ii}} \mathbf{U}_i) \mathbf{\Sigma}_{\mathbf{e}} (\mathbf{I} + \frac{b_{ii}}{1 - b_{ii}} \mathbf{U}_i)^{T}. \end{aligned}$$

The resulting model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ *is also weakly stable and yields the same observations at equilibrium in all experiments.*

Figure 5 shows explicitly the relation of edge strengths in the two models of the lemma. Since we are only rescaling some of the coefficients, the graph structure of the model stays intact, except for the deleted self-loop. The structure of the covariance matrix $\mathbf{\Sigma}_{\mathbf{e}}$ also remains unchanged, with only the ith row and the ith column rescaled. For a model $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$ with several self-loops we can apply Lemma 8 repeatedly to obtain a model without any self-loops, which is equivalent to the original model in the sense that it yields the same equilibrium data as the original model for all experiments.

Note that, as with marginalization, the standardization by removal of self-cycles may produce a canonical model that is only weakly stable, and not asymptotically stable, even if the original model was asymptotically stable.

Ultimately, self-loops affect the speed and path to convergence to the equilibrium, but not the equilibrium itself. Our approach will not yield any insight on self-loops, because we do not address the causal process in a time series. However, the indeterminacy regarding self-loops also means that any predictions at equilibrium are not affected by the learned model being represented in canonical form, that is, without the possibly existing self-loops. So, although self-loops are not strictly forbidden for the data generating model, we can present the theory in the following sections entirely in terms of models without them.

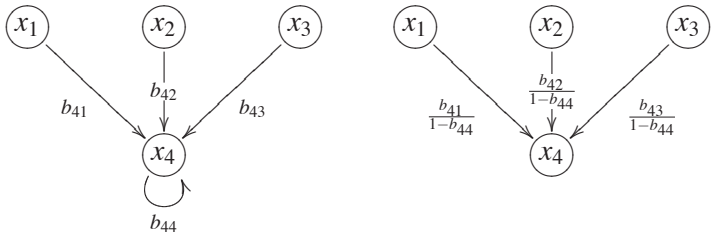


Figure 5: Perturbation of coefficients from a model with self-loops (on the left) to a model without self-loops (on the right). The two models are indistinguishable from equilibrium data.

3. Identifiability

The full characterization of the model under passive observational and experimental circumstances now allows us to specify conditions (on the set of experiments) that are sufficient (Section 3.1) and necessary (Section 3.2) to identify the model parameters. Throughout, for purposes of full identification (uniqueness of the solution) and notational simplicity, we assume that in each experiment we observe the covariance matrix in the infinite sample limit as described by Equation 8, and that both the underlying model and all experiments are canonical. For reasons discussed in the previous section we also assume that there is an underlying generating model that is asymptotically stable, even though the marginalized parts of the model we observe may only be weakly stable. Readers who are primarily interested in the learning algorithm we have developed can skip to Section 4 and return to the identifiability conditions of this section when required.

3.1 Sufficiency

Going back to our four variable example in Figure 3, in which x_1 and x_2 are subject to intervention, we already derived in Equations 10 and 11 the experimental effects $t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\})$ and $t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\})$ using the trek-rules. Taken together, these equations imply the following

$$\begin{aligned} t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) &= b_{31} + t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) b_{34} \\ &= t(x_1 \rightsquigarrow x_3 || \{x_1, x_2, x_4\}) + t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) t(x_4 \rightsquigarrow x_3 || \{x_1, x_2, x_4\}). \end{aligned} \tag{12}$$

Note that Equation 12 relates the experimental effects of intervening on $\{x_1, x_2\}$ to the experimental effects of intervening on $\{x_1, x_2, x_4\}$. It shows that the experimental effect $t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\})$ can be calculated by separating the single path *not* going through x_4 (with contribution b_{31}) from the remaining paths that all go through x_4. The last edge on these paths is always $x_4 \to x_3$. The total contribution of the paths through x_4 is therefore the product $t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) b_{34}$.

Equation 12 illustrates two separate but related approaches to identifying the full model parameters from a set of measured experimental effects: On the one hand, it provides an example of how experimental effects from one set of experiments can be used to identify experimental effects of a novel experiment (not in the existing set). Thus, if we had a set of experiments that allowed us to infer all the experimental effects of all the experiments that intervene on all but one variable, then we would have determined all the direct effects and would thereby have identified the **B**-matrix. On the other hand, Equation 12 shows how the measured experimental effects can be used to construct

linear constraints on the (unknown) direct effects b_{ji}. Thus, if we had a set of experiments that supplies constraints that would be sufficient for us to solve for all the direct effects, then we would again be able to identify the **B**-matrix. In either case, the question crucial for identifiability is: Which sets of experiments produce experimental effects that are sufficient to identify the model? Unsurprisingly, the answer is the same for both cases. For reasons of simplicity, we present the identifiability proof in this section in terms of the first approach. We use the second approach, involving a system of linear constraints, for the learning algorithm in Section 4.

The example in Equation 12 can be generalized in the following way: As stated earlier, for an asymptotically stable model, the experimental effect $t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ of $x_i \in \mathcal{J}_k$ on $x_u \in \mathcal{U}_k$ in experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ is the sum-product of coefficients on all directed paths from x_i to x_u. We can calculate the sum-product in two parts with respect to an observed variable $x_j \in \mathcal{U}_k$. First we consider all the paths that *do not* go through x_j. The sum-product of all those paths is equal to the experimental effect $t(x_i \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\})$, since all paths through x_j are intercepted by additionally intervening on x_j. Second, the remaining paths are all of the form $x_i \rightsquigarrow \tilde{x}_j \rightsquigarrow x_u$, where $\tilde{x}_j$ is the last occurrence of x_j on the path (recall that paths may contain cycles, so there may be multiple occurrences of x_j on the path). The sum-product of coefficients on all subpaths $x_i \rightsquigarrow \tilde{x}_j$ is given by $t(x_i \rightsquigarrow x_j || \mathcal{J}_k)$ and the sum-product of coefficients on all subpaths $\tilde{x}_j \rightsquigarrow x_u$ is $t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\})$. Taking all combinations of subpaths $x_i \rightsquigarrow \tilde{x}_j$ and $\tilde{x}_j \rightsquigarrow x_u$, we obtain the contribution of all the paths through x_j as the product $t(x_i \rightsquigarrow x_j || \mathcal{J}_k) t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\})$. We thus obtain

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k) \quad = \quad t(x_i \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\}) + t(x_i \rightsquigarrow x_j || \mathcal{J}_k) t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\}). \tag{13}$$

This equation is derived formally in Appendix F, where it is also shown that it holds for all weakly stable models (not only asymptotically stable models).

We now show that equations of the above type from two different experiments can be combined to determine the experimental effects of a novel third experiment. Consider for example the model in Figure 2 over variables $\mathcal{V} = \{x_1, x_2, x_3, x_4\}$. Say, we have conducted two single-intervention experiments $\mathcal{E}_1 = (\mathcal{J}_1, \mathcal{U}_1) = (\{x_1\}, \{x_2, x_3, x_4\})$ and $\mathcal{E}_2 = (\{x_2\}, \{x_1, x_3, x_4\})$. By making the following substitutions in Equation 13 for each experiment, respectively,

$$\begin{array}{cc} \mathcal{J}_k := \mathcal{J}_1 = \{x_1\} & \mathcal{J}_k := \mathcal{J}_2 = \{x_2\} \\ x_i := x_1 & x_i := x_2 \\ x_j := x_2 & x_j := x_1 \\ x_u := x_3 & x_u := x_3 \end{array}$$

we get two equations relating the experimental effects in the original two experiments to some experimental effects of the *union* experiment $\mathcal{E}_3 = (\{x_1, x_2\}, \{x_3, x_4\})$ (we denote it as the "union" experiment because $\mathcal{J}_3 = \mathcal{J}_1 \cup \mathcal{J}_2$):

$$\begin{bmatrix} 1 & t(x_1 \rightsquigarrow x_2 || \{x_1\}) \\ t(x_2 \rightsquigarrow x_1) || \{x_2\}) & 1 \end{bmatrix} \begin{bmatrix} t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) \\ t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) \end{bmatrix} \quad = \quad \begin{bmatrix} t(x_1 \rightsquigarrow x_3 || \{x_1\}) \\ t(x_2 \rightsquigarrow x_3 || \{x_2\}) \end{bmatrix}.$$

In the above equation, the quantities in the matrix on the left, and the elements of the vector on the right-hand-side, are experimental effects that are available from the experimental data. The unknown quantities are in the vector on the left-hand-side. Now, if the matrix on the left is invertible, we can directly solve for the experimental effects of the third experiment just from the experimental effects in the first two. (Similar equations hold for other experimental effects as well). The

following lemma shows that the matrix is invertible when the weak stability condition holds, and that in general, from experimental effects observed in two experiments, we can always estimate the experimental effects in their union and in their intersection experiments.

Lemma 9 (Union/Intersection Experiment) *For a weakly stable canonical model the experimental effects in two experiments $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ and $\mathcal{E}_l = (\mathcal{J}_l, \mathcal{U}_l)$ determine the experimental effects in*

- *the* union *experiment* $\mathcal{E}_{k\cup l} = (\mathcal{J}_k \cup \mathcal{J}_l,\ \mathcal{U}_k \cap \mathcal{U}_l)$*, and*
- *the* intersection *experiment* $\mathcal{E}_{k\cap l} = (\mathcal{J}_k \cap \mathcal{J}_l,\ \mathcal{U}_k \cup \mathcal{U}_l)$.

Since there are no experimental effects in experiments intervening on $\emptyset$ or $\mathcal{V}$, the experimental effects are considered to be determined trivially in those cases. In the case of union experiments, also the full covariance matrix $\mathbf{C}_{\mathbf{x}}^{k\cup l}$ of the experiment can be determined. For intersection experiments, $\mathbf{C}_{\mathbf{x}}^{k\cap l}$ can be fully determined if passive observational data is available (see Appendix J).

In a canonical model the coefficients $b(\bullet \to x_u)$ on the arcs into variable x_u (the direct effects of the other variables on that variable) are equal to the experimental effects when intervening on everything except x_u, that is, $b(\bullet \to x_u) = t(\bullet\rightsquigarrow x_u || \mathcal{V} \setminus \{x_u\})$. So in order to determine particular direct effects, it is sufficient to ensure that a given set of experiments provides the basis to apply Lemma 9 repeatedly so as to obtain the experimental effects of the experiments that intervene on all but one variable. In our example with four variables, we can first use Lemma 9 to calculate the experimental effects when intervening on $\{x_1\} \cup \{x_2\} = \{x_1, x_2\}$ (as suggested above), and given a further experiment that intervenes only on x_4, we can then determine the experimental effects of an experiment intervening on $\{x_1, x_2\} \cup \{x_4\} = \{x_1, x_2, x_4\}$. The experimental effects we obtain constitute the direct effects $b(\bullet \to x_3)$. Hence, if single-intervention experiments are available for each variable it is easy to see that all direct effects of the model are identified using the lemma.

What then is the general condition on the set of experiments such that we can derive all possible direct effects by iteratively applying Lemma 9? It turns out that we can determine all direct effects if the following *pair condition* is satisfied for all ordered pairs of variables.

Definition 10 (Pair Condition) *A set of experiments $\{\mathcal{E}_k\}_{k=1,\dots,K}$ satisfies the pair condition for an ordered pair of variables $(x_i, x_u) \in \mathcal{V} \times \mathcal{V}$ (with $x_i \neq x_u$) whenever there is an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ in $\{\mathcal{E}_k\}_{k=1,\dots,K}$ such that $x_i \in \mathcal{J}_k$ (x_i is intervened on) and $x_u \in \mathcal{U}_k$ (x_u is passively observed).*

It is not difficult to see that the pair condition holding for all ordered pairs of variables is sufficient to identify $\mathbf{B}$. Consider one variable x_u. From a set of experiments satisfying the pair condition for all ordered pairs, we can find for all $x_i \neq x_u$ an experiment satisfying the pair condition for the pair (x_i, x_u). We refer to such an experiment as $\tilde{\mathcal{E}}_i = (\tilde{\mathcal{J}}_i, \tilde{\mathcal{U}}_i)$ in the following. Now, by iteratively using Lemma 9, we can determine the experimental effects in the union experiment $\tilde{\mathcal{E}}_\cup = (\tilde{\mathcal{J}}_\cup, \tilde{\mathcal{U}}_\cup)$ of experiments $\{\tilde{\mathcal{E}}_i\}_{i\neq u}$, where variables in set $\tilde{\mathcal{J}}_\cup = \bigcup_{i\neq u} \tilde{\mathcal{J}}_i$ are intervened on. Each x_i was intervened on at least in one experiment, thus $\forall i \neq u : x_i \in \tilde{\mathcal{J}}_\cup$. Variable x_u was passively observed in each experiment, thus $x_u \notin \tilde{\mathcal{J}}_\cup$. The experimental effects of this union experiment intervening on $\tilde{\mathcal{J}}_\cup = \mathcal{V} \setminus \{x_u\}$ are thus the direct effects $b(\bullet \to x_u)$. Repeating the same procedure for each $x_u \in \mathcal{V}$ allows us to identify all direct effects.

Thus, if the pair condition is satisfied for all ordered pairs, we can determine all elements of $\mathbf{B}$, and only the covariance matrix $\mathbf{\Sigma_e}$ of the disturbances remains to be determined. The passive observational data covariance matrix $\mathbf{C_x^0}$ can be estimated from a null-experiment $\mathcal{E}_0 = (\emptyset, \mathcal{V})$. Given $\mathbf{B}$ and $\mathbf{C_x^0}$ we can solve for $\mathbf{\Sigma_e}$ using Equation 3:

$$\mathbf{\Sigma_e} \quad = \quad (\mathbf{I}-\mathbf{B})\mathbf{C_x^0}(\mathbf{I}-\mathbf{B})^T. \tag{14}$$

If there is no null-experiment, then the block $(\mathbf{\Sigma_e})_{\mathcal{U}_k,\mathcal{U}_k}$ of the covariance matrix can instead be determined from the covariance matrix in any experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ using Equation 8:

$$(\mathbf{\Sigma_e})_{\mathcal{U}_k\mathcal{U}_k} \quad = \quad [(\mathbf{I}-\mathbf{U}_k\mathbf{B})\mathbf{C}_\mathbf{x}^k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^T]_{\mathcal{U}_k\mathcal{U}_k}. \tag{15}$$

Consequently, given $\mathbf{B}$, we can determine $(\mathbf{\Sigma_e})_{ij} = \sigma_{ij}$ if the following covariance condition is met.

Definition 11 (Covariance Condition) *A set of experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ satisfies the covariance condition for an unordered pair of variables $\{x_i, x_j\} \subseteq \mathcal{V}$ whenever there is an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ in $(\mathcal{E}_k)_{k=1,\ldots,K}$ such that $x_i \in \mathcal{U}_k$ and $x_j \in \mathcal{U}_k$, that is, both variables are passively observed.*

Similarly to the pair condition, if we know $\mathbf{B}$, and if the covariance condition is satisfied for all pairs of variables, we can identify all covariances in $\mathbf{\Sigma_e}$. Notice that the variances $(\mathbf{\Sigma_e})_{ii}$ can be determined since the assumption includes that each variable x_i must be passively observed at least in one experiment.

Putting the results together we get a sufficient identifiability condition for a canonical model:

Theorem 12 (Identifiability–Sufficiency) *Given canonical experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ a weakly stable canonical model $(\mathbf{B}, \mathbf{\Sigma_e})$ over the variables $\mathcal{V}$ is identifiable if the set of experiments satisfies the pair condition for each ordered pair of variables $(x_i, x_j) \in \mathcal{V} \times \mathcal{V}$ (with $x_i \neq x_j$) and the covariance condition for each unordered pair of variables $\{x_i, x_j\} \subseteq \mathcal{V}$.*

The identifiability condition is satisfied for our four-variable case in Figure 2 by, for example, a set of experiments intervening on $\{x_1, x_2\}, \{x_2, x_4\}, \{x_1, x_4\}$ and $\{x_3\}$. Obviously, a full set of single-intervention experiments or a full set of all-but-one experiments together with a passive observational data set would also do. We return to this issue in Section 4.2.

3.2 Necessity

To show that the conditions of Theorem 12 are not only sufficient but in fact also necessary for identifiability, we consider what happens when the pair condition or the covariance condition is not satisfied for some variable pair. Since the covariance condition only ensures the identifiability of $\mathbf{\Sigma_e}$ when $\mathbf{B}$ is already identified, we start with the more fundamental *pair condition*.

Consider the two models in Figure 6. The models differ in their parameters, and even in their structure, yet produce the same observations in all experiments that do not satisfy the pair condition for the (ordered) pair (x_2, x_4). That is, for any experiment (including a passive observation), for which it is *not the case* that $x_2 \in \mathcal{J}_k$ and $x_4 \in \mathcal{U}_k$, the two models are indistinguishable, despite the fact that for an experiment that *satisfies* the pair condition for (x_2, x_4), the two models will in general have different experimental effects (due to the difference in the direct effect b_{42}). Since the effect due to b_{42} cannot be isolated in the left model without satisfying the pair condition for the pair

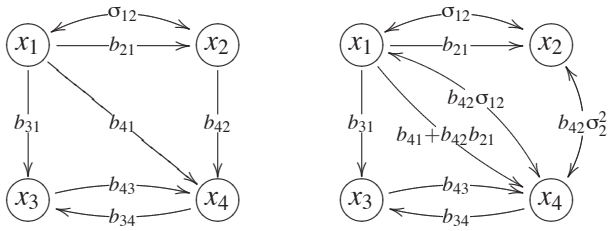


Figure 6: Underdetermination of the model. On the left: the data generating model $(\mathbf{B}, \mathbf{\Sigma_e})$. On the right: a model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_\mathbf{e})$ producing the same observations in all experiments not satisfying the pair condition for the ordered pair (x_2, x_4).

(x_2, x_4), its effect can be accounted for elsewhere in the right model, for example, the effect of the missing path $x_1 \to x_2 \to x_4$ is accounted for in the model on the right by the perturbed coefficient $b_{41} + b_{42}b_{21}$ on the arc $x_1 \to x_4$.

The $\tilde{\mathbf{B}}$-matrix for the model on the right was constructed from the one on the left by perturbing the coefficient b_{42} corresponding to the pair (x_2, x_4), for which the pair condition is not satisfied. The perturbation corresponds to setting $\delta := -b_{42}$ in the following lemma.

Lemma 13 (Perturbation of B) *Let* $\mathbf{B}$ *be the coefficient matrix of a weakly stable canonical model over* $\mathcal{V}$ *and let* $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ *be a set of experiments on* $\mathbf{B}$ *that does not satisfy the pair condition for some pair* (x_i, x_j)*. Denote the sets* $\mathcal{K} = \mathcal{V} \setminus \{x_i, x_j\}$ *and* $\mathcal{L} = \{x_i, x_j\}$*. Then a model with coefficient matrix* $\tilde{\mathbf{B}}$ *defined by*

$$\tilde{\mathbf{B}}_{\mathcal{K}\mathcal{V}} = \mathbf{B}_{\mathcal{K}\mathcal{V}}, \quad \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}} = \begin{bmatrix} 0 & b_{ij} \\ b_{ji} + \delta & 0 \end{bmatrix}, \quad \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{K}} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})(\mathbf{I} - \mathbf{B}_{\mathcal{L}\mathcal{L}})^{-1}\mathbf{B}_{\mathcal{L}\mathcal{K}}$$

will produce the same experimental effects *as* $\mathbf{B}$ *for* any *experiment that does not satisfy the pair condition for the pair* (x_i, x_j)*. The free parameter* δ *must be chosen such that* $\tilde{\mathbf{B}}$ *is weakly stable.*

Lemma 13 shows that if the pair condition is not satisfied for the pair (x_i, x_j), then b_{ji} cannot be identified on the basis of the measured experimental effects. As in our example, it is generally the case that for $\delta \neq 0$ the models $\mathbf{B}$ and $\tilde{\mathbf{B}}$ will produce different experimental effects in any experiment that satisfies the pair condition for the pair (x_i, x_j). The choice of δ is not crucial, since most choices will produce a weakly stable perturbed model.

To see the effect of the perturbation more clearly, we can write it explicitly as follows:

$$\begin{aligned}
\forall l \neq j,\ \forall k : \tilde{b}_{lk} &= b_{lk}, && \text{(no changes to any edges that do not end in } x_j\text{)} \\
\tilde{b}_{ji} &= b_{ji} + \delta, && \text{(perturb the direct effect of } x_i \text{ on } x_j \text{ by } \delta\text{)} \\
\tilde{b}_{jj} &= 0, && \text{(no self-loop at } x_j\text{)} \\
\forall k \notin \{i, j\} :\ \tilde{b}_{jk} &= b_{jk} - \delta \frac{b_{ik} + b_{ij}b_{jk}}{1 - b_{ij}b_{ji}}. && \text{(needed adjustments to incoming arcs to } x_j\text{)}
\end{aligned}$$

The above form makes it clear that if the pair condition is not satisfied for the pair (x_i, x_j), in general all coefficients on the jth row of $\mathbf{B}$ may be unidentified as well. Hence, to guarantee the

identifiability of coefficient b_{ji} we must have the pair condition satisfied for all pairs $(\bullet, x_j)$. In Figure 6 the coefficient b_{42} is unidentified because the pair condition for the pair (x_2, x_4) is not satisfied. But as a result, b_{41} is also unidentified. Nevertheless, in this particular example, the coefficient b_{43} happens to be identified, because of the structure of the graph.

If the pair condition is not satisfied for several pairs, then Lemma 13 can be applied iteratively for each missing pair to arrive at a model with different coefficients, that produces the same experimental effects as the original for all experiments not satisfying the pairs in question. Each missing pair adds an additional degree of freedom to the system.

We emphasize that Lemma 13 only ensures that the *experimental effects* of the original and perturbed model are the same. However, the following lemma shows that the covariance matrix of disturbances can always be perturbed such that the two models become completely indistinguishable for any experiment that does not satisfy the pair condition for some pair (x_i, x_j), as was the case in Figure 6.

Lemma 14 (Perturbation of $\boldsymbol{\Sigma_e}$) *Let the true model generating the data be $(\mathbf{B}, \boldsymbol{\Sigma}_\mathbf{e})$. For each of the experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$, let the obtained data covariance matrix be $\mathbf{C}^k_\mathbf{x}$. If there exists a coefficient matrix $\tilde{\mathbf{B}} \neq \mathbf{B}$ such that for all $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ and all $x_i \in \mathcal{I}_k$ and $x_j \in \mathcal{U}_k$ it produces the same experimental effects $t(x_i \rightsquigarrow x_j \,||\, \mathcal{I}_k)$, then the model $(\tilde{\mathbf{B}}, \tilde{\boldsymbol{\Sigma}}_\mathbf{e})$ with $\tilde{\boldsymbol{\Sigma}}_\mathbf{e} = (\mathbf{I} - \tilde{\mathbf{B}})(\mathbf{I} - \mathbf{B})^{-1} \boldsymbol{\Sigma}_\mathbf{e} (\mathbf{I} - \mathbf{B})^{-T} (\mathbf{I} - \tilde{\mathbf{B}})^T$ produces data covariance matrices $\tilde{\mathbf{C}}^k_\mathbf{x} = \mathbf{C}^k_\mathbf{x}$ for all $k = 1, \ldots, K$.*

Lemma 14, in combination with Lemma 13, shows that for identifiability the pair condition must be satisfied for all pairs. If the pair condition is not satisfied for some pair, then an alternative model (distinct from the true underlying model) can be constructed (using the two lemmas) which produces the exact same covariance matrices $\mathbf{C}^k_\mathbf{x}$ for all the available experiments. In Figure 6, the effect of the missing link $x_2 \to x_4$ is imitated by the additional covariance $b_{42}\sigma_2^2$ between e_2 and e_4 and by the covariance $b_{42}\sigma_{12}$ between e_1 and e_4.

The result implies that identifying the coefficient matrix $\mathbf{B}$ exclusively on the basis of constraints based on *experimental effects* already fully exploits the information summarized by the second order statistics. The covariances between the passively observed variables (corresponding to the lower right hand block in Equation 9) do not provide any further information. We thus obtain the result:

Theorem 15 (Completeness) *Given the covariance matrices in a set of experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ over the variables in $\mathcal{V}$, all coefficients $b(x_i \to x_j)$ of a weakly stable canonical model are identified if and only if the pair condition is satisfied for all ordered pairs of variables with respect to these experiments.*

Intuitively, the covariances between the passively observed variables do not help in identifying the coefficients $\mathbf{B}$ because they also depend on the unknowns $\boldsymbol{\Sigma}_\mathbf{e}$, and the additional unknowns swamp the gains of the additional covariance measures.

If $\mathbf{B}$ is known or the pair condition is satisfied for all pairs, but the *covariance condition* is not satisfied for a pair $\{x_i, x_j\}$, then in general the covariance σ_{ij} cannot be identified: In all the manipulated graphs of the experiments the arc $x_i \leftrightarrow x_j$ is cut, and thus σ_{ij} does not affect the data in any way. It follows that the covariance condition is necessary as well. However, unlike for the pair condition, not satisfying the covariance condition for some pair does not affect the identifiability of any of the other covariances.

We can now summarize the previous results in the form of a sufficient and necessary identifiability condition for the full model. Theorem 12 states that satisfying the pair condition and covariance

condition for all pairs is sufficient for model identifiability. Theorem 15 shows that the coefficients cannot be identified if the pair condition is not satisfied for all pairs of variables, and in the previous paragraph we showed that satisfying the covariance condition for all pairs is necessary to identify all covariances and variances of the disturbances. This yields the following main result.

Corollary 16 (Model Identifiability) *The parameters of a weakly stable canonical model* $(\mathbf{B}, \mathbf{\Sigma_e})$ *over the variables in* $\mathcal{V}$ *can be identified if and only if the set of experiments* $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ *satisfies the pair condition for all ordered pairs* $(x_i, x_j) \in \mathcal{V} \times \mathcal{V}$ *(such that* $x_i \neq x_j$*) and the covariance condition for all unordered pairs* $\{x_i, x_j\} \subseteq \mathcal{V}$.

Finally, note that all of our identifiability results and our learning algorithm (Section 4) are solely based on second-order statistics of the data and the stated model space assumptions. No additional background knowledge is included. When the data are multivariate Gaussian, these statistics exhaust the information available, and hence our identifiability conditions are (at least) in this case necessary.

4. Learning Method

In this section, we present an algorithm, termed LLC, for inferring a linear cyclic model with latent variables, provided finite sample data from a set of experiments over the given variable set. Although Lemma 9 (Union/Intersection Experiment) naturally suggests a procedure for model discovery given a set of canonical experiments that satisfy the conditions of Corollary 16 (Model Identifiability), we will pursue a slightly different route in this section. It allows us to not only identify the model when possible, but can also provide a more intuitive representation of the (common) situation when the true model is either over- or underdetermined by the given set of experiments. As before, we will continue to assume that we are considering a set of canonical experiments on a weakly stable canonical model (Definitions 2, 4 and 6). From the discussion in Section 2 it should now be clear that this assumption can be made essentially without loss of generality: Any asymptotically stable model can be converted into a weakly stable canonical model and any experiment can be redescribed as a canonical experiment, as long as the interventions in the original experiment were independent of the disturbances. As presented here, the basic LLC algorithm provides only estimates of the values of all the edge coefficients in $\mathbf{B}$, as well as estimates of the variances and covariances among the disturbances in $\mathbf{\Sigma_e}$. We later discuss how to obtain error estimates for the parameters and how to adapt the basic algorithm to different learning tasks such as structure discovery.

4.1 LLC Algorithm

To illustrate the derivation of the algorithm, we again start with Equation 12, which was derived from the experiment that intervenes on x_1 and x_2 in Figure 3,

$$t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) \quad = \quad b_{31} + t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) b_{34}.$$

This provides a linear constraint of the measured experimental effects $t(x_1 \rightsquigarrow x_j || \{x_1, x_2\})$ on the unknown direct effects b_{31} and b_{34} into x_3. In general, the experimental effects observed in an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ can be used to provide linear constraints on the unknown direct effects that, like Equation 12, have the form

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k) \quad = \quad b_{ui} + \sum_{x_j \in \mathcal{U}_k \setminus \{x_u\}} t(x_i \rightsquigarrow x_j || \mathcal{J}_k) b_{uj}, \tag{16}$$

where $x_i \in \mathcal{J}_k$ and $x_j, x_u \in \mathcal{U}_k$. Analogously to the equations in Section 3.1, for asymptotically stable models Equation 16 is also naturally interpretable in terms of the sum of paths connecting the variables: The experimental effect of x_i on x_u is a sum of the direct effect of x_i on x_u and the effect of each path from x_i to any other $x_j \in \mathcal{U} \setminus \{x_u\}$, multiplied by the direct connection from that x_j to x_u. (Alternatively, one can also see how Equation 16 is reached by iteratively applying Equation 13.)

Since the covariance matrix $\mathbf{C}_{\mathbf{x}}^k$ of an experiment $\mathcal{E}_k$ contains the experimental effects for all pairs (x_i, x_j) with $x_i \in \mathcal{J}_k$ and $x_j \in \mathcal{U}_k$, each experiment generates $m_k = |\mathcal{J}_k| \times |\mathcal{U}_k|$ constraints of the form of Equation 16. For a set of experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$ we can represent the constraints as a system of equations linear in the $(n^2 - n)$ unknown coefficients b_{ji} in $\mathbf{B}$. (Recall that $b_{ii} = 0$ for all i in canonical models.) We thus have a matrix equation

$$\mathbf{T}\mathbf{b} \quad = \quad \mathbf{t}, \tag{17}$$

where $\mathbf{T}$ is a $((\sum_{k=1}^{K} m_k) \times (n^2 - n))$-matrix of (measured) experimental effects, $\mathbf{b}$ is the $(n^2 - n)$-vector of unknown b_{ji} and $\mathbf{t}$ is a $(\sum_{k=1}^{K} m_k)$-ary vector corresponding to the (measured) experimental effects on the left-hand side of Equation 16.

Provided that matrix $\mathbf{T}$ has full column rank, we can solve this system of equations for $\mathbf{b}$ and rearrange $\mathbf{b}$ into $\mathbf{B}$ (including the diagonal of zeros). Since any one constraint (e.g., Equation 16) only includes unknowns of the type $b_{u\bullet}$, corresponding to edge-coefficients for edges into some node $x_u \in \mathcal{U}_k$, we can rearrange the equations such that the system of equations can be presented in the following form

$$\begin{bmatrix} \mathbf{T}_{11} & & & \\ & \mathbf{T}_{22} & & \\ & & \ddots & \\ & & & \mathbf{T}_{nn} \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_n \end{bmatrix} \quad = \quad \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \vdots \\ \mathbf{t}_n \end{bmatrix}, \tag{18}$$

where $\mathbf{T}$ is a block diagonal matrix with all entries outside the blocks equal to zero. Instead of solving the equation system in Equation 17 with $(n^2 - n)$ unknowns, Equation 18 allows us to separate the system into n blocks each constraining direct effects $b_{u\bullet}$ into a different x_u. We can thus separately solve n equation systems $\mathbf{T}_{uu}\mathbf{b}_u = \mathbf{t}_u$ with $(n-1)$ unknowns in each. The matrix $\mathbf{T}$ has full column rank if and only if all $\mathbf{T}_{uu}$ have full column rank as well.

For example, in the case of the experiment intervening on $\mathcal{J}_k = \{x_1, x_2\}$ of the 4-variable model in Figure 3, we obtain the following experimental covariance matrix:

$$\mathbf{C}_{\mathbf{x}}^k = \begin{bmatrix} 1 & 0 & t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) & t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) \\ 0 & 1 & t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) & t(x_2 \rightsquigarrow x_4 || \{x_1, x_2\}) \\ t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) & t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) & \mathrm{var}_k(x_3) & \mathrm{cov}_k(x_3, x_4) \\ t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) & t(x_2 \rightsquigarrow x_4 || \{x_1, x_2\}) & \mathrm{cov}_k(x_3, x_4) & \mathrm{var}_k(x_4) \end{bmatrix}.$$

This covariance matrix allows us to construct the following four linear constraints on the unknown b's:

$$t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) = b_{31} + t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) b_{34}, \tag{19}$$

$$t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) = b_{41} + t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) b_{43}, \tag{20}$$

$$t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) = b_{32} + t(x_2 \rightsquigarrow x_4 || \{x_1, x_2\}) b_{34}, \tag{21}$$

$$t(x_2 \rightsquigarrow x_4 || \{x_1, x_2\}) = b_{42} + t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) b_{43}. \tag{22}$$

If we have a further experiment $\mathcal{E}_l = (\mathcal{J}_l, \mathcal{U}_l)$ with $\mathcal{J}_l = \{x_4\}$ then we obtain the following three additional constraints:

$$t(x_4 \rightsquigarrow x_1 || \{x_4\}) = b_{14} + t(x_4 \rightsquigarrow x_2 || \{x_4\}) b_{12} + t(x_4 \rightsquigarrow x_3 || \{x_4\}) b_{13}, \quad (23)$$

$$t(x_4 \rightsquigarrow x_2 || \{x_4\}) = b_{24} + t(x_4 \rightsquigarrow x_1 || \{x_4\}) b_{21} + t(x_4 \rightsquigarrow x_3 || \{x_4\}) b_{23}, \quad (24)$$

$$t(x_4 \rightsquigarrow x_3 || \{x_4\}) = b_{34} + t(x_4 \rightsquigarrow x_1 || \{x_4\}) b_{31} + t(x_4 \rightsquigarrow x_2 || \{x_4\}) b_{32}. \quad (25)$$

Converting the Equations 19-25 to the form of the Equation 18, we see that Equations 19, 21 and 25 become part of $\mathbf{T}_{33}$, while Equations 20 and 22 become part of $\mathbf{T}_{44}$, and the remaining Equations 23 and 24 become part of $\mathbf{T}_{11}$ and $\mathbf{T}_{22}$, respectively. We will focus on $\mathbf{T}_{33}$ consisting of Equations 19, 21 and 25:

$$\begin{aligned} \mathbf{T}_{33}\mathbf{b}_3 &= \begin{bmatrix} 1 & 0 & t(x_1 \rightsquigarrow x_4 || \{x_1, x_2\}) \\ 0 & 1 & t(x_2 \rightsquigarrow x_4 || \{x_1, x_2\}) \\ t(x_4 \rightsquigarrow x_1 || \{x_4\}) & t(x_4 \rightsquigarrow x_2 || \{x_4\}) & 1 \end{bmatrix} \begin{bmatrix} b_{31} \\ b_{32} \\ b_{34} \end{bmatrix} \\ &= \begin{bmatrix} t(x_1 \rightsquigarrow x_3 || \{x_1, x_2\}) \\ t(x_2 \rightsquigarrow x_3 || \{x_1, x_2\}) \\ t(x_4 \rightsquigarrow x_3 || \{x_4\}) \end{bmatrix} = \mathbf{t}_3. \end{aligned}$$

Given Lemma 9 (Union/Intersection Experiment) it should now be clear that the experimental effects of experiments $\mathcal{E}_k$ and $\mathcal{E}_l$ are sufficient to determine the experimental effects of an experiment intervening on $\mathcal{J} = \mathcal{V} \setminus \{x_3\}$, which would directly specify the values for b_{31}, b_{32} and b_{34}. Unsurprisingly, the matrix $\mathbf{T}_{33}$ is invertible and the coefficients b_{31}, b_{32} and b_{34} can be solved also from the above equation system. In Appendix K we show formally that when the pair condition is satisfied for *all* ordered pairs, then $\mathbf{T}$ has full column rank.

Once we have obtained $\mathbf{B}$ using the above method, the covariance matrix $\mathbf{\Sigma_e}$ can be obtained easily using Equation 14 if a null-experiment $\mathcal{E}_0 = (\emptyset, \mathcal{V})$ is available, or else using Equation 15 in the more general case where only the covariance condition is satisfied for all pairs.

Until now, we have described the algorithm in terms of the covariances and the experimental effects 'observed' in a given experiment. In practice, of course, we only have finite sample data, and the above quantities must be *estimated* from the data, and the estimated covariances and experimental effects do not precisely equal their true underlying values. This naturally has practical ramifications that we describe in the context of the algorithm below.

The LLC algorithm (Algorithm 1), for models that are *linear*, may have *latent* variables and may contain *cycles*, gathers the ideas described so far in this section. It omits all but the most rudimentary handling of the inevitable sampling variability in the estimates. The algorithm minimizes the sum of squared errors in the available linear constraints by solving the equation system using the Moore-Penrose pseudo-inverse. Thus, whenever the linear constraints derived from different experiments are partly conflicting, the algorithm will find a compromise that comes as close as possible to satisfying all the available constraints. Similarly, to improve the statistical estimation of $\mathbf{\Sigma_e}$, we average over all the instances when a particular pair of variables was passively observed. When the covariance condition is not satisfied for a particular pair, then the covariance of the disturbances for that pair remains undefined.

There are several standard modifications that can be made to this basic algorithm in light of statistical variability of the finite sample data. Whenever the sample size differs substantially between experiments, a re-weighting of the constraint equations according to the sample size of the

experiment they were obtained from, favors the more precise constraints. Simple bootstrapping of the observed samples in each experiment separately, can be used to obtain rough estimates of error for the *identified* parameters. In Section 6.2 we calculate a Z-score from these error estimates, which in turn is used for structure discovery. Finally, some form of regularization can help to avoid overfitting (see Sections 6.2 and 6.3). Although we have presented the LLC algorithm here in its stripped down form to illustrate its main contribution, the code implementation[5] provides various options for using these additional features.

When the pair condition is not satisfied for all ordered pairs, then $\mathbf{T}$ does not provide a sufficient set of constraints and the model is underdetermined.[6] Nevertheless, some inferences about the model are still possible. We discuss the details in the following section on underdetermination. For now, note that the algorithm also outputs a list of pairs that satisfy the pair condition, and a list of pairs that satisfy the covariance condition. We will show that these can be used to characterize the underdetermination.

We thus have an algorithm that fully exploits the set of available experiments: When the model identifiability conditions are satisfied it returns an estimate of the true model, when the system is overdetermined it finds a compromise to the available constraints, and when the model is underdetermined we show in the next section what can and cannot be recovered, and how one may proceed in such circumstances.

4.2 Underdetermination

Even when the set of experiments does not satisfy the pair condition for all ordered pairs of variables, the LLC algorithm will nevertheless return a model with estimates for all the coefficients. If there were no sampling errors, one could then check the null-space of the $\mathbf{T}$-matrix to identify which entries of $\mathbf{B}$ are actually underdetermined: An element of $\mathbf{B}$ is determined if and only if it is orthogonal to the null-space of $\mathbf{T}$. In some cases one may find that specific coefficients are determined due to particular values of other coefficients even though that was not clear from the satisfied pair conditions. The coefficient b_{43} in the example in Figure 6 (see the discussion following Lemma 13) is a case in point.

In practice, however, using the null-space to identify the remaining underdetermination can be misleading. The constraints in $\mathbf{T}$ are based on estimates and so its null-space may not correctly identify which coefficients are determined. One can take a more conservative approach and treat any b_{jk} as undetermined for all k whenever there exists an i such that the pair condition is not fulfilled for the ordered pair (x_i, x_j). This follows from the fact that perturbing the model according to Lemma 13 (Perturbation of $\mathbf{B}$) with respect to pair (x_i, x_j), may change all coefficients of the form $b_{j\bullet}$, while leaving the observed experimental effects unchanged. Similarly, the fifth step of the algorithm implements a conservative condition for the identifiability of the covariance matrix: covariance σ_{ij} can be treated as determined if the covariance condition is satisfied for the pair $\{x_i, x_j\}$ *and* the direct effects $\mathbf{B}_{\{x_i,x_j\},\mathcal{V}}$ are determined. Depending on which parameters are identified, Lemma 9 (Union/Intersection Experiment) can be used to make consistent predictions of the

5. Code implementing the learning algorithm is available at `http://www.cs.helsinki.fi/u/ajhyttin/exp/`.

6. Because of statistical variability, $\mathbf{T}$ may well have full rank even in this case, but some of the dimensions it spans only represent errors in the estimates rather than information about the coefficients. See Section 4.2 for details.

Algorithm 1 LLC algorithm

1. Input data from a set of experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$. Initialize matrix $\mathbf{T}$ and vector $\mathbf{t}$ as empty.

2. Using $\{\mathcal{E}_k\}_{k=1,\ldots,K}$, determine which ordered pairs of variables satisfy the pair condition and which pairs of variables satisfy the covariance condition.

3. For each experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$:

   (a) Estimate the covariance matrix $\mathbf{C}_\mathbf{x}^k$.

   (b) From the estimated covariance matrix, extract the experimental effects $t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ for all $(x_i, x_u) \in \mathcal{J}_k \times \mathcal{U}_k$.

   (c) For each pair $(x_i, x_u) \in \mathcal{J}_k \times \mathcal{U}_k$ add an equation

   $$b_{ui} + \sum_{x_j \in \mathcal{U}_k \setminus \{x_u\}} t(x_i \rightsquigarrow x_j || \mathcal{J}_k) b_{uj} \quad = \quad t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$$

   into the system $\mathbf{Tb} = \mathbf{t}$.

4. Solve the equations by $\mathbf{b} = \mathbf{T}^\dagger \mathbf{t}$, where $\mathbf{T}^\dagger$ is the Moore-Penrose pseudo-inverse of $\mathbf{T}$, and rearrange $\mathbf{b}$ to get $\mathbf{B}$.

5. For any pair $\{x_i, x_j\} \subseteq \mathcal{V}$ calculate the covariance of the disturbances as a mean of the covariances estimated in those experiments $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ where $\{x_i, x_j\} \subseteq \mathcal{U}_k$, by

   $$(\mathbf{\Sigma_e})_{ij} \quad = \quad \text{mean}(\{((\mathbf{I} - \mathbf{U}_k\mathbf{B})\mathbf{C}_\mathbf{x}^k(\mathbf{I} - \mathbf{U}_k\mathbf{B})^T)_{ij} \,|\, \{x_i, x_j\} \subseteq \mathcal{U}_k\}),$$

   including variances when $x_i = x_j$. (The mean is undefined for a particular pair if the covariance condition is not satisfied for that pair.)

6. Output the estimated model $(\mathbf{B}, \mathbf{\Sigma_e})$, a list of ordered pairs of variables for which the pair condition is not satisfied, and a list of pairs of variables for which the covariance condition is not satisfied.

experimental effects or the entire covariance matrix for union- or intersection[7] experiments of the available experiments even if the set of experiments does not satisfy the identifiability conditions.

Instead of characterizing the underdetermination, one may consider how to satisfy the model identifiability conditions. There are two general approaches one could pursue. One approach is to strengthen the underlying assumptions, the other to perform additional experiments. Taking the first approach, the additional assumptions may be domain specific or domain general. In econometrics it is common to include background knowledge of the domain that excludes the presence of certain edges, that is, certain edge coefficients are known to be zero. *Faithfulness*, on the other hand, is an assumption we did not make, but that is widely used in causal discovery algorithms (Spirtes et al., 2000). For the linear models we consider here, the assumption of faithfulness requires that a zero-

7. We note that to fully determine the covariance matrix $\mathbf{C}_\mathbf{x}^{k \cap l}$ in an *intersection* experiment, one may require additional passive observational data. See the discussion following Lemma 9.

covariance between two variables entails the absence of a causal connection between the variables. While reasonable for many circumstances, there are well-known cases where faithfulness is not satisfied. For example, if two or more paths between two variables cancel each other out exactly, then one would find a zero-covariance between the variables despite the fact that the variables are (multiply!) causally connected. Moreover, if the data is noisy, a close to unfaithful causal relation may not be distinguishable from an unfaithful one unless a large amount of data or particular experiments are available. Nevertheless, if faithfulness is judged to be a reasonable assumption, then it can provide additional constraints. We have discussed the integration of faithfulness and background knowledge into the current framework in Hyttinen et al. (2010). It remains, however, an open task to develop a procedure for linear cyclic models with latent variables that is *complete* with regard to the additional inferences one can draw on the basis of faithfulness.

If one is able to perform additional experiments, an obvious strategy is to select the next experiment such that it maximizes the number of additional pair conditions that are satisfied. If experiments that intervene on multiple variables simultaneously are taken into consideration, a brute force search for such a best experiment will be exponential in the number of variables. In that case one may consider more efficient selection strategies or heuristics. In most cases any additional experiment will also repeat tests for pairs for which the pair condition is already satisfied. When included in Equation 18, constraints derived from such tests can make the inference more reliable, so one may deliberately select experiments to include particular repeats.

A selection of experiments that is greedy with respect to the satisfaction of additional pair conditions will not necessarily result in the minimum number of experiments overall. For example, if one has six variables $x_1,\ldots,x_6$, and no pair condition has been satisfied so far, that is, no experiment has been performed, then a greedy strategy may recommend a sequence of five intervention sets to fulfill the pair condition for all pairs:

$$\mathcal{J}_1 = \{x_1,x_2,x_3\}, \mathcal{J}_2 = \{x_4,x_5,x_6\}, \mathcal{J}_3 = \{x_1,x_4\}, \mathcal{J}_4 = \{x_2,x_5\}, \mathcal{J}_5 = \{x_3,x_6\}.$$

However, the following four intervention sets are sufficient to satisfy the pair condition for all pairs, but would not be selected by any procedure that is greedy in this respect:

$$\mathcal{J}_1 = \{x_1,x_2,x_3\}, \mathcal{J}_2 = \{x_3,x_4,x_5\}, \mathcal{J}_3 = \{x_5,x_6,x_1\}, \mathcal{J}_4 = \{x_2,x_4,x_6\}.$$

The optimal selection of experiments (given possible background knowledge) is closely related to the theory in combinatorics of finding so-called 'minimal completely separating systems' for directed graphs (see Hyttinen et al., 2012 and Spencer, 1970 for some relevant results). A full discussion here is beyond the scope of this paper.

From a statistical perspective we have found that intervening on more variables simultaneously leads to a higher accuracy of the estimates even if the total sample size across all experiments is maintained constant (Eberhardt et al., 2010). That is, for two sets of experiments that each satisfy the pair condition for all pairs of variables (e.g., the set of four experiments on six variables above versus a set of six experiments each intervening on a single variable), the sequence of experiments intervening on multiple variables simultaneously will provide a better estimate of the underlying model even if the total sample size is the same.

5. Simulations

We compared the performance of the LLC-algorithm against well-known learning algorithms able to exploit experimental data. Since there is no competing procedure that applies directly to the

search space including cyclic *and* latent variable models, we chose for our comparison two procedures that could easily be adapted to the experimental setting and that would provide a good contrast to illustrate the performance of LLC under different model space assumptions. As baseline we used the learning procedure by Geiger and Heckerman (1994) for acyclic Bayesian networks with linear Gaussian conditional probability distributions, referred to as GH. Experimental data is incorporated into the calculation of the local scores in GH using the technique described by Cooper and Yoo (1999). Given that GH assumes acyclicity and causal sufficiency (the absence of latent confounding), it provides a useful basis to assess the increased difficulty of the task when these assumptions are dropped. We also compare to an algorithm for learning Directed Cyclic Graphical models (DCG, Schmidt and Murphy, 2009), designed for discrete cyclic causal models without latent confounding. In this model, the passively observed distribution is represented as a globally normalized product of potentials

$$P(x_1,\ldots,x_n) = \frac{1}{Z}\prod_{i=1}^{n}\phi(x_i;x_{\mathrm{pa}(i)}),$$

where Z is a global normalizing constant. By using unnormalized potentials instead of normalized conditional probability distributions, cycles are allowed in the graph structure. Experimental data is then modeled by simply dropping the potentials corresponding to manipulated variables from the expression, resulting in a manipulated distribution, such as, for example,

$$P(x_2,\ldots,x_n||x_1) = \frac{1}{Z'}\prod_{i=2}^{n}\phi(x_i;x_{\mathrm{pa}(i)}),$$

with a new normalizing constant Z'. Schmidt and Murphy (2009) use potentials of the form $\phi(x_i;x_{\mathrm{pa}(i)}) = \exp(b_i(x_i) + \sum_{j\in\mathrm{pa}(i)} w_{ij}(x_i,x_j))$ to model discrete data and learn the model by maximizing the penalized likelihood function using numerical optimization techniques. To fit this approach we discretized the continuous data (at the very end of the data-generating process) to binary data using 0 as threshold value. While the DCG model may be useful in analyzing cyclic systems under intervention, one should note that the underlying causal generative process is not very clear. Certainly, our data generating processes do not in general yield distributions that fit the model family of DCG.

At first glance, it would appear natural to consider two further procedures for comparison: the Cyclic Causal Discovery algorithm (CCD, Richardson, 1996) that allows for cycles but not latent variables, and the Fast Causal Inference algorithm (FCI, Spirtes et al., 2000) that allows for latents but not for cycles. Both are based on conditional independence tests and return equivalence classes of causal models. However, while background knowledge can be integrated into both procedures to learn from a single experimental data set, it is not clear how (possibly conflicting) results from different experiments should be combined. Identifying the appropriate combining procedure for these algorithms would thus require a separate analysis. The approach by Claassen and Heskes (2010) provides some steps in this direction with regard to FCI, but their framework does not quite fit our context since in their framework the interventions are not targeted at particular variables. We considered a comparison with the recent proposal by Itani et al. (2008), but as of this writing no fully automated procedure was available to the present authors.

To compare the LLC- with the GH- and DCG-algorithms we considered models under five different conditions:

1. linear acyclic models without latent variables,
2. linear cyclic models without latent variables,
3. linear acyclic models with latent variables,
4. linear cyclic models with latent variables, and
5. non-linear acyclic models without latent variables.

For each condition we randomly generated 20 causal models with 10 observed variables each. In the underlying graphs each node had 0-3 parents. In models with latent variables, there were 5 additional latent variables, exogenous to the 10 observed variables. The structural equations were of the form

$$x_j \quad := \quad \sum_{i \in \mathrm{pa}(j)} (b_{ji} x_i + a_{ji} x_i^2) + e_j,$$

where[8] $e_j \sim N(0, \sigma_j^2)$, $b_{ji} \sim \pm\mathrm{Unif}(0.2, 0.8)$ and $a_{ji} = 0$ except for the fifth condition with non-linear models where $a_{ji} \sim \mathrm{Unif}(-0.2, 0.2)$. For the second and fourth condition we sampled until we obtained models that contained at least one cycle. From each model we collected samples in the passive observational setting (null experiment) and in ten additional experiments, each intervening on a single (but different) variable. The intervened variables were always randomized to a normal distribution with zero mean and unit variance. The total number of samples (1,000 to 100,000) were divided evenly among the 11 different experiments, so that adjustments to account for the fact that one experiment may provide more accurate estimates than another were unnecessary. Note that the described set of experiments satisfies the identifiability condition for the LLC-method in Theorem 12 (Identifiability–Sufficiency).

There are a variety of ways to assess the output of the algorithms. Given that every test condition violates at least one of the assumptions of one of the algorithms being tested, we decided against a direct comparison of the quantitative output of each procedure. Instead we used the same qualitative measure that is applied in the cellular network inference challenge that we consider as a case study in Section 6. Following Stolovitzky et al. (2009), the simulations were designed such that each method was required to output a list of all possible edges among the observed variables, sorted in decreasing order of confidence that an edge is in the true graph. To this end, we adapted the three algorithms in the following way. For LLC, the edges were simply ranked from highest to lowest according to the absolute value of their learned coefficients in $\mathbf{B}$. Although the magnitude of a coefficient does not directly represent the confidence in the presence of the edge, we found empirically that it worked quite well in the simulations. (See Section 6 for an alternative approach based on resampling.) For GH, we calculated the marginal edge probabilities over all DAG structures (with an in-degree bound of 3) using the dynamic programming algorithm of Koivisto and Sood (2004), thus obtaining a score for the confidence in each possible edge. Given that DCG uses binary variables, each edge is associated with four weights: $w_{ij}(0,0)$, $w_{ij}(0,1)$, $w_{ij}(1,0)$ and $w_{ij}(1,1)$. Since the weights were penalized (with regularization parameter λ), an edge $x_j \to x_i$ is absent whenever the four associated weights are zero. Following Schmidt and Murphy (2009), we used the L^2-norm of the weights for each edge to determine its strength and hence its rank. As with LLC, this seemed to work well to generate the order.

8. Although the disturbances e_j are uncorrelated in the data generating model, the disturbances of the learned model are in fact correlated when some of the original variables are considered unobserved.

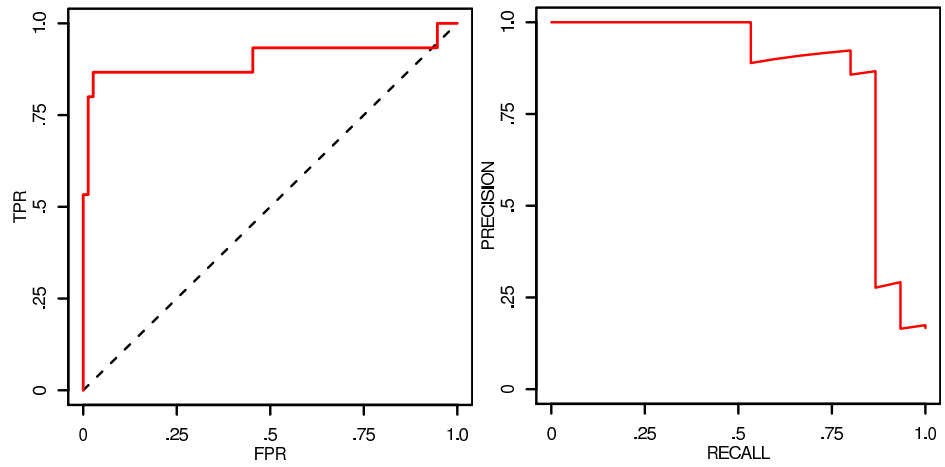


Figure 7: Examples of ROC- (left) and PR-curves (right) of the output of LLC run on 1,000 samples evenly divided over 11 experiments on a linear acyclic model without latents (condition 1).

Given the ordered lists of all possible edges, we can obtain a binary prediction for the presence or absence of an individual edge by simply defining a threshold above which edges would be predicted to be present. These binary predictions can then be compared with the ground truth of the underlying model. However, since the selection of the threshold is to some extent arbitrary (and requires domain specific knowledge of the general sparsity of the generating models), we follow the common approach of reporting Receiver Operating Characteristic (ROC) curves and Precision Recall (PR) curves, and areas under these curves, as explained below. This evaluation of the simulations is also consistent with the evaluation of the case study in Section 6.

A ROC-curve (Figure 7, left) is drawn by plotting the true positive rate (TPR) against the false positive rate (FPR) for different values of the threshold score, where

$$\begin{aligned} \text{TPR} &= \frac{\text{\# edges correctly predicted to be present}}{\text{\# edges in generating model}}, \\ \text{FPR} &= \frac{\text{\# edges incorrectly predicted to be present}}{\text{\# edges not in generating model}}. \end{aligned}$$

The ROC-curve for a powerful classification method should reach close to the top left corner (perfect classification) for some threshold value of the score, while classifying at random would result in the dashed curve in Figure 7. The area under the ROC-curve (AUROC) is often used as a simple one-figure score to assess the power of a classification algorithm. When discovering causal edges in our setting, the AUROC-value specifies the probability that a random edge present in the true model will obtain a higher score than a random absent edge. The AUROC-value usually ranges from 0.5 (random classification) to 1.0 (perfect classification).

Another measure of the quality of search algorithms examines the trade-off between Precision and Recall on a PR-curve (Figure 7, right), where

$$\begin{aligned} \text{Precision} &= \frac{\text{\# edges correctly predicted to be present}}{\text{\# edges predicted to be present}}, \\ \text{Recall} &= \frac{\text{\# edges correctly predicted to be present}}{\text{\# edges in generating model}}. \end{aligned}$$

A perfect classification algorithm should have a precision of 1 for all recall values. The area under the PR-curve (AUPR) specifies the average precision over different threshold values of the score, and can range from 0.0 to 1.0 (perfect classification).

Figure 8 shows the results of our simulations. For DCG we ran the algorithm with several regularization parameter values ($\lambda = 2^8, 2^7, \ldots, 2^{-7}, 2^{-8}$), and always report the best AUROC- and AUPR-score. LLC and GH are run without any further tuning. In the first condition (linear acyclic models without latents), all methods seem to learn the correct causal structure as the sample size increases. For small sample sizes the GH approach benefits from the use of Bayesian priors. Such priors could also be added to the LLC-algorithm, if better performance is needed for very low sample sizes. In the other conditions GH does not achieve good results even with large sample sizes. The performance of GH actually tends to get worse with increasing sample size because the method starts adding incorrect edges to account for measured correlations that cannot be fit otherwise, since the generating model is not included in the restricted model class GH uses. In contrast, LLC suffers at low sample sizes at least in part because of the larger model class it considers. In the second (cyclic models without latents), third (acyclic models with latents) and fourth condition (cyclic models with latent variables), both LLC and DCG find quite good estimates of the causal structure, when sufficient samples are available. Some inaccuracies of the DCG-method are due to the discretization of the data. The performance of DCG in the presence of latent confounding is surprisingly good given that the DCG model does not represent latent variables explicitly. The result may also suggest that the dependencies among the observed variables that were due to latent confounding may have been weak compared to the dependencies due to the causal relationships among the observed variables. For the non-linear data condition, the only discrete (and therefore non-linear) method DCG achieves the best results.

Without further adjustments GH and DCG cannot be scaled to larger sample sizes or a large number of variables (n). The super-exponential growth of the number of DAGs currently limits the GH approach to not more than 30-50 variables. Additionally, the calculation of local scores can be time consuming. On the other hand, DCG requires a numerical optimization over $n + 4n(n-1)$ parameters, which is also infeasible for large n.

In its most basic form (i.e., Algorithm 1), the LLC algorithm only requires the straightforward estimation of the covariance matrices and a calculation of a pseudo-inverse for n matrices with a dimensionality of $(n-1) \times (n-1)$ each. Such a procedure, as used in our simulations, can thus scale to a relatively high (e.g., $n = 100$) number of variables. However, as we see in the next section, it may be useful to add regularization to the basic procedure, and one may have to resort to resampling approaches to obtain estimates of the errors in the coefficients, needed to infer which edges are present and which are absent. Such adaptations and extensions of the basic method can, of course, add significantly to the complexity of the method, but may also pay off in terms of a higher accuracy on small sample sizes.

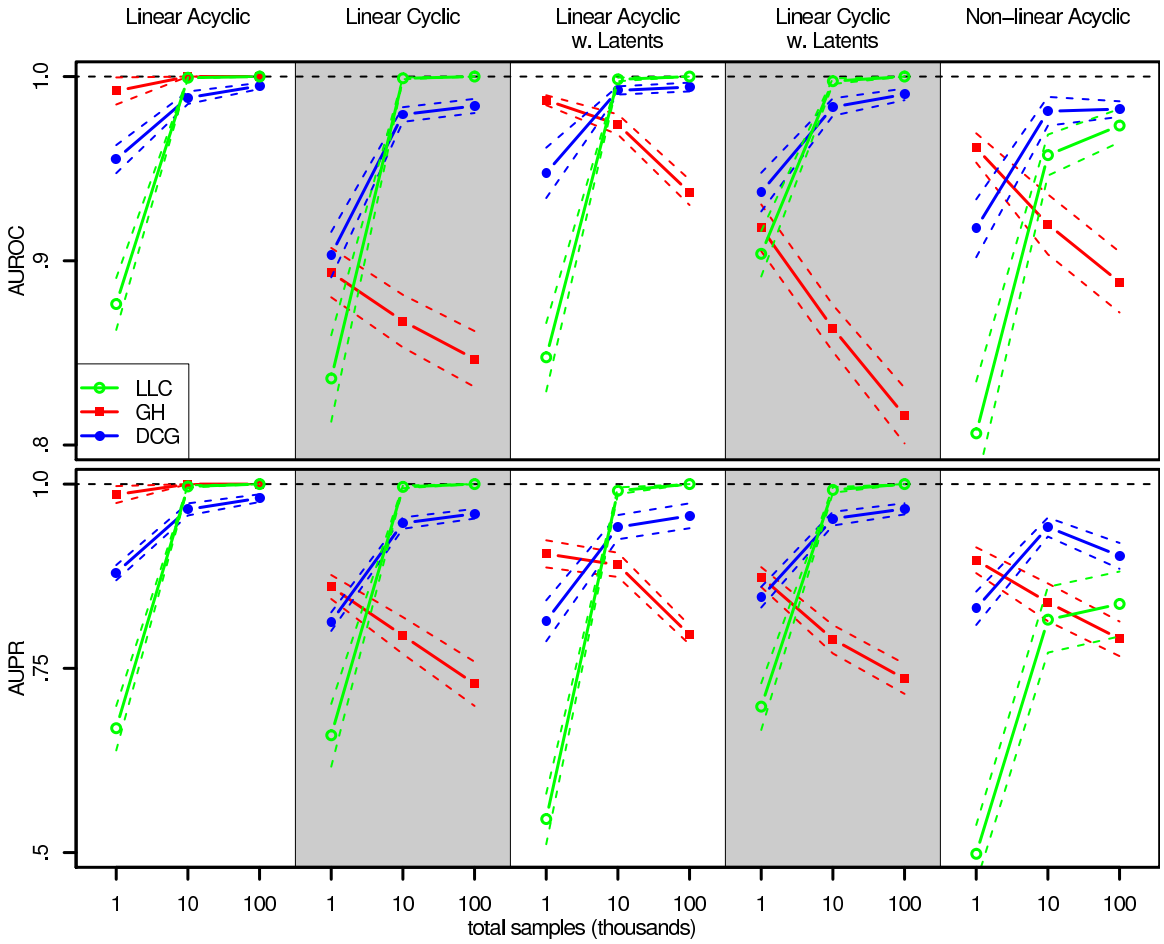


Figure 8: Simulation results: AUROC (top) and AUPR (bottom) values for the LLC-, GH- and DCG-algorithms in the five model conditions (columns, see main text for details) for a total sample size of 1,000-100,000 (x-axis) evenly divided over a passive observation and 10 single intervention experiments. Each point on the solid lines is an average over 20 models with 10 observed variables each, the dashed lines indicate the standard deviation of this average. The light gray shading in this and subsequent figures is used solely for visual distinction.

6. Case Study: DREAM Challenge Data

DREAM (Dialogue for Reverse Engineering Assessments and Methods) is a yearly held challenge for the fair evaluation of strengths and weaknesses of cellular network inference procedures. In this section, we describe how we applied an adapted version of the LLC-method to the *in silico* network challenges of DREAM 3 and DREAM 4, conducted in 2008 and 2009, respectively. The network sizes of the 25 individual models, divided into 5 sub-challenges, ranged from 10 to 100 nodes.

The participants were asked to learn the directed graph structure of a gene regulatory network in different types of cells, from experimental data. Data was *in silico*, or simulated, in order to

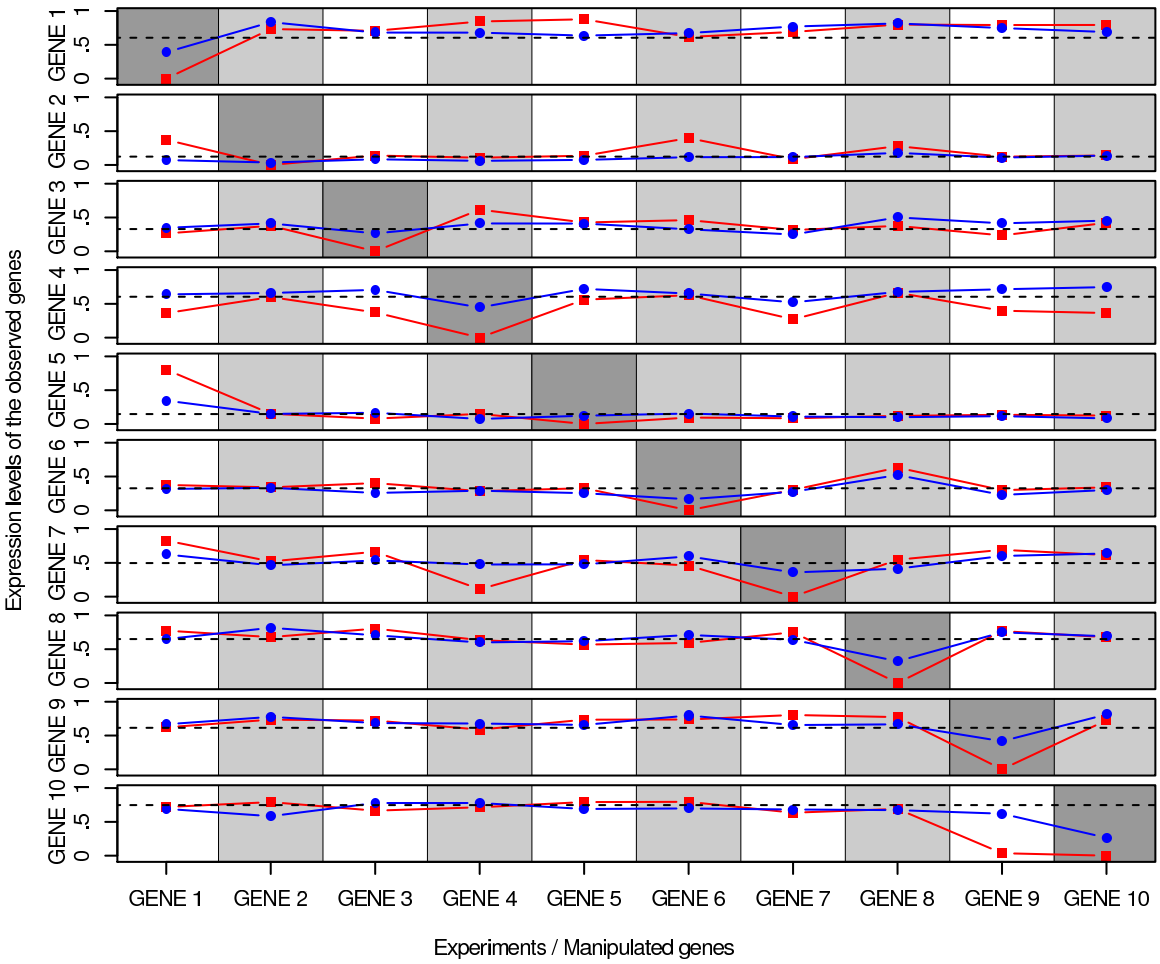


Figure 9: An example of the data provided for one of the 10 variable DREAM network inference challenge. Each row shows the steady state expression levels for each of the 10 genes when the gene indicated on the columns is knocked down (●) or knocked out (■). For each gene, the dashed line indicates the passively observed value. The dark gray shading highlights the diagonal elements, marking the measured levels when intervening on the respective gene. From the 10th row we see that the expression level of the 10th gene responds strongly only to the manipulation of the 9th gene or the 10th gene itself.

have access to the ground truth network structures. The data generating models were designed to be biologically plausible (Marbach et al., 2009) in order to achieve a realistic performance assessment of the network learning algorithms. The networks were based on modules extracted from known biological interaction networks, preserving functional and structural properties of the original networks. Data was then generated simulating a biologically plausible dynamical process and adding noise (Prill et al., 2010).

The data provided to the participants included two measures of the steady states of gene expression levels (the levels converge to these values over time) as mRNA concentrations, in several

different conditions. One data set is visualized in Figure 9. Given that only two data vectors were provided for each condition, GH and DCG, tested in Section 5, are not directly applicable. The challenges also provided several time series of how the modeled cell recovers from a perturbation back to its equilibrium state. We do not include the time series data in our analysis, since LLC (or the other procedures we considered) cannot straightforwardly exploit this data. Each team was supposed to output a confidence measure or a score for their belief in the existence of each possible edge in the model. The performance of the learning algorithms was compared using AUROC and AUPR scores for a single data set (Stolovitzky et al., 2009), in the same manner as explained in Section 5. Finally, in each sub-challenge of 5 models, the competing teams were compared using a total score averaging the individual network scores over all 5 networks.

Below, we discuss how we adapted LLC so that we could apply it to these challenges, and compare the results we obtained with the scores achieved by the teams that participated in the original challenge.

6.1 Estimating the Total Effects

When gene i is knocked down or knocked out, we can treat the result in our framework as an outcome of an experiment where variable x_i is intervened on. However, the DREAM data provides only the steady state values of the expression levels, and not the full covariance matrices. We can still find the total effects in the experiments by the following approach. First, we treat the steady state values as the expected values of the variables under the different interventions (or passive observation), rather than as individual samples. Second, the passive observational steady state values are deducted from all the interventional steady state values such that we can assume $E(\mathbf{x}) = \mathbf{0}$ and thus $E(\mathbf{e}) = \mathbf{0}$. Recall that the total effect $t(x_i \rightsquigarrow x_j)$ is just the regression coefficient of x_i when x_j is regressed over the only manipulated variable x_i. Thus, the expected or steady state value of x_j when x_i is manipulated to a value $x_i^{i,ko}$ (knocked out) is simply $t(x_i \rightsquigarrow x_j) \cdot x_i^{i,ko}$. Similar reasoning applies when x_i is manipulated to a value $x_i^{i,kd}$, and so we can estimate $t(x_i \rightsquigarrow x_j)$ by the least squares solution of the equation system:

$$\begin{aligned} t(x_i \rightsquigarrow x_j) \cdot x_i^{i,ko} &= x_j^{i,ko}, \\ t(x_i \rightsquigarrow x_j) \cdot x_i^{i,kd} &= x_j^{i,kd}. \end{aligned}$$

Given that the data set satisfies the pair condition for all ordered pairs, the DREAM experiments fulfill the requirements given in Section 3 for model identifiability and all total effects $t(x_i \rightsquigarrow x_j)$ can be estimated for all pairs (x_i, x_j).

6.2 Network Inference

Given the estimated total effects, we could directly apply the LLC algorithm to estimate the direct effects matrix $\mathbf{B}$. However, we found that to obtain strong results we had to adapt the algorithm in the following way.

First, unlike in the simulations in Section 5, we found that here the absolute value of a coefficient b_{ji} does not provide a good confidence measure for the existence of the edge $x_i \to x_j$, since it does not consider the possibly large variance of the estimate for b_{ji} in any way. As direct re-sampling approaches are not possible with the available data, we created K noisy data sets by adding noise

from a normal distribution with variance $\sigma^2 = 0.1$ to each raw data point. We then estimated the total effects as explained above.

Second, to estimate the direct effects $\mathbf{B}$ we solved the LLC equation system in Equation 17 using an L^1-norm penalization with weight $\lambda = 0.1$. An estimate of the direct effects $\mathbf{B}$ (vectorized as $\mathbf{b}$) from the noisy data set is thus calculated by

$$\min_{\mathbf{b}} \|\mathbf{T}\mathbf{b} - \mathbf{t}\|_{L^2}^2 + \lambda \|\mathbf{b}\|_{L^1}.$$

As explained in Section 4 the estimation can be done by n separate minimization problems. Note that the L^1-norm penalization can be thought of as a prior for sparse structures, in a way somewhat similar to the use of a faithfulness assumption.

Finally, we calculate the Z-scores for each link b_{ji} by

$$Z_{ji} = \text{mean}(\{b_{ji}^k\}_{k=1}^K)/\text{std}(\{b_{ji}^k\}_{k=1}^K).$$

The higher the Z-score the more confident we are of the existence of the edge. Using Z-scores allows for a high score for a small coefficient as long as its estimated variance is small as well.

Figure 10 summarizes the results. The first observation is that the DREAM 4 challenges were more competitive than the DREAM 3 challenges as the variation of the results for the 10 best teams is lower. Our overall ranks in the five challenges are 3rd, 9th, 3rd, 2nd and 10th among the approximately 30 teams that participated in the actual challenges. There is no clear difference in evaluation with either score metric. We take these results to be encouraging, especially since—unlike many other candidates—we did not use the available time series data. How to exploit the time series data remains an open question. The noise in the data, not having access to a sufficient number of samples and the possible non-linearity of the causal relations constitute additional sources of errors.

6.3 Prediction Accuracy

In addition to structure discovery, another important aspect of causal modeling is prediction under previously unseen experimental conditions. Thus, DREAM 4 featured a bonus round for predicting the steady state values of the gene expression levels in novel experimental settings. The data were the same as for the structure discovery challenges. For the five 10-variable models, the teams were asked to predict all steady state expression levels in 5 situations where always a pair of genes is knocked out. For the five 100-variable models predictions were requested for 20 double knockout settings each.

The knocked out values of variables x_i and x_j are defined by the data as $x_i^{i,ko}$ and $x_j^{j,ko}$. We can estimate the values of the variables x_u such that $u \neq i, j$ using the interpretation of the experimental effects as regression coefficients:

$$x_u^{i,j,ko} \quad = \quad t(x_i \rightsquigarrow x_u || \{x_i, x_j\}) \cdot x_i^{i,ko} + t(x_j \rightsquigarrow x_u || \{x_i, x_j\}) \cdot x_j^{j,ko}.$$

Since we can estimate $t(x_i \rightsquigarrow x_k || \{x_i\})$ and $t(x_j \rightsquigarrow x_k || \{x_j\})$ as described in the previous section, we can also estimate the quantities $t(x_i \rightsquigarrow x_k || \{x_i, x_j\})$ and $t(x_j \rightsquigarrow x_k || \{x_i, x_j\})$ using Lemma 9 (Union/Intersection). We solve the linear equation group (Equation 35 in Appendix G) for the experimental effects using an L_2 prior with regularization parameter λ. In other words, we assume that the data generating model is a linear cyclic model with latent variables and we predict the steady

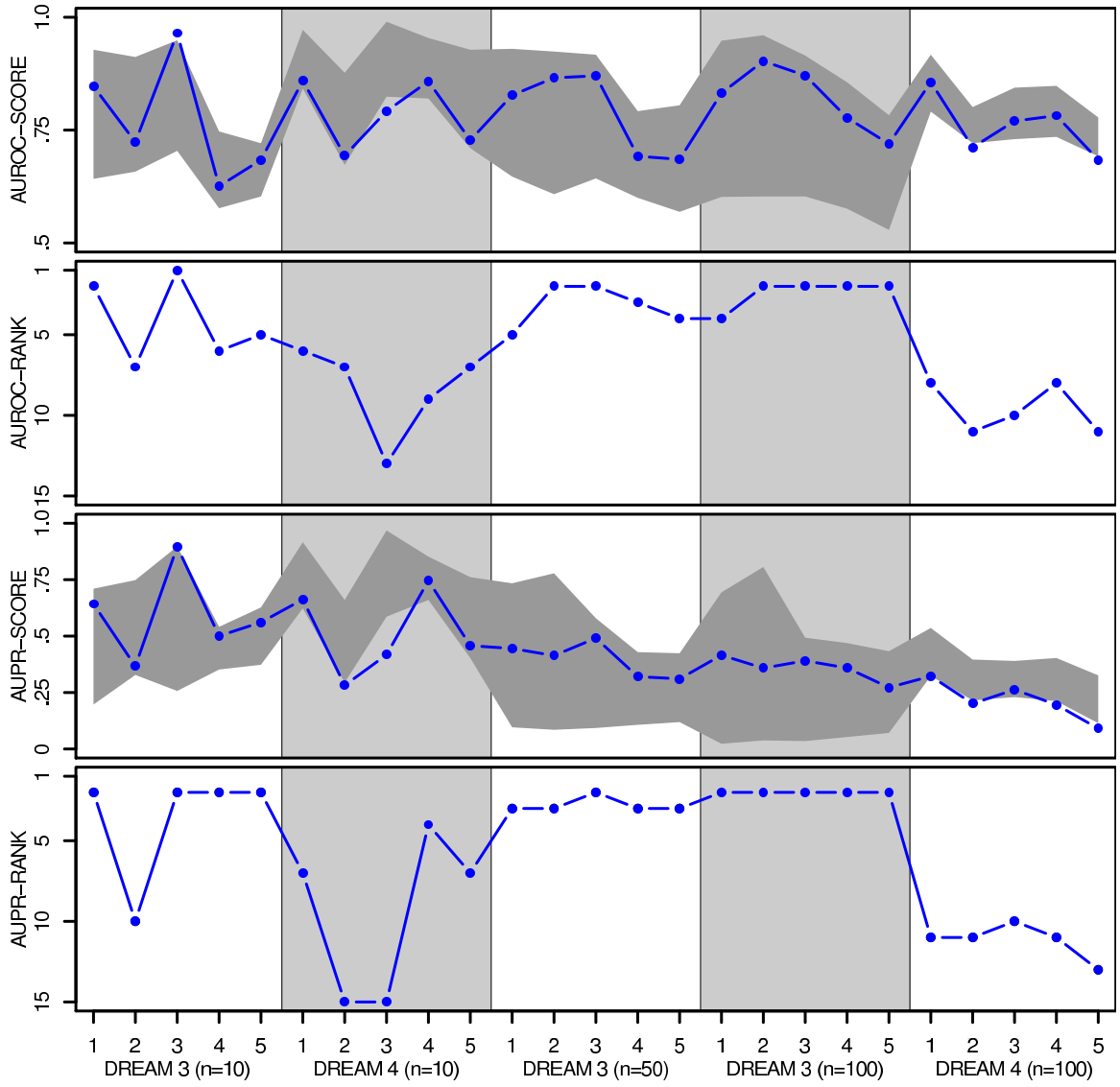


Figure 10: Summary of the results for the DREAM *in silico* network inference challenges: The AUROC- and AUPR-scores (first and third row) and the corresponding rank among the competitors, for each of the DREAM 3 and DREAM 4 challenges. The top of the dark gray area shows the best results among the competing teams for each individual data set, while the bottom always shows the 10th best result. Overall there were about 30 competitors in each of the challenges.

state values of the specific combined (double) knockout experiment on the basis of the relevant single knockout experimental data provided. (The double knockout effects are identified based on the single knockout experimental data by Lemma 9.) In this way, in each individual prediction task we disregard the data that is irrelevant to this specific prediction, and only use the data that is actually

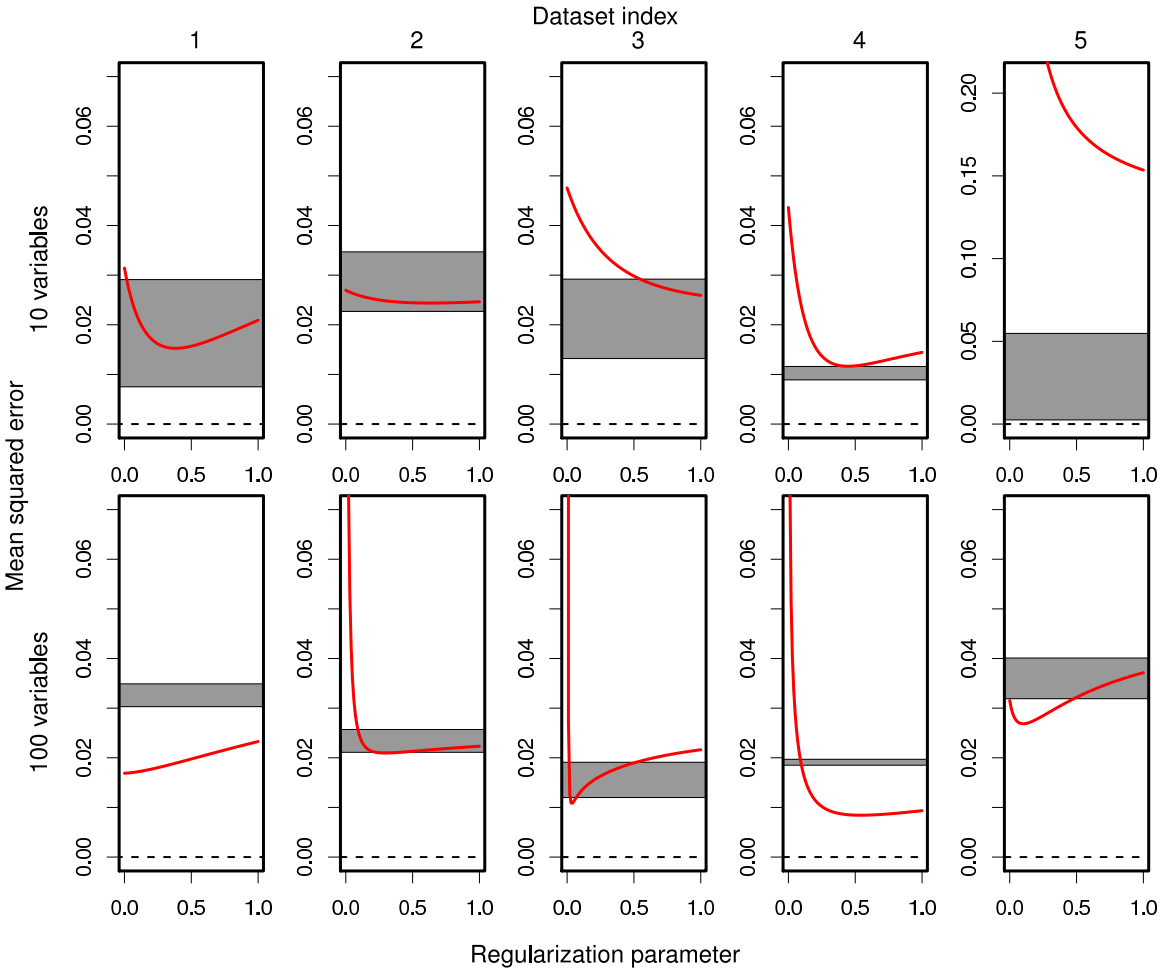


Figure 11: Predictive performance: Mean squared errors of predictions in double intervention experiments on five 10-variable models (top) and 100-variable models (bottom) plotted as a function of the regularization parameter. The red line shows the prediction errors for our procedure. The bottom of the dark gray area shows the best result among the competing teams for each individual data set, while the top always shows the third best result.

relevant. In practice, this means that the predictions are more robust to any slight violations of the modeling assumptions not crucial to the prediction task at hand.

Figure 11 assesses the quality of the predictions. The predictions are compared using the mean squared error from the ground truth, that is, the average sum of squared errors over the variables and over the different predictions requested. For the 10-variable models the results of our procedure are competitive with those of the seven participating teams. For the 100-variable models our procedure achieves in aggregate the best predictions among the five participating teams for a range of the regularization parameter values.

7. Extensions

We have presented and developed the theory in this paper in terms of the standard interpretation of linear non-recursive structural equation models, in which the vector of disturbances $\mathbf{e}$ is held constant throughout the equilibrating process. Following Lauritzen and Richardson (2002) we refer to this most common interpretation of cyclic models as the *deterministic equilibrium* interpretation, since the value of the observed variables $\mathbf{x}$ at equilibrium is a deterministic function of the disturbances $\mathbf{e}$. In this model, as defined in Section 2.1, different observed vectors $\mathbf{x}$ arise solely from different outside influences $\mathbf{e}$, yielding a covariance matrix $\mathbf{C}_\mathbf{x}^k$ for each experiment $\mathcal{E}_k$. In this section we discuss some preliminary ideas for extending the theory to other related linear cyclic models.

In Section 6 we have already seen an application of the method to data in which there is only a single passive-observational data vector $\mathbf{x}_0$ and two experimental data vectors $\mathbf{x}_k^{kd}, \mathbf{x}_k^{ko}$ (corresponding to gene knockdown and knockout experiments, respectively) for each experiment *intervening on a single variable at a time*. In this case, to make the LLC method applicable, one essentially must assume that there is a single (constant) disturbance vector $\mathbf{e}$ that does not change between the different experimental conditions, so that the experimental effects are given by the change in values (from the passive observational to the experimental data) of the non-intervened variables divided by the corresponding change in value of the intervened variable. Under this assumption, the theory presented in this paper is directly applicable to estimate the direct effects among the variables from the experimental effects.

If, however, one wants to apply the full machinery provided in this paper to data of the above kind, but in which each experiment intervenes on *multiple* variables simultaneously, it is not sufficient to obtain just one or two experimental data vectors $\mathbf{x}_k$. Rather, in general multiple data vectors may be needed to be able to disentangle the effects of each of the intervened-upon variables on the non-intervened ones. The details of the required experimental protocols, as well as sufficient and necessary identifiability conditions, are however left for future work.

A different extension considers models in which the observed data vectors arise from an equilibrium reached by a process with *stochastic* dynamics. Specifically, consider a time-series process

$$\mathbf{x}(t) \quad := \quad \mathbf{B}\mathbf{x}(t-1) + \mathbf{e}(t),$$

where $\mathbf{e}(t)$ is sampled anew at each time step t, always from the same distribution with mean $\boldsymbol{\mu}_\mathbf{e} = \mathbf{0}$ and variance-covariance matrix $\boldsymbol{\Sigma}_\mathbf{e}$. All the variables in $\mathbf{x}$ are updated simultaneously given their values of the previous time step and the new disturbance term $\mathbf{e}(t)$.[9] Obviously, this system no longer has a deterministic equilibrium, but for an asymptotically stable model $(\mathbf{B}, \boldsymbol{\Sigma}_\mathbf{e})$ the process converges to an equilibrium in which a sample vector $\mathbf{x}(t=\infty)$ is drawn from

$$\begin{aligned} \boldsymbol{\mu}_\mathbf{x} &= \mathbf{0}, \\ \mathbf{C}_\mathbf{x} &= \lim_{t\to\infty} \sum_{i=1}^{t} \mathbf{B}^{t-i} \boldsymbol{\Sigma}_\mathbf{e} (\mathbf{B}^T)^{t-i}. \end{aligned}$$

As in the deterministic model, the observed vector $\mathbf{x}$ drawn at equilibrium is independent of the initial values at the start of the process. Different observed data vectors $\mathbf{x}$ would be obtained by running multiple parallel chains. Interventions could be modeled as setting a given variable to a value

9. We note that this model differs from Lauritzen and Richardson (2002)'s stochastic equilibrium model, discussed in Sections 6 and 7 of their paper. They consider a sequential update of the variables in a particular order.

drawn from some distribution, and then keeping that variable constant throughout the equilibrating process.

In such a model the covariances between the intervened and non-intervened variables correspond to experimental effects, mirroring the deterministic case. Hence the theory presented in this paper could be used to estimate the direct effects matrix $\mathbf{B}$. Given the direct effects, and given a passive-observational covariance matrix $\mathbf{C_x}$, one could estimate $\mathbf{\Sigma_e}$ using the relation

$$\mathbf{\Sigma_e} \quad = \quad \mathbf{C_x} - \mathbf{B}\mathbf{C_x}\mathbf{B}^T.$$

Note, however, that the expression for the covariance among the non-intervened variables is *not* directly parallel to the deterministic case, so some of the theory presented in this paper would need to be adapted if this particular model were of primary interest.

In all the models discussed so far, we have been assuming that interventions take full control of the intervened variable by making it independent of its normal causes. This representation of an intervention is consistent with interventions in randomized controlled trials or in cases where a variable is "clamped" to a particular value. However, interventions needn't be "surgical" in this sense, but could instead only add an additional influence to the intervened variable without breaking the relations between the intervened variable and its causal parents. Such interventions are sometimes referred to as "soft" interventions. In linear models they are formally equivalent to instrumental variables, which are known to be useful for causal discovery. In our model a soft intervention is simply represented by an added influence that does not affect the coefficient matrix $\mathbf{B}$, nor the disturbance term $\mathbf{e}$. That is, the matrix $\mathbf{U}_k$ is deleted in both instances from Equation 4, but the influence $\mathbf{c}$ is still added. Assuming that the influence of the soft interventions on the intervened variables is known, that is, that $\mathbf{c}$ is measured, and that multiple simultaneous soft interventions are performed independently, it can be shown that one can still determine the experimental effects of the intervened variables. The entire machinery described here thus transfers with only some very minor adjustments. Given that soft interventions can be combined independently of one another, very efficient experimental protocols can be developed. In Eberhardt et al. (2010) we found that even from a statistical perspective, soft interventions appear to require the overall least number of samples for causal discovery.

Lastly, it is worth noting that the LLC-Algorithm presented here uses the measured experimental effects $t(x_i \rightsquigarrow x_u || \mathcal{J})$ to linearly constrain the unknown *direct* effects b_{ji} of $\mathbf{B}$. There may be circumstances in which it might be beneficial to instead use the experimental effects to linearly constrain the *total* effects $t(x_i \rightsquigarrow x_u)$.[10] In fact, such a representation was originally developed in Eberhardt et al. (2010). Given an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$, the linear constraint of the measured experimental effects on the unknown total effects $t(x_i \rightsquigarrow x_u)$ is then given by

$$t(x_i \rightsquigarrow x_u) \quad = \quad t(x_i \rightsquigarrow x_u || \mathcal{J}_k) + \sum_{x_j \in \mathcal{J}_k \setminus \{x_i\}} t(x_i \rightsquigarrow x_j) t(x_j \rightsquigarrow x_u || \mathcal{J}_k).$$

The constraint has a similar form to the constraint on direct effects in Equation 16, but combines a different set of experimental effects. Such a representation of the constraints in terms of total effects forms the basis for an algorithm analogous to LLC to identify the total effects. Once all the total effects are determined, one can, if needed, easily infer the direct effects (see Eberhardt et al., 2010).

10. Recall that the total effect corresponds to the experimental effect in the single-intervention experiment where only the cause is subject to intervention, that is, $t(x_i \rightsquigarrow x_u) = t(x_i \rightsquigarrow x_u || \{x_i\})$.

8. Conclusion

We have described a procedure that uses data from a set of experiments to identify linear causal models that may contain cycles and latent variables. While assuming linearity is a significant restriction, we are not aware of any other procedure that works with assumptions that are as weak in all other regards. Given this model space, we have shown how important the satisfaction of the pair condition and the covariance condition is for identifiability. Additionally, we have noted that when the identifiability conditions are not satisfied, the underdetermination of the model is generally fairly local.

Despite our analysis in terms of *canonical* models and sets of *canonical* experiments, we have indicated that these are in fact only very weak conditions: Any data from a non-conditional surgical experiment can be turned into data from a corresponding canonical one (if the experiment was not canonical to start with), and almost any linear cyclic model with latent variables can be represented by a canonical model that is completely equivalent with respect to the available data and any novel predictions produced. Thus, our procedure can handle a quite general model family and experimental setup.

We have shown that the LLC algorithm performs quite well in comparison with algorithms designed for solving similar inference problems. Moreover, within the DREAM challenges, we have a good comparison of how our algorithm (suitably adapted to the problem) performs for realistic data. It is competitive across all challenges despite the linearity assumption.

In Section 7 we have suggested how our model and search procedure can be generalized to models with stochastic dynamics; in Eberhardt et al. (2010) we also considered experiments with so-called "soft" interventions. An open question remains: What are the minimal conditions a model must satisfy such that a search procedure based on experiments that satisfy the pair condition for all ordered pairs of variables is sufficient for model identifiability? In Hyttinen et al. (2011) we showed that this condition is necessary and sufficient for identifiability in discrete acyclic models with a noisy-or parametrization. It is not known to what extent the condition generalizes to other model families.

Acknowledgments

The authors would like to thank the anonymous reviewers for extremely detailed and helpful comments. A.H. & P.O.H. were supported by the Academy of Finland and by University of Helsinki Research Funds. F.E. was supported by a grant from the James S. McDonnell Foundation on 'Experimental Planning and the Unification of Causal Knowledge'.

Appendix A. Centering the Data

Here we show how to center the data, so that it can be modeled with a linear cyclic model with latent variables that assumes a zero mean for the disturbances. We also consider how to translate the predictions of the model to predictions for the actual data generating process. Throughout, we assume that in each experiment we observe the mean and covariance matrix in the infinite sample limit.

Let the true data generating model be a linear cyclic model with latent variables $(\mathbf{B}, \mathbf{\Sigma_e}, \boldsymbol{\mu}_\mathbf{e})$ where $\boldsymbol{\mu}_\mathbf{e} \neq \mathbf{0}$. Say, we have observed passive observational data with mean $\boldsymbol{\mu}^0_\mathbf{X}$. In an arbitrary

experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ the data generating model produces data with the following mean and covariance matrix:

$$\begin{aligned} \boldsymbol{\mu}^k_{\mathbf{x}} &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\mathbf{U}_k\boldsymbol{\mu}_{\mathbf{e}} + \mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}}), \\ \mathbf{C}^k_{\mathbf{x}} &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\boldsymbol{\Sigma}^k_{\mathbf{c}} + \mathbf{U}_k\boldsymbol{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-T}. \end{aligned} \tag{26}$$

If we first center all data vectors by

$$\mathbf{x} = \mathbf{x} - \boldsymbol{\mu}^0_{\mathbf{x}}, \tag{27}$$

then the centered data has mean $\boldsymbol{\mu}^k_{\mathbf{x}} = \boldsymbol{\mu}^k_{\mathbf{x}} - \boldsymbol{\mu}^0_{\mathbf{x}}$ and unaltered covariance matrix $\mathbf{C}^k_{\mathbf{x}} = \mathbf{C}^k_{\mathbf{x}}$. The centering of Equation 27 implies that instead of randomizing the intervened variables in $\mathcal{J}_k$ with mean $(\boldsymbol{\mu}^k_{\mathbf{c}})_{\mathcal{J}_k}$ and covariance $(\boldsymbol{\Sigma}^k_{\mathbf{c}})_{\mathcal{J}_k\mathcal{J}_k}$, the centered variables are considered to be randomized with mean $(\boldsymbol{\mu}^k_{\mathbf{c}})_{\mathcal{J}_k} = (\boldsymbol{\mu}^k_{\mathbf{c}} - \boldsymbol{\mu}^0_{\mathbf{x}})_{\mathcal{J}_k}$ and covariance $(\boldsymbol{\Sigma}^k_{\mathbf{c}})_{\mathcal{J}_k\mathcal{J}_k} = (\boldsymbol{\Sigma}^k_{\mathbf{c}})_{\mathcal{J}_k\mathcal{J}_k}$. The subsequent equations show that the corresponding model with zero mean disturbance $(\tilde{\mathbf{B}}, \tilde{\boldsymbol{\Sigma}}_{\mathbf{e}}, \tilde{\boldsymbol{\mu}}_{\mathbf{e}})$, where $\tilde{\mathbf{B}} = \mathbf{B}$, $\tilde{\boldsymbol{\Sigma}}_{\mathbf{e}} = \boldsymbol{\Sigma}_{\mathbf{e}}$ and $\tilde{\boldsymbol{\mu}}_{\mathbf{e}} = \mathbf{0}_n$, generates the centered data when the intervened variables are randomized with mean $(\boldsymbol{\mu}^k_{\mathbf{c}})_{\mathcal{J}_k}$ and covariance $\mathbf{C}^k_{\mathbf{x}}$:

$$\begin{aligned} \tilde{\mathbf{C}}^k_{\mathbf{x}} &= (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\boldsymbol{\Sigma}^k_{\mathbf{c}} + \mathbf{U}_k\tilde{\boldsymbol{\Sigma}}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-T} \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\boldsymbol{\Sigma}^k_{\mathbf{c}} + \mathbf{U}_k\boldsymbol{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-T} = \mathbf{C}^k_{\mathbf{x}} = \mathbf{C}^k_{\mathbf{x}}, \\ \tilde{\boldsymbol{\mu}}^k_{\mathbf{x}} &= (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{U}_k\tilde{\boldsymbol{\mu}}_{\mathbf{e}} + \mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}}) \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\boldsymbol{\mu}^k_{\mathbf{c}} - \boldsymbol{\mu}^0_{\mathbf{x}}) \qquad\qquad ||\boldsymbol{\mu}^0_{\mathbf{x}} = (\mathbf{I} - \mathbf{B})^{-1}\boldsymbol{\mu}_{\mathbf{e}} \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}} - (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I} - \mathbf{B})^{-1}\boldsymbol{\mu}_{\mathbf{e}} + (\mathbf{I} - \mathbf{B})^{-1}\boldsymbol{\mu}_{\mathbf{e}} - \boldsymbol{\mu}^0_{\mathbf{x}} \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}} + (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(-\mathbf{J}_k + \mathbf{I} - \mathbf{U}_k\mathbf{B})(\mathbf{I} - \mathbf{B})^{-1}\boldsymbol{\mu}_{\mathbf{e}} - \boldsymbol{\mu}^0_{\mathbf{x}} \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}} + (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(-(\mathbf{I} - \mathbf{U}_k) + \mathbf{I} - \mathbf{U}_k\mathbf{B})(\mathbf{I} - \mathbf{B})^{-1}\boldsymbol{\mu}_{\mathbf{e}} - \boldsymbol{\mu}^0_{\mathbf{x}} \\ &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\mathbf{U}_k\boldsymbol{\mu}_{\mathbf{e}} + \mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}}) - \boldsymbol{\mu}^0_{\mathbf{x}} = \boldsymbol{\mu}^k_{\mathbf{x}} - \boldsymbol{\mu}^0_{\mathbf{x}} = \boldsymbol{\mu}^k_{\mathbf{x}}. \end{aligned}$$

Thus, the centering is innocuous with regard to $\mathbf{B}$ and $\boldsymbol{\Sigma}_{\mathbf{e}}$. The identities show also how to translate the predictions of the zero-mean disturbance model in some novel experiment $\mathcal{E}_k$ to the predictions of the actual data generating model: $\boldsymbol{\mu}^k_{\mathbf{x}} = \boldsymbol{\mu}^0_{\mathbf{x}} + \tilde{\boldsymbol{\mu}}^k_{\mathbf{x}}$ and $\mathbf{C}^k_{\mathbf{x}} = \tilde{\mathbf{C}}^k_{\mathbf{x}}$.

In the unlikely case that passive observational data is not available, we can simply center the data vectors $\mathbf{x}$ observed in experiment $\mathcal{E}_k$:

$$\mathbf{x} := \mathbf{x} - \boldsymbol{\mu}^k_{\mathbf{x}}.$$

This essentially corresponds to just ignoring the observed mean in each experiment. The theory in the paper can be used to estimate the direct effects matrix $\mathbf{B}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{e}}$, as the data covariance matrices are independent of the mean of the disturbances. This is sufficient for structure discovery, but if we want to achieve consistent predictions of the observed mean in novel experimental setups, $\boldsymbol{\mu}_{\mathbf{e}}$ also needs to be estimated. In experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$, the data has mean $\boldsymbol{\mu}^k_{\mathbf{x}}$, so $(\boldsymbol{\mu}_{\mathbf{e}})_{\mathcal{U}_k}$ can be estimated by $((\mathbf{I} - \mathbf{U}_k\mathbf{B})\boldsymbol{\mu}^k_{\mathbf{x}})_{\mathcal{U}_k}$, since

$$\begin{aligned} ((\mathbf{I} - \mathbf{U}_k\mathbf{B})\boldsymbol{\mu}^k_{\mathbf{x}})_{\mathcal{U}_k} &= ((\mathbf{I} - \mathbf{U}_k\mathbf{B})(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}(\mathbf{U}_k\boldsymbol{\mu}_{\mathbf{e}} + \mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}}))_{\mathcal{U}_k} \\ &= (\mathbf{U}_k\boldsymbol{\mu}_{\mathbf{e}} + \mathbf{J}_k\boldsymbol{\mu}^k_{\mathbf{c}})_{\mathcal{U}_k} = (\boldsymbol{\mu}_{\mathbf{e}})_{\mathcal{U}_k}. \end{aligned}$$

Thus, if each variable x_i is observed unmanipulated in some experiment *and* $\mathbf{B}$ is identified, then the whole vector $\boldsymbol{\mu}_\mathbf{e}$ can be estimated. The predicted mean $\boldsymbol{\mu}^k_\mathbf{x}$ for an arbitrary novel experiment $\mathcal{E}_k$ can then be obtained using Equation 26. See Appendices B and J for additional discussion on predicting means.

Appendix B. Proof of Lemma 5 (Correlated Experiment)

In a correlated experiment $\mathcal{E}_k$, where $\mathbf{c}$ is randomized with mean $\boldsymbol{\mu}^k_\mathbf{c}$ and covariance matrix $\boldsymbol{\Sigma}^k_\mathbf{c}$ such that $(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}$ is symmetric positive-definite, the model $(\mathbf{B}, \boldsymbol{\Sigma}_\mathbf{e})$ produces the following observations:

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}^k_\mathbf{x} &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\boldsymbol{\mu}^k_\mathbf{c},\\
\tilde{\mathbf{C}}^k_\mathbf{x} &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\boldsymbol{\Sigma}^k_\mathbf{c}+\mathbf{U}_k\boldsymbol{\Sigma}_\mathbf{e}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\\
&= \begin{bmatrix} (\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k} & (\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}\mathbf{B}^T_{\mathcal{U}_k\mathcal{J}_k}(\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-T} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k}(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k} & * \end{bmatrix},\\
* &= (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}((\boldsymbol{\Sigma}_\mathbf{e})_{\mathcal{U}_k,\mathcal{U}_k}+\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k}(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}\mathbf{B}^T_{\mathcal{U}_k\mathcal{J}_k})(\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-T}.
\end{aligned}$$

Then, matrix $\tilde{\mathbf{T}}^k_\mathbf{x}$, defined in the lemma in terms of the observed covariance matrix $\tilde{\mathbf{C}}^k_\mathbf{x}$, can be expressed solely in terms of the model parameters $\mathbf{B}$:

$$\begin{aligned}
\tilde{\mathbf{T}}^k_\mathbf{x} &= (\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{V}\mathcal{J}_k}((\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k})^{-1} = \begin{bmatrix} (\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k}(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k} \end{bmatrix}\begin{bmatrix} (\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k} \end{bmatrix}^{-1}\\
&= \begin{bmatrix} \mathbf{I} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k} \end{bmatrix} = ((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{J}_k},
\end{aligned}$$

where matrix $(\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k} = (\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}$ is invertible, since it is a positive-definite matrix. The following identities apply:

$$\begin{aligned}
\tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{T}}^k_\mathbf{x})^T &= ((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{J}_k}(((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{J}_k})^T\\
&= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T},\\
\tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k}(\tilde{\mathbf{T}}^k_\mathbf{x})^T &= ((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{J}_k}(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}(((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{J}_k})^T\\
&= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\boldsymbol{\Sigma}^k_\mathbf{c}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}.
\end{aligned}$$

Now from Equations 5 and 6 we can calculate the statistics of the experiment if the intervened variables had been randomized with zero mean and unit variance (appearing in Equations 7 and 8):

$$\begin{aligned}
\boldsymbol{\mu}^k_\mathbf{x} &= \mathbf{0},\\
\mathbf{C}^k_\mathbf{x} &= \tilde{\mathbf{C}}^k_\mathbf{x}-\tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{C}}^k_\mathbf{x})_{\mathcal{J}_k\mathcal{J}_k}(\tilde{\mathbf{T}}^k_\mathbf{x})^T+\tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{T}}^k_\mathbf{x})^T\\
&= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\boldsymbol{\Sigma}^k_\mathbf{c}+\mathbf{U}_k\boldsymbol{\Sigma}_\mathbf{e}\mathbf{U}_k-\boldsymbol{\Sigma}^k_\mathbf{c}+\mathbf{J}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\\
&= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{U}_k\boldsymbol{\Sigma}_\mathbf{e}\mathbf{U}_k+\mathbf{J}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}.
\end{aligned}$$

Notice that the formulas in the lemma can also be used to transform the predictions $\mathbf{0}$ and $\mathbf{C}^k_\mathbf{x}$ in a canonical experiment to predictions $\tilde{\boldsymbol{\mu}}^k_\mathbf{x}$ and $\tilde{\mathbf{C}}^k_\mathbf{x}$ in a non-canonical experiment, where $\mathbf{c}$ is randomized with mean $\boldsymbol{\mu}^k_\mathbf{c}$ and covariance $\boldsymbol{\Sigma}^k_\mathbf{c}$:

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}^k_\mathbf{x} &= \tilde{\mathbf{T}}^k_\mathbf{x}\boldsymbol{\mu}^k_\mathbf{c},\\
\tilde{\mathbf{C}}^k_\mathbf{x} &= \mathbf{C}^k_\mathbf{x}+\tilde{\mathbf{T}}^k_\mathbf{x}(\boldsymbol{\Sigma}^k_\mathbf{c})_{\mathcal{J}_k\mathcal{J}_k}(\tilde{\mathbf{T}}^k_\mathbf{x})^T-\tilde{\mathbf{T}}^k_\mathbf{x}(\tilde{\mathbf{T}}^k_\mathbf{x})^T,
\end{aligned}$$

where $\tilde{\mathbf{T}}^k_{\mathbf{x}} = (\mathbf{C}^k_{\mathbf{x}})_{\mathcal{V}\mathcal{J}_k}$.

Appendix C. Derivation of the Trek Rule for Asymptotically Stable Models

From the definition of asymptotic stability it follows that the eigenvalues of $\mathbf{U}_k\mathbf{B}$ are all less than one in absolute value. As the eigenvalues of matrix $\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k}$ are equal to those of $\mathbf{U}_k\mathbf{B}$, matrix $(\mathbf{I} - \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}$ can be written as the following geometric series:

$$(\mathbf{I} - \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1} \quad = \quad \mathbf{I} + \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k} + \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k}\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k} + \cdots .$$

Now, the experimental effect $t(x_i \rightsquigarrow x_u || \mathcal{J}_k)$ can be expressed as the sum-product implied by the trek rules:

$$\begin{aligned} t(x_i \rightsquigarrow x_u || \mathcal{J}_k) &= (\mathbf{T}^k_{\mathbf{x}})_{\{x_u\}\{x_i\}} \\ &= ((\mathbf{I} - \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k})_{\{x_u\}\{x_i\}} \\ &= ((\mathbf{I} + \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k} + \mathbf{B}_{\mathcal{U}_k\mathcal{U}_k}\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k} + \cdots)\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k})_{\{x_u\}\{x_i\}} \\ &= b_{ui} + \sum_{j \in \mathcal{U}_k} b_{uj}b_{ji} + \sum_{j \in \mathcal{U}_k}\sum_{l \in \mathcal{U}_k} b_{uj}b_{jl}b_{li} + \cdots \\ &= \sum_{p \in \mathcal{P}(x_i \rightsquigarrow x_u || \mathcal{J}_k)} \prod_{(x_l \to x_m) \in p} b_{ml}. \end{aligned}$$

Appendix D. Proof of Lemma 7 (Marginalization)

In the following, note that the experiment of the marginalized model $\tilde{\mathcal{E}}_k = (\tilde{\mathcal{J}}_k, \tilde{\mathcal{U}}_k)$ and the corresponding experiment of the full model $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ satisfy $\mathcal{J}_k = \tilde{\mathcal{J}}_k$ and $\mathcal{U}_k = \tilde{\mathcal{U}}_k \cup \mathcal{M}$. Without loss of generality the variables are labeled such that $x_1, \ldots, x_i \in \tilde{\mathcal{J}}_k, x_{i+1}, \ldots, x_j \in \tilde{\mathcal{U}}_k$ and $x_{j+1}, \ldots, x_n \in \mathcal{M}$ to allow for easy block matrix manipulation.

D.1 Weak Stability

We show that if the full model $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$ is weakly stable then the marginalized model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ is also weakly stable. Make the counter-assumption that $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ is weakly unstable, thus there exists an experiment $\tilde{\mathcal{E}}_k$ such that $(\mathbf{I} - \tilde{\mathbf{U}}_k\tilde{\mathbf{B}})$ is singular, or equivalently matrix $\tilde{\mathbf{U}}_k\tilde{\mathbf{B}}$ has a unit eigenvalue: $\exists \tilde{\mathbf{v}} \neq \mathbf{0}$ such that $\tilde{\mathbf{U}}_k\tilde{\mathbf{B}}\tilde{\mathbf{v}} = \tilde{\mathbf{v}}$. The following shows that then $\mathbf{U}_k\mathbf{B}$ also has a unit eigenvalue corresponding to the eigenvector $\mathbf{v}$ defined below:[11]

11. Invertibility of $(\mathbf{I} - \mathbf{B}_{\mathcal{M}\mathcal{M}})$ follows from the weak stability of $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$ in experiment $(\tilde{\mathcal{V}}, \mathcal{M})$.

$$
\begin{aligned}
\mathbf{U}_k\mathbf{B}\mathbf{v} &= \begin{bmatrix} \tilde{\mathbf{U}}_k & \\ & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{B}_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}} & \mathbf{B}_{\tilde{\mathcal{V}}\mathcal{M}} \\ \mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}} & \mathbf{B}_{\mathcal{M}\mathcal{M}} \end{bmatrix} \mathbf{v} \qquad ||\mathbf{v} = \begin{bmatrix} \tilde{\mathbf{v}} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} \\
&= \begin{bmatrix} \tilde{\mathbf{U}}_k\mathbf{B}_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}} & \tilde{\mathbf{U}}_k\mathbf{B}_{\tilde{\mathcal{V}}\mathcal{M}} \\ \mathbf{I}\cdot\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}} & \mathbf{I}\cdot\mathbf{B}_{\mathcal{M}\mathcal{M}} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{v}} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} \\
&= \begin{bmatrix} \tilde{\mathbf{U}}_k\mathbf{B}_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} + \tilde{\mathbf{U}}_k\mathbf{B}_{\tilde{\mathcal{V}}\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \\ \mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} + \mathbf{B}_{\mathcal{M}\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} \\
&= \begin{bmatrix} \tilde{\mathbf{U}}_k(\mathbf{B}_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}} + \mathbf{B}_{\tilde{\mathcal{V}}\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}})\tilde{\mathbf{v}} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} + \mathbf{B}_{\mathcal{M}\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} \quad ||\text{Def. of } \tilde{\mathbf{B}} \\
&= \begin{bmatrix} \tilde{\mathbf{U}}_k\tilde{\mathbf{B}}\tilde{\mathbf{v}} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}}+\mathbf{B}_{\mathcal{M}\mathcal{M}})(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{v}} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{V}}}\tilde{\mathbf{v}} \end{bmatrix} = \mathbf{v}.
\end{aligned}
$$

Thus, $(\mathbf{I}-\mathbf{U}_k\mathbf{B})$ is singular and the full model $(\mathbf{B},\mathbf{\Sigma_e})$ is not weakly stable. Because this is contradictory to the assumptions, $(\tilde{\mathbf{B}},\tilde{\mathbf{\Sigma}}_\mathbf{e})$ must be weakly stable.

D.2 Equal Covariance Matrices

We need to show that in experiment $\mathcal{E}_k$ the covariance matrix $(\mathbf{C}^k_\mathbf{x})_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}}$ produced by $(\mathbf{B},\mathbf{\Sigma_e})$ is equal to the covariance matrix $\tilde{\mathbf{C}}^k_\mathbf{x}$ produced by $(\tilde{\mathbf{B}},\tilde{\mathbf{\Sigma}}_\mathbf{e})$. This requires us first to derive the following identities:

$$
\begin{aligned}
(\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1} &= (\mathbf{I}-\mathbf{B}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k}-\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{U}}_k})^{-1}, && (28)\\
\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{J}}_k} &= \mathbf{B}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{J}}_k}+\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{J}}_k}, && (29)\\
(\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{J}}_k} &= ((\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{J}_k})_{\tilde{\mathcal{U}}_k\tilde{\mathcal{J}}_k}, && (30)\\
((\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1})_{\tilde{\mathcal{V}}\tilde{\mathcal{J}}_k} &= ((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\tilde{\mathcal{V}}\tilde{\mathcal{J}}_k}. && (31)
\end{aligned}
$$

The goal is to derive Equation 31, which means that both models produce the same experimental effects from $x_i \in \tilde{\mathcal{J}}_k$ to $x_u \in \tilde{\mathcal{U}}_k$.

Equations 28 and 29 follow directly from the marginalized model definition in Lemma 7. To show Equation 30, we invert the matrix $(\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})$ in blocks (the unneeded blocks on rows corresponding to the marginalized variables are replaced with a '$\cdot$'-symbol):

$$
\begin{aligned}
(\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1} &= \begin{bmatrix} \mathbf{I}-\mathbf{B}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k} & -\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}} \\ -\mathbf{B}_{\mathcal{M}\tilde{\mathcal{U}}_k} & \mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}} \end{bmatrix}^{-1} \quad ||\text{block matrix inversion \& Eq. 28} \\
&= \begin{bmatrix} (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1} & (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1} \\ \cdot & \cdot \end{bmatrix}.
\end{aligned}
$$

Then, we can verify Equation 30:

$$
\begin{aligned}
& ((\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{I}_k})_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \\
= & \left(\begin{bmatrix} (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1} & (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1} \\ \cdot & \cdot \end{bmatrix} \begin{bmatrix} \mathbf{B}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \\ \mathbf{B}_{\mathcal{M}\tilde{\mathcal{I}}_k} \end{bmatrix} \right)_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \\
= & \begin{bmatrix} (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}(\mathbf{B}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k}+\mathbf{B}_{\tilde{\mathcal{U}}_k\mathcal{M}}(\mathbf{I}-\mathbf{B}_{\mathcal{M}\mathcal{M}})^{-1}\mathbf{B}_{\mathcal{M}\tilde{\mathcal{I}}_k}) \\ \cdot \end{bmatrix}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \quad ||\text{Eq. 29} \\
= & \begin{bmatrix} (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \\ \cdot \end{bmatrix}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} = (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k}.
\end{aligned}
$$

Equation 31 follows quite directly from Equation 30:

$$
\begin{aligned}
((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\tilde{\mathcal{V}}\tilde{\mathcal{I}}_k} &= (((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{V}\mathcal{I}_k})_{\tilde{\mathcal{V}}\tilde{\mathcal{I}}_k} = \begin{bmatrix} \mathbf{I} \\ (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{I}_k} \end{bmatrix}_{\tilde{\mathcal{V}}\tilde{\mathcal{I}}_k} \\
&= \begin{bmatrix} \mathbf{I} \\ ((\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{I}_k})_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \end{bmatrix} \quad ||\text{Eq. 30} \\
&= \begin{bmatrix} \mathbf{I} \\ (\mathbf{I}-\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{U}}_k})^{-1}\tilde{\mathbf{B}}_{\tilde{\mathcal{U}}_k\tilde{\mathcal{I}}_k} \end{bmatrix} = ((\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1})_{\tilde{\mathcal{V}}\tilde{\mathcal{I}}_k}.
\end{aligned}
$$

Next, we use matrix $\tilde{\mathbf{v}} = [\mathbf{I}_{j\times j}\ \mathbf{0}_{j\times(n-j)}]$ to avoid the complicated block matrix notation. Multiplication from the left by $\tilde{\mathbf{v}}$ just selects the rows corresponding to variables in $\tilde{\mathcal{V}}$, multiplication from the right by $\tilde{\mathbf{v}}^T$ selects the columns corresponding to variables in $\tilde{\mathcal{V}}$. We prove the following identities.:

$$
\begin{aligned}
(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{J}}_k &= \tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k\tilde{\mathbf{v}}^T, && (32) \\
(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{U}}_k(\mathbf{I}-\tilde{\mathbf{B}})\tilde{\mathbf{v}} &= \tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I}-\mathbf{B}). && (33)
\end{aligned}
$$

Equation 32 just restates Equation 31 using matrix $\tilde{\mathbf{v}}$. Equation 33 is verified by the following derivation:

$$
\begin{aligned}
& (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{U}}_k(\mathbf{I}-\tilde{\mathbf{B}})\tilde{\mathbf{v}}-\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I}-\mathbf{B}) \quad ||\mathbf{U}_k=\mathbf{I}-\mathbf{J}_k, \tilde{\mathbf{U}}_k=\mathbf{I}-\tilde{\mathbf{J}}_k \\
= & (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}}-\tilde{\mathbf{J}}_k)\tilde{\mathbf{v}}-\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{I}-\mathbf{U}_k\mathbf{B}-\mathbf{J}_k) \\
= & \tilde{\mathbf{v}}-(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{J}}_k\tilde{\mathbf{v}}-\tilde{\mathbf{v}}+\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k \ ||\text{Eq. 32} \\
= & -\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k\tilde{\mathbf{v}}^T\tilde{\mathbf{v}}+\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k \quad ||\mathbf{J}_k\tilde{\mathbf{v}}^T\tilde{\mathbf{v}}=\mathbf{J}_k \\
= & \mathbf{0}.
\end{aligned}
$$

Finally, we can show that the covariance matrix $\tilde{\mathbf{C}}_{\mathbf{x}}^k$ of the marginalized model matches the marginalized covariance matrix $(\mathbf{C}_{\mathbf{x}}^k)_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}}$ of the original model:

$$\begin{aligned}
\tilde{\mathbf{C}}_{\mathbf{x}}^k &= (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}(\tilde{\mathbf{J}}_k+\tilde{\mathbf{U}}_k\tilde{\mathbf{\Sigma}}_{\mathbf{e}}\tilde{\mathbf{U}}_k)(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-T} \quad \text{||definition of } \tilde{\mathbf{\Sigma}}_{\mathbf{e}} \\
&= (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}(\tilde{\mathbf{J}}_k+\tilde{\mathbf{U}}_k(\mathbf{I}-\tilde{\mathbf{B}})\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}-\mathbf{B})^{-T}\tilde{\mathbf{v}}^T(\mathbf{I}-\tilde{\mathbf{B}})^T\tilde{\mathbf{U}}_k)(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-T} \\
&= (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{J}}_k(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-T}+ \\
&\quad (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{U}}_k(\mathbf{I}-\tilde{\mathbf{B}})\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}-\mathbf{B})^{-T}\tilde{\mathbf{v}}^T(\mathbf{I}-\tilde{\mathbf{B}})^T\tilde{\mathbf{U}}_k(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-T} \text{ ||Eq. 33} \\
&= (\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{J}}_k\tilde{\mathbf{J}}_k(\mathbf{I}-\tilde{\mathbf{U}}_k\tilde{\mathbf{B}})^{-T}+ \\
&\quad \tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}-\mathbf{B})^{-T}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\tilde{\mathbf{v}}^T \text{ ||Eq. 32} \\
&= \tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\tilde{\mathbf{v}}^T+\tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\tilde{\mathbf{v}}^T \\
&= \tilde{\mathbf{v}}(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\tilde{\mathbf{v}}^T=(\mathbf{C}_{\mathbf{x}}^k)_{\tilde{\mathcal{V}}\tilde{\mathcal{V}}}.
\end{aligned}$$

Appendix E. Proof of Lemma 8 (Self Cycles)

Again, we first show weak stability and then confirm that the covariance matrices are equal.

E.1 Weak Stability

First, we show that the model $(\tilde{\mathbf{B}},\tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ without the self-loop is weakly stable, if the model $(\mathbf{B},\mathbf{\Sigma}_{\mathbf{e}})$ with the self-loop is weakly stable. Notice that the weak stability of $(\mathbf{B},\mathbf{\Sigma}_{\mathbf{e}})$ in experiment $(\mathcal{V}\setminus\{x_i\},\{x_i\})$ implies that $b_{ii}\neq 1$. So, assume that $(\mathbf{B},\mathbf{\Sigma}_{\mathbf{e}})$ is weakly stable. Make the counter-assumption that $(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})$ is not invertible in some experiment $\mathcal{E}_k$, then $\exists\mathbf{v}\neq\mathbf{0}$ such that $\mathbf{U}_k\tilde{\mathbf{B}}\mathbf{v}=\mathbf{v}$. Matrix $\mathbf{B}$ can be written as a function of matrix $\tilde{\mathbf{B}}$ by inverting the definition of $\tilde{\mathbf{B}}$ in the lemma:

$$\mathbf{B} = (\mathbf{I}-b_{ii}\mathbf{U}_i)\tilde{\mathbf{B}}+b_{ii}\mathbf{U}_i.$$

If $x_i\in\mathcal{J}_k$ we have that $\mathbf{U}_k\mathbf{U}_i=\mathbf{0}_{n\times n}$, then

$$\mathbf{U}_k\mathbf{B} = \mathbf{U}_k(\mathbf{I}-b_{ii}\mathbf{U}_i)\tilde{\mathbf{B}}+b_{ii}\mathbf{U}_k\mathbf{U}_i=\mathbf{U}_k\tilde{\mathbf{B}}$$

and $\mathbf{U}_k\mathbf{B}\mathbf{v}=\mathbf{U}_k\tilde{\mathbf{B}}\mathbf{v}=\mathbf{v}$. Alternatively if $x_i\in\mathcal{U}_k$, we have that $\mathbf{U}_k\mathbf{U}_i=\mathbf{U}_i$, then

$$\begin{aligned}
\mathbf{U}_k\mathbf{B}\mathbf{v} &= \mathbf{U}_k(\mathbf{I}-b_{ii}\mathbf{U}_i)\tilde{\mathbf{B}}\mathbf{v}+b_{ii}\mathbf{U}_k\mathbf{U}_i\mathbf{v} \quad \text{||Multiplication of diagonal matrices commutes} \\
&= (\mathbf{I}-b_{ii}\mathbf{U}_i)\mathbf{U}_k\tilde{\mathbf{B}}\mathbf{v}+b_{ii}\mathbf{U}_k\mathbf{U}_i\mathbf{v} \\
&= (\mathbf{I}-b_{ii}\mathbf{U}_i)\mathbf{v}+b_{ii}\mathbf{U}_i\mathbf{v}=\mathbf{v}.
\end{aligned}$$

In both cases matrix $\mathbf{U}_k\mathbf{B}$ has a unit eigenvalue, and thus $\mathbf{I}-\mathbf{U}_k\mathbf{B}$ is singular. This is contradictory to the assumption that the model $(\mathbf{B},\mathbf{\Sigma}_{\mathbf{e}})$ is weakly stable, and so the model $(\tilde{\mathbf{B}},\tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ must be weakly stable.

E.2 Equal Covariance Matrices

Then we show that in an arbitrary experiment $\mathcal{E}_k$ the two models produce data with the same covariance matrices. First, if variable $x_i\in\mathcal{J}_k$, then $\mathbf{U}_k\mathbf{U}_i=\mathbf{0}_{n\times n}$, $\mathbf{U}_k\mathbf{B}=\mathbf{U}_k\tilde{\mathbf{B}}$ (as shown above) and

$$\mathbf{U}_k\tilde{\mathbf{\Sigma}}_{\mathbf{e}}\mathbf{U}_k = \mathbf{U}_k(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)^T\mathbf{U}_k=\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k.$$

The covariance matrices are trivially equal:

$$\begin{aligned}\tilde{\mathbf{C}}^k_{\mathbf{x}} &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{J}_k+\mathbf{U}_k\tilde{\mathbf{\Sigma}}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}=\mathbf{C}^k_{\mathbf{x}}.\end{aligned}$$

Alternatively, if variable $x_i \in \mathcal{U}_k$, then $\mathbf{U}_k\mathbf{U}_i = \mathbf{U}_i$, and because

$$\begin{aligned}(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1} &= (\mathbf{I}-\mathbf{U}_k\mathbf{B}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_k\mathbf{U}_i(\mathbf{I}-\mathbf{B}))(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\\ &= \mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i(\mathbf{I}-\mathbf{U}_k\mathbf{B}-\mathbf{J}_k\mathbf{B})(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1} \quad ||\mathbf{U}_i\mathbf{J}_k=\mathbf{0}_{n\times n}\\ &= \mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i,\end{aligned}$$

the covariance matrices are also equal:

$$\begin{aligned}\tilde{\mathbf{C}}^k_{\mathbf{x}} &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{J}_k+\mathbf{U}_k\tilde{\mathbf{\Sigma}}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T} \quad ||\text{definition of } \tilde{\mathbf{\Sigma}}_{\mathbf{e}}\\ &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{J}_k+\mathbf{U}_k(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)^T\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}((\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)\mathbf{J}_k(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)^T \quad ||\text{Multip. of diag. mat. commutes}\\ &\quad +\mathbf{U}_k(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)^T\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}+\frac{b_{ii}}{1-b_{ii}}\mathbf{U}_i)^T(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T} \ ||\text{id. above}\\ &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)\\ &\quad \cdot(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^T(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}=\mathbf{C}^k_{\mathbf{x}}.\end{aligned}$$

Appendix F. Derivation of Equation 13

Lemma 7 (Marginalization) showed that weak stability and experimental effects from an intervened variable $x_i \in \mathcal{I}_k$ to an observed variable $x_u \in \mathcal{U}_k$ are preserved (as part of the covariance matrix) when some variables in $\mathcal{U}_k$ are marginalized. Then, it is sufficient to show that Equation 13 applies in a weakly stable model where variables $\mathcal{U}_k \setminus \{x_j, x_u\}$ are marginalized. Lemma 8 (Self cycles) allows us to assume without loss of generality that there are no self-loops in this model.

Examine experiment $\mathcal{E}_k = (\mathcal{I}_k, \mathcal{U}_k)$ where $\mathcal{U}_k = \{x_j, x_u\}$ in the marginalized model $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$. The experimental effects in the experiment intervening on $\mathcal{I}_k \cup \{x_j\}$ are just the direct effects $t(x_i \rightsquigarrow x_u || \mathcal{I}_k \cup \{x_j\}) = b_{ui}$ and $t(x_j \rightsquigarrow x_u || \mathcal{I}_k \cup \{x_j\}) = b_{uj}$. The remaining experimental effects $t(x_i \rightsquigarrow x_u || \mathcal{I}_k)$ and $t(x_i \rightsquigarrow x_j || \mathcal{I}_k)$ appear in the matrix $((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{U}_k\mathcal{I}_k}$:

$$\begin{aligned}((\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1})_{\mathcal{U}_k\mathcal{I}_k} &= (\mathbf{I}-\mathbf{B}_{\mathcal{U}_k\mathcal{U}_k})^{-1}\mathbf{B}_{\mathcal{U}_k\mathcal{I}_k} = \begin{bmatrix} 1 & -b_{ju} \\ -b_{uj} & 1 \end{bmatrix}^{-1}\begin{bmatrix} \cdots & b_{ji} & \cdots \\ \cdots & b_{ui} & \cdots \end{bmatrix}\\ &= \frac{1}{1-b_{uj}b_{ju}}\begin{bmatrix} 1 & b_{ju} \\ b_{uj} & 1 \end{bmatrix}\begin{bmatrix} \cdots & b_{ji} & \cdots \\ \cdots & b_{ui} & \cdots \end{bmatrix} = \begin{bmatrix} \cdots & \frac{b_{ji}+b_{ju}b_{ui}}{1-b_{uj}b_{ju}} & \cdots \\ \cdots & \frac{b_{ui}+b_{uj}b_{ji}}{1-b_{uj}b_{ju}} & \cdots \end{bmatrix}.\end{aligned}$$

Now Equation 13 can be verified:

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\}) + t(x_i \rightsquigarrow x_j || \mathcal{J}_k) t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \{x_j\}) = b_{ui} + \frac{b_{ji} + b_{ju} b_{ui}}{1 - b_{uj} b_{ju}} b_{uj}$$
$$= \frac{b_{ui} - b_{uj} b_{ju} b_{ui} + b_{uj} b_{ji} + b_{uj} b_{ju} b_{ui}}{1 - b_{uj} b_{ju}} = \frac{b_{ui} + b_{uj} b_{ji}}{1 - b_{uj} b_{ju}} = t(x_i \rightsquigarrow x_u || \mathcal{J}_k).$$

Appendix G. Proof of Lemma 9 (Union/Intersection Experiment)

In this proof, we first derive a linear equation system on the unknown experimental effects and then show that it has a unique solution under weak stability.

G.1 Generalizations of Equation 13

Equation 13 can be generalized to relate some experimental effects in $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ to some experimental effects in $\mathcal{E}_{k \cup l} = (\mathcal{J}_k \cup \mathcal{J}_l, \mathcal{U}_k \cap \mathcal{U}_l)$ by applying Equation 13 iteratively:

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k) \quad = \quad t(x_i \rightsquigarrow x_u || \mathcal{J}_k \cup \mathcal{J}_l) + \sum_{x_j \in \mathcal{J}_l \setminus \mathcal{J}_k} t(x_i \rightsquigarrow x_j || \mathcal{J}_k) t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \mathcal{J}_l). \tag{34}$$

Here $x_i \in \mathcal{J}_k$, $x_u \in \mathcal{U}_k \cap \mathcal{U}_l$. Another way of writing the generalization relates some experimental effects in $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ to experimental effects in $\mathcal{E}_{k \cap l} = (\mathcal{J}_k \cap \mathcal{J}_l, \mathcal{U}_k \cup \mathcal{U}_l)$:

$$t(x_i \rightsquigarrow x_u || \mathcal{J}_k \cap \mathcal{J}_l) \quad = \quad t(x_i \rightsquigarrow x_u || \mathcal{J}_k) + \sum_{x_j \in \mathcal{J}_k \setminus \mathcal{J}_l} t(x_i \rightsquigarrow x_j || \mathcal{J}_k \cap \mathcal{J}_l) t(x_j \rightsquigarrow x_u || \mathcal{J}_k).$$

Here $x_i \in \mathcal{J}_k \cap \mathcal{J}_l$, $x_u \in \mathcal{U}_k$.

G.2 Equations for the Experimental Effects in the Union Experiment

First, partition $\mathcal{V}$ into the following disjoint sets: $I = \mathcal{J}_k \cap \mathcal{J}_l$ (intervened in both experiments), $\mathcal{K} = \mathcal{J}_k \setminus \mathcal{J}_l$ (intervened only in $\mathcal{E}_k$), $\mathcal{L} = \mathcal{J}_l \setminus \mathcal{J}_k$ (intervened only in $\mathcal{E}_l$) and $O = \mathcal{U}_k \cap \mathcal{U}_l$ (passively observed in both experiments). For each pair (x_k, x_u) with $x_k \in \mathcal{K}$ and $x_u \in O$ we can form an equation of the form of Equation 34 using experimental effects from experiment $\mathcal{E}_k$:

$$t(x_k \rightsquigarrow x_u || \mathcal{J}_k \cup \mathcal{J}_l) + \sum_{x_j \in \mathcal{L}} t(x_k \rightsquigarrow x_j || \mathcal{J}_k) t(x_j \rightsquigarrow x_u || \mathcal{J}_k \cup \mathcal{J}_l) \quad = \quad t(x_k \rightsquigarrow x_u || \mathcal{J}_k).$$

Equations for all such pairs can be represented neatly by block matrices:

$$(\mathbf{T}_{\mathbf{x}}^{k \cup l})_{O\mathcal{K}} + (\mathbf{T}_{\mathbf{x}}^{k \cup l})_{O\mathcal{L}} (\mathbf{T}_{\mathbf{x}}^{k})_{\mathcal{L}\mathcal{K}} \quad = \quad (\mathbf{T}_{\mathbf{x}}^{k})_{O\mathcal{K}}.$$

Similarly, equations can be formed for all pairs (x_k, x_u) with $x_k \in \mathcal{L}$ and $x_u \in O$ using experimental effects from experiment $\mathcal{E}_l$. For pairs (x_k, x_u) with $x_k \in I$ and $x_u \in O$, equations could be formed using the experimental effects from either experiments, but it turns out that only equations using the experimental effects of experiment $\mathcal{E}_k$ are needed. The equations form the following system:

$$\left[(\mathbf{T}_{\mathbf{x}}^{k \cup l})_{OI} \; (\mathbf{T}_{\mathbf{x}}^{k \cup l})_{O\mathcal{K}} \; (\mathbf{T}_{\mathbf{x}}^{k \cup l})_{O\mathcal{L}} \right] \underbrace{\begin{bmatrix} \mathbf{I}_{|I|} & & \\ & \mathbf{I}_{|\mathcal{K}|} & (\mathbf{T}_{\mathbf{x}}^{l})_{\mathcal{K}\mathcal{L}} \\ (\mathbf{T}_{\mathbf{x}}^{k})_{\mathcal{L}I} & (\mathbf{T}_{\mathbf{x}}^{k})_{\mathcal{L}\mathcal{K}} & \mathbf{I}_{|\mathcal{L}|} \end{bmatrix}}_{\mathbf{Q}} = \left[(\mathbf{T}_{\mathbf{x}}^{k})_{OI} \; (\mathbf{T}_{\mathbf{x}}^{k})_{O\mathcal{K}} \; (\mathbf{T}_{\mathbf{x}}^{l})_{O\mathcal{L}} \right]. \tag{35}$$

G.3 Invertibility

Now, we know the matrix on the right and matrix $\mathbf{Q}$, and we would like to solve for the matrix on the left by multiplying from the right by $\mathbf{Q}^{-1}$. Thus, we need to show that $\mathbf{Q}$ is invertible. Since the variables in O do not appear in matrix $\mathbf{Q}$ in any way, consider a marginalized model $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ over $\tilde{\mathcal{V}} = \mathcal{V} \setminus O$, where variables O are marginalized. The marginalized experiments corresponding to experiments $\mathcal{E}_k$ and $\mathcal{E}_l$ are $\tilde{\mathcal{E}}_k = (I \cup \mathcal{K}, \mathcal{L})$ and $\tilde{\mathcal{E}}_l = (I \cup \mathcal{L}, \mathcal{K})$ respectively. If $(\mathbf{B}, \mathbf{\Sigma}_{\mathbf{e}})$ is weakly stable as we assume, also $(\tilde{\mathbf{B}}, \tilde{\mathbf{\Sigma}}_{\mathbf{e}})$ is weakly stable by Lemma 7 (Marginalization). All the experimental effects in $\mathbf{Q}$ are preserved in the marginalization. The blocks can be now expressed using Equation 9:

$$
\begin{aligned}
(\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}I} &= (\tilde{\mathbf{T}}^k_{\mathbf{x}})_{\mathcal{L}I} = ((\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}, I \cup \mathcal{K}})_{\mathcal{L}I} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}I}, \\
(\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}\mathcal{K}} &= (\tilde{\mathbf{T}}^k_{\mathbf{x}})_{\mathcal{L}\mathcal{K}} = ((\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}, I \cup \mathcal{K}})_{\mathcal{L}\mathcal{K}} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{K}}, \\
(\mathbf{T}^l_{\mathbf{x}})_{\mathcal{K}\mathcal{L}} &= (\tilde{\mathbf{T}}^l_{\mathbf{x}})_{\mathcal{K}\mathcal{L}} = ((\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{K}})^{-1} \tilde{\mathbf{B}}_{\mathcal{K}, I \cup \mathcal{L}})_{\mathcal{K}\mathcal{L}} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{K}})^{-1} \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{L}}.
\end{aligned}
$$

The matrices inverted in the expressions are invertible, because the marginalized model is weakly stable. Now $\mathbf{Q}$ can be written as a product of 3 simple square matrices:

$$
\mathbf{Q} = \begin{bmatrix} \mathbf{I}_{|I|} & & \\ & \mathbf{I}_{|\mathcal{K}|} & (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{K}})^{-1} \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{L}} \\ (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}I} & (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{K}} & \mathbf{I}_{|\mathcal{L}|} \end{bmatrix} =
$$

$$
\begin{bmatrix} \mathbf{I}_{|I|} & & \\ & -(\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{K}})^{-1} & \\ & & (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1} \end{bmatrix} \left[\begin{array}{c|cc} \mathbf{I}_{|I|} & & \\ \hline & \mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{K}\mathcal{K}} & -\tilde{\mathbf{B}}_{\mathcal{K}\mathcal{L}} \\ \tilde{\mathbf{B}}_{\mathcal{L}I} & -\tilde{\mathbf{B}}_{\mathcal{L}\mathcal{K}} & \mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}} \end{array}\right] \begin{bmatrix} \mathbf{I}_{|I|} & & \\ & -\mathbf{I}_{|\mathcal{K}|} & \\ & & \mathbf{I}_{|\mathcal{L}|} \end{bmatrix}.
$$

The matrices on the left and on the right are invertible as block diagonal matrices with invertible blocks. Consider the middle matrix in the blocks indicated by the lines. Because the upper right-hand block is just zeros, the matrix is invertible if the two diagonal blocks are invertible. The lower right-hand block is invertible since the marginalized model is weakly stable in the experiment $(I, \mathcal{K} \cup \mathcal{L})$. As a product of 3 invertible matrices matrix $\mathbf{Q}$ is invertible. Note that the factorization is valid also in the case where $I = \emptyset$.

G.4 Matrix Equations for the Experimental Effects

The derivation of the equations and proof of invertibility for the intersection experiment proceeds very similarly. Here the formulas for solving the experimental effects in the union and intersection experiment are presented for completeness:

$$
\left[(\mathbf{T}^{k\cup l}_{\mathbf{x}})_{OI}\,(\mathbf{T}^{k\cup l}_{\mathbf{x}})_{O\mathcal{K}}\,(\mathbf{T}^{k\cup l}_{\mathbf{x}})_{O\mathcal{L}}\right] = \left[(\mathbf{T}^k_{\mathbf{x}})_{OI}\,(\mathbf{T}^k_{\mathbf{x}})_{O\mathcal{K}}\,(\mathbf{T}^l_{\mathbf{x}})_{O\mathcal{L}}\right] \begin{bmatrix} \mathbf{I} & & \\ & \mathbf{I} & (\mathbf{T}^l_{\mathbf{x}})_{\mathcal{K}\mathcal{L}} \\ (\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}I} & (\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}\mathcal{K}} & \mathbf{I} \end{bmatrix}^{-1},
$$

$$
\begin{bmatrix} (\mathbf{T}^{k\cap l}_{\mathbf{x}})_{\mathcal{K}I} \\ (\mathbf{T}^{k\cap l}_{\mathbf{x}})_{\mathcal{L}I} \\ (\mathbf{T}^{k\cap l}_{\mathbf{x}})_{OI} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & -(\mathbf{T}^l_{\mathbf{x}})_{\mathcal{K}\mathcal{L}} & \\ -(\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}\mathcal{K}} & \mathbf{I} & \\ -(\mathbf{T}^k_{\mathbf{x}})_{O\mathcal{K}} & & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} (\mathbf{T}^l_{\mathbf{x}})_{\mathcal{K}I} \\ (\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}I} \\ (\mathbf{T}^k_{\mathbf{x}})_{OI} \end{bmatrix}.
$$

See Appendix J on how to determine the full covariance matrices in the union and intersection experiments.

Appendix H. Proof of Lemma 13 (Perturbation of B)

Experiments $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ with $x_i \in \mathcal{J}_k, x_j \in \mathcal{U}_k$ do not have to be considered as the pair condition is not satisfied for the pair (x_i, x_j). Consider then experiments $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ with $x_j \in \mathcal{J}_k$. As explained in the text after Lemma 13, $\mathbf{B}$ and $\tilde{\mathbf{B}}$ differ only on the j:th row. Then, if $x_j \in \mathcal{J}_k$, we have that $\mathbf{U}_k\tilde{\mathbf{B}} = \mathbf{U}_k\mathbf{B}$ and the experimental effects must be equal.

That leaves us with experiments $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ with $x_i \in \mathcal{U}_k$ and $x_j \in \mathcal{U}_k$. In the special case of experiment $\mathcal{E}_{k'} = (\mathcal{K}, \mathcal{L}) = (\mathcal{V} \setminus \{x_i, x_j\}, \{x_i, x_j\})$, the experimental effects are the same by the definition of the alternative coefficient matrix $\tilde{\mathbf{B}}$:

$$\tilde{\mathbf{T}}^{k'}_{\mathbf{x}} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1}\tilde{\mathbf{B}}_{\mathcal{L}\mathcal{K}} = (\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})^{-1}(\mathbf{I} - \tilde{\mathbf{B}}_{\mathcal{L}\mathcal{L}})(\mathbf{I} - \mathbf{B}_{\mathcal{L}\mathcal{L}})^{-1}\mathbf{B}_{\mathcal{L}\mathcal{K}} = (\mathbf{I} - \mathbf{B}_{\mathcal{L}\mathcal{L}})^{-1}\mathbf{B}_{\mathcal{L}\mathcal{K}} = \mathbf{T}^{k'}_{\mathbf{x}}.$$

Otherwise the intervention set $\mathcal{J}_k$ has a presentation $\mathcal{J}_k = \mathcal{K} \cap (\mathcal{J}_k \cup \mathcal{L})$. We just noted that the experimental effects are the same in experiment $(\mathcal{K}, \mathcal{L})$. Earlier we showed that experimental effects are equal when x_j is intervened on, this holds in particular for experiment $(\mathcal{J}_k \cup \mathcal{L}, \mathcal{U}_k \setminus \mathcal{L})$. By Lemma 9 (Union/Intersection Experiment) the effects of an intersection experiment $\mathcal{E}_k$ are defined by the experimental effects of the two original experiments, so the experimental effects must be equal in experiment $\mathcal{E}_k$.

Appendix I. Proof of Lemma 14 (Perturbation of $\boldsymbol{\Sigma}_{\mathbf{e}}$)

Take any experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$. The two models $(\mathbf{B}, \boldsymbol{\Sigma}_{\mathbf{e}})$ and $(\tilde{\mathbf{B}}, \tilde{\boldsymbol{\Sigma}}_{\mathbf{e}})$ produce the same experimental effects. Then, we can prove the following identities:

$$\begin{aligned}
\mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k &= \mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k, && (36)\\
(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k, && (37)\\
(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-T} &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-T}, && (38)\\
(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{U}_k(\mathbf{I} - \tilde{\mathbf{B}}) &= (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I} - \mathbf{B}). && (39)
\end{aligned}$$

Equation 36 follows directly from the fact that the experimental effects of the two models are the same in experiment $\mathcal{E}_k$. Equation 37 is proven by the following:

$$\begin{aligned}
& (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k && ||\mathbf{U}_k + \mathbf{J}_k = \mathbf{I}\\
= \; & \mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k + \mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k && ||\mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}}) = \mathbf{J}_k\\
= \; & \mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k + \mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k\\
= \; & \mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k + \mathbf{J}_k && ||\text{Eq. 36}\\
= \; & \mathbf{U}_k(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k + \mathbf{J}_k = (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k.
\end{aligned}$$

Equation 38 follows from Equation 37:

$$\begin{aligned}
& (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-T} && ||\mathbf{J}_k\mathbf{J}_k = \mathbf{J}_k, \mathbf{J}_k = \mathbf{J}_k^T\\
= \; & (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k\mathbf{J}_k^T(\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\
= \; & (\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k((\mathbf{I} - \mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k)^T && ||\text{Eq. 37}\\
= \; & (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k((\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k)^T\\
= \; & (\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I} - \mathbf{U}_k\mathbf{B})^{-T}.
\end{aligned}$$

Equation 39 is proven by the following:

$$\begin{aligned}(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{U}_k(\mathbf{I}-\tilde{\mathbf{B}}) &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}}-\mathbf{J}_k)\\ &= \mathbf{I}-(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k \quad ||\text{Eq. 37}\\ &= \mathbf{I}-(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k = (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I}-\mathbf{B}).\end{aligned}$$

Finally, the covariance matrices produced by the two models can be shown to be equal:

$$\begin{aligned}\tilde{\mathbf{C}}^k_{\mathbf{x}} &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}(\mathbf{J}_k+\mathbf{U}_k\tilde{\mathbf{\Sigma}}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T} \quad ||\text{Definition of } \tilde{\mathbf{\Sigma}}_{\mathbf{e}}\\ &= (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{J}_k(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}+ \quad ||\text{Eq. 38 and 39}\\ &\quad (\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-1}\mathbf{U}_k(\mathbf{I}-\tilde{\mathbf{B}})(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}-\mathbf{B})^{-T}(\mathbf{I}-\tilde{\mathbf{B}})^T\mathbf{U}_k(\mathbf{I}-\mathbf{U}_k\tilde{\mathbf{B}})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{J}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}+\\ &\quad (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\mathbf{U}_k(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_{\mathbf{e}}(\mathbf{I}-\mathbf{B})^{-T}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_k(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}\\ &= (\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}=\mathbf{C}^k_{\mathbf{x}}.\end{aligned}$$

Appendix J. Covariance Matrices of Union and Intersection Experiments

Even if the set of experiments does not allow for the identification of the full model, consistent predictions are still possible in some unseen experimental settings assuming the data generating model is a linear cyclic model with latent variables. Lemma 9 already showed that the experimental effects can be predicted in the union and intersection experiments of any two already conducted experiments. In the following we extend this result to the prediction of the entire covariance matrices.

Let the data generating model be $(\mathbf{B},\mathbf{\Sigma}_{\mathbf{e}})$. Say we have conducted experiment $\mathcal{E}_k$ observing covariance matrix $\mathbf{C}^k_{\mathbf{x}}$ and experiment $\mathcal{E}_l$ observing covariance matrix $\mathbf{C}^l_{\mathbf{x}}$. By solving Equation 17 using the pseudoinverse we can find a matrix $\tilde{\mathbf{B}}$ that produces the same experimental effects in the two experiments. Now define

$$\begin{aligned}\tilde{\mathbf{M}}_1 &:= (\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-T},\\ \tilde{\mathbf{M}}_2 &:= (\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-1}\mathbf{U}_{k\cup l}(\mathbf{I}-\tilde{\mathbf{B}}).\end{aligned}$$

using the estimate $\tilde{\mathbf{B}}$. Now, we can show that matrix $\tilde{\mathbf{M}}_1+\tilde{\mathbf{M}}_2\mathbf{C}^k_{\mathbf{x}}\tilde{\mathbf{M}}_2^T$ is equal to the covariance matrix $\mathbf{C}^{k\cup l}_{\mathbf{x}}$ that the true data generating model would produce in experiment $\mathcal{E}_{k\cup l}=(\mathcal{J}_{k\cup l},\mathcal{U}_{k\cup l})=(\mathcal{J}_k\cup\mathcal{J}_l,\mathcal{U}_k\cap\mathcal{U}_l)$:

$$\begin{aligned}&\tilde{\mathbf{M}}_1+\tilde{\mathbf{M}}_2\mathbf{C}^k_{\mathbf{x}}\tilde{\mathbf{M}}_2^T\\ =\ &(\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-T} \quad ||\text{Eq. 38 and 39}\\ &+(\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-1}\mathbf{U}_{k\cup l}(\mathbf{I}-\tilde{\mathbf{B}})\mathbf{C}^k_{\mathbf{x}}(\mathbf{I}-\tilde{\mathbf{B}})^T\mathbf{U}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\tilde{\mathbf{B}})^{-T}\\ =\ &(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \quad ||\text{Eq. 8}\\ &+(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{U}_{k\cup l}(\mathbf{I}-\mathbf{B})\mathbf{C}^k_{\mathbf{x}}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T}\\ =\ &(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T}+(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{U}_{k\cup l}(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\\ &\cdot(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \quad ||\mathbf{U}_{k\cup l}=\mathbf{U}_l\mathbf{U}_k\mathbf{U}_k\\ =\ &(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T}+(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{U}_l\mathbf{U}_k\mathbf{U}_k(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1}\\ &\cdot(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_{\mathbf{e}}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_k\mathbf{U}_k\mathbf{U}_l(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \quad ||\mathbf{U}_k=\mathbf{I}-\mathbf{J}_k\end{aligned}$$

$$
\begin{aligned}
&= (\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} + (\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{U}_l\mathbf{U}_k((\mathbf{I}-\mathbf{U}_k\mathbf{B})-\mathbf{J}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-1} \\
&\quad \cdot(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_\mathbf{e}\mathbf{U}_k)(\mathbf{I}-\mathbf{U}_k\mathbf{B})^{-T}((\mathbf{I}-\mathbf{U}_k\mathbf{B})-\mathbf{J}_k)^T\mathbf{U}_k\mathbf{U}_l(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \quad ||\mathbf{U}_k\mathbf{J}_k=\mathbf{0}_{n\times n} \\
&= (\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{J}_{k\cup l}(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \\
&\quad +(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}\mathbf{U}_l\mathbf{U}_k(\mathbf{J}_k+\mathbf{U}_k\mathbf{\Sigma}_\mathbf{e}\mathbf{U}_k)\mathbf{U}_k\mathbf{U}_l(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T} \quad ||\mathbf{U}_l\mathbf{U}_k\mathbf{U}_k=\mathbf{U}_{k\cup l} \\
&= (\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-1}(\mathbf{J}_{k\cup l}+\mathbf{U}_{k\cup l}\mathbf{\Sigma}_\mathbf{e}\mathbf{U}_{k\cup l})(\mathbf{I}-\mathbf{U}_{k\cup l}\mathbf{B})^{-T}=\mathbf{C}_\mathbf{x}^{k\cup l}.
\end{aligned}
$$

To predict the whole covariance matrix in the intersection experiment, we need the passive observational data covariance matrix $\mathbf{C}_\mathbf{x}^0$ in addition to the observations in experiments $\mathcal{E}_k$ and $\mathcal{E}_l$. Now, define matrices

$$
\begin{aligned}
\tilde{\mathbf{M}}_3 &:= (\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-1}\mathbf{J}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-T}, \\
\tilde{\mathbf{M}}_4 &:= (\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-1}\mathbf{U}_{k\cap l}(\mathbf{I}-\tilde{\mathbf{B}}).
\end{aligned}
$$

Then, we can show that $\tilde{\mathbf{M}}_3+\tilde{\mathbf{M}}_4\mathbf{C}_\mathbf{x}^0\tilde{\mathbf{M}}_4^T$ is equal to the covariance matrix $\mathbf{C}_\mathbf{x}^{k\cap l}$ that the data generating model would produce in experiment $\mathcal{E}_{k\cap l}=(\mathcal{J}_{k\cap l},\mathcal{U}_{k\cap l})=(\mathcal{J}_k\cap\mathcal{J}_l,\mathcal{U}_k\cup\mathcal{U}_l)$:

$$
\begin{aligned}
&\tilde{\mathbf{M}}_3+\tilde{\mathbf{M}}_4\mathbf{C}_\mathbf{x}^0\tilde{\mathbf{M}}_4^T \\
&= (\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-1}\mathbf{J}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-T} \quad ||\text{Eq. 38 and 39} \\
&\quad +(\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-1}\mathbf{U}_{k\cap l}(\mathbf{I}-\tilde{\mathbf{B}})\mathbf{C}_\mathbf{x}^0(\mathbf{I}-\tilde{\mathbf{B}})^T\mathbf{U}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\tilde{\mathbf{B}})^{-T} \\
&= (\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-1}\mathbf{J}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-T} \quad ||\text{Eq. 3} \\
&\quad +(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-1}\mathbf{U}_{k\cap l}(\mathbf{I}-\mathbf{B})\mathbf{C}_\mathbf{x}^0(\mathbf{I}-\mathbf{B})^T\mathbf{U}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-T} \\
&= (\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-1}\mathbf{J}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-T} \\
&\quad +(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-1}\mathbf{U}_{k\cap l}(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{B})^{-1}\mathbf{\Sigma}_\mathbf{e}(\mathbf{I}-\mathbf{B})^{-T}(\mathbf{I}-\mathbf{B})^T\mathbf{U}_{k\cap l}(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-T} \\
&= (\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-1}(\mathbf{J}_{k\cap l}+\mathbf{U}_{k\cap l}\mathbf{\Sigma}_\mathbf{e}\mathbf{U}_{k\cap l})(\mathbf{I}-\mathbf{U}_{k\cap l}\mathbf{B})^{-T}=\mathbf{C}_\mathbf{x}^{k\cap l}.
\end{aligned}
$$

The above formulas for the prediction of covariance matrices can be used iteratively to find consistent estimates for the covariance matrices in different experiments, as long as the intervention set of the experiment can be reached by taking successive unions and intersections from the intervention sets of the actually conducted experiments.[12]

Appendix K. LLC Algorithm

We show here that matrix $\mathbf{T}$ of the LLC learning method is full column rank if the pair condition is satisfied for all pairs. This implies that the coefficients or direct effects are fully identified.

First we show that the equations of the type of Equation 16 obtained in the union experiment $\mathcal{E}_{k\cup l}$ are merely linear combinations of equations obtained in experiment $\mathcal{E}_k$ and $\mathcal{E}_l$. This is a rather direct consequence of Lemma 9 and its proof in Appendix G. In an arbitrary experiment $\mathcal{E}_k$, equations for all pairs (x_i,x_u) with $x_i\in\mathcal{J}_k$ and $x_u\in\mathcal{U}_k$, can be represented neatly in matrix notation:

$$
\begin{aligned}
\mathbf{B}_{\{x_u\}\mathcal{J}_k}+\mathbf{B}_{\{x_u\}(\mathcal{U}_k\setminus\{x_u\})}(\mathbf{T}_\mathbf{x}^k)_{(\mathcal{U}_k\setminus\{x_u\})\mathcal{J}_k} &= (\mathbf{T}_\mathbf{x}^k)_{\{x_u\}\mathcal{J}_k} \quad \Leftrightarrow \\
(\mathbf{B}_{\{x_u\}\mathcal{J}_k})^T+((\mathbf{T}_\mathbf{x}^k)_{(\mathcal{U}_k\setminus\{x_u\})\mathcal{J}_k})^T(\mathbf{B}_{\{x_u\}(\mathcal{U}_k\setminus\{x_u\})})^T &= ((\mathbf{T}_\mathbf{x}^k)_{\{x_u\}\mathcal{J}_k})^T.
\end{aligned}
$$

12. Note that if $\boldsymbol{\mu}_\mathbf{e}\neq\mathbf{0}$, $\tilde{\mathbf{M}}_2\boldsymbol{\mu}_\mathbf{x}^k$ and $\tilde{\mathbf{M}}_4\boldsymbol{\mu}_\mathbf{x}^0$ provide estimates for the observed means in the union and intersection experiments.

Now, partition $\mathcal{V}$ similarly as in Appendix G. Consider an arbitrary $x_u \in O$ (observed in both experiments). Define $\tilde{O} = O \setminus \{x_u\}$. Equations corresponding to pairs (x_i, x_u) with $x_i \in I \cup \mathcal{K}$ obtained in experiment $\mathcal{E}_k$ and equations corresponding to pairs (x_j, x_u) with $x_j \in \mathcal{L}$ obtained in experiment $\mathcal{E}_l$ can be collected into a single system constraining coefficients $b_{u\bullet}$:

$$\left[\begin{array}{ccc|c} \mathbf{I} & & ((\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}I})^T & ((\mathbf{T}^k_{\mathbf{x}})_{\tilde{O}I})^T \\ & \mathbf{I} & ((\mathbf{T}^k_{\mathbf{x}})_{\mathcal{L}\mathcal{K}})^T & ((\mathbf{T}^k_{\mathbf{x}})_{\tilde{O}\mathcal{K}})^T \\ & ((\mathbf{T}^l_{\mathbf{x}})_{\mathcal{K}\mathcal{L}})^T & \mathbf{I} & ((\mathbf{T}^l_{\mathbf{x}})_{\tilde{O}\mathcal{L}})^T \end{array}\right] \left[\begin{array}{c} (\mathbf{B}_{\{x_u\}I})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{K}})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{L}})^T \\ (\mathbf{B}_{\{x_u\}\tilde{O}})^T \end{array}\right] = \left[\begin{array}{c} ((\mathbf{T}^k_{\mathbf{x}})_{\{x_u\}I})^T \\ ((\mathbf{T}^k_{\mathbf{x}})_{\{x_u\}\mathcal{K}})^T \\ ((\mathbf{T}^l_{\mathbf{x}})_{\{x_u\}\mathcal{L}})^T \end{array}\right]. \quad (40)$$

Notice, that the left-hand block of the matrix on the left is just the transpose of the $\mathbf{Q}$ matrix introduced in Appendix G. As $\mathbf{Q}$ was shown to be invertible under the assumption that the data generating model is weakly stable, we can multiply the equation group by $\mathbf{Q}^{-T}$ from the left. As blocks of Equation 35 in Appendix G we get the following identities:

$$\mathbf{Q}^{-T}\left[\begin{array}{c} ((\mathbf{T}^k_{\mathbf{x}})_{\{x_u\}I})^T \\ ((\mathbf{T}^k_{\mathbf{x}})_{\{x_u\}\mathcal{K}})^T \\ ((\mathbf{T}^l_{\mathbf{x}})_{\{x_u\}\mathcal{L}})^T \end{array}\right] = \left[\begin{array}{c} ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}I})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{K}})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{L}})^T \end{array}\right],$$

$$\mathbf{Q}^{-T}\left[\begin{array}{c} ((\mathbf{T}^k_{\mathbf{x}})_{\tilde{O}I})^T \\ ((\mathbf{T}^k_{\mathbf{x}})_{\tilde{O}\mathcal{K}})^T \\ ((\mathbf{T}^l_{\mathbf{x}})_{\tilde{O}\mathcal{L}})^T \end{array}\right] = \left[\begin{array}{c} ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}I})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{K}})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{L}})^T \end{array}\right].$$

Thus, multiplying the Equation 40 from the left by $\mathbf{Q}^{-T}$ produces the following equation system:

$$\left[\begin{array}{ccc|c} \mathbf{I} & & & ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}I})^T \\ & \mathbf{I} & & ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{K}})^T \\ & & \mathbf{I} & ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{L}})^T \end{array}\right] \left[\begin{array}{c} (\mathbf{B}_{\{x_u\}I})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{K}})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{L}})^T \\ (\mathbf{B}_{\{x_u\}\tilde{O}})^T \end{array}\right] = \left[\begin{array}{c} ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}I})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{K}})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{L}})^T \end{array}\right]$$

$$\Leftrightarrow \quad \left[\begin{array}{c} (\mathbf{B}_{\{x_u\}I})^T + ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}I})^T(\mathbf{B}_{\{x_u\}\tilde{O}})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{K}})^T + ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{K}})^T(\mathbf{B}_{\{x_u\}\tilde{O}})^T \\ (\mathbf{B}_{\{x_u\}\mathcal{L}})^T + ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\tilde{O}\mathcal{L}})^T(\mathbf{B}_{\{x_u\}\tilde{O}})^T \end{array}\right] = \left[\begin{array}{c} ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}I})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{K}})^T \\ ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{L}})^T \end{array}\right].$$

For the union experiment $\mathcal{E}_{k\cup l} = (\mathcal{J}_{k\cup l}, \mathcal{U}_{k\cup l})$ we have that $I \cup \mathcal{K} \cup \mathcal{L} = \mathcal{J}_{k\cup l}$ and $\tilde{O} = \mathcal{U}_{k\cup l} \setminus \{x_u\}$. The equation system can be written in in the following simple form:

$$(\mathbf{B}_{\{x_u\}\mathcal{J}_{k\cup l}})^T + ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{(\mathcal{U}_{k\cup l}\setminus\{x_u\})\mathcal{J}_{k\cup l}})^T(\mathbf{B}_{\{x_u\}(\mathcal{U}_{k\cup l}\setminus\{x_u\})})^T = ((\mathbf{T}^{k\cup l}_{\mathbf{x}})_{\{x_u\}\mathcal{J}_{k\cup l}})^T.$$

These are all of the equations from experiment $\mathcal{E}_{k\cup l}$ constraining coefficients $b_{u\bullet}$. As we considered arbitrary $x_u \in O$, the same procedure can be repeated for each $x_u \in O$. This exhausts all equations obtained in the union experiment. All of the equations obtained in experiment $\mathcal{E}_{k\cup l}$ are thus linear combinations of some of the equations obtained in the original two experiments $\mathcal{E}_k$ and $\mathcal{E}_l$.

Finally, matrix $\mathbf{T}$ can be verified to have full column rank as follows. Matrix $\mathbf{T}$ being full column rank is equivalent to system $\mathbf{Tb} = \mathbf{t}$ having at most a unique solution. The original equation system $\mathbf{Tb} = \mathbf{t}$ consists of all the equations (like Equation 16) gathered in experiments $\{\mathcal{E}_k\}_{k=1,\ldots,K}$. We can always add equations that would be obtained in the union experiment $\mathcal{E}_{k\cup l}$ of two experiments $\mathcal{E}_k$

and $\mathcal{E}_l$ whose equations are already in the system, without further restricting the possible solutions of the system. This is because the added equations are merely linear combinations of some of the equations already in the system. If the pair condition is satisfied for all pairs, by adding always equations from the union experiments of two experiments, whose equations are already in the system, we are eventually able to add equations for experiments intervening on sets $\mathcal{V} \setminus \{x_u\}$, for all variables $x_u \in \mathcal{V}$ (this follows the rationale discussed after Definition 10). These equations specify the direct effects $\mathbf{b}$ directly and uniquely. Since the solution space was not restricted throughout the procedure of adding new equations, we can deduce that the original system had at most a unique solution, which implies that the original matrix $\mathbf{T}$ has full column rank.

References

K. A. Bollen. *Structural Equations with Latent Variables*. John Wiley & Sons, 1989.

D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, 2002a.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002b.

T. Claassen and T. Heskes. Causal discovery in multiple models from different experiments. In *Advances in Neural Information Processing Systems 23*, pages 415–423, 2010.

G. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 116–125, 1999.

D. Eaton and K. Murphy. Exact Bayesian structure learning from uncertain interventions. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, 2007.

F. Eberhardt and R. Scheines. Interventions and causal inference. *Philosophy of Science*, 74:5: 981–995, 2007.

F. Eberhardt, C. Glymour, and R. Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Proceedings of the Twenty-First Conference Conference on Uncertainty in Artificial Intelligence*, pages 178–184, 2005.

F. Eberhardt, P. O. Hoyer, and R. Scheines. Combining experiments to discover linear cyclic models with latent variables. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

F. M. Fisher. A correspondence principle for simultaneous equation models. *Econometrica*, 38(1): pp. 73–92, 1970.

R. A. Fisher. *The Design of Experiments*. Hafner, 1935.

D. Geiger and D. Heckerman. Learning Gaussian networks. Technical Report MSR-TR-94-10, Microsoft Research, 1994.

C. Glymour, R. Scheines, P. Spirtes, and K. Kelly. *Discovering Causal Structure: Artificial Intelligence, Philosophy of Science and Statistical Modeling*. Academic Press, 1987.

G. H. Hardy. *Divergent Series*. Oxford: Clarendon Press, 1949.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Causal discovery for linear cyclic models with latent variables. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models*, 2010.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Noisy-or models with latent confounding. In *Proceedings of the Twenty-Seventh Conference Conference on Uncertainty in Artificial Intelligence*, 2011.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Experiment selection for causal discovery. Submitted, 2012.

S. Itani, M. Ohannessian, K. Sachs, G. P. Nolan, and M. A. Dahleh. Structure learning in causal cyclic networks. In *JMLR Workshop & Conference Proceedings*, volume 6, pages 165–176, 2008.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

S. L. Lauritzen and T. S. Richardson. Chain graph models and their causal interpretations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64:321–348, 2002.

D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239, 2009.

S. J. Mason. Feedback theory : further properties of signal flow graphs. Technical Report 303, Research Laboratory of Electronics, Massachusetts Institute of Technology, 1956.

S. Meganck, B. Manderick, and P. Leray. A decision theoretic approach to learning Bayesian networks. Technical report, Vrije Universiteit Brussels, 2005.

K. P. Murphy. Active learning of causal Bayes net structure. Technical report, U.C. Berkeley, 2001.

E. Nyberg and K. Korb. Informative interventions. In *Causality and Probability in the Sciences*. College Publications, London, 2006.

J. Pearl. *Causality*. Oxford University Press, 2000.

R. J. Prill, D. Marbach, J. Saez-Rodriguez, P. K. Sorger, L. G. Alexopoulos, X. Xue, N. D. Clarke, G. Altan-Bonnet, and G. Stolovitzky. Towards a rigorous assessment of systems biology models: The DREAM 3 challenges. *PLoS ONE*, 5(2):e9202, 2010.

T. S. Richardson. *Feedback Models: Interpretation and Discovery*. PhD thesis, Carnegie Mellon, 1996.

M. Schmidt and K. Murphy. Modeling discrete interventional data using directed cyclic graphical models. In *Proceedings of the Twenty-Fifth Conference Conference on Uncertainty in Artificial Intelligence*, 2009.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. J. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

J. Spencer. Minimal completely separating systems. *Journal of Combinatorial Theory*, 8(4):446 – 447, 1970.

P. Spirtes. Directed cyclic graphical representation of feedback models. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, 1995.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. MIT Press, 2nd edition, 2000.

G. Stolovitzky, R. J. Prill, and A. Califano. Lessons from the DREAM 2 challenges. *Annals of the New York Academy of Sciences*, 1158(1):159–195, 2009.

S. Tong and D. Koller. Active learning for structure in Bayesian networks. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 863–869, 2001.

T. Verma and J. Pearl. Causal networks: Semantics and expressiveness. In *Proceedings of the Fourth Conference Conference on Uncertainty in Artificial Intelligence*, pages 352–359, 1988.

S. Wright. The method of path coefficients. *The Annals of Mathematical Statistics*, 5(3):pp. 161–215, 1934.

Journal of Machine Learning Research 13 (2012) 3441-3473 Submitted 9/11; Revised 5/12; Published 11/12

Iterative Reweighted Algorithms for Matrix Rank Minimization

Karthik Mohan KARNA@UW.EDU
Maryam Fazel MFAZEL@UW.EDU
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-4322, USA

Editor: Tong Zhang

Abstract

The problem of minimizing the rank of a matrix subject to affine constraints has applications in several areas including machine learning, and is known to be NP-hard. A tractable relaxation for this problem is nuclear norm (or trace norm) minimization, which is guaranteed to find the minimum rank matrix under suitable assumptions. In this paper, we propose a family of Iterative Reweighted Least Squares algorithms IRLS-p (with $0 \leq p \leq 1$), as a computationally efficient way to improve over the performance of nuclear norm minimization. The algorithms can be viewed as (locally) minimizing certain smooth approximations to the rank function. When $p = 1$, we give theoretical guarantees similar to those for nuclear norm minimization, that is, recovery of low-rank matrices under certain assumptions on the operator defining the constraints. For $p < 1$, IRLS-p shows better empirical performance in terms of recovering low-rank matrices than nuclear norm minimization. We provide an efficient implementation for IRLS-p, and also present a related family of algorithms, sIRLS-p. These algorithms exhibit competitive run times and improved recovery when compared to existing algorithms for random instances of the matrix completion problem, as well as on the MovieLens movie recommendation data set.

Keywords: matrix rank minimization, matrix completion, iterative algorithms, null-space property

1. Introduction

The Affine Rank Minimization Problem (ARMP), or the problem of finding the minimum rank matrix in an affine set, arises in many engineering applications such as collaborative filtering (e.g., Candes and Recht, 2009; Srebro et al., 2005), low-dimensional Euclidean embedding (e.g., Fazel et al., 2003), low-order model fitting and system identification (e.g., Liu and Vandenberghe, 2008), and quantum tomography (e.g., Gross et al., 2010). The problem is as follows,

$$\begin{array}{ll} \text{minimize} & \text{rank}(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{1}$$

where $X \in \mathbb{R}^{m \times n}$ is the optimization variable, $\mathcal{A} : \mathbb{R}^{m \times n} \to \mathbb{R}^q$ is a linear map, and $b \in \mathbb{R}^q$ denotes the measurements. When X is restricted to be a diagonal matrix, ARMP reduces to the cardinality

minimization or sparse vector recovery problem,

$$\begin{array}{ll} \text{minimize} & \text{card}(x) \\ \text{subject to} & Ax = b, \end{array}$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $\text{card}(x)$ denotes the number of non-zeros entries of x. A commonly used convex heuristic for this problem is ℓ_1 minimization. The field of compressed sensing has shown that under certain conditions on the matrix A, this heuristic solves the original cardinality minimization problem (Candes and Tao, 2004). There are also geometric arguments in favor of ℓ_1 norm as a good convex relaxation (see, e.g., Chandrasekaran et al., 2010b, for a unifying analysis for general linear inverse problems). It has also been observed empirically (e.g., Candes et al., 2008; Lobo et al., 2006) that by appropriately weighting the ℓ_1 norm and iteratively updating the weights, the recovery performance of the algorithm is enhanced. It is shown theoretically that for sparse recovery from noisy measurements, this algorithm has a better recovery error than ℓ_1 minimization under suitable assumptions on the matrix A (Needell, 2009; Zhang, 2010). The Reweighted ℓ_1 algorithm has also been generalized to the recovery of low-rank matrices (Fazel et al., 2003; Mohan and Fazel, 2010a).

Another simple and computationally efficient reweighted algorithm for sparse recovery that has been proposed in the literature (Daubechies et al., 2010), is the Iterative Reweighted Least Squares algorithm (IRLS-p, for any $0 < p \leq 1$). Its kth iteration is given by

$$x^{k+1} \quad = \quad \underset{x}{\text{argmin}} \left\{ \sum_i w_i^k x_i^2 : Ax = b, \right\},$$

where $w^k \in \mathbb{R}^n$ is a weight vector with $w_i^k = (|x_i^k|^2 + \gamma)^{p/2-1}$ ($\gamma > 0$ being a regularization parameter added to ensure that w^k is well defined). For $p = 1$, a theoretical guarantee for sparse recovery has been given for IRLS-p (Daubechies et al., 2010), similar to the guarantees for ℓ_1 minimization. The starting point of the algorithm, x^0 is set to zero. So the first iteration gives the least norm solution to $Ax = b$. It was empirically observed (Chartrand and Staneva, 2008; Chartrand and Yin, 2008) that IRLS-p shows a better recovery performance than ℓ_1 minimization for $p < 1$ and has a similar performance as the reweighted ℓ_1 algorithm when p is set to zero. The computational benefits of IRLS-1, as well as the above mentioned theoretical and empirical improvements of IRLS-p, $p < 1$, over ℓ_1 minimization, motivate us to ask: Would an iterative reweighted algorithm bring similar benefits for recovery of low rank matrices, and would an improved performance be at the price of additional computational cost relative to the standard nuclear norm minimization?

1.1 Iterative Reweighted Least Squares for ARMP

Towards answering this question, we propose the iterative reweighted least squares algorithm for rank minimization, outlined below.

Data: $\mathcal{A}$, b
Set $k = 1$. Initialize $W_p^0 = I, \gamma^1 > 0$;
while *not converged* **do**
  $X^k = \underset{X}{\operatorname{argmin}} \{\operatorname{Tr}(W_p^{k-1} X^T X) : \mathcal{A}(X) = b\}$;
  $W_p^k = (X^{k^T} X^k + \gamma^k I)^{\frac{p}{2}-1}$;
  Choose $0 < \gamma^{k+1} \leq \gamma^k$;
  $k = k+1$;
end

Algorithm 1: IRLS-p Algorithm for Matrix Rank Minimization with $0 \leq p \leq 1$

Each iteration of Algorithm 1 minimizes a weighted Frobenius norm of the matrix X, since $\operatorname{Tr}(W_p^{k-1} X^T X) = \|(W_p^{k-1})^{1/2} X\|_F^2$. While minimizing the Frobenius norm subject to affine constraints doesn't lead to low-rank solutions in general, through a careful reweighting of this norm we show that Algorithm 1 does indeed produce low-rank solutions under suitable assumptions. Usually a reweighted algorithm trades off computational time for improved recovery performance when compared to the unweighted convex heuristic. As an example, the reweighted ℓ_1 algorithm for sparse recovery (Candes et al., 2008) and the reweighted nuclear norm algorithm (Mohan and Fazel, 2010a) for matrix rank minimization solve the corresponding standard convex relaxations (ℓ_1 and nuclear norm minimization respectively) in their first iteration. Thus these algorithms take at least as much time as the corresponding convex algorithms. However, the iterates of the IRLS-p family of algorithms simply minimize a weighted Frobenius norm, and have run-times comparable with the nuclear norm heuristic, while (for $p < 1$) also enjoy improved recovery performance. In the $p = 1$ case, we show that the algorithm minimizes a certain smooth approximation to the nuclear norm, allowing for efficient implementations while having theoretical guarantees similar to nuclear norm minimization.

1.2 Contributions

Contributions of this paper are as follows.

- We give a convergence analysis for IRLS-p for all $0 \leq p \leq 1$. We also propose a related algorithm, sIRLS-p (or short IRLS), which can be seen as a first-order method for locally minimizing a smooth approximation to the rank function, and give convergence results for this algorithm as well. The results exploit the fact that these algorithms can be derived from the KKT conditions for minimization problems whose objectives are suitable smooth approximations to the rank function.

- We prove that the IRLS-1 algorithm is guaranteed to recover a low-rank matrix if the linear map $\mathcal{A}$ satisfies the following *Null space property* (NSP). We show this property is both necessary and sufficient for low-rank recovery. The specific NSP that we use first appeared in the literateure recently (Oymak and Hassibi, 2010; Oymak et al., 2011), and is expressed only in terms of the singular values of the matrices in the null space of $\mathcal{A}$.

Definition 1 *Given $\tau > 0$, the linear map $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ satisfies the* τ-Null space Property *(τ-NSP) of order r if for every $Z \in \mathcal{N}(\mathcal{A})\backslash\{0\}$, we have*

$$\sum_{i=1}^{r} \sigma_i(Z) \quad < \quad \tau \sum_{i=r+1}^{n} \sigma_i(Z) \tag{2}$$

where $\mathcal{N}(\mathcal{A})$ denotes the null space of $\mathcal{A}$ and $\sigma_i(Z)$ denotes the i^{th} largest singular value of Z.

It has been shown in the literature (Oymak and Hassibi, 2010) that certain random Gaussian maps $\mathcal{A}$ satisfy this property with high probability.

- We give a gradient projection algorithm to implement IRLS-p. We extensively compare these algorithms with other state of the art algorithms on both 'easy' and 'hard' randomly picked instances of the matrix completion problem (these notions are made precise in the numerical section). We also present comparisons on the MovieLens movie recommendation data set. Numerical experiments demonstrate that IRLS-0 and sIRLS-0 applied to the matrix completion problem have a better recovery performance than the Singular Value Thresholding algorithm, an implementation of Nuclear Norm Minimization (Cai et al., 2008), on both easy and hard instances. Importantly, in the case where there is no apriori information on the rank of the low rank solution (which is common in practice), our algorithm has a significantly better recovery performance on hard problem instances as compared to other state of the art algorithms for matrix completion including IHT, FPCA (Goldfarb and Ma, 2011), and Optspace (Keshavan and Oh, 2009).

1.3 Related Work

We review related algorithms for recovering sparse vectors and low rank matrices. Many approaches have been proposed for recovery of sparse vectors from linear measurements including ℓ_1 minimization and greedy algorithms (see, e.g., Needell and Tropp, 2008; Goldfarb and Ma, 2011; Garg and Khandekar, 2009, for CoSaMP, IHT, and GraDes respectively). As mentioned earlier, reweighted algorithms including Iterative reweighted ℓ_1 (Candes et al., 2008) and Iterative Reweighted Least Squares (Rao and Kreutz-Delgado, 1999; Wipf and Nagarajan, 2010; Daubechies et al., 2010) with $0 < p \leq 1$ have been proposed to improve on the recovery performance of ℓ_1 minimization.

For the ARMP, analogous algorithms have been proposed including nuclear norm minimization (Fazel et al., 2001), reweighted nuclear norm minimization (Mohan and Fazel, 2010a; Fazel et al., 2003), as well as greedy algorithms such as AdMiRA (Lee and Bresler, 2010) which generalizes CoSaMP, SVP (Meka et al., 2010), a hard-thresholding algorithm that we also refer to as IHT, and Optspace (Keshavan and Oh, 2009). Developing efficient implementations for nuclear norm minimization is an important research area since standard semidefinite programming solvers cannot handle large problem sizes. Towards this end, algorithms including SVT (Cai et al., 2008), NNLS (Toh and Yun, 2010), FPCA (Goldfarb and Ma, 2011) have been proposed. A spectral regularization algorithm (Mazumder et al., 2010) has also been proposed for the specific problem of matrix completion.

In a preliminary conference version of this paper (Mohan and Fazel, 2010b), we proposed the IRLS-p family of algorithms for ARMP, analogous to IRLS for sparse vector recovery (Daubechies et al., 2010). The present paper gives a new and improved theoretical analysis for IRLS-1 via a simpler NSP condition, obtains complete convergence results for all $0 \leq p \leq 1$, and gives more extensive numerical experiments.

Independent of us and at around the same time as the publication of our conference paper, Fornasier et al. (2010) proposed the IRLSM algorithm, which is similar to our IRLS-1 algorithm but with a different weight update (a thresholding operation is used to ensure the weight matrix is invertible). The authors employ the Woodbury matrix inversion lemma to speed up the implementation of IRLSM and compare it to two other algorithms, Optspace and FPCA, showing that IRLSM has a lower relative error and comparable computational times. However, their analysis of low-rank recovery uses a Strong Null Space Property (SRNSP) (which is equivalent to the condition in, Mohan and Fazel, 2010b) and is less general than the condition we consider in this paper, as discussed in Section 3.

Also, the authors have only considered recovery of matrices of size 500×500 in their experiments. We present extensive comparisons of our algorithms with algorithms including Optspace, FPCA, and IHT for recovery of matrices of different sizes. We observe that our IRLS and sIRLS implementations run faster than IRLSM in practice. IRLSM solves a quadratic program in each iteration using a sequence of inversions, which can be expensive for large problems, even after exploiting the matrix inversion lemma. Finally the authors of the IRLSM paper only consider the $p = 1$ case.

The rest of the paper is organized as follows. We introduce the IRLS-p algorithm for ARMP in Section 2 and give convergence and performance guarantees in Section 3. In Section 4, we discuss an implementation for IRLS-p, tailored for the *matrix completion problem*, and in Section 5 we present the related algorithm sIRLS-p. Numerical experiments for IRLS-0 and sIRLS-0 for the matrix completion problem, as well as comparisons with SVT, IHT, IRLSM, Optspace and FPCA are given in Section 6. The last section summarizes the paper along with future research directions.

1.3.1 NOTATION

Let $\mathcal{N}(\mathcal{A})$ denote the null space of the operator $\mathcal{A}$ and $\mathrm{Ran}(\mathcal{A}^*)$ denote the range space of the adjoint of $\mathcal{A}$. Let $\sigma(X)$ denote the vector of decreasingly ordered singular values of X so that $\sigma_i(X)$ denotes the i^{th} largest singular value of X. Also let $\|X\|, \|X\|_F$ denote the spectral norm and Frobenius norm of X respectively. The nuclear norm is defined as $\|X\|_\star = \sum_i \sigma_i(X)$. I_k denotes the identity matrix of size $k \times k$.

2. Iterative Reweighted Least Squares (IRLS-p)

In this section, we describe the IRLS-p family of algorithms (Algorithm 1). Recall that replacing the rank function in (1) by $\|X\|_\star$ yields the nuclear norm heuristic,

$$\begin{array}{ll} \text{minimize} & \|X\|_\star \\ \text{subject to} & \mathcal{A}(X) = b. \end{array}$$

We now consider other (convex and non-convex) smooth approximations to the rank function. Define the *smooth Schatten-p function* as

$$\begin{aligned} f_p(X) &= \mathrm{Tr}(X^TX+\gamma I)^{p/2} \\ &= \textstyle\sum_{i=1}^n(\sigma_i^2(X)+\gamma)^{\frac{p}{2}}. \end{aligned}$$

Note that $f_p(X)$ is differentiable for $p>0$ and convex for $p\geq 1$. With $\gamma=0$, $f_1(X)=\|X\|_\star$, which is also known as the Schatten-1 norm. With $\gamma=0$ and $p\to 0$, $f_p(X)\to \mathrm{rank}(X)$. For $p=1$, one can also derive the smooth Schatten-1 function as follows.

$$f_1(X)=\max_{Z:\|Z\|_2\leq 1}\langle Z,X\rangle-\sqrt{\gamma}d(Z)+n\sqrt{\gamma},$$

where $d(Z)$ is a smoothing term given by $d(Z)=\sum_{i=1}^n(1-\sqrt{1-\sigma_i^2(Z)})=n-\mathrm{Tr}(I-Z^TZ)^{1/2}$. Plugging in $d(Z)=0$ in (3) gives $\|X\|_\star+n\sqrt{\gamma}$. Hence $f_1(X)$ is a smooth approximation of $\|X\|_\star$ obtained by smoothing the conjugate of $\|X\|_\star$. By virtue of the smoothing, it is easy to see that

$$\|X\|_\star\leq f_1(X)\leq\|X\|_\star+n\sqrt{\gamma}.$$

As γ approaches 0, $f_1(X)$ becomes tightly bounded around $\|X\|_\star$. Also let X^* be the optimal solution to minimizing $\|X\|_\star$ subject to $\mathcal{A}(X)=b$ and let X^s be the optimal solution to minimizing $f_1(X)$ subject to the same constraints. Then it holds that

$$(\|X\|_\star-\|X^*\|_\star)\leq(f_1(X)-f_1(X^s))+n\sqrt{\gamma}\quad\forall X:\mathcal{A}(X)=b.$$

Thus, to minimize $\|X\|_\star$ to a precision of ε, we need to minimize the smooth approximation $f_1(X)$ to a precision of $\varepsilon-n\sqrt{\gamma}$.

It is therefore of interest to consider the problem

$$\begin{array}{ll} \text{minimize} & f_p(X) \\ \text{subject to} & \mathcal{A}(X)=b, \end{array} \tag{3}$$

the optimality conditions of which motivate the IRLS-p algorithms for $0\leq p\leq 1$. We show that IRLS-1 solves the smooth Schatten-1 norm or nuclear norm minimization problem, that is, finds a globally optimal solution to (3) with $p=1$. For $p<1$, we show that IRLS-p finds a stationary point of (3).

We now give an intuitive way to derive the IRLS-p algorithm from the KKT conditions of (3). The Lagrangian corresponding to (3) is

$$L(X,\lambda)=f_p(X)+\langle\widehat{\lambda},\ \mathcal{A}(X)-b\rangle,$$

and the KKT conditions are $\nabla_X L(X,\widehat{\lambda})=0$, $\mathcal{A}(X)=b$. Note that $\nabla f_p(X)=pX(X^TX+\gamma I)^{p/2-1}$ (see, e.g., Lewis, 1996). Letting $\lambda=\frac{\widehat{\lambda}}{p}$, we have that the KKT conditions for (3) are given by

$$\begin{aligned} 2X(X^TX+\gamma I)^{p/2-1}+\mathcal{A}^*(\lambda) &= 0 \\ \mathcal{A}(X) &= b. \end{aligned} \tag{4}$$

Let $W_p^k = ({X^k}^T X^k + \gamma I)^{p/2-1}$. The first condition in (4) can be written as

$$X = -\frac{1}{2}\mathcal{A}^*(\lambda)(X^T X + \gamma I)^{1-p/2}.$$

This is a fixed point equation, and a solution can be obtained by iteratively solving for X as $X^{k+1} = \frac{1}{2}\mathcal{A}^*(\lambda)(W_p^k)^{-1}$, along with the condition $\mathcal{A}(X^{k+1}) = b$. Note that X^{k+1} and the dual variable λ satisfy the KKT conditions for the convex optimization problem,

$$\begin{array}{ll} \text{minimize} & \operatorname{Tr} W_p^k X^T X \\ \text{subject to} & \mathcal{A}(X) = b. \end{array}$$

This idea leads to the IRLS-p algorithm described in Algorithm 1. Note that we also let $p = 0$ in Algorithm 1; to derive IRLS-0, we define another non-convex surrogate function by taking limits over $f_p(X)$. For any positive scalar x, it holds that $\lim_{p\to 0} \frac{1}{p}(x^p - 1) = \log x$. Therefore,

$$\begin{array}{rcl} \lim\limits_{p\to 0} \dfrac{f_p(X) - n}{p} & = & \frac{1}{2}\log\det(X^T X + \gamma I) \\ & = & \sum_i \frac{1}{2}\log(\sigma_i^2(X) + \gamma). \end{array}$$

Thus IRLS-0 can be seen as iteratively solving (as outlined previously) the KKT conditions for the non-convex problem,

$$\begin{array}{ll} \text{minimize} & \log\det(X^T X + \gamma I) \\ \text{subject to} & \mathcal{A}(X) = b. \end{array} \tag{5}$$

Another way to derive the IRLS-p algorithm uses an alternative characterization of the Smooth Schatten-p function; see appendix A.

3. IRLS-p: Theoretical Results

In this section, convergence properties for the IRLS-p family of algorithms are studied. We also give a matrix recovery guarantee for IRLS-1, under suitable assumptions on the null space of the linear operator $\mathcal{A}$.

3.1 Convergence of IRLS-p

We show that the difference between successive iterates of the IRLS-p ($0 \le p \le 1$) algorithm converges to zero and that every cluster point of the iterates is a stationary point of (3). These results generalize the convergence results given for IRLS-1 in previous literature (Mohan and Fazel, 2010b; Fornasier et al., 2010) to IRLS-p with $0 < p \le 1$. In this section, we drop the subscript on W_p^k for ease of notation. Our convergence analysis relies on useful auxiliary functions defined as

$$\mathcal{J}^p(X, W, \gamma) := \begin{cases} \frac{p}{2}(\operatorname{Tr}(W(X^T X + \gamma I)) + \frac{2-p}{p}\operatorname{Tr}((W)^{\frac{p}{p-2}})) & \text{if } 0 < p \le 1 \\ \operatorname{Tr}(W(X^T X + \gamma I)) - \log\det W - n & \text{if } p = 0. \end{cases}$$

These functions can be obtained from the alternative characterization of Smooth Schatten-p function with details in Appendix A. We can express the iterates of IRLS-p as

$$\begin{aligned} X^{k+1} &= \operatorname*{argmin}_{X:\mathcal{A}(X)=b} \mathcal{J}^p(X, W^k, \gamma^k) \\ W^{k+1} &= \operatorname*{argmin}_{W\succ 0} \mathcal{J}^p(X^{k+1}, W, \gamma^{k+1}), \end{aligned}$$

and it follows that

$$\begin{aligned} \mathcal{J}^p(X^{k+1}, W^{k+1}, \gamma^{k+1}) &\leq \mathcal{J}^p(X^{k+1}, W^k, \gamma^{k+1}) \\ &\leq \mathcal{J}^p(X^{k+1}, W^k, \gamma^k) \\ &\leq \mathcal{J}^p(X^k, W^k, \gamma^k). \end{aligned} \tag{6}$$

The following theorem shows that the difference between successive iterates converges to zero. The proof is given in Appendix B.

Theorem 2 *Given any $b \in \mathbb{R}^q$, the iterates $\{X^k\}$ of IRLS-p $(0 < p \leq 1)$ satisfy*

$$\sum_{k=1}^{\infty} \|X^{k+1} - X^k\|_F^2 \leq 2D^{\frac{2}{p}},$$

where $D := \mathcal{J}^p(X^1, W^0, \gamma^0)$. In particular, we have that $\lim_{k\to\infty} \left(X^k - X^{k+1}\right) = 0$.

Theorem 3 *Let $\gamma_{\min} := \lim_{k\to\infty} \gamma^k > 0$. Then the sequence of iterates $\{X^k\}$ of IRLS-p $(0 \leq p \leq 1)$ is bounded, and every cluster point of the sequence is a stationary point of* (3) *(when $0 < p \leq 1$), or a stationary point of* (5) *(when $p = 0$).*

Proof Let

$$g(X, \gamma) = \begin{cases} \operatorname{Tr}(X^T X + \gamma I)^{\frac{p}{2}} & \text{if } 0 < p \leq 1 \\ \log\det(X^T X + \gamma I) & \text{if } p = 0 \end{cases}$$

Then $g(X^k, \gamma^k) = \mathcal{J}^p(X^k, W^k, \gamma^k)$ and it follows from (6) that $g(X^{k+1}, \gamma^{k+1}) \leq g(X^k, \gamma^k)$ for all $k \geq 1$. Hence the sequence $\{g(X^k, \gamma^k)\}$ converges. This fact together with $\gamma_{\min} > 0$ implies that the sequence $\{X^k\}$ is bounded.

We now show that every cluster point of $\{X^k\}$ is a stationary point of (3). Suppose to the contrary and let $\widetilde{X}$ be a cluster point of $\{X^k\}$ that is not a stationary point. By the definition of cluster point, there exists a subsequence $\{X^{n_i}\}$ of $\{X^k\}$ converging to $\widetilde{X}$. By passing to a further subsequence if necessary, we can assume that $\{X^{n_i+1}\}$ is also convergent and we denote its limit by $\widehat{X}$. By definition, X^{n_i+1} is the minimizer of

$$\begin{aligned} \text{minimize} \quad & \operatorname{Tr} W^{n_i} X^T X \\ \text{subject to} \quad & \mathcal{A}(X) = b. \end{aligned} \tag{7}$$

Thus, X^{n_i+1} satisfies the KKT conditions of (7), that is,

$$X^{n_i+1} W^{n_i} \in \operatorname{Ran}(\mathcal{A}^*) \text{ and } \mathcal{A}(X^{n_i+1}) = b,$$

where $\mathrm{Ran}(\mathcal{A}^*)$ denotes the range space of $\mathcal{A}^*$. Passing to limits, we see that

$$\widehat{X}\widetilde{W} \in \mathrm{Ran}(\mathcal{A}^*) \text{ and } \mathcal{A}(\widehat{X}) = b, \tag{8}$$

where $\widetilde{W} = (\widetilde{X}^T\widetilde{X} + \gamma_{\min} I)^{-1}$. From (8), we conclude that $\widehat{X}$ is a minimizer of the following convex optimization problem,

$$\begin{array}{ll} \text{minimize} & \mathrm{Tr}\widetilde{W}X^TX \\ \text{subject to} & \mathcal{A}(X) = b. \end{array} \tag{9}$$

Next, by assumption, $\widetilde{X}$ is not a stationary point of (3) (for $0 < p \leq 1$) nor (5) (for $p = 0$). This implies that $\widetilde{X}$ is not a minimizer of (9) and thus $\mathrm{Tr}\widetilde{W}\widehat{X}^T\widehat{X} < \mathrm{Tr}\widetilde{W}\widetilde{X}^T\widetilde{X}$. This is equivalent to $\mathcal{J}^p(\widehat{X},\widetilde{W},\gamma_{\min}) < \mathcal{J}^p(\widetilde{X},\widetilde{W},\gamma_{\min})$. From this last relation and (6) it follows that,

$$\begin{array}{rcl} \mathcal{J}^p(\widehat{X},\widehat{W},\gamma_{\min}) & < & \mathcal{J}^p(\widetilde{X},\widetilde{W},\gamma_{\min}), \\ g(\widehat{X},\gamma_{\min}) & < & g(\widetilde{X},\gamma_{\min}). \end{array} \tag{10}$$

On the other hand, since the sequence $\{g(X^k,\gamma^k)\}$ converges, we have that

$$\lim g(X^i,\gamma^j) = \lim g(X^{n_i},\gamma^{n_i}) = g(\widetilde{X},\gamma_{\min}) = \lim g(X^{n_{i+1}},\gamma^{n_{i+1}}) = g(\widehat{X},\gamma_{\min})$$

which contradicts (10). Hence, every cluster point of $\{X^k\}$ is a stationary point of (3) (when $0 < p \leq 1$) and a stationary point of (5) (when $p = 0$). ■

3.2 Performance Guarantee for IRLS-1

In this section, we discuss necessary and sufficient conditions for low-rank recovery using IRLS-1. We show that any low-rank matrix satisfying $\mathcal{A}(X) = b$ can be recovered via IRLS-1, if the null space of $\mathcal{A}$ satisfies a certain property. If the desired matrix is not low-rank, we show IRLS-1 recovers it to within an error that is a constant times the best rank-r approximation error, for any r. We first give a few definitions and lemmas.

Definition 4 *Given $x \in \mathbb{R}^n$, let $x_{[i]}$ denote the i^{th} largest element of x so that $x_{[1]} \geq x_{[2]} \geq \ldots x_{[n-1]} \geq x_{[n]}$. A vector $x \in \mathbb{R}^n$ is said to be* majorized *by $y \in \mathbb{R}^n$ (denoted as $x \prec y$) if $\sum_{i=1}^k x_{[i]} \leq \sum_{i=1}^k y_{[i]}$ for $k = 1,2,\ldots,n-1$ and $\sum_{i=1}^n x_{[i]} = \sum_{i=1}^n y_{[i]}$. A vector $x \in \mathbb{R}^n$ is said to be* weakly sub-majorized *by $y \in \mathbb{R}^n$ (denoted as $x \prec_w y$) if $\sum_{i=1}^k x_{[i]} \leq \sum_{i=1}^k y_{[i]}$ for $k = 1,2,\ldots,n$.*

Lemma 5 *(Horn and Johnson, 1990) For any two matrices, $A,B \in \mathbb{R}^{m\times n}$ it holds that $|\sigma(A) - \sigma(B)| \prec_w \sigma(A-B)$.*

Lemma 6 *(Horn and Johnson, 1991; Marshall and Olkin, 1979) Let $g : D \to \mathbb{R}$ be a convex and increasing function where $D \subset \mathbb{R}$. Let $x,y \in D^n$. Then if $x \prec_w y$, we have $(g(x_1),g(x_2),\ldots,g(x_n)) \prec_w (g(y_1),g(y_2),\ldots,g(y_n))$.*

Since $g(x) = (x^2+\gamma)^{\frac{1}{2}}$ is a convex and increasing function on $\mathbb{R}_+$, applying Lemma 6 to the majorizaiton inequality in Lemma 5 we have

$$\sum_{i=1}^{n}(|\sigma_i(A)-\sigma_i(B)|^2+\gamma)^{\frac{1}{2}} \quad \leq \quad \sum_{i=1}^{n}(\sigma_i^2(A-B)+\gamma)^{\frac{1}{2}}. \tag{11}$$

Let X_0 be a matrix (that is not necessarily low-rank) and let the measurements be given by $b = \mathcal{A}(X_0)$. In this section, we give necessary and sufficient conditions for recovering X_0 using IRLS-1. Let $X_0^\gamma := (X_0^T X_0 + \gamma I)^{\frac{1}{2}}$ be the γ-approximation of X_0, and let $X_{0,r}$, $X_{0,r}^\gamma$ be the best rank-r approximations of X_0, X_0^γ respectively.

Recall the Null space Property τ-NSP defined earlier in (2). This condition requires that every nonzero matrix in the null space of $\mathcal{A}$ has a rank larger than $2r$.

Theorem 7 *Assume that $\mathcal{A}$ satisfies τ-NSP of order r for some $0<\tau<1$. For every X_0 satisfying $\mathcal{A}(X_0) = b$ it holds that*

$$f_1(X_0+Z) > f_1(X_0), \text{ for all } \ Z \in \mathcal{N}(\mathcal{A})\backslash\{0\} \text{ satisfying } \ \|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star. \tag{12}$$

Furthermore, we have the following bounds,

$$\begin{aligned} \|\bar{X}-X_0\|_\star &\leq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star \\ \|\bar{X}_r - X_0\|_\star &\leq (2C+1)\|X_0^\gamma - X_{0,r}^\gamma\|_\star, \end{aligned}$$

where $C = 2\frac{(1+\tau)}{1-\tau}$, and $\bar{X}$ is the output of IRLS-1. Conversely, if (12) *holds, then $\mathcal{A}$ satisfies δ-NSP of order r, where $\delta > \tau$.*

Proof

Let $Z \in \mathcal{N}(A)\backslash\{0\}$ and $\|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$. We see that

$$\begin{aligned} f_1(X_0+Z) &= \mathrm{Tr}((X_0+Z)^T(X_0+Z)+\gamma I)^{\frac{1}{2}} = \sum_{i=1}^{n}(\sigma_i^2(X_0+Z)+\gamma)^{\frac{1}{2}} \\ &\geq \sum_{i=1}^{r}((\sigma_i(X_0)-\sigma_i(Z))^2+\gamma)^{\frac{1}{2}} + \sum_{i=r+1}^{n}((\sigma_i(Z)-\sigma_i(X_0))^2+\gamma)^{\frac{1}{2}} \\ &= \sum_{i=1}^{r}((\sigma_i^2(X_0)+\gamma)+\sigma_i^2(Z)-2\sigma_i(X_0)\sigma_i(Z))^{\frac{1}{2}} \\ &\quad + \sum_{i=r+1}^{n}((\sigma_i^2(Z)+\gamma)+\sigma_i^2(X_0)-2\sigma_i(Z)\sigma_i(X_0))^{\frac{1}{2}} \\ &\geq \sum_{i=1}^{r}|(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}-\sigma_i(Z)| + \sum_{i=r+1}^{n}|(\sigma_i^2(Z)+\gamma)^{\frac{1}{2}}-\sigma_i(X_0)| \\ &\geq \sum_{i=1}^{r}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} - \sum_{i=1}^{r}\sigma_i(Z) + \sum_{i=r+1}^{n}(\sigma_i^2(Z)+\gamma)^{\frac{1}{2}} - \sum_{i=r+1}^{n}\sigma_i(X_0) \\ &\geq f_1(X_0) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}} - \sum_{i=1}^{r}\sigma_i(Z) \\ &\quad + \sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=r+1}^{n}\sigma_i(X_0), \end{aligned} \tag{13}$$

where the first inequality follows from (11). Since τ-NSP holds, we have

$$\begin{aligned}\sum_{i=1}^{r}\sigma_i(Z) &< \tau\sum_{i=r+1}^{n}\sigma_i(Z) = \sum_{i=r+1}^{n}\sigma_i(Z) - (1-\tau)\sum_{i=r+1}^{n}\sigma_i(Z)\\ &\le \sum_{i=r+1}^{n}\sigma_i(Z) - C\frac{1-\tau}{1+\tau}\|X_0^\gamma - X_{0,r}^\gamma\|_\star\\ &= \sum_{i=r+1}^{n}\sigma_i(Z) - 2\|X_0^\gamma - X_{0,r}^\gamma\|_\star, \end{aligned} \tag{14}$$

where the second inequality uses $\|Z\|_\star \ge C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$ and τ-NSP. Combining (13) and (14), we obtain

$$\begin{aligned} f_1(X_0+Z) &= f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}\\ &\quad - \sum_{i=r+1}^{n}\sigma_i(X_0)\\ &\ge f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - \sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}\\ &\quad - \textstyle\sum_{i=r+1}^{n}(\sigma_i^2(X_0)+\gamma)^{\frac{1}{2}}\\ &= f_1(X_0) + \textstyle\sum_{i=r+1}^{n}\sigma_i(Z) - \sum_{i=1}^{r}\sigma_i(Z) - 2\|X_0^\gamma - X_{0,r}^\gamma\|_\star\\ &> f_1(X_0). \end{aligned} \tag{15}$$

This proves (12).

Next, if $\widehat{X}$ is an optimal solution of (3) with $p=1$, then $Z = \widehat{X} - X_0 \in \mathcal{N}(\mathcal{A})$ and it follows immediately from (15) that

$$\|\widehat{X} - X_0\|_\star = \|Z\|_\star \le C\|X_0^\gamma - X_{0,r}^\gamma\|_\star.$$

Note that problem (3) is convex when $p=1$, and every stationary point of (3) is a global minimum. Hence, by Theorem 3, IRLS-1 converges to a global minimum of the problem (3). It follows that $\bar{X} = \widehat{X}$ and

$$\|\bar{X} - X_0\|_\star \le C\|X_0^\gamma - X_{0,r}^\gamma\|_\star. \tag{16}$$

Finally, since $|\sigma(\bar{X}) - \sigma(X_0)| \prec_w \sigma(\bar{X} - X_0)$,

$$\sum_{i=r+1}^{n}\sigma_i(\bar{X}) - \sum_{i=r+1}^{n}\sigma_i(X_0) \le \sum_{i=1}^{n}|\sigma_i(\bar{X}) - \sigma_i(X_0)| \le \|\bar{X} - X_0\|_\star. \tag{17}$$

Thus,

$$\begin{aligned}\|\bar{X}_r - X_0\|_\star &\le \|\bar{X} - \bar{X}_r\|_\star + \|\bar{X} - X_0\|_\star\\ &\le \|X_0 - X_{0r}\|_\star + \|\bar{X} - X_0\|_\star + \|\bar{X} - X_0\|_\star\\ &\le (2C+1)\|X_0^\gamma - X_{0,r}^\gamma\|_\star, \end{aligned}$$

where the second inequality follows from (17) and the third inequality from (16).

Conversely, suppose that (12) holds, that is, $f_1(X_0+Z) > f_1(X_0)$ for all $\|Z\|_\star \geq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$, $Z \in \mathcal{N}(A)\backslash\{0\}$. We would like to show that $\mathcal{A}$ satisfies δ-NSP of order r. Assume to the contrary that there exists $Z \in \mathcal{N}(A)$ such that

$$\sum_{i=1}^{r} \sigma_i(Z) \geq \delta \sum_{i=r+1}^{n} \sigma_i(Z). \tag{18}$$

Let $\alpha = \frac{1-\tau\delta}{2(1+\tau)}$ and set $X_0 = -Z_r - \alpha(Z - Z_r)$, where Z_r denotes the best rank-r approximation to Z. Note that $\alpha < (1+\delta)/C$. Assume that Z satisfies

$$\left(\frac{1+\delta}{C} - \alpha\right) \sum_{i=r+1}^{n} \sigma_i(Z) \geq (n-r)\sqrt{\gamma}. \tag{19}$$

If not, Z can be multiplied by a large enough positive constant so that it satisfies both (19) and (18). Note that (18) can be rewritten as,

$$\|Z\|_\star \geq (1+\delta) \sum_{i=r+1}^{n} \sigma_i(Z). \tag{20}$$

Combining (19) and (20), we obtain that

$$\begin{aligned} \|X_0^\gamma - X_{0,r}^\gamma\|_\star &= \sum_{i=r+1}^{n} (\alpha^2\sigma_i^2(Z) + \gamma)^{\frac{1}{2}} \leq (n-r)\sqrt{\gamma} + \alpha \sum_{i=r+1}^{n} \sigma_i(Z) \\ &\leq \frac{\|Z\|_\star}{C} \end{aligned} \tag{21}$$

Moreover, it follows from (18) and the choice of X_0 that

$$\begin{array}{rcl} f_1(X_0) & = & \sum_{i=1}^{n} (\sigma_i^2(X_0) + \gamma)^{\frac{1}{2}} \\ & \geq & \sum_{i=1}^{r} \sigma_i(X_0) + \sum_{i=r+1}^{n} \sigma_i(X_0) \\ & = & \sum_{i=1}^{r} \sigma_i(Z) + \alpha \sum_{i=r+1}^{n} \sigma_i(Z) \\ & \geq & (\alpha+\delta) \sum_{i=r+1}^{n} \sigma_i(Z). \end{array} \tag{22}$$

On the other hand, notice by the definition of α that $\alpha > \frac{1-\delta}{2}$. Also assume Z satisfies

$$(2\alpha + \delta - 1) \sum_{i=r+1}^{n} \sigma_i(Z) \geq n\sqrt{r}. \tag{23}$$

If not, Z can be multiplied by a large enough positive constant to satisfy (23) and also (18). Combining (23) and (22), we obtain further that

$$\begin{aligned} f_1(X_0+Z) = \sum_{i=1}^{n} (\sigma_i^2(X_0+Z) + \gamma)^{\frac{1}{2}} &= r\sqrt{\gamma} + \sum_{i=r+1}^{n} ((1-\alpha)^2\sigma_i^2(Z) + \gamma)^{\frac{1}{2}} \\ &\leq n\sqrt{\gamma} + (1-\alpha) \sum_{i=r+1}^{n} \sigma_i(Z) \leq f_1(X_0). \end{aligned} \tag{24}$$

Now it is easy to see that (21) and (24) together contradicts (12), which completes the proof. ■

Thus when the sufficient condition (τ-NSP of order r) holds, we have shown that the best rank-r approximation of the IRLS-1 solution is not far away from $\widetilde{X}$, the solution we wish to recover, and the distance between the two is bounded by a (γ-approximate) rank-r approximation error of X_0. It has been shown (Oymak and Hassibi, 2010) that 1-NSP of order r holds for a random Gaussian map $\mathcal{A}$ with high probability when the number of measurements is large enough. The necessity statement can be rephrased as follows. δ-NSP with $\delta > \tau$ is a necessary condition for the following to hold: whenever there is an X such that $f_1(X) \leq f_1(X_0)$, we have that $\|X - X_0\|_\star \leq C\|X_0^\gamma - X_{0,r}^\gamma\|_\star$. Note that the necessary and sufficient conditions for recovery of approximately low-rank matrices using IRLS-1 mirror the conditions for recovery of approximately low-rank matrices using nuclear norm minimization which is that $\mathcal{A}$ satisfy 1-NSP. We never let the weight regularization parameter γ go to 0, thus theoretically the solution of IRLS-1 may never exactly equal the solution of Nuclear norm minimization. However, we note that γ can be made very small, for example in the numerical experiments we let γ go to 10^{-10}.

Also note that the null space property SRNSP of order r, considered in Fornasier et al. (2010) and shown to be sufficient for low-rank recovery using IRLSM, is equivalent to 1-NSP of order $2r$. In this paper, we show that τ-NSP of order r (as contrasted with the order $2r$ NSP) is both necessary and sufficient for recovery of approximately low-rank matrices using IRLS-1. Thus, our condition for the recovery of approximately low-rank matrices using IRLS-1 generalizes those stated in previous literature (Mohan and Fazel, 2010b; Fornasier et al., 2010), since it places a weaker requirement on the linear map $\mathcal{A}$.

4. A Gradient Projection Based Implementation of IRLS

In this section, we describe IRLS-GP, a gradient projection based implementation of IRLS-p for the Matrix Completion Problem.

4.1 The Matrix Completion Problem

The matrix completion problem is a special case of the affine rank minimization problem with constraints that restrict some of the entries of the matrix variable X to equal given values. The problem can be written as

$$\begin{array}{ll} \text{minimize} & \text{rank}(X) \\ \text{subject to} & X_{ij} = (X_0)_{ij}, \quad (i,j) \in \Omega, \end{array}$$

where X_0 is the matrix we would like to recover and Ω denotes the set of entries which are revealed. We define the operator $\mathcal{P}_\Omega : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ as

$$(\mathcal{P}_\Omega(X))_{ij} = \begin{cases} X_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

Also, Ω^c denotes the complement of Ω, that is, all index pairs (i,j) except those in Ω.

4.2 IRLS-GP

To apply Algorithm 1 to the matrix completion problem (25), we replace the constraint $\mathcal{A}(X) = b$ by $\mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0)$. Each iteration of IRLS solves a quadratic program (QP). A gradient projection algorithm could be used to solve the quadratic program (QP) in each iteration of IRLS. We call this implementation IRLS-GP (Algorithm 2).

Data: Ω, $\mathcal{P}_\Omega(X_0)$
Result: $\widehat{X} : \mathcal{P}_\Omega(\widehat{X}) = b$
Set $k = 0$. Initialize $X^0 = 0, \gamma^0 > 0$, $s^0 = (\gamma^0)^{(1-\frac{p}{2})}$;
while *IRLS iterates not converged* **do**
  $W^k = ({X^k}^T X^k + \gamma^k I)^{\frac{p}{2}-1}$. Set $X_{\text{temp}} = X^k$;
  while *Gradient projection iterates not converged* **do**
    $X_{\text{temp}} = \mathcal{P}_{\Omega^c}(X_{\text{temp}} - s^k X_{\text{temp}} W^k) + \mathcal{P}_\Omega(X_0)$;
  end
  Set $X^{k+1} = X_{\text{temp}}$;
  Choose $0 < \gamma^{k+1} \leq \gamma^k$, $s^{k+1} = (\gamma^{k+1})^{(1-\frac{p}{2})}$;
  $k = k + 1$;
end

Algorithm 2: IRLS-GP for Matrix Completion

The step size used in the gradient descent step is $s^k = 1/L^k$, where $L^k = 2\|W^k\|$ is the Lipschitz constant of the gradient of the quadratic objective $\text{Tr}(W^k X^T X)$ at the k^{th} iteration. We also warm-start the gradient projection algorithm to solve for the $(k+1)^{th}$ iterate of IRLS with the solution of the k^{th} iterate and find that this speeds up the convergence of the gradient projection algorithm in subsequent iterations. At each iteration of IRLS, computing the weighting matrix involves an inversion operation which can be expensive for large n. To work around this, we observe that the singular values of subsequent iterates of IRLS cluster into two distinct groups, so a low rank approximation of the iterates (obtained by setting the smaller set of singular values to zero) can be used to compute the weighting matrix efficiently. Computing the singular value decomposition (SVD) can be expensive. Randomized algorithms (e.g., Halko et al., 2011) can be used to compute the top r singular vectors and singular values of a matrix X efficiently, with small approximation errors, if $\sigma_{r+1}(X)$ is small. We describe our computations of the weighting matrix below.

Computing the weighting matrix efficiently. Let $U\Sigma V^T$ be the truncated SVD of X^k (keeping top r terms in the SVD with r being determined at each iteration), so that $U \in \mathbb{R}^{m\times r}, \Sigma \in \mathbb{R}^{r\times r}, V \in \mathbb{R}^{n\times r}$. Then for $p = 0$, ${W^k}^{-1} = \frac{1}{\gamma^k}(U\Sigma V^T)^T(U\Sigma V^T) + \gamma^k I_n$. It is easy to check that $W^k = V(\gamma^k(\Sigma^2 + \gamma^k I)^{-1} - I_r)V^T + I_n$. Thus the cost of computing the weighting matrix given the truncated SVD is $O(nr^2)$, saving significant computational costs. At each iteration, we choose r to be $\min\{r_{\max}, \widehat{r}\}$ where $\widehat{r}$ is the largest integer such that $\sigma_{\widehat{r}}(X^k) > 10^{-2} \times \sigma_1(X^k)$. Also, since the singular values of X^k tend to separate into two clusters, we observe that this choice eliminates the cluster with smaller singular

values and gives a good estimate of the rank r to which X^k can be well approximated. We find that combining warm-starts for the gradient projection algorithm along with the use of randomized algorithms for SVD computations speeds up the overall computational time of the gradient projection implementation considerably.

5. sIRLS-p: A First-order Algorithm for Minimizing the Smooth Schatten-p Function

In this section, we present the sIRLS-p family of algorithms that are related to the IRLS-GP implementation discussed in the previous section. We first describe the algorithm before discussing its connection to IRLS-p.

Data: $\mathcal{P}_\Omega$, b
Result: $\widehat{X} : \mathcal{P}_\Omega(\widehat{X}) = b$
Set $k = 0$. Initialize $X^0 = \mathcal{P}_\Omega(X_0), \gamma^0 > 0$, $s^0 = (\gamma^0)^{(1-\frac{p}{2})}$;
while *not converged* **do**
  $W_p^k = ({X^k}^T X^k + \gamma^k I)^{\frac{p}{2}-1}$;
  $X^{k+1} = \mathcal{P}_{\Omega^c}(X^k - s^k X^k W_p^k) + \mathcal{P}_\Omega(X_0)$;
  Choose $0 < \gamma^{k+1} \leq \gamma^k$, $s^{k+1} = (\gamma^{k+1})^{(1-\frac{p}{2})}$;
  $k = k+1$
end

Algorithm 3: sIRLS-p for Matrix Completion Problem

We note that sIRLS-p (Algorithm 3) is a gradient projection algorithm applied to

$$\begin{array}{ll} \text{minimize} & f_p(X) = \text{Tr}(X^T X + \gamma I)^{\frac{p}{2}} \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0), \end{array} \tag{26}$$

while sIRLS-0 is a gradient projection algorithm applied to

$$\begin{array}{ll} \text{minimize} & \log\det(X^T X + \gamma I) \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0). \end{array} \tag{27}$$

Indeed $\nabla f_p(X^k) = X^k W_p^k$ and the gradient projection iterates,

$$\begin{aligned} X^{k+1} &= \mathcal{P}_{\{X:\mathcal{P}_\Omega(X)=\mathcal{P}_\Omega(X_0)\}}(X^k - s^k \nabla f_p(X^k)) \\ &= \mathcal{P}_{\Omega^c}(X^k - s^k \nabla f_p(X^k)) + \mathcal{P}_\Omega(X_0) \end{aligned}$$

are exactly the same as the sIRLS-p (Algorithm 3) iterates when γ^k is a constant. In other words, for large enough k (i.e., k such that $\gamma^k = \gamma_{\min}$), the iterates of sIRLS-p and sIRLS-0 are nothing but gradient projection applied to (26) and (27) respectively, with $\gamma = \gamma_{\min}$. We have the following convergence results for sIRLS.

Theorem 8 *Every cluster point of sIRLS-p $(0 < p \leq 1)$ is a stationary point of (26) with $\gamma = \gamma_{\min}$.*

Theorem 9 *Every cluster point of sIRLS-0 is a stationary point of (27) with* $\gamma = \gamma_{\min}$.

The proof of Theorem 8 can be found in appendix C, and Theorem 9 can be proved in a similar way. With $p = 1$, Theorem 8 implies that sIRLS-1 has the same performance guarantee as IRLS-1 given in Theorem 7. Note that despite sIRLS-p with $p < 1$ being a gradient projection algorithm applied to non-convex problems (26) and (27), a simple step-size suffices for convergence, and we do not consider a potentially expensive line search at each iteration of the algorithm.

We now relate the sIRLS-p family to the IRLS-p family of algorithms. Each iteration of the IRLS-p algorithm is a quadratic program, and IRLS-GP uses iterative gradient projection to solve this quadratic program. We note that sIRLS-p is nothing but IRLS-GP, with each quadratic program solved approximately by terminating at the first iteration in the gradient projection inner loop. Barring this connection, the IRLS-p and sIRLS-p algorithms can be seen as two different approaches to solving the smooth Schatten-p minimization problem (26). We examine the trade off between these two algorithms along with comparisons to other state of the art algorithms for matrix completion in the next section.

6. Numerical Results

In this section, we give numerical comparisons of sIRLS-$0,1$ and IRLS-$0,1$ with other algorithms. We begin by examining the behavior of IRLS-0 (through the IRLS-GP implementation) and its sensitivity to γ^k (regularization parameter in the weighting matrix, W^k).

6.1 Choice of Regularization γ

We find that IRLS-0 converges faster when the regularization parameter in the weighting matrix, γ^k, is chosen appropriately. We consider an exponentially decreasing model $\gamma^k = \gamma^0/(\eta)^k$, where γ^0 is the initial value and η is a scaling parameter. We run sensitivity experiments to determine good choices of γ^0 and η for the matrix completion problem. For this and subsequent experiments, the indices corresponding to the known entries Ω are generated using i.i.d Bernoulli $\{0,1\}$ random variables with a mean support size $|\Omega| = q$, where q/n^2 is the probability for an index (i,j) to belong to the support set. The completed and unknown matrix X_0 of rank r is generated as YY^T, where $Y \in \mathbb{R}^{n\times r}$ is generated using i.i.d gaussian entries. All experiments are conducted in Matlab 7.12.0 (R2011 a) on a Intel 3 Ghz core 2 duo processor with 4 GB RAM.

As will be seen from the results, the regularization parameter γ^k plays an important role in the recovery. We let $\gamma^0 = \gamma_c \|X_0\|^2$ where γ_c is a proportional parameter that needs to be estimated. For the sensitivity analysis of IRLS-0 (with respect to γ^0 and η), we consider matrices of size 500×500.

6.1.1 Choice of γ_c

As can be seen from Figures 1 a) and b), choosing γ_c appropriately leads to a better convergence rate for IRLS-0. Small values of γ_c ($< 10^{-3}$) don't give good recovery results (premature convergence to a larger relative error). However larger values of γ_c (> 1) might lead to a delayed convergence. As a heuristic, we observe that $\gamma_c = 10^{-2}$ works well. We also note that this choice of γ_c works well

even if the spectral norm of X_0 varies from 1 to 1000. Thus for future experiments, we normalize X_0 to have a spectral norm of 1.

6.1.2 CHOICE OF η

Figures 2 a),b),c), and d) look at the sensitivity of the IRLS-0 algorithm to the scaling parameter, η. We observe that for a good choice of γ^0 (described earlier), η depends on the rank of X_0 to be recovered. More specifically, η seems to have an inverse relationship with the rank of X_0. From Figures 2 a) and d), it is clear that $\eta = 1.3$ works well if rank of X_0 equals 2 and $\eta = 1.05$ works well when rank of X_0 equals 15. More generally, the choice of η seems to depend on the hardness of the problem instance being considered. We formalize this notion in the next section.

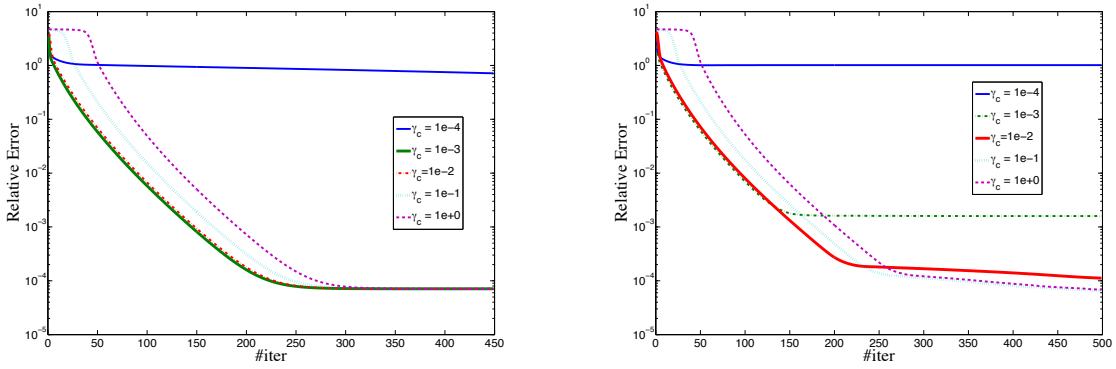


Figure 1: $n = 500$, $r = 5$, $\eta = 1.15$. $\gamma^0 = \gamma_c \|X_0\|^2$. *From left to right*: Recovery error using IRLS-0 for $\|X_0\| = 1, \|X_0\| = 1000$.

6.2 Numerical Experiments

We classify our numerical experiments into two categories based on the *degrees of freedom ratio*, given by FR $= r(2n-r)/q$. Note that for a $n \times n$ matrix of rank r, $r(2n-r)$ is the number of degrees of freedom in the matrix. Thus if FR is large (close to 1), recovering X_0 becomes harder (as the number of measurements is close to the degrees of freedom) and conversely if FR is close to zero, recovering X_0 becomes easier.

We conduct subsequent numerical experiments over what we refer to in this paper as *Easy problems* ($FR < 0.4$) and *Hard problems*($FR > 0.4$). We define the recovery to be successful when the relative error, $\|X - X_0\|_F / \|X_0\|_F \leq 10^{-3}$ (with X being the output of the algorithm considered) and unsuccessful recovery otherwise. For each problem (easy or hard), the results are reported over 10 random generations of the support set, Ω and X_0. We use NS to denote the number of successful recoveries for a given problem. Also, computation times are reported in seconds. For sIRLS and IRLS-GP (implementation of IRLS), we fix $\eta = 1.1$ if $FR < 0.4$ and $\eta = 1.03$ if $FR > 0.4$ based

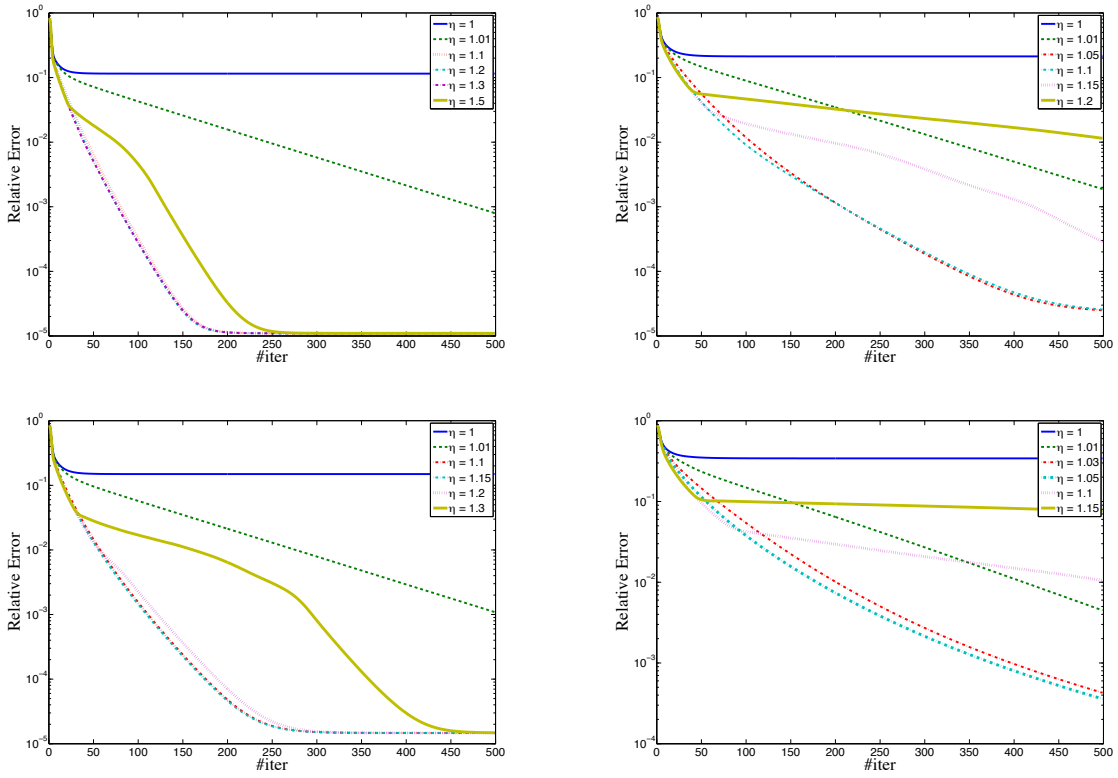


Figure 2: $n = 500$, $\gamma_c = 10^{-2}$. *Clockwise from top left*: Recovery error using IRLS-0 for ranks $2, 5, 10, 15$ respectively.

on our earlier observations. In the next few sections, we compare the IRLS implementations with other state of the art algorithms on both exact and noisy matrix completion problems.

6.3 Comparison of (s)IRLS and Nuclear Norm Minimization

In this section, we compare the gradient projection implementation IRLS-GP (of IRLS-0,1) and the algorithm sIRLS-0,1 with the *Singular Value Thresholding* (SVT) algorithm (an implementation for nuclear norm minimization Cai et al., 2008) on both easy and hard problem sets. Note that SVT is not the only implementation of nuclear norm minimization. Other implementations include NNLS (Toh and Yun, 2010) and Spectral Regularization (Mazumder et al., 2010).

When we refer to IRLS-0,1 in the tables and in subsequent paragraphs, we mean their gradient projection implementation, IRLS-GP. We compare (s)IRLS-0,1 and SVT in Tables 1 and 3. A few aspects of these comparisons are highlighted below.

Problem				IRLS-1		sIRLS-1		IRLS-0	
n	r	$\frac{q}{n^2}$	FR	# iter	Time	# iter	Time	# iter	Time
100	10	0.57	0.34	133	4.49	132	1.63	54	0.79
200	10	0.39	0.25	140	4.49	140	2.41	60	1.34
500	10	0.2	0.2	160	24.46	163	8	77	9.63
500	10	0.12	0.33	271	37.47	336	13.86	220	22.74
1000	10	0.12	0.17	180	113.72	195	32.21	109	55.42
1000	50	0.39	0.25	140	134.30	140	102.64	51	59.74
1000	20	0.12	0.33	241	156.09	284	57.85	188	96.20
2000	20	0.12	0.17	180	485.24	190	166.28	100	235.94
2000	40	0.12	0.33	236	810.13	270	322.96	170	432.34

Table 1: Comparison of IRLS-0,1 and sIRLS-1. Performance on Easy Problems $FR < 0.4$.

Problem				sIRLS-1		sIRLS-0		SVT	
n	r	$\frac{q}{n^2}$	FR	# iter	Time	# iter	Time	# iter	Time
100	10	0.57	0.34	132	1.63	59	0.84	170	5.69
200	10	0.39	0.25	140	2.41	63	1.31	109	3.74
500	10	0.2	0.2	163	8	98	4.97	95	5.9
500	10	0.12	0.33	336	13.86	280	11.03	-	-
1000	10	0.12	0.17	195	32.21	140	20.80	85	10.71
1000	50	0.39	0.25	140	102.64	60	61.32	81	49.17
1000	20	0.12	0.33	284	57.85	241	43.11	-	-
2000	20	0.12	0.17	190	166.28	130	98.55	73	42.31
2000	40	0.12	0.33	270	322.96	220	227.07	-	-

Table 2: Comparison of sIRLS-0,1 with SVT. Performance on Easy Problems $FR < 0.4$.

Problem				sIRLS-1			IRLS-0			sIRLS-0		
n	r	$\frac{q}{n^2}$	FR	# iter	NS	Time	# iter	NS	Time	# iter	NS	Time
40	9	0.5	0.8	4705	4	163.2	1385	10	17.36	2364	9	30.22
100	14	0.3	0.87	10000	0	545.91	4811	10	89.51	5039	7	114.54
500	20	0.1	0.78	10000	0	723.58	4646	8	389.66	5140	10	315.57
1000	20	0.1	0.4	645	10	142.84	340	10	182.78	406	10	97.15
1000	20	0.06	0.66	10000	0	1830.98	2679	10	921.15	2925	10	484.84
1000	30	0.1	0.59	1152	10	295.56	781	10	401.98	915	10	244.23
1000	50	0.2	0.49	550	10	342	191	10	239.77	270	10	234.25

Table 3: Comparison of sIRLS-1, IRLS-0 and sIRLS-0. Performance on Hard Problems $FR \geq 0.4$

6.3.1 IRLS-0 VS IRLS-1

Between IRLS-0 and IRLS-1, IRLS-0 takes fewer iterations to converge successfully and has a lower computational time (Table 1). The same holds true between sIRLS-0 and sIRLS-1. sIRLS-0 is also successful on more hard problem instances than sIRLS-1 (Table 3). This indicates that (s)IRLS-p with $p = 0$ has a better recovery performance and computational time as compared to $p = 1$.

6.3.2 IRLS VS SIRLS

Between sIRLS and IRLS, sIRLS-1 takes more iterations to converge as compared to IRLS-1. However because it has a lower per iteration cost, sIRLS-1 takes significantly lower computational time than IRLS-1 (Table 1). The same holds true for sIRLS-0. Thus sIRLS-0,1 are not only simpler algorithms, they also have a lower overall run time as compared to IRLS-0,1.

6.3.3 COMPARISON ON EASY PROBLEMS

Table 2 shows that sIRLS-0 and sIRLS-1 have competitive computational times as compared to SVT (implementation available at, Candes and Becker, 2010). There are also certain instances where SVT fails to have successful recovery while sIRLS-1 succeeds. Thus sIRLS-1 is competitive and in some instances better than SVT.

6.3.4 COMPARISON ON HARD PROBLEMS

For hard problems, Table 3 shows that sIRLS-0 and IRLS-0 are successful in almost all problems considered, while sIRLS-1 is not successful in 4 problems. We also found that SVT was not successful in recovery for any of the hard problems. (s)IRLS-0 also compares favorably with FPCA (Goldfarb and Ma, 2011) and Optspace (Keshavan and Oh, 2009) in terms of recovery and computational time on the easy and hard problem sets. These results are given subsequently.

In summary, (s)IRLS-1,(s)IRLS-0 have a better recovery performance than a nuclear norm minimization implementation (SVT) as evidenced by successful recovery in both easy and hard problem sets. We note that (s)IRLS-1 converges to the Nuclear Norm Minimizer (when the regularization, $\gamma \to 0$) and empirically has a better recovery performance than SVT. We also note that among the family of (s)IRLS-p algorithms tested, sIRLS-0 and IRLS-0 are better in both recovery performance and computational times.

6.4 Comparison of Algorithms for Exact Matrix Completion

As observed in the previous section, sIRLS has a lower total run time compared to IRLS-GP. Thus in subsequent experiments we compare other algorithms only with sIRLS.

6.4.1 DESIGN OF EXPERIMENTS

In this section, we report results from two sets of experiments. In the first set, we compare sIRLS-0 (henceforth referred to as sIRLS), *Iterative Hard Thresholding* algorithm (IHT) (Goldfarb and

Problem				sIRLS		IRLSM		IHT		Optspace	
n	r	$\frac{q}{n^2}$	FR	# iter	Time	#iter	Time	#iter	Time	#iter	Time
100	10	0.57	0.34	56	0.8	68	0.58	44	0.51	27	0.6
200	10	0.39	0.25	61	0.96	78	1.53	53	0.95	19	1.28
500	10	0.2	0.2	99	4.5	106	15	105	3.63	18	8.08
500	10	0.12	0.33	285	13.24	240	160.62	344	12.50	29	12.45
1000	10	0.12	0.17	143	21.17	152	106.18	192	19.30	16	28.93
1000	50	0.39	0.25	60	27.39	60	429.25	46	19.58	17	1755.44
1000	20	0.12	0.33	244	45.33	264	396.49	289	40.39	38	241.94
2000	20	0.12	0.17	130	82.47	140	916.48	179	80.76	14	428.37
2000	40	0.12	0.33	230	229.53	220	4213.4	270	225.46	28	4513

Table 4: Comparison of sIRLS-0, IRLSM, IHT and Optspace on Easy Problems with rank of the matrix to be recovered known apriori.

Ma, 2011; Meka et al., 2010), Optspace and IRLSM (Fornasier et al., 2010) over easy and hard problems with the assumption that the rank of the matrix to be recovered (X_0) is known. We use the implementations of IRLSM, Optspace available on the authors webpage (Fornasier et al , 2012; Keshavan et al., 2009a). When the rank of X_0 (denoted as r) is known, the weighting matrix W^k for sIRLS is computed using a rank r approximation of X^k (also see Section 3.1). The second set of experiments correspond to the case where the rank of X_0 is unknown, which is a more practical assumption.

6.4.2 RANK OF X_0 KNOWN APRIORI

All the algorithms are fully successful (NS = 10) on the easy problem sets. As seen in Table 4, Optspace takes fewer iterations to converge as compared to sIRLS, IRLSM and IHT. On the other hand, sIRLS and IHT are significantly faster than Optspace and much faster than IRLSM (Fornasier et al., 2010). Although IRLSM uses a sub-routine implemented in C to speed up inverse matrix computations inside a Matlab code, it is still about 10 times slower than sIRLS on easy problems and even slower on hard problems. We note that for the hard problems in Table 5, the rank of the true solution is given by (9,14,20,20,20,30,50) respectively. IRLSM is also not successful on any of the instances for two of the hard problems. Also while sIRLS, Optspace and IHT are fully successful on most of the hard problems (see Table 5), Optspace takes considerably higher time as compared to IHT and sIRLS. Thus, when the rank of X_0 is known, sIRLS is competitive with IHT in performance and computational time and much faster than Optspace and IRLSM.

6.4.3 RANK OF X_0 UNKNOWN

A possible disadvantage of IHT and Optspace could be their sensitivity to the knowledge of the rank of X_0. Thus, our second set of experiments compare sIRLS, IHT, Optspace and FPCA (Goldfarb and Ma, 2011) over easy and hard problems when the rank of X_0 is unknown. We use a heuristic for

Problem		sIRLS-0			IRLSM			IHT			Optspace		
n	$\frac{q}{n^2}$	# iter	NS	Time	# iter	NS	Time	# iter	NS	Time	# iter	NS	Time
40	0.5	1718	10	12.67	-	0	-	1635	10	12.16	1543	7	6.82
100	0.3	4298	8	60.18	-	0	-	4868	10	68.56	4011	5	131.62
1000	0.1	417	10	78.26	428	10	642.53	466	10	65.84	69	10	409.66
1000	0.08	814	10	151.86	1120	10	1674.2	947	10	134.30	103	10	580.09
1000	0.07	1368	10	251.46	2180	8	3399.2	1564	10	225.13	147	10	806.32
1000	0.1	949	10	226.12	1536	10	4451.1	1006	10	189.33	134	10	1904.47
1000	0.2	270	10	123.84	244	10	1736	254	10	105.88	46	10	2968

Table 5: Comparison of sIRLS-0, IHT and Optspace on Hard Problems with rank of the matrix to be recovered known apriori.

Problem				sIRLS		IHT		FPCA	Optspace	
n	r	$\frac{q}{n^2}$	FR	# iter	Time	# iter	Time	Time	# iter	Time
100	10	0.57	0.34	59	1.61	38	1.15	0.13	25	0.62
200	10	0.39	0.25	62	2.55	44	1.96	0.37	17	1.4
500	10	0.2	0.2	98	9.39	71	8.65	2.52	17	8.46
500	10	0.12	0.33	283	16.07	225	23.23	71.26	30	11.63
1000	10	0.12	0.17	140	38.44	104	31.47	11.24	14	26.31
1000	50	0.39	0.25	60	217.79	35	132.36	15.10	17	1774.08
1000	20	0.12	0.33	241	77.52	177	70.13	18.51	30	199.98
2000	20	0.12	0.17	130	236.19	98	152.36	42.06	12	374.15
2000	40	0.12	0.33	220	234.44	167	323.67	76.26	26	3466

Table 6: Comparison of sIRLS-0, IHT, FPCA and Optspace on Easy Problems when no prior information is available on the rank of the matrix to be recovered.

determining the approximate rank of X^k at each iteration for sIRLS, IHT. Heuristics for determining approximate rank are also used in the respective implementations for FPCA and Optspace. We note that computing the approximate rank is important for speeding up the SVD computations in all of these algorithms.

Choice of rank. We choose r (the rank at which the SVD of X^k is truncated) to be $\min\{r_{\max}, \widehat{r}\}$ where $\widehat{r}$ is the largest integer such that $\sigma_{\widehat{r}}(X^k) > \alpha \times \sigma_1(X^k)$. For IHT we find that $\alpha = 5 \times 10^{-2}$ works well while for sIRLS and FPCA, $\alpha = 10^{-2}$ works well. The SVD computations in IHT, sIRLS are based on a randomized algorithm (Halko et al., 2011). We note that Linear-Time SVD (Drineas et al., 2006) is used to compute the SVD in the FPCA implementation (Goldfarb and Ma, 2009), and although faster than the randomized SVD algorithm we use for sIRLS, it can be significantly less accurate.

Problem		sIRLS			IHT			FPCA		Optspace		
n	FR	# iter	NS	Time	# iter	NS	Time	NS	Time	# iter	NS	Time
40	0.8	1498	10	12.91	-	0	-	5	1.69	-	0	-
100	0.87	4934	5	72.36	-	0	-	0	-	-	0	-
500	0.78	4859	9	326.06	-	0	-	0	-	-	0	-
1000	0.40	406	10	115.73	280	10	72.67	10	26.54	40	10	256.92
1000	0.57	1368	10	237.22	1059	10	244.49	0	-	133	5	769.29
1000	0.66	2961	10	554.25	-	0	-	0	-	-	0	-
1000	0.59	897	10	276.08	660	10	213.95	10	62.43	89	5	1420.81
1000	0.49	270	10	263.45	203	10	186.15	10	25.21	45	10	2924.68

Table 7: Comparison of sIRLS-0, IHT, FPCA and Optspace on Hard Problems when no prior information is available on the rank of the matrix to be recovered.

Comparison of algorithms. All the algorithms compared are successful on the easy problems. However, Optspace takes much more time to converge on recovering matrices with high rank as can be seen from from Table 6. sIRLS, FPCA and IHT have competitive run times on all the problems. For hard problems, however, sIRLS has a clear advantage over IHT, Optspace and FPCA in successful recovery (Table 7). sIRLS is fully successful on all problems except the second and third on which it has a success rate of 5 and 9 respectively. On the other hand IHT, Optspace and FPCA have partial or unsuccessful recovery in many problems. sIRLS is competitive with IHT and FPCA on computational times while Optspace is much slower than all the other algorithms. Thus, when the rank of X_0 is not known apriori, sIRLS has a distinct advantage over IHT, Optspace and FPCA in successfully recovering X_0 for hard problems.

6.5 Comparison of Algorithms for Noisy Matrix Completion

In this subsection, we compare IHT, FPCA and sIRLS on randomly generated noisy matrix completion problems. We consider the following noisy matrix completion problem,

$$\begin{array}{ll} \text{minimize} & \text{rank}(X) \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(B), \end{array}$$

where $\mathcal{P}_\Omega(B) = \mathcal{P}_\Omega(X_0) + \mathcal{P}_\Omega(Z)$, X_0 is a low rank matrix of rank r that we wish to recover and $\mathcal{P}_\Omega(Z)$ is the measurement noise. Note that this noise model has been used before for matrix completion (see Cai et al., 2008). Let Z_{ij} be i.i.d Gaussian random variables with distribution $\mathcal{N}(0,\sigma^2)$. We would like the noise to be such that $\|\mathcal{P}_\Omega(Z)\|_F \leq \varepsilon \|\mathcal{P}_\Omega(X_0)\|_F$ for a noise parameter ε. This would be true if $\sigma \sim \varepsilon\sqrt{r}$ (Cai et al., 2008).

We adapt sIRLS for noisy matrix completion by replacing $\mathcal{P}_\Omega(X_0)$ by $\mathcal{P}_\Omega(B)$ in Algorithm 3. For all the algorithms tested in Table 8, we declare the recovery to be successful if $\|X - X_0\|_F / \|X_0\|_F \leq \varepsilon = 10^{-3}$, where X is the output of the algorithms. Table 8 shows that sIRLS has successful recovery for easy noisy matrix completion problems with apriori knowledge of rank. The same holds true for

Problem				sIRLS			IHT			Optspace		
n	r	$\frac{q}{n^2}$	FR	# iter	NS	Time	#iter	NS	Time	#iter	NS	Time
100	10	0.57	0.34	51	10	0.55	41	10	0.44	28	10	0.29
200	10	0.39	0.25	56	10	0.78	48	10	0.60	19	10	0.56
500	10	0.2	0.2	96	10	4.11	88	10	2.76	18	10	6.82
500	10	0.12	0.33	298	10	12.60	298	10	9.56	29	10	12.45
1000	10	0.12	0.17	141	10	19.66	132	10	12.28	15	10	37.17
1000	10	0.39	0.25	50	10	20.94	40	10	15.53	18	10	1197.48
1000	20	0.12	0.33	254	10	44.35	247	10	32.03	25	10	220.60
2000	20	0.12	0.17	130	10	77.30	121	10	84.23	12	10	469.67
2000	40	0.12	0.33	236	10	221.31	227	10	170.88	28	10	4515.06

Table 8: Comparison of sIRLS, IHT and Optspace on the Noisy Matrix Completion problem.

hard problems with the true rank known apriori. Thus sIRLS has a competitive performance even for noisy recovery.

6.6 Application to Movie Lens Data Set

In this section, we consider the movie lens data sets (Dahlen et al, 1998) with 100,000 ratings. In particular, we consider four different splits of the 100k ratings into (training set, test set): (`u1.base`,`u1.test`), (`u2.base`,`u2.test`), (`u3.base`,`u3.test`), (`u4.base`,`u4.test`) for our numerical experiments. Any given set of ratings (e.g., from a data split) can be represented as a matrix. This matrix has rows representing the users and columns representing the movies and an entry (i, j) of the matrix is non-zero if we know the rating of user i for movie j. Thus estimating the remaining ratings in the matrix corresponds to a matrix completion problem. For each data split, we train sIRLS, IHT, and Optspace on the training set and compare their performance on the corresponding test set. The performance metric here is *Normalized Mean Absolute Error* or NMAE given as follows. Let M be the matrix representation corresponding to the actual test ratings and X be the ratings matrix output by an algorithm when input the training set. Then

$$\text{NMAE} = \left(\sum_{i,j \in \text{supp}(M)} \frac{|M_{ij} - X_{ij}|}{|\text{supp}(M)|} \right) / (rt_{\max} - rt_{\min}),$$

where $rt_{\min}$ and $rt_{\max}$ are the minimum and maximum movie ratings possible. The choice of γ^0, η for sIRLS is the same as for the random experiments (described in previous sections). sIRLS is terminated if the maximum number of iterations exceeds 700 or if the relative error between the successive iterates is less than 10^{-3}. We set the rank of the unknown ratings matrix to be equal to 5 while running all the three algorithms. Table 9 shows that the NMAE for sIRLS, IHT, and Optspace are almost the same across different splits of the data. Keshavan et al. (2009b) reported the NMAE results for different algorithms when tested on data split 1. These were reported to be 0.186,0.19 and 0.242 for Optspace (Keshavan and Oh, 2009), FPCA (Goldfarb and Ma, 2011), and AdMiRA

	sIRLS	IHT	Optspace
split 1	0.1919	0.1925	0.1887
split 2	0.1878	0.1883	0.1878
split 3	0.1870	0.1872	0.1881
split 4	0.1899	0.1896	0.1882

Table 9: NMAE for sIRLS-0 for different splits of the 100k movie-lens data set.

(Lee and Bresler, 2010) respectively. Thus sIRLS has a NMAE that is as good as Optspace, FPCA, IHT and has a better NMAE than AdMiRA.

7. Conclusions and Future Directions

We proposed a family of Iterative Reweighted Least Squares algorithms (IRLS-p) for the affine rank minimization problem. We showed that IRLS-1 converges to the global minimum of the smoothed nuclear norm, and that IRLS-p with $p < 1$ converges to a stationary point of the corresponding non-convex yet smooth approximation to the rank function. We gave a matrix recovery guarantee for IRLS-1, showing that it approximately recovers the true low-rank solution (within a small error depending on the algorithm parameter γ), if the null space of the affine measurement map satisfies certain properties. This null space condition is both necessary and sufficient for low-rank recovery, thus improving on and simplifying the previous analysis for IRLS-1 (Mohan and Fazel, 2010b).

We then focused on the matrix completion problem, a special case of affine rank minimization arising in collaborative filtering among other applications, and presented efficient implementations specialized to this problem. We gave an implementation for IRLP-p for this problem using gradient projections. We also presented a related first-order algorithm, sIRLS-p, for minimizing the smooth Schatten-p function, which serves as a smooth approximation of the rank. Our first set of numerical experiments show that (s)IRLS-0 has a better recovery performance than nuclear norm minimization via SVT. We show that sIRLS-0 has a good recovery performance even when noise is present. Our second set of experiments demonstrate that sIRLS-0 compares favorably in terms of performance and run time with IHT, Optspace, and IRLSM when the rank of the low rank matrix to be recovered is known. When the rank information is absent, sIRLS-0 shows a distinct advantage in performance over IHT, Optspace and FPCA.

7.1 Future Directions

Low-rank recovery problems have recently been pursued in machine learning motivated by applications including collaborative filtering. Iterative reweighted algorithms for low-rank matrix recovery have empirically exhibited improved performance compared to unweighted convex relaxations. However, there has been a relative lack of theoretical results, as well as efficient implementations for these algorithms. This paper takes a step in addressing both of these issues, and opens up several directions for future research.

Low-rank plus sparse decomposition. The problem of decomposing a matrix into a low-rank component and a sparse component has received much attention (Chandrasekaran et al., 2011; Tan et al., 2011), and arises in graphical model identification (Chandrasekaran et al., 2010a) as well as a version of robust PCA (Candes et al., 2011), where problem sizes of practical interest are often very large. The convex relaxation proposed for this problem minimizes a combination of nuclear norm and ℓ_1 norm. An interesting direction for future work is to extend the IRLS algorithms family to this problem, by combining the vector and the matrix weighted updates. A potential feature of such an algorithm can be that the value of p for the vector part and the matrix part (and hence the weights) can be chosen separately, allowing control over how aggressively to promote the sparsity and the low-rank features.

Distributed IRLS. In the IRLS family, the least squares problem that is solved in every iteration is in fact *separable* in the columns of the matrix X (as also pointed out in Fornasier et al., 2010), so it can be solved completely in parallel. This opens the door not just to a fast parallel implementation, but also to the possibility of a partially distributed algorithm. Noting that the weight update step does not appear easy to decompose, an interesting question is whether we can use approximate but decomposable weights, so that the updates would require only local information.

Other applications for the NSP. The simple Null space Property used here, being based on only the singular values of elements in the null space, makes the connection between associated vector and matrix recovery proofs clear and transparent, and may be of independent interest (see Oymak et al., 2011).

Acknowledgments

We would like to acknowledge Ting Kei Pong for helpful discussions on the algorithms and Samet Oymak for pointing out an important singular value inequality. M. F. also acknowledges helpful discussions with Holger Rauhut.

Appendix A. IRLS-p from Characterization of Smooth Schatten-p Function

We can also derive the IRLS-p algorithm for $0 < p \leq 1$ by defining the following function,

$$\mathcal{F}^p(X,W,\gamma) \quad = \quad \mathrm{Tr} W^{\frac{p-2}{p}}(X^TX+\gamma I)$$

The following lemma (Argyriou et al., 2007; Argyriou, 2010) proves useful for our analysis.

Lemma 10 *Let* $\gamma > 0$ *and*

$$W^* = \frac{(X^TX+\gamma I)^{\frac{p}{2}}}{\mathrm{Tr}(X^TX+\gamma I)^{\frac{p}{2}}}.$$

Then

$$W^* = \underset{W}{\mathrm{argmin}} \left\{ \mathcal{F}^p(X,W,\gamma) : \; W \succ 0, \mathrm{Tr}(W) \leq 1 \right\}.$$

Note that $\mathcal{F}^p(X,W^*,\gamma) = (f_p(X))^{\frac{2}{p}}$. Hence the problem of minimizing the *Smooth Schatten-p function* $f_p(X)$ (3) is equivalent to the following

$$\begin{array}{ll} \text{minimize} & \mathcal{F}^p(X,W,\gamma) \\ \text{subject to} & \mathcal{A}(X) = b, W \succ 0, \operatorname{Tr} W \leq 1, \end{array} \tag{28}$$

where the variables are X and W. As a relaxation to minimizing (28) jointly in X,W, one can consider minimizing (28) alternately with respect to X and W as in Algorithm 4.

Data: $\mathcal{A}$, b
Result: $\widehat{X} : \mathcal{A}(\widehat{X}) = b$
Set $k = 0$. Initialize $W_p^0 = I, \gamma^1 > 0$;
while *not converged* **do**

$$X^{k+1} = \begin{array}{l} \operatorname*{argmin}_X \mathcal{F}^p(X,W^k) \\ \text{s.t. } \mathcal{A}(X) = b \end{array} ;$$

$$W^{k+1} = \begin{array}{l} \operatorname*{argmin}_W \mathcal{F}^p(X^{k+1},W) \\ \text{s.t. } W \succ 0, \operatorname{Tr}(W) \leq 1 \end{array} ;$$

Choose $0 < \gamma^{k+2} \leq \gamma^{k+1}$;
$k = k+1$;
end

Algorithm 4: Alternative representation of the IRLS-p algorithm.

We note (and shall prove) that Algorithm 4 gives us the same X updates as the IRLS-p algorithm. This gives an interpretation to the IRLS-p algorithm as alternately minimizing an equivalent *Smooth Schatten-p problem* (28). Consider minimization with respect to W of $\mathcal{F}^p(X,W,\gamma)$ with X fixed. This problem can be re-formulated as

$$\begin{array}{ll} \text{minimize} & \operatorname{Tr} \widehat{W}(X^TX + \gamma I) \\ \text{subject to} & \operatorname{Tr} \widehat{W}^{\frac{p}{p-2}} \leq 1 \\ & \widehat{W} \succ 0, \end{array} \tag{29}$$

where $\widehat{W} = W^{\frac{p-2}{p}}$. The following lemma relates $\mathcal{F}^p(X,W,\gamma)$ with $\mathcal{J}^p(X,W,\gamma)$ (defined in (6)).

Lemma 11 *Let*

$$\widehat{\mathcal{W}^*} = \frac{(X^TX + \gamma I)^{\frac{p}{2}-1}}{(\operatorname{Tr}(X^TX + \gamma I)^{\frac{p}{2}})^{\frac{p-2}{p}}}.$$

Then $\widehat{\mathcal{W}^*}$ *is the optimal solution to* (29) *as well as the following problem:*

$$\begin{array}{ll} \textit{minimize} & \operatorname{Tr} \widehat{W}(X^TX + \gamma I) + \lambda(\operatorname{Tr} \widehat{W}^{\frac{p}{p-2}} - 1), \\ \textit{subject to} & \widehat{W} \succ 0, \end{array}$$

where $\lambda = \frac{2-p}{p}(\operatorname{Tr}(X^TX + \gamma I)^{\frac{p}{2}})^{\frac{2}{p}}$. *Furthermore, let*

$$\widetilde{W} = \operatorname*{argmin}_{W \succ 0} \mathcal{J}^p(X,W,\gamma).$$

Then $\mathcal{J}^p(X,\widetilde{W},\gamma) = f_p(X)$ *and*

$$\operatorname*{argmin}_X \left\{ \mathcal{F}^p(X,(\widehat{\mathcal{W}}^*)^{\frac{p}{p-2}},\gamma) :\ \mathcal{A}(X)=b \right\} = \operatorname*{argmin}_Z \left\{ \mathcal{J}^p(Z,\widetilde{W},\gamma) :\ \mathcal{A}(Z)=b \right\}.$$

Thus Lemma (11) shows that alternately minimizing $\mathcal{F}^p(X,W,\gamma)$ with respect to W followed by X (with constraints on W as in (29) and affine constraints on X) is equivalent to alternately minimizing $\mathcal{J}^p(X,W,\gamma)$ with respect to W followed by X (with affine constraints on X and $\{W : W \succ 0\}$).

Proof The Lagrangian for (29) is given by

$$L(\widehat{W},\lambda) \quad = \quad \operatorname{Tr}\widehat{W}(X^TX+\gamma I)+\lambda(\operatorname{Tr}\widehat{W}^{\frac{p}{p-2}}-1)$$

Note that $\lambda^* = \frac{2-p}{p}(\operatorname{Tr}(X^TX+\gamma I)^{\frac{p}{2}})^{\frac{2}{p}}$, $\widehat{\mathcal{W}}^* = \frac{(X^TX+\gamma I)^{\frac{p}{2}-1}}{(\operatorname{Tr}(X^TX+\gamma I)^{\frac{p}{2}})^{\frac{p-2}{p}}}$ satisfy the KKT conditions to (29). This is because $\operatorname{Tr}(\widehat{\mathcal{W}}^*)^{\frac{p}{p-2}} = 1$, $\widehat{\mathcal{W}}^* \succ 0$ and $\lambda^* > 0$. The complementary slackness is also true since the primal inequality constraint is tight. Since (29) is a convex problem and $\widehat{\mathcal{W}}^*$ satisfies the KKT conditions, it is the optimal solution to (29). It is also easy to see that $\widetilde{W} = \operatorname*{argmin}_{W\succ 0} \mathcal{J}^p(X,W,\gamma)$ where $\widetilde{W} = (X^TX+\gamma I)^{\frac{p}{2}-1}$. Also note that $\mathcal{J}^p(X,\widetilde{W},\gamma) = f_p(X)$. Now,

$$\begin{aligned}
\operatorname*{argmin}_{Z:\mathcal{A}(Z)=b} \mathcal{J}^p(Z,\widetilde{W},\gamma) &= \operatorname*{argmin}_{Z:\mathcal{A}(Z)=b} \operatorname{Tr}\widetilde{W}(Z^TZ+\gamma I) \\
&= \operatorname*{argmin}_{Z:\mathcal{A}(Z)=b} \operatorname{Tr}(X^TX+\gamma I)^{\frac{p}{2}-1}(Z^TZ+\gamma I) \\
&= \operatorname*{argmin}_{Z:\mathcal{A}(Z)=b} \operatorname{Tr}\widehat{\mathcal{W}}^*(Z^TZ+\gamma I) \\
&= \operatorname*{argmin}_{Z:\mathcal{A}(Z)=b} \mathcal{F}^p(Z,(\widehat{\mathcal{W}}^*)^{\frac{p}{p-2}},\gamma).
\end{aligned}$$

■

Appendix B. Proof of Theorem 2

We first present two useful lemmas.

Lemma 12 *For each* $k \geq 1$*, we have*

$$\operatorname{Tr}(X^{k^T}X^k)^{\frac{p}{2}} \leq \mathcal{J}^p(X^1,W^0,\gamma^0) := D \tag{30}$$

where $W^0 = I$, $\gamma^0 = 1$. *Also,* $\lambda_j(W^k) \geq D^{(1-\frac{2}{p})},\ j = 1,2,\ldots,\min\{m,n\}$

Proof First, notice that

$$\begin{aligned}
\operatorname{Tr}(X^{k^T}X^k)^{\frac{p}{2}} &\leq \operatorname{Tr}(X^{k^T}X^k+\gamma I)^{p/2} = \mathcal{J}^p(X^k,W^k,\gamma^k) \\
&\leq \mathcal{J}^p(X^1,W^1,\gamma^1) \leq \mathcal{J}^p(X^1,W^0,\gamma^0) = D,
\end{aligned}$$

where the second and third inequalities follow from (6). This proves (30). Furthermore, from the above chain of inequalities,

$$(\|X^{k^T}X^k\|+\gamma)^{\frac{p}{2}} = \|(X^{k^T}X^k+\gamma I)^{\frac{p}{2}}\| \le D.$$

Using this and the definition of W^k, we obtain

$$\|(W^k)^{-1}\| = \|(X^{k^T}X^k+\gamma I)^{1-\frac{p}{2}}\| = (\|X^{k^T}X^k\|+\gamma)^{1-\frac{p}{2}} \le D^{(\frac{2}{p}-1)}.$$

This last relation shows that $\lambda_j(W^k) = \sigma_j(W^k) \ge 1/\|(W^k)^{-1}\| \ge D^{(1-\frac{2}{p})}$ for all j. ■

Lemma 13 *A matrix X^* is a minimizer of*

$$\begin{array}{ll} \textit{minimize} & \mathrm{Tr}WX^TX \\ \textit{subject to} & \mathcal{A}(X) = b \end{array}$$

if and only if $\mathrm{Tr}(WX^{*T}Z) = 0$ *for all* $Z \in \mathcal{N}(A)$.

Proof [of Theorem (2)] For each $k \ge 1$, we have that

$$\begin{array}{rcl} 2[\mathcal{J}^p(X^k,W^k,\gamma^k) - \mathcal{J}^p(X^{k+1},W^{k+1},\gamma^{k+1})] & \ge & 2[\mathcal{J}^1(X^k,W^k,\gamma^k) - \mathcal{J}^1(X^{k+1},W^k,\gamma^k)] \\ & = & \langle X^k,X^k\rangle_{W^k} - \langle X^{k+1},X^{k+1}\rangle_{W^k} \\ & = & \langle X^k+X^{k+1},X^k-X^{k+1}\rangle_{W^k} \\ & = & \langle X^k-X^{k+1},X^k-X^{k+1}\rangle_{W^k} \\ & = & \mathrm{Tr}W^k(X^k-X^{k+1})^T(X^k-X^{k+1}) \\ & \ge & D^{(1-\frac{2}{p})}\|X^k-X^{k+1}\|_F^2 \end{array}$$

where the above expressions use Lemma 13 and Lemma 12. Summing the above inequalities over all $k \ge 1$, we have that $\lim_{n\to\infty}\left(X^n - X^{n+1}\right) = 0$. ■

Appendix C. Proof of Theorem 8

For any two matrices X, Y we denote $\langle X,Y\rangle_W = \mathrm{Tr}WX^TY$. We first note that the iterates of sIRLS-p satisfy

$$\begin{array}{rcl} \mathcal{J}^p(X^{k+1},W^{k+1},\gamma^{k+1}) & \le & \mathcal{J}^p(X^{k+1},W^k,\gamma^{k+1}) \\ & \le & \mathcal{J}^p(X^{k+1},W^k,\gamma^k) \\ & \le & \mathcal{J}^p(X^k,W^k,\gamma^k). \end{array}$$

The last inequality follows from the Lipschitz continuity (with $L^k = 2(\gamma^k)^{\frac{p}{2}-1}$) of the gradient of $\mathrm{Tr}(W^kX^TX)$, that is,

$$\mathrm{Tr}W^kX^TX \le \mathrm{Tr}W^kX^{k^T}X^k + \langle 2X^kW^k, X-X^k\rangle + \frac{L^k}{2}\|X-X^k\|_F^2 \quad \forall X, X^k$$

and the fact that

$$\begin{aligned} X^{k+1} &= \arg\min_X \|X - (X^k - X^k W^k)\|_F^2 \\ &\quad \text{s.t. } \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(X_0). \end{aligned}$$

The convergence of $\{\mathcal{J}^p(X^k, W^k, \gamma^k)\}$ follows from Monotone convergence theorem. This also implies that the sequence $\{X^k\}$ is bounded. Hence there exists a convergent subsequence, $\{X^{n_i}\} \to X^*$. Also let $\{X^{n_i+1}\} \to \widehat{X}$. If X^* is a stationary point, we are done. Conversely, if X^* is not a stationary point to (26) then it follows that, $\widehat{X} \neq X^*$. But $\widehat{X} \neq X^*$ implies (using strict convexity) that $\mathrm{Tr}(W^*\widehat{X}^T\widehat{X}) < \mathrm{Tr}(W^*X^{*T}X^*)$ which also implies that $\mathcal{J}^p(\widehat{X}, \widehat{W}, \gamma_{\min}) < \mathcal{J}^p(X^*, W^*, \gamma_{\min})$. However since

$$\begin{aligned} \lim \mathcal{J}^p(X^i, W^i, \gamma^i) &= \lim \mathcal{J}^p(X^{n_i}, W^{n_i}, \gamma^{n_i}) \\ &= \lim \mathcal{J}^p(X^{n_{i+1}}, W^{n_{i+1}}, \gamma^{n_{i+1}}), \end{aligned}$$

we have a contradiction. Therefore, X^* is a stationary point to (26) and the theorem follows.

References

B.J. Dahlen, J.A. Konstan, J. Herlocker, N. Good, A. Borchers, and J. Riedl. "Movie lens data". 1998. http://www.grouplens.org/node/73.

A. Argyriou. A study of convex regularizers for sparse recovery and feature selection. 2010. Technical Report. Available at http://ttic.uchicago.edu/~argyriou/papers/sparse_report.pdf.

A. Argyriou, C.A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In *Proc. of Neural Information Processing Systems (NIPS)*, 2007.

J.F. Cai, E.J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, 2008.

E. J. Candes and S. Becker. Software for singular value thresholding algorithm for matrix completion. 2010. Available at http://svt.caltech.edu/code.html.

E. J. Candes and B. Recht. Exact matrix completion via convex optimziation. *Foundations of Computational Mathematics*, 9:717–772, 2009.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans Info. Theory*, 2004.

E. J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), 2011.

E.J. Candes, M.B. Wakin, and S. Boyd. Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, 14:877–905, 2008.

V. Chandrasekaran, P.A. Parrilo, and A.S. Willsky. Latent variable graphical model selection via convex optimization. *Annals of Statistics*, forthcoming.

V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky. The convex geometry of linear inverse problems. *Journal of the Foundations of Compuataional Mathematics*, forthcoming.

V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

R. Chartrand and V. Staneva. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24(035020):1–14, 2008.

R. Chartrand and W. Yin. Iteratively reweighted algorithms for compressive sensing. In *33rd International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

I. Daubechies, R. DeVore, M. Fornasier, and C.S. Gunturk. Iteratively re-weighted least squares minimization for sparse recovery. *Commun. Pure Appl. Math*, 63(1):1–38, 2010.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low rank approximation to a matrix. *SIAM Journal on Computing*, 36:158–183, 2006.

M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proc. American Control Conference*, Arlington, VA, 2001.

M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proc. American Control Conference*, pages 2156–2162, Denver, CO, 2003.

M. Fornasier, H. Rauhut, and R. Ward. Low-rank matrix recovery via iteratively reweighted least squares minimization. *SIAM Journal of Optimization*, 21(4), 2011.

M. Fornasier, H. Rauhut and R. Ward. Code for IRLSM. Available at `http://www.ma.utexas.edu/users/rward/`.

R. Garg and R. Khandekar. Gradient descent with sparsification: An iterative algorithm for sparse recovery with restricted isometry property. In *Proc. of 26th Intl. Conf. on Machine Learning (ICML)*, 2009.

D. Goldfarb and S. Ma. Convergence of fixed point continuation algorithms for matrix rank minimization. *Foundations of Computational Mathematics*, 11(2), 2011.

D. Goldfarb and S. Ma. FPCA code. 2009. Available at `http://www.columbia.edu/~sm2756/FPCA.htm`.

D. Gross, Y. K. Liu, S. T. Flammia, S. Becker, and J. Eisert. Quantum state tomography via compressed sensing. *Physical Review Letters*, 105, 2010.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

R. H. Keshavan and S. Oh. A gradient descent algorithm on the grassman manifold for matrix completion. 2009. Available at `http://arxiv.org/abs/0910.5260`.

R. H. Keshavan, A. Montanari, and S. Oh. Optspace algorithm. 2009a. `http://www.stanford.edu/~raghuram/optspace/index.html`.

R. H. Keshavan, A. Montanari, and S. Oh. Low-rank matrix completion with noisy observations: a quantitative comparison. In *Proc. 47th Annual Allerton Conference on Communication, Control, and Computing*, 2009b.

K. Lee and Y. Bresler. Admira: Atomic decomposition for minimum rank approximation. *IEEE Tran. Info. Theory*, 56(9), 2010.

A.S. Lewis. Derivatives of spectral functions. *Mathematics of Operations Research*, 21(3):576–588, 1996.

Z. Liu and L. Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM J. Matrix Analysis and Appl.*, 31(3), 2008.

M. S. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152:341–365, 2006.

A. W. Marshall and I. Olkin. *Inequalities: Theory of Majorization and Its Applications*. 1979.

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287 – 2322, 2010.

R. Meka, P. Jain, and I. S. Dhillon. Guaranteed rank minimization via singular value projection. In *Proc. of Neural Information Processing Systems (NIPS)*, 2010.

K. Mohan and M. Fazel. Reweighted nuclear norm minimization with application to system identification. In *Proc. American Control Conference*, Baltimore, MA, 2010a.

K. Mohan and M. Fazel. Iterative reweighted least squares for matrix rank minimization. In *Proc. 48th Allerton Conference on Controls and Communications*, Allerton, IL, 2010b.

D. Needell. Noisy signal recovery via iterative reweighted l1-minimization. In *Proc. Asilomar Conference on Signals, Systems and Computers*, 2009.

D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2008.

S. Oymak and B. Hassibi. New null space results and recovery thresholds for matrix rank minimization. 2010. Available at `http://arxiv.org/abs/1011.6326`.

S. Oymak, K. Mohan, M. Fazel, and B. Hassibi. A simplified approach to recovery conditions for low rank matrices. In *Proc. International Symposium on Information Theory*, 2011.

B. D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Transactions on Signal Processing*, 47:187–200, 1999.

N. Srebro, J.D.M. Rennie, and T.S. Jakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, 2005b.

W. Tan, G. Cheung, and Y. Ma. Face recovery in conference video streaming using robust principal component analysis. In *Proc. IEEE Intl. Conf. on Image Processing*, 2011.

K. C. Toh and S. W. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problem. *Pacific Journal of Optimization*, 6:615–640, 2010.

D. P. Wipf and S. Nagarajan. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *Journal of Selected Topics in Signal Proessing (Special Issue on Compressive Sensing)*, 4(2), 2010.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010.

Journal of Machine Learning Research 13 (2012) 3475-3506 Submitted 7/12; Published 12/12

Fast Approximation of Matrix Coherence and Statistical Leverage

Petros Drineas DRINEP@CS.RPI.EDU
Malik Magdon-Ismail MAGDON@CS.RPI.EDU
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180

Michael W. Mahoney MMAHONEY@CS.STANFORD.EDU
Department of Mathematics
Stanford University
Stanford, CA 94305

David P. Woodruff DPWOODRU@US.IBM.COM
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120

Editor: Mehryar Mohri

Abstract

The *statistical leverage scores* of a matrix A are the squared row-norms of the matrix containing its (top) left singular vectors and the *coherence* is the largest leverage score. These quantities are of interest in recently-popular problems such as matrix completion and Nyström-based low-rank matrix approximation as well as in large-scale statistical data analysis applications more generally; moreover, they are of interest since they define the key structural nonuniformity that must be dealt with in developing fast randomized matrix algorithms. Our main result is a randomized algorithm that takes as input an arbitrary $n \times d$ matrix A, with $n \gg d$, and that returns as output relative-error approximations to *all* n of the statistical leverage scores. The proposed algorithm runs (under assumptions on the precise values of n and d) in $O(nd \log n)$ time, as opposed to the $O(nd^2)$ time required by the naïve algorithm that involves computing an orthogonal basis for the range of A. Our analysis may be viewed in terms of computing a relative-error approximation to an *under*constrained least-squares approximation problem, or, relatedly, it may be viewed as an application of Johnson-Lindenstrauss type ideas. Several practically-important extensions of our basic result are also described, including the approximation of so-called cross-leverage scores, the extension of these ideas to matrices with $n \approx d$, and the extension to streaming environments.

Keywords: matrix coherence, statistical leverage, randomized algorithm

1. Introduction

The concept of *statistical leverage* measures the extent to which the singular vectors of a matrix are correlated with the standard basis and as such it has found usefulness recently in large-scale data analysis and in the analysis of randomized matrix algorithms (Velleman and Welsch, 1981; Mahoney and Drineas, 2009; Drineas et al., 2008). A related notion is that of *matrix coherence*, which has been of interest in recently popular problems such as matrix completion and Nyström-based low-rank matrix approximation (Candes and Recht, 2008; Talwalkar and Rostamizadeh, 2010). Defined

more precisely below, the statistical leverage scores may be computed as the squared Euclidean norms of the rows of the matrix containing the top left singular vectors and the coherence of the matrix is the largest statistical leverage score. Statistical leverage scores have a long history in statistical data analysis, where they have been used for outlier detection in regression diagnostics (Hoaglin and Welsch, 1978; Chatterjee and Hadi, 1986). Statistical leverage scores have also proved crucial recently in the development of improved worst-case randomized matrix algorithms that are also amenable to high-quality numerical implementation and that are useful to domain scientists (Drineas et al., 2008; Mahoney and Drineas, 2009; Boutsidis et al., 2009; Drineas et al., 2006b; Sarlós, 2006; Drineas et al., 2010b); see Mahoney (2011) for a detailed discussion. The naïve and best previously existing algorithm to compute these scores would compute an orthogonal basis for the dominant part of the spectrum of A, for example, the basis provided by the Singular Value Decomposition (SVD) or a basis provided by a QR decomposition (Golub and Loan, 1996), and then use that basis to compute diagonal elements of the projection matrix onto the span of that basis.

We present a randomized algorithm to compute relative-error approximations to every statistical leverage score in time qualitatively faster than the time required to compute an orthogonal basis. For the case of an arbitrary $n \times d$ matrix A, with $n \gg d$, our main algorithm runs (under assumptions on the precise values of n and d, see Theorem 2 for an exact statement) in $O(nd \log n/\varepsilon^2)$ time, as opposed to the $\Theta(nd^2)$ time required by the naïve algorithm. As a corollary, our algorithm provides a relative-error approximation to the coherence of an arbitrary matrix in the same time. In addition, several practically-important extensions of the basic idea underlying our main algorithm are also described in this paper.

1.1 Overview and Definitions

We start with the following definition of the *statistical leverage scores* of a matrix.

Definition 1 *Given an arbitrary $n \times d$ matrix A, with $n > d$, let U denote the $n \times d$ matrix consisting of the d left singular vectors of A, and let $U_{(i)}$ denote the i-th row of the matrix U as a row vector. Then, the* statistical leverage scores *of the rows of A are given by*

$$\ell_i = \left\|U_{(i)}\right\|_2^2,$$

for $i \in \{1,\ldots,n\}$; the coherence γ *of the rows of A is*

$$\gamma = \max_{i \in \{1,\ldots,n\}} \ell_i,$$

that is, it is the largest statistical leverage score of A; and the (i,j)-cross-leverage scores c_{ij} *are*

$$c_{ij} = \left\langle U_{(i)}, U_{(j)} \right\rangle,$$

that is, they are the dot products between the i^{th} row and the j^{th} row of U.

Although we have defined these quantities in terms of a particular basis, they clearly do not depend on that particular basis, but only on the space spanned by that basis. To see this, let P_A denote the projection matrix onto the span of the columns of A. Then,

$$\ell_i = \left\|U_{(i)}\right\|_2^2 = \left(UU^T\right)_{ii} = (P_A)_{ii}. \tag{1}$$

That is, the statistical leverage scores of a matrix A are equal to the diagonal elements of the projection matrix onto the span of its columns.[1] Similarly, the (i, j)-cross-leverage scores are equal to the off-diagonal elements of this projection matrix, that is,

$$c_{ij} = (P_A)_{ij} = \left\langle U_{(i)}, U_{(j)} \right\rangle. \tag{2}$$

Clearly, $O(nd^2)$ time suffices to compute all the statistical leverage scores exactly: simply perform the SVD or compute a QR decomposition of A in order to obtain *any* orthogonal basis for the range of A and then compute the Euclidean norm of the rows of the resulting matrix. Thus, in this paper, we are interested in algorithms that run in $o(nd^2)$ time.

Several additional comments are worth making regarding this definition. First, since $\sum_{i=1}^{n} \ell_i = \|U\|_F^2 = d$, we can define a probability distribution over the rows of A as $p_i = \ell_i/d$. As discussed below, these probabilities have played an important role in recent work on randomized matrix algorithms and an important algorithmic question is the degree to which they are uniform or nonuniform.[2] Second, one could also define leverage scores for the columns of a "tall" matrix A, but clearly those are all equal to one unless $n < d$ or A is rank-deficient. Third, and more generally, given a rank parameter k, one can define the *statistical leverage scores relative to the best rank-k approximation to A* to be the n diagonal elements of the projection matrix onto the span of A_k, the best rank-k approximation to A.

1.2 Our Main Result

Our main result is a randomized algorithm for computing relative-error approximations to every statistical leverage score, as well as an additive-error approximation to all of the large cross-leverage scores, of an arbitrary $n \times d$ matrix, with $n \gg d$, in time qualitatively faster than the time required to compute an orthogonal basis for the range of that matrix. Our main algorithm for computing approximations to the statistical leverage scores (see Algorithm 1 in Section 3) will amount to constructing a "randomized sketch" of the input matrix and then computing the Euclidean norms of the rows of that sketch. This sketch can also be used to compute approximations to the large cross-leverage scores (see Algorithm 2 of Section 3).

The following theorem provides our main quality-of-approximation and running time result for Algorithm 1.

Theorem 2 *Let A be a full-rank $n \times d$ matrix, with $n \gg d$; let $\varepsilon \in (0, 1/2]$ be an error parameter; and recall the definition of the statistical leverage scores ℓ_i from Definition 1. Then, there exists a randomized algorithm (Algorithm 1 of Section 3 below) that returns values $\tilde{\ell}_i$, for all $i \in \{1, \ldots, n\}$, such that with probability at least* 0.8*,*

$$\left|\ell_i - \tilde{\ell}_i\right| \leq \varepsilon \ell_i$$

1. In this paper, for simplicity of exposition, we consider the case that the matrix A has rank equal to d, that is, has full column rank. Theoretically, the extension to rank-deficient matrices A is straightforward—simply modify Definition 1 and thus Equations (1) and (2) to let U be any orthogonal matrix (clearly, with fewer than d columns) spanning the column space of A. From a numerical perspective, things are substantially more subtle, and we leave this for future work that considers numerical implementations of our algorithms.
2. Observe that if U consists of d columns from the identity, then the leverage scores are extremely nonuniform: d of them are equal to one and the remainder are equal to zero. On the other hand, if U consists of d columns from a normalized Hadamard transform (see Section 2.3 for a definition), then the leverage scores are very uniform: all n of them are equal to d/n.

holds for all $i \in \{1,\ldots,n\}$. *Assuming* $d \leq n \leq e^d$, *the running time of the algorithm is*

$$O\left(nd\ln\left(d\varepsilon^{-1}\right)+nd\varepsilon^{-2}\ln n+d^3\varepsilon^{-2}\left(\ln n\right)\left(\ln\left(d\varepsilon^{-1}\right)\right)\right).$$

Algorithm 1 provides a relative-error approximation to all of the statistical leverage scores ℓ_i of A and, assuming $d\ln d = o\left(\frac{n}{\ln n}\right)$, $\ln n = o(d)$, and treating ε as a constant, its running time is $o(nd^2)$, as desired. As a corollary, the largest leverage score (and thus the coherence) is approximated to relative-error in $o(nd^2)$ time.

The following theorem provides our main quality-of-approximation and running time result for Algorithm 2.

Theorem 3 *Let* A *be a full-rank* $n \times d$ *matrix, with* $n \gg d$; *let* $\varepsilon \in (0,1/2]$ *be an error parameter; let* κ *be a parameter; and recall the definition of the cross-leverage scores* c_{ij} *from Definition 1. Then, there exists a randomized algorithm (Algorithm 2 of Section 3 below) that returns the pairs* $\{(i,j)\}$ *together with estimates* $\{\tilde{c}_{ij}\}$ *such that, with probability at least* 0.8,

i. If $c_{ij}^2 \geq \dfrac{d}{\kappa}+12\varepsilon\ell_i\ell_j$, *then* (i,j) *is returned; if* (i,j) *is returned, then* $c_{ij}^2 \geq \dfrac{d}{\kappa}-30\varepsilon\ell_i\ell_j$.

ii. For all pairs (i,j) *that are returned,* $\tilde{c}_{ij}^2-30\varepsilon\ell_i\ell_j \leq c_{ij}^2 \leq \tilde{c}_{ij}^2+12\varepsilon\ell_i\ell_j$.

This algorithm runs in $O(\varepsilon^{-2}n\ln n+\varepsilon^{-3}\kappa d\ln^2 n)$ *time.*

Note that by setting $\kappa = n\ln n$, we can compute all the "large" cross-leverage scores, that is, those satisfying $c_{ij}^2 \geq \frac{d}{n\ln n}$, to within additive-error in $O\left(nd\ln^3 n\right)$ time (treating ε as a constant). If $\ln^3 n = o(d)$ the overall running time is $o(nd^2)$, as desired.

1.3 Significance and Related Work

Our results are important for their applications to fast randomized matrix algorithms, as well as their applications in numerical linear algebra and large-scale data analysis more generally.

Significance in theoretical computer science. The statistical leverage scores define the key structural nonuniformity that must be dealt with (i.e., either rapidly approximated or rapidly uniformized at the preprocessing step) in developing fast randomized algorithms for matrix problems such as least-squares regression (Sarlós, 2006; Drineas et al., 2010b) and low-rank matrix approximation (Papadimitriou et al., 2000; Sarlós, 2006; Drineas et al., 2008; Mahoney and Drineas, 2009; Boutsidis et al., 2009). Roughly, the best random sampling algorithms use these scores (or the generalized leverage scores relative to the best rank-k approximation to A) as an importance sampling distribution to sample with respect to. On the other hand, the best random projection algorithms rotate to a basis where these scores are approximately uniform and thus in which uniform sampling is appropriate. See Mahoney (2011) for a detailed discussion.

As an example, the CUR decomposition of Drineas et al. (2008) and Mahoney and Drineas (2009) essentially computes $p_i = \ell_i/k$, for all $i \in \{1,\ldots,n\}$ and for a rank parameter k, and it uses these as an importance sampling distribution. The computational bottleneck for these and related random sampling algorithms is the computation of the importance sampling probabilities. On the other hand, the computational bottleneck for random projection algorithms is the application of the random projection, which is sped up by using variants of the Fast Johnson-Lindenstrauss Transform (Ailon and Chazelle, 2006, 2009). By our main result, the leverage scores (and thus

these probabilities) can be approximated in time that depends on an application of a Fast Johnson-Lindenstrauss Transform. In particular, the random sampling algorithms of Drineas et al. (2006b), Drineas et al. (2008) and Mahoney and Drineas (2009) for least-squares approximation and low-rank matrix approximation now run in time that is essentially the same as the best corresponding random projection algorithm for those problems (Sarlós, 2006).

Applications to numerical linear algebra. Recently, high-quality numerical implementations of variants of the basic randomized matrix algorithms have proven superior to traditional deterministic algorithms (Rokhlin and Tygert, 2008; Rokhlin et al., 2009; Avron et al., 2010). An important question raised by our main results is how these will compare with an implementation of our main algorithm. More generally, density functional theory (Bekas et al., 2007) and uncertainty quantification (Bekas et al., 2009) are two scientific computing areas where computing the diagonal elements of functions (such as a projection or inverse) of very large input matrices is common. For example, in the former case, "heuristic" methods based on using Chebychev polynomials have been used in numerical linear algebra to compute the diagonal elements of the projector (Bekas et al., 2007). Our main algorithm should have implications in both of these areas.

Applications in large-scale data analysis. The statistical leverage scores and the scores relative to the best rank-k approximation to A are equal to the diagonal elements of the so-called "hat matrix" (Hoaglin and Welsch, 1978; Chatterjee and Hadi, 1988). As such, they have a natural statistical interpretation in terms of the "leverage" or "influence" associated with each of the data points (Hoaglin and Welsch, 1978; Chatterjee and Hadi, 1986, 1988). In the context of regression problems, the i^{th} leverage score quantifies the leverage or influence of the i^{th} constraint/row of A on the solution of the overconstrained least squares optimization problem $\min_x \|Ax-b\|_2$ and the (i,j)-th cross leverage score quantifies how much influence or leverage the i^{th} data point has on the j^{th} least-squares fit (see Hoaglin and Welsch, 1978; Chatterjee and Hadi, 1986, 1988, for details). When applied to low-rank matrix approximation problems, the leverage score ℓ_j quantifies the amount of leverage or influence exerted by the j^{th} column of A on its optimal low-rank approximation. Historically, these quantities have been widely-used for outlier identification in diagnostic regression analysis (Velleman and Welsch, 1981; Chatterjee et al., 2000).

More recently, these scores (usually the largest scores) often have an interpretation in terms of the data and processes generating the data that can be exploited. For example, depending on the setting, they can have an interpretation in terms of high-degree nodes in data graphs, very small clusters in noisy data, coherence of information, articulation points between clusters, the value of a customer in a network, space localization in sensor networks, etc. (Bonacich, 1987; Richardson and Domingos, 2002; Newman, 2005; Jonckheere et al., 2007; Mahoney, 2011). In genetics, dense matrices of size thousands by hundreds of thousands (a size scale at which even traditional deterministic QR algorithms fail to run) constructed from DNA Single Nucleotide Polymorphisms (SNP) data are increasingly common, and the statistical leverage scores can correlate strongly with other metrics of genetic interest (Paschou et al., 2007; Mahoney and Drineas, 2009; Drineas et al., 2010a; Paschou et al., 2010). Our main result will permit the computation of these scores and related quantities for significantly larger SNP data sets than has been possible previously (Paschou et al., 2007; Drineas et al., 2010a; Paschou et al., 2010; Georgiev and Mukherjee, 2011).

Remark. Lest there be any confusion, we should emphasize our main contributions. First, note that statistical leverage and matrix coherence are important concepts in statistics and machine learning. Second, recall that several random sampling algorithms for ubiquitous matrix problems such as least-squares approximation and low-rank matrix approximation use leverage scores in a

crucial manner; but until now these algorithms were $\Omega(T_{SVD})$, where T_{SVD} is the time required to compute a QR decomposition or a partial SVD of the input matrix. Third, note that, in some cases, $o(T_{SVD})$ algorithms exist for these problems based on fast random projections. But recall that the existence of those projection algorithms *in no way implies* that it is easy or obvious how to compute the statistical leverage scores efficiently. Fourth, one implication of our main result is that those random sampling algorithms can now be performed *just as efficiently* as those random projection algorithms; thus, the solution for those matrix problems can now be obtained while preserving the identity of the rows. That is, these problems can now be solved just as efficiently by using actual rows, rather than the arbitrary linear combinations of rows that are returned by random projections. Fifth, we provide a generalization to "fat" matrices and to obtaining the cross-leverage scores. Sixth, we develop algorithms that can compute leverage scores and related statistics even in streaming environments.

1.4 Empirical Discussion of Our Algorithms

Although the main contribution of our paper is to provide a rigorous theoretical understanding of fast leverage score approximation, our paper does analyze the theoretical performance of what is meant to be a practical algorithm. Thus, one might wonder about the empirical performance of our algorithms—for example, whether hidden constants render the algorithms useless for data of realistic size. Not surprisingly, this depends heavily on the quality of the numerical implementation, whether one is interested in "tall" or more general matrices, etc. We will consider empirical and numerical aspects of these algorithms in forthcoming papers, for example, Gittens and Mahoney (2012). We will, however, provide here a brief summary of several numerical issues for the reader interested in these issues.

Empirically, the running time bottleneck for our main algorithm (Algorithm 1 of Section 3) applied to "tall" matrices is the application of the random projection Π_1. Thus, empirically the running time is similar to the running time of random projection based methods for computing approximations to the least-squares problem, which is also dominated by the application of the random projection. The state of the art here is the Blendenpik algorithm of Avron et al. (2010) and the LSRN algorithm of Meng et al. (2011). In their Blendenpik paper, Avron, Maymounkov, and Toledo showed that their high-quality numerical implementation of a Hadamard-based random projection (and associated least-squares computation) "beats LAPACK's[3] direct dense least-squares solver by a large margin on essentially any dense tall matrix," and they concluded that their empirical results "suggest that random projection algorithms should be incorporated into future versions of LAPACK" (Avron et al., 2010). The LSRN algorithm of Meng, Saunders, and Mahoney improves Blendenpik in several respects, for example, providing better handling of sparsity and rank deficiency, but most notably the random projection underlying LSRN is particularly appropriate for solving large problems on clusters with high communication cost, for example, it has been shown to scale well on Amazon Elastic Cloud Compute clusters. Thus, our main algorithm should extend easily to these environments with the use of the random projection underlying LSRN. Moreover, for both Blendenpik and LSRN (when implemented with a Hadamard-based random projection), the hidden constants in the Hadamard-based random projection are so *small* that the random projection algorithm (and thus the empirical running time of our main algorithm for approximating leverage

3. LAPACK (short for Linear Algebra PACKage) is a high-quality and widely-used software library of numerical routines for solving a wide range of numerical linear algebra problems.

scores) beats the traditional $O(nd^2)$ time algorithm for dense matrices as small as thousands of rows by hundreds of columns.

1.5 Outline

In Section 2, we will provide a brief review of relevant notation and concepts from linear algebra. Then, in Sections 3 and 4, we will present our main results: Section 3 will contain our main algorithm and Section 4 will contain the proof of our main theorem. Section 5 will then describe extensions of our main result to general "fat" matrices, that is, those with $n \approx d$. Section 6 will conclude by describing the relationship of our main result with another related estimator for the statistical leverage scores, an application of our main algorithm to the under-constrained least-squares approximation problem, and extensions of our main algorithm to streaming environments.

2. Preliminaries on Linear Algebra and Fast Random Projections

We will start with a review of basic linear algebra and notation, and then we will describe the Fast Johnson-Lindenstrauss Transform and the related Subsampled Randomized Hadamard Transform.

2.1 Basic Linear Algebra and Notation

Let $[n]$ denote the set of integers $\{1,2,\ldots,n\}$. For any matrix $A \in \mathbb{R}^{n\times d}$, let $A_{(i)}$, $i \in [n]$, denote the i-th row of A as a row vector, and let $A^{(j)}$, $j \in [d]$ denote the j-th column of A as a column vector. Let $\|A\|_F^2 = \sum_{i=1}^n \sum_{j=1}^d A_{ij}^2$ denote the square of the Frobenius norm of A, and let $\|A\|_2 = \sup_{\|x\|_2=1} \|Ax\|_2$ denote the spectral norm of A. Relatedly, for any vector $x \in \mathbb{R}^n$, its Euclidean norm (or ℓ_2-norm) is the square root of the sum of the squares of its elements. The dot product between two vectors $x, y \in \mathbb{R}^n$ will be denoted $\langle x, y\rangle$, or alternatively as $x^T y$. Finally, let $e_i \in \mathbb{R}^n$, for all $i \in [n]$, denote the standard basis vectors for $\mathbb{R}^n$ and let I_n denote the $n \times n$ identity matrix.

Let the rank of A be $\rho \leq \min\{n,d\}$, in which case the "compact" or "thin" SVD of A is denoted by $A = U\Sigma V^T$, where $U \in \mathbb{R}^{n\times\rho}$, $\Sigma \in \mathbb{R}^{\rho\times\rho}$, and $V \in \mathbb{R}^{d\times\rho}$. (For a general matrix X, we will write $X = U_X \Sigma_X V_X^T$.) Let $\sigma_i(A), i \in [\rho]$ denote the i-th singular value of A, and let $\sigma_{\max}(A)$ and $\sigma_{\min}(A)$ denote the maximum and minimum singular values of A, respectively. The Moore-Penrose pseudoinverse of A is the $d \times n$ matrix defined by $A^\dagger = V\Sigma^{-1}U^T$ (Nashed, 1976). Finally, for any orthogonal matrix $U \in \mathbb{R}^{n\times\ell}$, let $U^\perp \in \mathbb{R}^{n\times(n-\ell)}$ denote an orthogonal matrix whose columns are an orthonormal basis spanning the subspace of $\mathbb{R}^n$ that is orthogonal to the subspace spanned by the columns of U (i.e., the range of U). It is always possible to extend an orthogonal matrix U to a full orthonormal basis of $\mathbb{R}^n$ as $[U \quad U^\perp]$.

The SVD is important for a number of reasons (Golub and Loan, 1996). For example, the projection of the columns of A onto the k left singular vectors associated with the top k singular values gives the best rank-k approximation to A in the spectral and Frobenius norms. Relatedly, the solution to the least-squares (LS) approximation problem is provided by the SVD: given an $n \times d$ matrix A and an n-vector b, the LS problem is to compute the minimum ℓ_2-norm vector x such that $\|Ax-b\|_2$ is minimized over all vectors $x \in \mathbb{R}^d$. This optimal vector is given by $x_{opt} = A^\dagger b$. We call a LS problem *overconstrained* (or *overdetermined*) if $n > d$ and *underconstrained* (or *underdetermined*) if $n < d$.

2.2 The Fast Johnson-Lindenstrauss Transform (FJLT)

Given $\varepsilon > 0$ and a set of points $x_1, \ldots, x_n$ with $x_i \in \mathbb{R}^d$, a ε-Johnson-Lindenstrauss Transform (ε-JLT), denoted $\Pi \in \mathbb{R}^{r \times d}$, is a projection of the points into $\mathbb{R}^r$ such that

$$(1-\varepsilon)\|x_i\|_2^2 \leq \|\Pi x_i\|_2^2 \leq (1+\varepsilon)\|x_i\|_2^2.$$

To construct an ε-JLT with high probability, simply choose every entry of Π independently, equal to $\pm\sqrt{3/r}$ with probability $1/6$ each and zero otherwise (with probability $2/3$) (Achlioptas, 2003). Let Π_{JLT} be a matrix drawn from such a distribution over $r \times d$ matrices.[4] Then, the following lemma holds.

Lemma 4 (Theorem 1.1 of Achlioptas (2003)) *Let $x_1, \ldots, x_n$ be an arbitrary (but fixed) set of points, where $x_i \in \mathbb{R}^d$ and let $0 < \varepsilon \leq 1/2$ be an accuracy parameter. If*

$$r \geq \frac{1}{\varepsilon^2}\left(12\ln n + 6\ln\frac{1}{\delta}\right)$$

then, with probability at least $1-\delta$, $\Pi_{JLT} \in \mathbb{R}^{r \times d}$ is an ε-JLT.

For our main results, we will also need a stronger requirement than the simple ε-JLT and so we will use a version of the Fast Johnson-Lindenstrauss Transform (FJLT), which was originally introduced in Ailon and Chazelle (2006) and Ailon and Chazelle (2009). Consider an orthogonal matrix $U \in \mathbb{R}^{n \times d}$, viewed as d vectors in $\mathbb{R}^n$. A FJLT projects the vectors from $\mathbb{R}^n$ to $\mathbb{R}^r$, while preserving the orthogonality of U; moreover, it does so very quickly. Specifically, given $\varepsilon > 0$, $\Pi \in \mathbb{R}^{r \times n}$ is an ε-FJLT for U if

- $\left\|I_d - U^T\Pi^T\Pi U\right\|_2 \leq \varepsilon$, and
- given any $X \in \mathbb{R}^{n \times d}$, the matrix product ΠX can be computed in $O(nd\ln r)$ time.

The next lemma follows from the definition of an ε-FJLT, and its proof can be found in Drineas et al. (2006b) and Drineas et al. (2010b).

Lemma 5 *Let A be any matrix in $\mathbb{R}^{n \times d}$ with $n \gg d$ and $\mathbf{rank}(A) = d$. Let the SVD of A be $A = U\Sigma V^T$, let Π be an ε-FJLT for U (with $0 < \varepsilon \leq 1/2$) and let $\Psi = \Pi U = U_\Psi \Sigma_\Psi V_\Psi^T$. Then, all the following hold:*

$$\begin{aligned}
\mathbf{rank}(\Pi A) &= \mathbf{rank}(\Pi U) = \mathbf{rank}(U) = \mathbf{rank}(A) = d, \\
\left\|I - \Sigma_\Psi^{-2}\right\|_2 &\leq \varepsilon/(1-\varepsilon), \text{ and} \qquad (3) \\
(\Pi A)^\dagger &= V\Sigma^{-1}(\Pi U)^\dagger. \qquad (4)
\end{aligned}$$

4. When no confusion can arise, we will use Π_{JLT} to refer to this distribution over matrices as well as to a specific matrix drawn from this distribution.

2.3 The Subsampled Randomized Hadamard Transform (SRHT)

One can use a Randomized Hadamard Transform (RHT) to construct, with high probability, an ε-FJLT. Our main algorithm will use this efficient construction in a crucial way.[5] Recall that the (unnormalized) $n \times n$ matrix of the Hadamard transform $\hat{H}_n$ is defined recursively by

$$\hat{H}_{2n} = \begin{bmatrix} \hat{H}_n & \hat{H}_n \\ \hat{H}_n & -\hat{H}_n \end{bmatrix},$$

with $\hat{H}_1 = 1$. The $n \times n$ normalized matrix of the Hadamard transform is equal to

$$H_n = \hat{H}_n / \sqrt{n}.$$

From now on, for simplicity and without loss of generality, we assume that n is a power of 2 and we will suppress n and just write H. (Variants of this basic construction that relax this assumption and that are more appropriate for numerical implementation have been described and evaluated in Avron et al. (2010).) Let $D \in \mathbb{R}^{n \times n}$ be a random diagonal matrix with independent diagonal entries $D_{ii} = +1$ with probability $1/2$ and $D_{ii} = -1$ with probability $1/2$. The product HD is a RHT and it has three useful properties. First, when applied to a vector, it "spreads out" its energy. Second, computing the product HDx for any vector $x \in \mathbb{R}^n$ takes $O(n \log_2 n)$ time. Third, if we only need to access r elements in the transformed vector, then those r elements can be computed in $O(n \log_2 r)$ time (Ailon and Liberty, 2008). The Subsampled Randomized Hadamard Transform (SRHT) randomly samples (according to the uniform distribution) a set of r rows of a RHT.

Using the sampling matrix formalism described previously (Drineas et al., 2006a,b, 2008, 2010b), we will represent the operation of randomly sampling r rows of an $n \times d$ matrix A using an $r \times n$ linear sampling operator S^T. Let the matrix $\Pi_{FJLT} = S^T HD$ be generated using the SRHT.[6] The most important property about the distribution Π_{FJLT} is that if r is large enough, then, with high probability, Π_{FJLT} generates an ε-FJLT. We summarize this discussion in the following lemma (which is essentially a combination of Lemmas 3 and 4 from Drineas et al. (2010b), restated to fit our notation).

Lemma 6 *Let $\Pi_{FJLT} \in \mathbb{R}^{r \times n}$ be generated using the SRHT as described above and let $U \in \mathbb{R}^{n \times d}$ ($n \gg d$) be an (arbitrary but fixed) orthogonal matrix. If*

$$r \geq \frac{14^2 d \ln(40nd)}{\varepsilon^2} \ln\left(\frac{30^2 d \ln(40nd)}{\varepsilon^2}\right),$$

then, with probability at least 0.9, Π_{FJLT} is an ε-FJLT for U.

3. Our Main Algorithmic Results

In this section, we will describe our main results for computing relative-error approximations to every statistical leverage score (see Algorithm 1) as well as additive-error approximations to all of the large cross-leverage scores (see Algorithm 2) of an arbitrary matrix $A \in \mathbb{R}^{n \times d}$, with $n \gg d$. Both algorithms make use of a "randomized sketch" of A of the form $A(\Pi_1 A)^{\dagger}\Pi_2$, where Π_1 is an ε-FJLT and Π_2 is an ε-JLT. We start with a high-level description of the basic ideas.

5. Note that the RHT has also been crucial in the development of $o(nd^2)$ randomized algorithms for the general over-constrained LS problem (Drineas et al., 2010b) and its variants have been used to provide high-quality numerical implementations of such randomized algorithms (Rokhlin and Tygert, 2008; Avron et al., 2010).

6. Again, when no confusion can arise, we will use Π_{FJLT} to denote a specific SRHT or the distribution on matrices implied by the randomized process for constructing an SRHT.

3.1 Outline of Our Basic Approach

Recall that our first goal is to approximate, for all $i \in [n]$, the quantities

$$\ell_i = \left\|U_{(i)}\right\|_2^2 = \left\|e_i^T U\right\|_2^2, \tag{5}$$

where e_i is a standard basis vector. The hard part of computing the scores ℓ_i according to Equation (5) is computing an orthogonal matrix U spanning the range of A, which takes $O(nd^2)$ time. Since $UU^T = AA^\dagger$, it follows that

$$\ell_i = \left\|e_i^T UU^T\right\|_2^2 = \left\|e_i^T AA^\dagger\right\|_2^2 = \left\|(AA^\dagger)_{(i)}\right\|_2^2, \tag{6}$$

where the first equality follows from the orthogonality of (the columns of) U. The hard part of computing the scores ℓ_i according to Equation (6) is two-fold: first, computing the pseudoinverse; and second, performing the matrix-matrix multiplication of A and $A^\dagger$. Both of these procedures take $O(nd^2)$ time. As we will see, we can get around both of these bottlenecks by the judicious application of random projections to Equation (6).

To get around the bottleneck of $O(nd^2)$ time due to computing $A^\dagger$ in Equation (6), we will compute the pseudoinverse of a "smaller" matrix that approximates A. A necessary condition for such a smaller matrix is that it preserves rank. So, naïve ideas such as uniformly sampling $r_1 \ll n$ rows from A and computing the pseudoinverse of this sampled matrix will not work well for an arbitrary A. For example, this idea will fail (with high probability) to return a meaningful approximation for matrices consisting of $n-1$ identical rows and a single row with a nonzero component in the direction perpendicular to that the identical rows; finding that "outlying" row is crucial to obtaining a relative-error approximation. This is where the SRHT enters, since it preserves important structures of A, in particular its rank, by first rotating A to a random basis and then uniformly sampling rows from the rotated matrix (see Drineas et al., 2010b for more details). More formally, recall that the SVD of A is $U\Sigma V^T$ and let $\Pi_1 \in \mathbb{R}^{r_1 \times n}$ be an ε-FJLT for U (using, for example, the SRHT of Lemma 6 with the appropriate choice for r_1). Then, one could approximate the ℓ_i's of Equation (6) by

$$\hat{\ell}_i = \left\|e_i^T A \left(\Pi_1 A\right)^\dagger\right\|_2^2, \tag{7}$$

where we approximated the $n \times d$ matrix A by the $r_1 \times d$ matrix $\Pi_1 A$. Computing $A\left(\Pi_1 A\right)^\dagger$ in this way takes $O\left(ndr_1\right)$ time, which is not efficient because $r_1 > d$ (from Lemma 6).

To get around this bottleneck, recall that we only need the Euclidean norms of the rows of the matrix $A\left(\Pi_1 A\right)^\dagger \in \mathbb{R}^{n \times r_1}$. Thus, we can further reduce the dimensionality of this matrix by using an ε-JLT to reduce the dimension $r_1 = \Omega(d)$ to $r_2 = O(\ln n)$. Specifically, let $\Pi_2^T \in \mathbb{R}^{r_2 \times r_1}$ be an ε-JLT for the rows of $A\left(\Pi_1 A\right)^\dagger$ (viewed as n vectors in $\mathbb{R}^{r_1}$) and consider the matrix $\Omega = A\left(\Pi_1 A\right)^\dagger \Pi_2$. This $n \times r_2$ matrix Ω may be viewed as our "randomized sketch" of the rows of $AA^\dagger$. Then, we can compute and return

$$\tilde{\ell}_i = \left\|e_i^T A \left(\Pi_1 A\right)^\dagger \Pi_2\right\|_2^2,$$

for each $i \in [n]$, which is essentially what Algorithm 1 does. Not surprisingly, the sketch $A\left(\Pi_1 A\right)^\dagger \Pi_2$ can be used in other ways: for example, by considering the dot product between two different rows of this randomized sketching matrix (and some additional manipulations) Algorithm 2 approximates the large cross-leverage scores of A.

Input: $A \in \mathbb{R}^{n\times d}$ (with SVD $A = U\Sigma V^T$), error parameter $\varepsilon \in (0, 1/2]$.

Output: $\tilde{\ell}_i, i \in [n]$.

1. Let $\Pi_1 \in \mathbb{R}^{r_1\times n}$ be an ε-FJLT for U, using Lemma 6 with
$$r_1 = \Omega\left(\frac{d\ln n}{\varepsilon^2}\ln\left(\frac{d\ln n}{\varepsilon^2}\right)\right).$$
2. Compute $\Pi_1 A \in \mathbb{R}^{r_1\times d}$ and its SVD, $\Pi_1 A = U_{\Pi_1 A}\Sigma_{\Pi_1 A}V_{\Pi_1 A}^T$. Let $R^{-1} = V_{\Pi_1 A}\Sigma_{\Pi_1 A}^{-1} \in \mathbb{R}^{d\times d}$.
(Alternatively, R could be computed by a QR factorization of $\Pi_1 A$.)
3. View the normalized rows of $AR^{-1} \in \mathbb{R}^{n\times d}$ as n vectors in $\mathbb{R}^d$, and construct $\Pi_2 \in \mathbb{R}^{d\times r_2}$ to be an ε-JLT for n^2 vectors (the aforementioned n vectors and their $n^2 - n$ pairwise sums), using Lemma 4 with
$$r_2 = O\left(\varepsilon^{-2}\ln n\right).$$
4. Construct the matrix product $\Omega = AR^{-1}\Pi_2$.
5. **For all** $i \in [n]$ **compute and return** $\tilde{\ell}_i = \left\|\Omega_{(i)}\right\|_2^2$.

Algorithm 1: Approximating the (diagonal) statistical leverage scores ℓ_i.

3.2 Approximating All the Statistical Leverage Scores

Our first main result is Algorithm 1, which takes as input an $n \times d$ matrix A and an error parameter $\varepsilon \in (0, 1/2]$, and returns as output numbers $\tilde{\ell}_i$, $i \in [n]$. Although the basic idea to approximate $\left\|(AA^\dagger)_{(i)}\right\|^2$ was described in the previous section, we can improve the efficiency of our approach by avoiding the full sketch of the pseudoinverse. In particular, let $\hat{A} = \Pi_1 A$ and let its SVD be $\hat{A} = U_{\hat{A}}\Sigma_{\hat{A}}V_{\hat{A}}^T$. Let $R^{-1} = V_{\hat{A}}\Sigma_{\hat{A}}^{-1}$ and note that $R^{-1} \in \mathbb{R}^{d\times d}$ is an orthogonalizer for $\hat{A}$ since $U_{\hat{A}} = \hat{A}R^{-1}$ is an orthogonal matrix.[7] In addition, note that AR^{-1} is approximately orthogonal. Thus, we can compute AR^{-1} and use it as an approximate orthogonal basis for A and then compute $\hat{\ell}_i$ as the squared row-norms of AR^{-1}. The next lemma states that this is exactly what our main algorithm does; even more, we could get the same estimates by using any "orthogonalizer" of $\Pi_1 A$.

Lemma 7 *Let R^{-1} be such that $Q = \Pi_1 AR^{-1}$ is an orthogonal matrix with* $\mathbf{rank}(Q) = \mathbf{rank}(\Pi_1 A)$. *Then,* $\left\|(AR^{-1})_{(i)}\right\|_2^2 = \hat{\ell}_i$.

7. This preprocessing is reminiscent of how Rokhlin and Tygert (2008) and Avron et al. (2010) preprocessed the input to provide numerical implementations of the fast relative-error algorithm (Drineas et al., 2010b) for approximate LS approximation. From this perspective, Algorithm 1 can be viewed as specifying a particular basis Q, that is, as choosing Q to be the left singular vectors of $\Pi_1 A$.

Proof Since $\hat{A} = \Pi_1 A$ has rank d (by Lemma 5) and R^{-1} preserves this rank, R^{-1} is a $d \times d$ invertible matrix. Using $\hat{A} = QR$ and properties of the pseudoinverse, we get $\left(\hat{A}\right)^{\dagger} = R^{-1}Q^T$. Thus,

$$\hat{\ell}_i = \left\| \left(A\left(\Pi_1 A\right)^{\dagger}\right)_{(i)} \right\|_2^2 = \left\| \left(AR^{-1}Q^T\right)_{(i)} \right\|_2^2 = \left\| \left(AR^{-1}\right)_{(i)} Q^T \right\|_2^2 = \left\| \left(AR^{-1}\right)_{(i)} \right\|_2^2.$$

■

This lemma says that the $\hat{\ell}_i$ of Equation (7) can be computed with any QR decomposition, rather than with the SVD; but note that one would still have to post-multiply by Π_2, as in Algorithm 1, in order to compute "quickly" the approximations of the leverage scores.

3.3 Approximating the Large Cross-leverage Scores

By combining Lemmas 9 and 10 (in Section 4.2 below) with the triangle inequality, one immediately obtains the following lemma.

Lemma 8 *Let Ω be either the sketching matrix constructed by Algorithm 1, that is, $\Omega = AR^{-1}\Pi_2$, or $\Omega = A\left(\Pi_1 A\right)^{\dagger}\Pi_2$ as described in Section 3.1. Then, the pairwise dot-products of the rows of Ω are additive-error approximations to the leverage scores and cross-leverage scores:*

$$\left|\langle U_{(i)}, U_{(j)}\rangle - \langle \Omega_{(i)}, \Omega_{(j)}\rangle\right| \leq \frac{3\varepsilon}{1-\varepsilon}\left\|U_{(i)}\right\|_2\left\|U_{(j)}\right\|_2.$$

That is, if one were interested in obtaining an approximation to all the cross-leverage scores to within additive error (and thus the diagonal statistical leverage scores to relative-error), then the algorithm which first computes Ω followed by all the pairwise inner products achieves this in time $T(\Omega) + O\left(n^2 r_2\right)$, where $T(\Omega)$ is the time to compute Ω from Section 3.2 and $r_2 = O(\varepsilon^{-2}\ln n)$.[8] The challenge is to avoid the n^2 computational complexity and this can be done if one is interested only in the large cross-leverage scores.

Our second main result is provided by Algorithms 2 and 3. Algorithm 2 takes as input an $n \times d$ matrix A, a parameter $\kappa > 1$, and an error parameter $\varepsilon \in (0, 1/2]$, and returns as output a subset of $[n] \times [n]$ and estimates $\tilde{c}_{ij}$ satisfying Theorem 3. The first step of the algorithm is to compute the matrix $\Omega = AR^{-1}\Pi_2$ constructed by Algorithm 1. Then, Algorithm 2 uses Algorithm 3 as a subroutine to compute "heavy hitter" pairs of rows from a matrix.

4. Proofs of Our Main Theorems

We will start with a sketch of the proofs of Theorems 2 and 3, and then we will provide the details.

4.1 Sketch of the Proof of Theorems 2 and 3

We will start by providing a sketch of the proof of Theorems 2 and 3. A detailed proof is provided in the next two subsections. In our analysis, we will condition on the events that $\Pi_1 \in \mathbb{R}^{r_1 \times n}$ is an ε-FJLT for U and $\Pi_2 \in \mathbb{R}^{r_1 \times r_2}$ is an ε-JLT for n^2 points in $\mathbb{R}^{r_1}$. Note that by setting $\delta = 0.1$ in Lemma 4, both events hold with probability at least 0.8, which is equal to the success probability of Theorems 2 and 3. The algorithm estimates $\tilde{\ell}_i = \|\tilde{u}_i\|_2^2$, where $\tilde{u}_i = e_i^T A(\Pi_1 A)^{\dagger}\Pi_2$. First, observe

8. The exact algorithm which computes a basis first and then the pairwise inner products requires $O(nd^2 + n^2 d)$ time. Thus, by using the sketch, we can already improve on this running time by a factor of $d/\ln n$.

Input: $A \in \mathbb{R}^{n\times d}$ and parameters $\kappa > 1$, $\varepsilon \in (0, 1/2]$.

Output: The set $\mathcal{H}$ consisting of pairs (i, j) together with estimates $\tilde{c}_{ij}$ satisfying Theorem 3.

1. Compute the $n \times r_2$ matrix $\Omega = AR^{-1}\Pi_2$ from Algorithm 1.
2. Use Algorithm 3 with inputs Ω and $\kappa' = \kappa(1 + 30d\varepsilon)$ to obtain the set $\mathcal{H}$ containing all the κ'-heavy pairs of Ω.
3. **Return** the pairs in $\mathcal{H}$ as the κ-heavy pairs of A.

Algorithm 2: Approximating the large (off-diagonal) cross-leverage scores c_{ij}.

Input: $X \in \mathbb{R}^{n\times r}$ with rows $x_1, \ldots, x_n$ and a parameter $\kappa > 1$.

Output: $\mathcal{H} = \{(i, j), \tilde{c}_{ij}\}$ containing all heavy (unordered) pairs. The pair $(i, j), \tilde{c}_{ij} \in \mathcal{H}$ if and only if $\tilde{c}_{ij}^2 = \langle x_i, x_j\rangle^2 \geq \left\|X^T X\right\|_F^2/\kappa$.

Compute the norms $\|x_i\|_2$ and sort the rows according to norm, so that $\|x_1\|_2 \leq \cdots \leq \|x_n\|_2$.
$\mathcal{H} \leftarrow \{\}$; $z_1 \leftarrow n$; $z_2 \leftarrow 1$.
while $z_2 \leq z_1$ **do**
  while $\|x_{z_1}\|_2^2 \|x_{z_2}\|_2^2 < \left\|X^T X\right\|_F^2/\kappa$ **do**
    $z_2 \leftarrow z_2 + 1$.
    if $z_2 > z_1$ **then**
      return $\mathcal{H}$.
    end if
  end while
  for each pair (i, j) where $i = z_1$ and $j \in \{z_2, z_2 + 1, \ldots, z_1\}$ **do**
    $\tilde{c}_{ij}^2 = \langle x_i, x_j\rangle^2$.
    if $\tilde{c}_{ij}^2 \geq \left\|X^T X\right\|_F^2/\kappa$ **then**
      add (i, j) and $\tilde{c}_{ij}$ to $\mathcal{H}$.
    end if
    $z_1 \leftarrow z_1 - 1$.
  end for
end while
return $\mathcal{H}$.

Algorithm 3: Computing heavy pairs of a matrix.

that the sole purpose of Π_2 is to improve the running time while preserving pairwise inner products;

this is achieved because Π_2 is an ε-JLT for n^2 points. So, the results will follow if

$$e_i^T A(\Pi_1 A)^\dagger((\Pi_1 A)^\dagger)^T A^T e_j \approx e_i^T U U^T e_j$$

and $(\Pi_1 A)^\dagger$ can be computed efficiently. Since Π_1 is an ε-FJLT for U, where $A = U\Sigma V^T$, $(\Pi_1 A)^\dagger$ can be computed in $O(nd\ln r_1 + r_1 d^2)$ time. By Lemma 5, $(\Pi_1 A)^\dagger = V\Sigma^{-1}(\Pi_1 U)^\dagger$, and so

$$e_i^T A(\Pi_1 A)^\dagger((\Pi_1 A)^\dagger)^T A^T e_j = e_i^T U(\Pi_1 U)^\dagger(\Pi_1 U)^{\dagger T} U^T e_j.$$

Since Π_1 is an ε-FJLT for U, it follows that $(\Pi_1 U)^\dagger(\Pi_1 U)^{\dagger T} \approx I_d$, that is, that $\Pi_1 U$ is approximately orthogonal. Theorem 2 follows from this basic idea. However, in order to prove Theorem 3, having a sketch which preserves inner products alone is not sufficient. We also need a fast algorithm to identify the large inner products and to relate these to the actual cross-leverage scores. Indeed, it is possible to efficiently find pairs of rows in a general matrix with large inner products. Combining this with the fact that the inner products are preserved, we obtain Theorem 3.

4.2 Proof of Theorem 2

We condition all our analysis on the events that $\Pi_1 \in \mathbb{R}^{r_1 \times n}$ is an ε-FJLT for U and $\Pi_2 \in \mathbb{R}^{r_1 \times r_2}$ is an ε-JLT for n^2 points in $\mathbb{R}^{r_1}$. Define

$$\begin{aligned} \hat{u}_i &= e_i^T A(\Pi_1 A)^\dagger, \text{ and} \\ \tilde{u}_i &= e_i^T A(\Pi_1 A)^\dagger \Pi_2. \end{aligned}$$

Then, $\hat{\ell}_i = \|\hat{u}_i\|_2^2$ and $\tilde{\ell}_i = \|\tilde{u}_i\|_2^2$. The proof will follow from the following two lemmas.

Lemma 9 *For $i, j \in [n]$,*

$$\left|\langle U_{(i)}, U_{(j)}\rangle - \langle \hat{u}_i, \hat{u}_j\rangle\right| \leq \frac{\varepsilon}{1-\varepsilon}\left\|U_{(i)}\right\|_2\left\|U_{(j)}\right\|_2.$$

Lemma 10 *For $i, j \in [n]$,*

$$\left|\langle \hat{u}_i, \hat{u}_j\rangle - \langle \tilde{u}_i, \tilde{u}_j\rangle\right| \leq 2\varepsilon\|\hat{u}_i\|_2\left\|\hat{u}_j\right\|_2.$$

Lemma 9 states that $\langle \hat{u}_i, \hat{u}_j\rangle$ is an additive error approximation to all the cross-leverage scores ($i \neq j$) and a relative error approximation for the diagonals ($i = j$). Similarly, Lemma 10 shows that these cross-leverage scores are preserved by Π_2. Indeed, with $i = j$, from Lemma 9 we have $|\hat{\ell}_i - \ell_i| \leq \frac{\varepsilon}{1-\varepsilon}\ell_i$, and from Lemma 10 we have $|\hat{\ell}_i - \tilde{\ell}_i| \leq 2\varepsilon\hat{\ell}_i$. Using the triangle inequality and $\varepsilon \leq 1/2$:

$$\left|\ell_i - \tilde{\ell}_i\right| = \left|\ell_i - \hat{\ell}_i + \hat{\ell}_i - \tilde{\ell}_i\right| \leq \left|\ell_i - \hat{\ell}_i\right| + \left|\hat{\ell}_i - \tilde{\ell}_i\right| \leq \left(\frac{\varepsilon}{1-\varepsilon} + 2\varepsilon\right)\ell_i \leq 4\varepsilon\ell_i.$$

The theorem follows after rescaling ε.

Proof of Lemma 9 Let $A = U\Sigma V^T$. Using this SVD of A and Equation (4) in Lemma 5,

$$\langle \hat{u}_i, \hat{u}_j \rangle = e_i^T U\Sigma V^T V\Sigma^{-1} (\Pi_1 U)^\dagger (\Pi_1 U)^{\dagger T} \Sigma^{-1} V^T V\Sigma U^T e_j = e_i^T U (\Pi_1 U)^\dagger (\Pi_1 U)^{\dagger T} U^T e_j.$$

By performing standard manipulations, we can now bound $\left|\langle U_{(i)}, U_{(j)} \rangle - \langle \hat{u}_i, \hat{u}_j \rangle\right|$:

$$\begin{aligned}
\left|\langle U_{(i)}, U_{(j)} \rangle - \langle \hat{u}_i, \hat{u}_j \rangle\right| &= e_i^T UU^T e_j - e_i^T U (\Pi_1 U)^\dagger (\Pi_1 U)^{\dagger T} U^T e_j \\
&= e_i^T U \left(I_d - (\Pi_1 U)^\dagger (\Pi_1 U)^{\dagger T} \right) U^T e_j \\
&\leq \left\| I_d - (\Pi_1 U)^\dagger (\Pi_1 U)^{\dagger T} \right\|_2 \left\| U_{(i)} \right\|_2 \left\| U_{(j)} \right\|_2.
\end{aligned}$$

Let the SVD of $\Psi = \Pi_1 U$ be $\Psi = U_\Psi \Sigma_\Psi V_\Psi^T$, where V_Ψ is a full rotation in d dimensions (because $\mathbf{rank}(A) = \mathbf{rank}(\Pi_1 U)$). Then, $\Psi^\dagger \Psi^{\dagger T} = V_\Psi \Sigma_\Psi^{-2} V_\Psi^T$. Thus,

$$\begin{aligned}
\left|\langle U_{(i)}, U_{(j)} \rangle - \langle \hat{u}_i, \hat{u}_j \rangle\right| &\leq \left\| I_d - V_\Psi \Sigma_\Psi^{-2} V_\Psi^T \right\|_2 \left\| U_{(i)} \right\|_2 \left\| U_{(j)} \right\|_2 \\
&= \left\| V_\Psi V_\Psi^T - V_\Psi \Sigma_\Psi^{-2} V_\Psi^T \right\|_2 \left\| U_{(i)} \right\|_2 \left\| U_{(j)} \right\|_2 \\
&= \left\| I_d - \Sigma_\Psi^{-2} \right\|_2 \left\| U_{(i)} \right\|_2 \left\| U_{(j)} \right\|_2,
\end{aligned}$$

where we used the fact that $V_\Psi V_\Psi^T = V_\Psi^T V_\Psi = I_d$ and the unitary invariance of the spectral norm. Finally, using Equation (3) of Lemma 5 the result follows.

Proof of Lemma 10. Since Π_2 is an ε-JLT for n^2 vectors, it preserves the norms of an arbitrary (but fixed) collection of n^2 vectors. Let $x_i = \hat{u}_i / \|\hat{u}_i\|_2$. Consider the following n^2 vectors:

$$\begin{aligned}
x_i &\quad \text{for } i \in [n], \text{ and} \\
x_i + x_j &\quad \text{for } i, j \in [n], i \neq j.
\end{aligned}$$

By the ε-JLT property of Π_2 and the fact that $\|x_i\|_2 = 1$,

$$1 - \varepsilon \leq \|x_i \Pi_2\|_2^2 \leq 1 + \varepsilon \qquad \text{for } i \in [n], \text{ and} \tag{8}$$

$$(1-\varepsilon)\left\|x_i + x_j\right\|_2^2 \leq \left\|x_i\Pi_2 + x_j\Pi_2\right\|_2^2 \leq (1+\varepsilon)\left\|x_i + x_j\right\|_2^2 \quad \text{for } i, j \in [n], i \neq j. \tag{9}$$

Combining Equations (8) and (9) after expanding the squares using the identity $\|a+b\|^2 = \|a\|^2 + \|b\|^2 + 2\langle a, b \rangle$, substituting $\|x_i\| = 1$, and after some algebra, we obtain

$$\langle x_i, x_j \rangle - 2\varepsilon \leq \langle x_i\Pi_2, x_j\Pi_2 \rangle \leq \langle x_i, x_j \rangle + 2\varepsilon.$$

To conclude the proof, multiply throughout by $\|\hat{u}_i\|\left\|\hat{u}_j\right\|$ and use the homogeneity of the inner product, together with the linearity of Π_2, to obtain:

$$\langle \hat{u}_i, \hat{u}_j \rangle - 2\varepsilon\|\hat{u}_i\|\left\|\hat{u}_j\right\| \leq \langle \hat{u}_i\Pi_2, \hat{u}_j\Pi_2 \rangle \leq \langle \hat{u}_i, \hat{u}_j \rangle + 2\varepsilon\|\hat{u}_i\|\left\|\hat{u}_j\right\|.$$

Running Times. By Lemma 7, we can use $V_{\Pi_1 A}\Sigma_{\Pi_1 A}^{-1}$ instead of $(\Pi_1 A)^\dagger$ and obtain the same estimates. Since Π_1 is an ε-FJLT, the product $\Pi_1 A$ can be computed in $O(nd \ln r_1)$ while its SVD takes an additional $O(r_1 d^2)$ time to return $V_{\Pi_1 A}\Sigma_{\Pi_1 A}^{-1} \in \mathbb{R}^{d \times d}$. Since $\Pi_2 \in \mathbb{R}^{d \times r_2}$, we obtain $V_{\Pi_1 A}\Sigma_{\Pi_1 A}^{-1}\Pi_2 \in \mathbb{R}^{d \times r_2}$ in an additional $O(r_2 d^2)$ time. Finally, premultiplying by A takes $O(ndr_2)$

time, and computing and returning the squared row-norms of $\Omega = AV_{\Pi_1 A}\Sigma^{-1}_{\Pi_1 A}\Pi_2 \in \mathbb{R}^{n\times r_2}$ takes $O(nr_2)$ time. So, the total running time is the sum of all these operations, which is

$$O(nd\ln r_1 + ndr_2 + r_1 d^2 + r_2 d^2).$$

Recall that for our implementations of the ε-JLTs and ε-FJLTs, we have $\delta = 0.1$, and we have $r_1 = O\left(\varepsilon^{-2} d\,(\ln n)\left(\ln\left(\varepsilon^{-2} d\ln n\right)\right)\right)$ and $r_2 = O(\varepsilon^{-2}\ln n)$. It follows that the asymptotic running time is

$$O\left(nd\ln\left(d\varepsilon^{-1}\right) + nd\varepsilon^{-2}\ln n + d^3\varepsilon^{-2}\,(\ln n)\left(\ln\left(d\varepsilon^{-1}\right)\right)\right).$$

To simplify, suppose that $d \le n \le e^d$ and treat ε as a constant. Then, the asymptotic running time is

$$O\left(nd\ln n + d^3\,(\ln n)\,(\ln d)\right).$$

4.3 Proof of Theorem 3

We first construct an algorithm to estimate the large inner products among the rows of an arbitrary matrix $X \in \mathbb{R}^{n\times r}$ with $n > r$. This general algorithm will be applied to the matrix $\Omega = AV_{\Pi_1 A}\Sigma^{-1}_{\Pi_1 A}\Pi_2$. Let $x_1,\ldots,x_n$ denote the rows of X; for a given $\kappa > 1$, the pair (i,j) is *heavy* if

$$\langle x_i, x_j\rangle^2 \ge \frac{1}{\kappa}\left\|X^T X\right\|_F^2.$$

By the Cauchy-Schwarz inequality, this implies that

$$\|x_i\|_2^2\|x_j\|_2^2 \ge \frac{1}{\kappa}\left\|X^T X\right\|_F^2, \tag{10}$$

so it suffices to find all the pairs (i,j) for which Equation (10) holds. We will call such pairs *norm-heavy*. Let s be the number of norm-heavy pairs satisfying Equation (10). We first bound the number of such pairs.

Lemma 11 *Using the above notation, $s \le \kappa r$.*

Proof Observe that

$$\sum_{i,j=1}^{n}\|x_i\|_2^2\|x_j\|_2^2 = \left(\sum_{i=1}^{n}\|x_i\|_2^2\right)^2 = \|X\|_F^4 = \left(\sum_{i=1}^{r}\sigma_i^2\right)^2,$$

where $\sigma_1,\ldots,\sigma_r$ are the singular values of X. To conclude, by the definition of a heavy pair,

$$\sum_{i,j}\|x_i\|_2^2\|x_j\|_2^2 \ge \frac{s}{\kappa}\left\|X^T X\right\|_F^2 = \frac{s}{\kappa}\sum_{i=1}^{r}\sigma_i^4 \ge \frac{s}{\kappa r}\left(\sum_{i=1}^{r}\sigma_i^2\right)^2,$$

where the last inequality follows by Cauchy-Schwarz. ■

Algorithm 3 starts by computing the norms $\|x_i\|_2^2$ for all $i \in [n]$ and sorting them (in $O(nr + n\ln n)$ time) so that we can assume that $\|x_1\|_2 \le \cdots \le \|x_n\|_2$. Then, we initialize the set of norm-heavy pairs to $\mathcal{H} = \{\}$ and we also initialize two pointers $z_1 = n$ and $z_2 = 1$. The basic loop in the algorithm checks if $z_2 > z_1$ and stops if that is the case. Otherwise, we increment z_2 to the first pair (z_1, z_2) that

is norm-heavy. If none of pairs are norm heavy (i.e., $z_2 > z_1$ occurs), then we stop and output $\mathcal{H}$; otherwise, we add $(z_1, z_2), (z_1, z_2+1), \ldots, (z_1, z_1)$ to $\mathcal{H}$. This basic loop computes all pairs (z_1, i) with $i \leq z_1$ that are norm-heavy. Next, we decrease z_1 by one and if $z_1 < z_2$ we stop and output $\mathcal{H}$; otherwise, we repeat the basic loop. Note that in the basic loop z_2 is always *incremented*. This occurs whenever the pair (z_1, z_2) is not norm-heavy. Since z_2 can be incremented at most n times, the number of times we check whether a pair is norm-heavy and fail is at most n. Every successful check results in the addition of at least one norm-heavy pair into $\mathcal{H}$ and thus the number of times we check if a pair is norm heavy (a constant-time operation) is at most $n+s$. The number of pair additions into $\mathcal{H}$ is exactly s and thus the total running time is $O(nr + n\ln n + s)$. Finally, we must check each norm-heavy pair to verify whether or not it is actually heavy by computing s inner products vectors in $\mathbb{R}^r$; this can be done in $O(sr)$ time. Using $s \leq \kappa r$ we get the following lemma.

Lemma 12 *Algorithm 3 returns $\mathcal{H}$ including all the heavy pairs of X in $O(nr + \kappa r^2 + n\ln n)$ time.*

To complete the proof, we apply Algorithm 3 with $\Omega = AV_{\Pi_1 A}\Sigma^{-1}_{\Pi_1 A}\Pi_2 \in \mathbb{R}^{n\times r_2}$, where $r_2 = O(\varepsilon^{-2}\ln n)$. Let $\tilde{u}_1, \ldots, \tilde{u}_n$ denote the rows of Ω and recall that $A = U\Sigma V^T$. Let $u_1, \ldots, u_n$ denote the rows of U; then, from Lemma 8,

$$\langle u_i, u_j\rangle - \frac{3\varepsilon}{1-\varepsilon}\|u_i\|\|u_j\| \leq \langle \tilde{u}_i, \tilde{u}_j\rangle \leq \langle u_i, u_j\rangle + \frac{3\varepsilon}{1-\varepsilon}\|u_i\|\|u_j\|. \tag{11}$$

Given ε, κ, assume that for the pair of vectors u_i and u_j

$$\langle u_i, u_j\rangle^2 \geq \frac{1}{\kappa}\|U^TU\|_F^2 + 12\varepsilon\|u_i\|^2\|u_j\|^2 = \frac{d}{\kappa} + 12\varepsilon\|u_i\|^2\|u_j\|^2,$$

where the last equality follows from $\|U^TU\|_F^2 = \|I_d\|_F^2 = d$. By Equation (11), after squaring and using $\varepsilon < 0.5$,

$$\langle u_i, u_j\rangle^2 - 12\varepsilon\|u_i\|^2\varepsilon\|u_j\|^2 \leq \langle \tilde{u}_i, \tilde{u}_j\rangle^2 \leq \langle u_i, u_j\rangle^2 + 30\varepsilon\|u_i\|^2\|u_j\|^2. \tag{12}$$

Thus, $\langle \tilde{u}_i, \tilde{u}_j\rangle^2 \geq d/\kappa$ and summing Equation (12) over all i, j we get $\|\Omega^T\Omega\|_F^2 \leq d + 30\varepsilon d^2$, or, equivalently,

$$d \geq \frac{\|\Omega^T\Omega\|_F^2}{1+30d\varepsilon}.$$

We conclude that

$$\langle u_i, u_j\rangle^2 \geq \frac{d}{\kappa} + 12\varepsilon\|u_i\|^2\|u_j\|^2 \implies \langle \tilde{u}_i, \tilde{u}_j\rangle^2 \geq \frac{d}{\kappa} \geq \frac{\|\Omega^T\Omega\|_F^2}{\kappa(1+30d\varepsilon)}. \tag{13}$$

By construction, Algorithm 3 is invoked with $\kappa' = \kappa\|\Omega^T\Omega\|_F^2/d$ and thus it finds all pairs with $\langle \tilde{u}_i, \tilde{u}_j\rangle^2 \geq \|\Omega^T\Omega\|_F^2/\kappa' = d/\kappa$. This set contains all pairs for which

$$\langle u_i, u_j\rangle^2 \geq \frac{d}{\kappa} + 12\varepsilon\|u_i\|^2\|u_j\|^2.$$

Further, since every pair returned satisfies $\langle \tilde{u}_i, \tilde{u}_j\rangle^2 \geq d/\kappa$, by Equation (12), $c_{ij} \geq d/\kappa - 30\varepsilon\ell_i\ell_j$. This proves the first claim of the Theorem; the second claim follows analogously from Equation (12).

Using Lemma 12, the running time of our approach is $O\left(nr_2 + \kappa' r_2^2 + n\ln n\right)$. Since $r_2 = O\left(\varepsilon^{-2}\ln n\right)$, and, by Equation (13), $\kappa' = \kappa\|\Omega^T\Omega\|_F^2/d \leq \kappa(1+30d\varepsilon)$, the overall running time is $O\left(\varepsilon^{-2}n\ln n + \varepsilon^{-3}\kappa d\ln^2 n\right)$.

5. Extending Our Algorithm to General Matrices

In this section, we will describe an important extension of our main result, namely the computation of the statistical leverage scores relative to the best rank-k approximation to a general matrix A. More specifically, we consider the estimation of leverage scores for the case of general "fat" matrices, namely input matrices $A \in \mathbb{R}^{n\times d}$, where both n and d are large, for example, when $d = n$ or $d = \Theta(n)$. Clearly, the leverage scores of any full rank $n \times n$ matrix are exactly uniform. The problem becomes interesting if one specifies a rank parameter $k \ll \min\{n,d\}$. This may arise when the numerical rank of A is small (e.g., in some scientific computing applications, more than 99% of the spectral norm of A may be captured by some $k \ll \min\{n,d\}$ directions), or, more generally, when one is interested in some low rank approximation to A (e.g., in some data analysis applications, a reasonable fraction or even the majority of the Frobenius norm of A may be captured by some $k \ll \min\{n,d\}$ directions, where k is determined by some exogenously-specified model selection criterion). Thus, assume that in addition to a general $n \times d$ matrix A, a rank parameter $k < \min\{n,d\}$ is specified. In this case, we wish to obtain the statistical leverage scores $\ell_i = \left\|(U_k)_{(i)}\right\|_2^2$ for $A_k = U_k \Sigma_k V_k^T$, the best rank-k approximation to A. Equivalently, we seek the normalized leverage scores

$$p_i = \frac{\ell_i}{k}. \tag{14}$$

Note that $\sum_{i=1}^n p_i = 1$ since $\sum_{i=1}^n \ell_i = \|U_k\|_F^2 = k$.

Unfortunately, as stated, this is an ill-posed problem. Indeed, consider the degenerate case when $A = I_n$ (i.e., the $n \times n$ identity matrix). In this case, U_k is not unique and the leverage scores are not well-defined. Moreover, for the obvious $\binom{n}{k}$ equivalent choices for U_k, the leverage scores defined according to any one of these choices do not provide a relative error approximation to the leverage scores defined according to any other choices. More generally, removing this trivial degeneracy does not help. Consider the matrix

$$A = \begin{pmatrix} I_k & 0 \\ 0 & (1-\gamma)I_{n-k} \end{pmatrix} \in \mathbb{R}^{n\times n}.$$

In this example, the leverage scores for A_k are well defined. However, as $\gamma \to 0$, it is not possible to distinguish between the top-k singular space and its complement. This example suggests that it should be possible to obtain some result conditioning on the spectral gap at the k^{th} singular value. For example, one might assume that $\sigma_k^2 - \sigma_{k+1}^2 \geq \gamma > 0$, in which case the parameter γ would play an important role in the ability to solve this problem. Any algorithm which cannot distinguish the singular values with an error less than γ will confuse the k-th and $(k+1)$-th singular vectors and consequently will fail to get an accurate approximation to the leverage scores for A_k.

In the following, we take a more natural approach which leads to a clean problem formulation. To do so, recall that the leverage scores and the related normalized leverage scores of Equation (14) are used to approximate the matrix in some way, for example, we might be seeking a low-rank approximation to the matrix with respect to the spectral (Drineas et al., 2008) or the Frobenius (Boutsidis et al., 2009) norm, or we might be seeking useful features or data points in downstream data analysis applications (Paschou et al., 2007; Mahoney and Drineas, 2009), or we might be seeking to develop high-quality numerical implementations of low-rank matrix approximation algorithms (Halko et al., 2011), etc. In all these cases, we only care that the estimated leverage scores are a good approximation to the leverage scores of some "good" low-rank approximation to A. The

following definition captures the notion of a set of rank-k matrices that are good approximations to A.

Definition 13 *Given $A \in \mathbb{R}^{n\times d}$ and a rank parameter $k \ll \min\{n,d\}$, let A_k be the best rank-k approximation to A. Define the set $\mathcal{S}_\varepsilon$ of rank-k matrices that are good approximations to A as follows (for $\xi = 2, F$):*

$$\mathcal{S}_\varepsilon = \left\{ X \in \mathbb{R}^{n\times d} : \ \mathbf{rank}(X) = k \ \ and \ \ \|A - X\|_\xi \leq (1+\varepsilon)\|A - A_k\|_\xi \right\}.$$

We are now ready to define our approximations to the normalized leverage scores of any matrix $A \in \mathbb{R}^{n\times d}$ given a rank parameter $k \ll \min\{n,d\}$. Instead of seeking to approximate the p_i of Equation (14) (a problem that is ill-posed as discussed above), we will be satisfied if we can approximate the normalized leverage scores of some matrix $X \in \mathcal{S}_\varepsilon$. This is an interesting relaxation of the task at hand: all matrices X that are sufficiently close to A_k are essentially equivalent, since they can be used instead of A_k in applications.

Definition 14 *Given $A \in \mathbb{R}^{n\times d}$ and a rank parameter $k \ll \min\{n,d\}$, let $\mathcal{S}_\varepsilon$ be the set of matrices of Definition 13. We call the numbers $\hat{p}_i$ (for all $i \in [n]$) β-approximations to the normalized leverage scores of A_k (the best rank-k approximation to A) if, for some matrix $X \in \mathcal{S}_\varepsilon$,*

$$\hat{p}_i \geq \frac{\beta \left\| (U_X)_{(i)} \right\|_2^2}{k} \qquad and \qquad \sum_{i=1}^{n} \hat{p}_i = 1.$$

Here $U_X \in \mathbb{R}^{n\times k}$ is the matrix of the left singular vectors of X.

Thus, we will seek algorithms whose output is a set of numbers, with the requirement that those numbers are good approximations to the normalized leverage scores of some matrix $X \in \mathcal{S}_\varepsilon$ (instead of A_k). This removes the ill-posedness of the original problem. Next, we will give two examples of algorithms that compute such β-approximations to the normalized leverage scores of a general matrix A with a rank parameter k for two popular norms, the spectral norm and the Frobenius norm.[9]

5.1 Leverage Scores for Spectral Norm Approximators

Algorithm 4 approximates the statistical leverage scores of a general matrix A with rank parameter k in the spectral norm case. It takes as inputs a matrix $A \in \mathrm{R}^{n\times d}$ with $\mathbf{rank}(A) = \rho$ and a rank parameter $k \ll \rho$, and outputs a set of numbers $\hat{p}_i$ for all $i \in [n]$, namely our approximations to the normalized leverage scores of A with rank parameter k.

The next lemma argues that there exists a matrix $X \in \mathbb{R}^{n\times d}$ of rank k that is sufficiently close to A (in particular, it is a member of $\mathcal{S}_\varepsilon$ with constant probability) and, additionally, can be written as $X = BY$, where $Y \in \mathbb{R}^{2k\times d}$ is a matrix of rank k. A version of this lemma was essentially proven in Halko et al. (2011), but see also Rokhlin et al. (2009) for computational details; we will use the version of the lemma that appeared in Boutsidis et al. (2011b). (See also the conference version (Boutsidis et al., 2011a), but in the remainder we refer to the technical report version (Boutsidis et al., 2011b) for consistency of numbering.) Note that for our purposes in this section, the computation of Y is not relevant and we defer the reader to Halko et al. (2011) and Boutsidis et al. (2011b) for details.

9. Note that we will not compute S_ε, but our algorithms will compute a matrix in that set. Moreover, that matrix can be used for high-quality low-rank matrix approximation. See the comments in Section 1.4 for more details.

Input: $A \in \mathbb{R}^{n\times d}$ with $\mathbf{rank}(A) = \rho$ and a rank parameter $k \ll \rho$

Output: $\hat{p}_i, i \in [n]$

1. Construct $\Pi \in \mathbb{R}^{d\times 2k}$ with entries drawn in i.i.d. trials from the normal distribution $\mathcal{N}(0,1)$.
2. Compute $B = \left(AA^T\right)^q A\Pi \in \mathbb{R}^{n\times 2k}$, with q as in Equation (16).
3. Approximately compute the statistical leverage scores of the "tall" matrix B by calling Algorithm 1 with inputs B and ε; let $\hat{\ell}_i$ (for all $i \in [n]$) be the outputs of Algorithm 1.
4. Return
$$\hat{p}_i = \frac{\hat{\ell}_i}{\sum_{j=1}^n \hat{\ell}_j} \tag{15}$$
for all $i \in [n]$.

Algorithm 4: Approximating the statistical leverage scores of a general matrix A (spectral norm case).

Lemma 15 (Spectral Sketch) *Given $A \in \mathbb{R}^{n\times d}$ of rank ρ, a rank parameter k such that $2 \le k < \rho$, and an error parameter ε such that $0 < \varepsilon < 1$, let $\Pi \in \mathbb{R}^{d\times 2k}$ be a standard Gaussian matrix (with entries selected in i.i.d. trials from $\mathcal{N}(0,1)$). If $B = \left(AA^T\right)^q A\Pi$, where*

$$q \ge \left\lceil \frac{\ln\left(1 + \sqrt{\frac{k}{k-1}} + e\sqrt{\frac{2}{k}}\sqrt{\min\{n,d\} - k}\right)}{2\ln(1+\varepsilon/10) - 1/2} \right\rceil, \tag{16}$$

then there exists a matrix $X \in \mathbb{R}^{n\times d}$ of rank k satisfying $X = BY$ (with $Y \in \mathbb{R}^{2k\times d}$) such that

$$\mathbf{E}\left[\|A - X\|_2\right] \le \left(1 + \frac{\varepsilon}{10}\right)\|A - A_k\|_2.$$

The matrix B can be computed in $O(ndkq)$ time.

This version of the above lemma is proven in Boutsidis et al. (2011b).[10] Now, since X has rank k, it follows that $\|A - X\|_2 \ge \|A - A_k\|_2$ and thus we can consider the non-negative random variable $\|A - X\|_2 - \|A - A_k\|_2$ and apply Markov's inequality to get that

$$\|A - X\|_2 - \|A - A_k\|_2 \le \varepsilon\|A - A_k\|_2$$

holds with probability at least 0.9. Thus, $X \in \mathcal{S}_\varepsilon$ with probability at least 0.9.

10. More specifically, the proof may be found in Lemma 32 and in particular in Equation (14) in Section A.2; note that for our purposes here we replaced $\varepsilon/\sqrt{2}$ by $\varepsilon/10$ after adjusting q accordingly.

The next step of the proposed algorithm is to approximately compute the leverage scores of $B \in \mathbb{R}^{n\times 2k}$ via Algorithm 1. Under the assumptions of Theorem 2, this step runs in $O\left(nk\varepsilon^{-2}\ln n\right)$ time. Let $U_X \in \mathbb{R}^{n\times k}$ be the matrix containing the left singular vectors of the matrix X of Lemma 15. Then, since $X = BY$ by Lemma 15, it follows that

$$U_B = [U_X \;\; U_R]$$

is a basis for the subspace spanned by the columns of B. Here $U_R \in \mathbb{R}^{n\times k}$ is an orthogonal matrix whose columns are perpendicular to the columns of U_X. Now consider the approximate leverage scores $\hat{\ell}_i$ computed by Algorithm 1 and note that (by Theorem 2),

$$\left|\hat{\ell}_i - \left\|(U_B)_{(i)}\right\|_2^2\right| \leq \varepsilon\left\|(U_B)_{(i)}\right\|_2^2$$

holds with probability at least 0.8 for all $i \in [n]$. It follows that

$$\sum_{j=1}^{n}\hat{\ell}_j \leq (1+\varepsilon)\sum_{j=1}^{n}\left\|(U_B)_{(j)}\right\|_2^2 = (1+\varepsilon)\sum_{j=1}^{n}\|U_B\|_F^2 = 2(1+\varepsilon)k.$$

Finally,

$$\begin{aligned}
\hat{p}_i = \frac{\hat{\ell}_i}{\sum_{j=1}^{n}\hat{\ell}_j} &\geq (1-\varepsilon)\frac{\left\|(U_B)_{(i)}\right\|_2^2}{\sum_{j=1}^{n}\hat{\ell}_j} \\
&\geq (1-\varepsilon)\frac{\left\|(U_X)_{(i)}\right\|_2^2 + \left\|(U_R)_{(i)}\right\|_2^2}{\sum_{j=1}^{n}\hat{\ell}_j} \\
&\geq \frac{1-\varepsilon}{2}\frac{\left\|(U_X)_{(i)}\right\|_2^2}{\sum_{j=1}^{n}\hat{\ell}_j} \\
&\geq \frac{1-\varepsilon}{2(1+\varepsilon)}\frac{\left\|(U_X)_{(i)}\right\|_2^2}{k}.
\end{aligned}$$

Clearly, $\left\|(U_X)_{(i)}\right\|_2^2/k$ are the normalized leverage scores of the matrix X. Recall that $X \in \mathcal{S}_\varepsilon$ with probability at least 0.9 and use Definition 14 to conclude that the scores $\hat{p}_i$ of Equation (15) are $\left(\frac{1-\varepsilon}{2(1+\varepsilon)}\right)$-approximations to the normalized leverage scores of A with rank parameter k. The following Theorem summarizes the above discussion:

Theorem 16 *Given $A \in \mathbb{R}^{n\times d}$, a rank parameter k, and an accuracy parameter ε, Algorithm 4 computes a set of normalized leverage scores $\hat{p}_i$ that are $\left(\frac{1-\varepsilon}{2(1+\varepsilon)}\right)$-approximations to the normalized leverage scores of A with rank parameter k with probability at least 0.7. The proposed algorithm runs in*

$$O\left(ndk\frac{\ln(\min\{n,d\})}{\ln(1+\varepsilon)} + nk\varepsilon^{-2}\ln n\right)$$

time.

Input: $A \in \mathbb{R}^{n\times d}$ with $\mathbf{rank}(A) = \rho$ and a rank parameter $k \ll \rho$

Output: $\hat{p}_i, i \in [n]$

1. Let r be as in Equation (18) and construct $\Pi \in \mathbb{R}^{d\times r}$ whose entries are drawn in i.i.d. trials from the normal distribution $\mathcal{N}(0,1)$.
2. Compute $B = A\Pi \in \mathbb{R}^{n\times r}$.
3. Compute a matrix $Q \in \mathbb{R}^{n\times r}$ whose columns form an orthonormal basis for the column space of B.
4. Compute the matrix $Q^T A \in \mathbb{R}^{r\times d}$ and its left singular vectors $U_{Q^T A} \in \mathbb{R}^{r\times d}$.
5. Let $U_{Q^T A,k} \in \mathbb{R}^{r\times k}$ denote the top k left singular vectors of the matrix $Q^T A$ (the first k columns of $U_{Q^T A}$) and compute, for all $i \in [n]$,
$$\hat{\ell}_i = \left\| \left(Q U_{Q^T A,k} \right)_{(i)} \right\|_2^2. \tag{17}$$
6. Return $\hat{p}_i = \hat{\ell}_i / k$ for all $i \in [n]$.

Algorithm 5: Approximating the statistical leverage scores of a general matrix A (Frobenius norm case).

5.2 Leverage Scores for Frobenius Norm Approximators

Algorithm 5 approximates the statistical leverage scores of a general matrix A with rank parameter k in the Frobenius norm case. It takes as inputs a matrix $A \in \mathrm{R}^{n\times d}$ with $\mathbf{rank}(A) = \rho$ and a rank parameter $k \ll \rho$, and outputs a set of numbers $\hat{p}_i$ for all $i \in [n]$, namely our approximations to the normalized leverage scores of A with rank parameter k. It is worth noting that $\sum_{i=1}^n \hat{\ell}_i = \left\| Q U_{Q^T A,k} \right\|_F^2 = \left\| U_{Q^T A,k} \right\|_F^2 = k$ and thus the $\hat{p}_i$ sum up to one. The next lemma argues that there exists a matrix $X \in \mathbb{R}^{n\times d}$ of rank k that is sufficiently close to A (in particular, it is a member of $\mathcal{S}_\varepsilon$ with constant probability). Unlike the previous section (the spectral norm case), we will now be able to provide a closed-form formula for this matrix X and, more importantly, the normalized leverage scores of X will be *exactly equal* to the $\hat{p}_i$ returned by our algorithm. Thus, in the parlance of Definition 14, we will get a 1-approximation to the normalized leverage scores of A with rank parameter k.

Lemma 17 (Frobenius Sketch) *Given $A \in \mathbb{R}^{n\times d}$ of rank ρ, a rank parameter k such that $2 \le k < \rho$, and an error parameter ε such that $0 < \varepsilon < 1$, let $\Pi \in \mathbb{R}^{d\times r}$ be a standard Gaussian matrix (with entries selected in i.i.d. trials from $\mathcal{N}(0,1)$) with*
$$r \ge k + \left\lceil \frac{10k}{\varepsilon} + 1 \right\rceil. \tag{18}$$

Let $B = A\Pi$ and let X be as in Equation (19). Then,

$$\mathbf{E}\left[\|A-X\|_F^2\right] \leq \left(1+\frac{\varepsilon}{10}\right)\|A-A_k\|_F^2.$$

The matrix B can be computed in $O\left(ndk\varepsilon^{-1}\right)$ time.

Let

$$X = Q\left(Q^TA\right)_k \in \mathbb{R}^{n\times d}, \tag{19}$$

where $\left(Q^TA\right)_k$ is the best rank-k approximation to the matrix Q^TA; from standard linear algebra, $\left(Q^TA\right)_k = U_{Q^TA,k}U_{Q^TA,k}^TQ^TA$. Then, the above lemma is proven in Boutsidis et al. (2011b).[11] Now, since X has rank k, it follows that $\|A-X\|_F^2 \geq \|A-A_k\|_F^2$ and thus we can consider the non-negative random variable $\|A-X\|_F^2 - \|A-A_k\|_F^2$ and apply Markov's inequality to get that

$$\|A-X\|_F^2 - \|A-A_k\|_F^2 \leq \varepsilon\|A-A_k\|_F^2$$

holds with probability at least 0.9. Rearranging terms and taking square roots of both sides implies that

$$\|A-X\|_F \leq \sqrt{1+\varepsilon}\|A-A_k\|_F \leq (1+\varepsilon)\|A-A_k\|_F.$$

Thus, $X \in \mathcal{S}_\varepsilon$ with probability at least 0.9. To conclude our proof, recall that Q is an orthonormal basis for the columns of B. From Equation (19),

$$X = Q\left(Q^TA\right)_k = QU_{Q^TA,k}U_{Q^TA,k}^TQ^TA = QU_{Q^TA,k}\Sigma_{Q^TA,k}V_{Q^TA,k}^T.$$

In the above, $\Sigma_{Q^TA,k} \in \mathbb{R}^{k\times k}$ is the diagonal matrix containing the top k singular values of Q^TA and $V_{Q^TA,k}^T \in \mathbb{R}^{k\times d}$ is the matrix whose rows are the top k right singular vectors of Q^TA. Thus, the left singular vectors of the matrix X are exactly equal to the columns of the orthogonal matrix $QU_{Q^TA,k}$; it now follows that the $\hat{\ell}_i$ of Equation (17) are the leverage scores of the matrix X and, finally, that the $\hat{p}_i$ returned by the proposed algorithm are the normalized leverage scores of the matrix X.

We briefly discuss the running time of the proposed algorithm. First, we can compute B in $O(ndr)$ time. Then, the computation of Q takes $O(nr^2)$ time. The computation of Q^TA takes $O(ndr)$ time and the computation of U_{Q^TA} takes $O(dr^2)$ time. Thus, the total time is equal to $O\left(ndr+(n+d)r^2\right)$. The following Theorem summarizes the above discussion.

Theorem 18 *Given $A \in \mathbb{R}^{n\times d}$, a rank parameter k, and an accuracy parameter ε, Algorithm 5 computes a set of normalized leverage scores $\hat{p}_i$ that are 1-approximations to the normalized leverage scores of A with rank parameter k with probability at least 0.7. The proposed algorithm runs in $O\left(ndk\varepsilon^{-1}+(n+d)k^2\varepsilon^{-2}\right)$ time.*

6. Discussion

We will conclude with a discussion of our main results in a broader context: understanding the relationship between our main algorithm and a related estimator for the statistical leverage scores; applying our main algorithm to solve under-constrained least squares problems; and implementing variants of the basic algorithm in streaming environments.

11. More specifically, the proof may be found in Lemma 33 in Section A.3; note that for our purposes here we set $p = \left\lceil \frac{10k}{\varepsilon}+1 \right\rceil$.

6.1 A Related Estimator for the Leverage Scores

Magdon-Ismail (2010) presented the following algorithm to estimate the statistical leverage scores: given as input an $n \times d$ matrix A, with $n \gg d$, the algorithm proceeds as follows.

- Compute ΠA, where the $O\left(\frac{n \ln d}{\ln^2 n}\right) \times n$ matrix Π is a SRHT or another FJLT.
- Compute $X = (\Pi A)^\dagger \Pi$.
- For $t = 1, \ldots, n$, compute the estimate $\tilde{w}_t = A_{(t)}^T X^{(t)}$ and set $w_t = \max\left\{\frac{d \ln^2 n}{4n}, \tilde{w}_t\right\}$.
- Return the quantities $\tilde{p}_i = w_i / \sum_{i'=1}^n w_{i'}$, for $i \in [n]$.

Magdon-Ismail (2010) argued that the output $\tilde{p}_i$ achieves an $O(\ln^2 n)$ approximation to all of the (normalized) statistical leverage scores of A in roughly $O(nd^2/\ln n)$ time. (To our knowledge, prior to our work here, this is the only known estimator that obtains any nontrivial provable approximation to the leverage scores of a matrix in $o(nd^2)$ time.) To see the relationship between this estimator and our main result, recall that

$$\ell_i = e_i^T U U^T e_i = e_i^T A A^\dagger e_i = x_i^T y_i,$$

where the vector $x_i^T = e_i^T A$ is cheap to compute and the vector $y_i = A^\dagger e_i$ is expensive to compute. The above algorithm effectively approximates $y_i = A^\dagger e_i$ via a random projection as $\tilde{y}_i = (\Pi A)^\dagger \Pi e_i$, where Π is a SRHT or another FJLT. Since the estimates $x_i^T \tilde{y}_i$ are not necessarily positive, a truncation at the negative tail, followed by a renormalization step, must be performed in order to arrive at the final estimator returned by the algorithm. This truncation-renormalization step has the effect of inflating the estimates of the small leverage scores by an $O(\ln^2 n)$ factor. By way of comparison, Algorithm 1 essentially computes a sketch of $AA^\dagger$ of the form $A(\Pi A)^\dagger \Pi^T$ that maintains positivity for each of the row norm estimates.

Although both Algorithm 1 and the algorithm of this subsection estimate $AA^\dagger$ by a matrix of the form $A(\Pi A)^\dagger \Pi^T$, there are notable differences. The algorithm of this subsection does not actually compute or approximate AA^T directly; instead, it separates the matrix into two parts and computes the dot product between $e_i^T A$ and $(\Pi A)^\dagger \Pi e_i$. Positivity is sacrificed and this leads to some complications in the estimator; however, the truncation step is interesting, since, despite the fact that the estimates are "biased" (in a manner somewhat akin to what is obtained with "thresholding" or "regularization" procedures), we still obtain provable approximation guarantees. The algorithm of this subsection is simpler (since it uses an application of only one random projection), albeit at the cost of weaker theoretical guarantees and a worse running time than our main algorithm. A direction of considerable practical interest is to evaluate empirically the performance of these two estimators, either for estimating all the leverage scores or (more interestingly) for estimating the largest leverage scores for data matrices for which the leverage scores are quite nonuniform.

6.2 An Application to Under-constrained Least-squares Problems

Consider the following under-constrained least-squares problem:

$$\min_{x \in \mathbb{R}^d} \|Ax - b\|_2, \tag{20}$$

where $A \in \mathbb{R}^{n\times d}$ has much fewer rows than columns, that is, $n \ll d$. It is well-known that we can solve this problem exactly in $O(n^2 d)$ time and that the minimal ℓ_2-norm solution is given by $x_{opt} = A^\dagger b$. For simplicity, let's assume that the input matrix A has full rank (i.e., $\mathbf{rank}(A) = n$) and thus $\|Ax_{opt} - b\|_2 = 0$.

In this section, we will argue that Algorithm 6 computes a simple, accurate estimator $\tilde{x}_{opt}$ for x_{opt}. In words, Algorithm 6 samples a small number of columns from A (note that the columns of A correspond to variables in our under-constrained problem) and uses the sampled columns to compute $\tilde{x}_{opt}$. However, in order to determine which columns will be included in the sample, the algorithm will make use of the statistical leverage scores of the matrix A^T; more specifically, columns (and thus variables) will be chosen with probability proportional to the corresponding statistical leverage score. We will state Algorithm 6 assuming that these probabilities are parts of the input; the following theorem is our main quality-of-approximation result for Algorithm 6.

Theorem 19 *Let $A \in \mathbb{R}^{n\times d}$ be a full-rank matrix with $n \ll d$; let $\varepsilon \in (0, 0.5]$ be an accuracy parameter; let $\delta \in (0,1)$ be a failure probability; and let $x_{opt} = A^\dagger b$ be the minimal ℓ_2-norm solution to the least-squares problem of Equation (20). Let $p_i \geq 0$, $i \in [d]$, be a set of probabilities satisfying $\sum_{i=1}^d p_i = 1$ and*

$$p_i \geq \frac{\beta \left\|V_{(i)}\right\|_2^2}{n} \tag{21}$$

for some constant $\beta \in (0,1]$. (Here $V \in \mathbb{R}^{d\times n}$ is the matrix of the right singular vectors of A.) If $\tilde{x}_{opt}$ is computed via Algorithm 6 then, with probability at least $1-\delta$,

$$\|x_{opt} - \tilde{x}_{opt}\|_2 \leq 2\varepsilon \|x_{opt}\|_2 .$$

Algorithm 6 runs in $O\left(n^3\varepsilon^{-2}\beta^{-1}\ln(n/\varepsilon\beta\delta) + nd\right)$ time.

Proof Let the singular value decomposition of the full-rank matrix A be $A = U\Sigma V^T$, with $U \in \mathbb{R}^{n\times n}$, $\Sigma \in \mathbb{R}^{n\times n}$, and $V \in \mathbb{R}^{d\times n}$; note that all the diagonal entries of Σ are strictly positive since A has full rank. We can now apply Theorem 4 of Section 6.1 of Drineas et al. (2010b) to get[12] that

$$\left\|I_n - V^T SS^T V\right\|_2 = \left\|V^T V - V^T SS^T V\right\|_2 \leq \varepsilon$$

for our choice of r with probability at least $1-\delta$. Note that $V^T S \in \mathbb{R}^{n\times r}$ (with $r \geq n$) and let $\sigma_i\left(V^T S\right)$ denote the singular values of $V^T S$ for all $i \in [n]$; the above inequality implies that for all $i \in [n]$

$$\left|1 - \sigma_i^2\left(V^T S\right)\right| \leq \left\|I_n - V^T SS^T V\right\|_2 \leq \varepsilon \leq 0.5.$$

Thus, all the singular values of $V^T S$ are strictly positive and hence $V^T S$ has full rank equal to n. Also, using $\varepsilon \leq 0.5$,

$$\left|1 - \sigma_i^{-2}\left(V^T S\right)\right| \leq \frac{\varepsilon}{1-\varepsilon} \leq 2\varepsilon. \tag{22}$$

12. We apply Theorem 4 of Section 6.1 of Drineas et al. (2010b) with $A = V^T$ and note that $\left\|V^T\right\|_F^2 = n \geq 1$, $\left\|V^T\right\|_2 = 1$, and $\left(V^T\right)^{(i)} = V_{(i)}$.

We are now ready to prove our theorem:

$$\begin{aligned}
\|x_{opt} - \tilde{x}_{opt}\|_2 &= \left\| A^T (AS)^{\dagger T} (AS)^{\dagger} b - A^{\dagger} b \right\|_2 \\
&= \left\| V\Sigma U^T \left(U\Sigma V^T S\right)^{\dagger T} \left(U\Sigma V^T S\right)^{\dagger} b - V\Sigma^{-1} U^T b \right\|_2 \\
&= \left\| \Sigma U^T U \Sigma^{-1} \left(V^T S\right)^{\dagger T} \left(V^T S\right)^{\dagger} \Sigma^{-1} U^T b - \Sigma^{-1} U^T b \right\|_2 \\
&= \left\| \left(V^T S\right)^{\dagger T} \left(V^T S\right)^{\dagger} \Sigma^{-1} U^T b - \Sigma^{-1} U^T b \right\|_2 .
\end{aligned}$$

In the above derivations we substituted the SVD of A, dropped terms that do not change unitarily invariant norms, and used the fact that $V^T S$ and Σ have full rank in order to simplify the pseudoinverse. Now let $\left(V^T S\right)^{\dagger T} \left(V^T S\right)^{\dagger} = I_n + E$ and note that Equation (22) and the fact that $V^T S$ has full rank imply

$$\|E\|_2 = \left\| I_n - \left(V^T S\right)^{\dagger T} \left(V^T S\right)^{\dagger} \right\|_2 = \max_{i \in [n]} \left| 1 - \sigma_i^{-2} \left(V^T S\right) \right| \leq 2\varepsilon.$$

Thus, we conclude our proof by observing that

$$\begin{aligned}
\|x_{opt} - \tilde{x}_{opt}\|_2 &= \left\| (I_n + E) \Sigma^{-1} U^T b - \Sigma^{-1} U^T b \right\|_2 \\
&= \left\| E \Sigma^{-1} U^T b \right\|_2 \\
&\leq \|E\|_2 \left\| \Sigma^{-1} U^T b \right\|_2 \\
&\leq 2\varepsilon \|x_{opt}\|_2 .
\end{aligned}$$

In the above we used the fact that $\|x_{opt}\|_2 = \left\|A^{\dagger} b\right\|_2 = \left\|V\Sigma^{-1} U^T b\right\|_2 = \left\|\Sigma^{-1} U^T b\right\|_2$. The running time of the algorithm follows by observing that AS is an $n \times r$ matrix and thus computing its pseudoinverse takes $O(n^2 r)$ time; computing x_{opt} takes an additional $O(nr + dn)$ time. ■

We conclude the section with a few remarks. First, assuming that ε, β, and δ are constants and $n \ln n = o(d)$, it immediately follows that Algorithm 6 runs in $o(n^2 d)$ time. It should be clear that we can use Theorem 2 and the related Algorithm 1 to approximate the statistical leverage scores, thus bypassing the need to exactly compute them. Second, instead of approximating the statistical leverage scores needed in Algorithm 6, we could use the randomized Hadamard transform (essentially post-multiply A by a randomized Hadamard transform to make all statistical leverage scores uniform). The resulting algorithm could be theoretically analyzed following the lines of Drineas et al. (2010b). It would be interesting to evaluate experimentally the performance of the two approaches in real data.

6.3 Extension to Streaming Environments

In this section, we consider the estimation of the leverage scores and of related statistics when the input data set is so large that an appropriate way to view the data is as a data stream (Muthukrishnan, 2005). In this context, one is interested in computing statistics of the data stream while making one pass (or occasionally a few additional passes) over the data from external storage and using only a small amount of additional space. For an $n \times d$ matrix A, with $n \gg d$, small additional space means that the space complexity only depends *logarithmically* on the high dimension n and *polynomially* on the low dimension d. When we discuss bits of space, we assume that the entries of A can be

Input: $A \in \mathbb{R}^{n\times d}$, $b \in \mathbb{R}^n$, error parameter $\varepsilon \in (0,.5]$, failure probability δ, and a set of probabilities p_i (for all $i \in [d]$) summing up to one and satisfying Equation (21).

Output: $\tilde{x}_{opt} \in \mathbb{R}^d$.

1. Let $r = \frac{96n}{\beta\varepsilon^2} \ln\left(\frac{96n}{\beta\varepsilon^2\sqrt{\delta}}\right)$.
2. Let $S \in \mathbb{R}^{d\times r}$ be an all-zeros matrix.
3. **For** $t = 1,\ldots,r$ **do**
   - Pick $i_t \in [d]$ such that $\mathbf{Pr}(i_t = i) = p_i$.
   - $S_{i_t t} = 1/\sqrt{r p_{i_t}}$.
4. Return $\tilde{x}_{opt} = A^T (AS)^{\dagger T} (AS)^{\dagger} b$.

Algorithm 6: Approximately solving under-constrained least squares problems.

discretized to $O(\log n)$ bit integers, though all of our results can be generalized to arbitrary word sizes. The general strategy behind our algorithms is as follows.

- As the data streams by, compute TA, for an appropriate problem-dependent linear sketching matrix T, and also compute ΠA, for a random projection matrix Π.[13]
- After the first pass over the data, compute the matrix R^{-1}, as described in Algorithm 1, corresponding to ΠA (or compute the pseudoinverse of ΠA or the R matrix from any other QR decomposition of A).
- Compute $TAR^{-1}\Pi_2$, for a random projection matrix Π_2, such as the one used by Algorithm 1.

With the procedure outlined above, the matrix T is effectively applied to the rows of $AR^{-1}\Pi_2$, that is, to the sketch of A that has rows with Euclidean norms approximately equal to the row norms of U, and pairwise inner products approximately equal to those in U. Thus statistics related to U can be extracted.

Large Leverage Scores. Given any $n \times d$ matrix A in a streaming setting, it is known how to find the indices of all rows $A_{(i)}$ of A for which $\|A_{(i)}\|_2^2 \geq \tau\|A\|_F^2$, for a parameter τ, and in addition it is known how to compute a $(1+\varepsilon)$-approximation to $\|A_{(i)}\|_2^2$ for these large rows. The basic idea is to use the notion of ℓ_2-sampling on matrix A, namely, to sample random entries A_{ij} with probability $A_{ij}^2/\|A\|_F^2$. A single entry can be sampled from this distribution in a single pass using $O(\varepsilon^{-2}\log^3(nd))$ bits of space (Monemizadeh and Woodruff, 2010; Andoni et al., 2010). More precisely, these references demonstrate that there is a distribution over $O(d\varepsilon^{-2}\log^3(nd)) \times n$ matrices

13. In the offline setting, one would use an SRHT or another FJLT, while in the streaming setting one could use either of the following. If the stream is such that one sees each entire column of A at once, then one could do an FJLT on the column. Alternatively, if one see updates to the individual entries of A in an arbitrary order, then one could apply any sketching matrix, such as those of Achlioptas (2003) or of Dasgupta et al. (2010).

T for which for any fixed matrix $A \in \mathbb{R}^{n\times d}$, there is a procedure which given TA, outputs a sample $(i,j) \in [n]\times[d]$ with probability $(1\pm\varepsilon)\frac{A_{i,j}^2}{\|A\|_F^2} \pm n^{-O(1)}$. Technically, these references concern sampling from vectors rather than matrices, so $T(A)$ is a linear operator which treats A as a length-nd vector and applies the algorithm of Monemizadeh and Woodruff (2010) and Andoni et al. (2010). However, by simply increasing the number of rows in T by a factor of the small dimension d, we can assume T is left matrix multiplication. By considering the marginal along $[n]$, the probability that $i = a$, for any $a \in [n]$, is

$$(1\pm\varepsilon)\frac{\|U_{(a)}\|_2^2}{\|U\|_F^2} \pm (nd)^{-O(1)}.$$

By the coupon collector problem, running $O(\tau^{-1}\log\tau^{-1})$ independent copies is enough to find a set containing all rows $A_{(i)}$ for which $\|A_{(i)}\|_2^2 \geq \tau\|A\|_F^2$, and no rows $A_{(i)}$ for which $\|A_{(i)}\|_2^2 < \frac{\tau}{2}\|A\|_F^2$ with probability at least 0.99.

When applied to our setting, we can apply a random projection matrix Π and a linear sketching matrix T which has $O(d\tau^{-1}\varepsilon^{-2}\log^3(n)\log\tau^{-1})$ rows in the following manner. First, TA and ΠA are computed in the first pass over the data; then, at the end of the first pass, we compute R^{-1}; and finally, we compute $TAR^{-1}\Pi_2$, for a random projection matrix Π_2. This procedure effectively applies the matrix T to the rows of $AR^{-1}\Pi_2$, which have norms equal to the row norms of U, up to a factor of $1+\varepsilon$. The multiplication at the end by Π_2 serves only to speed up the time for processing TAR^{-1}. Thus, by the results of Monemizadeh and Woodruff (2010) and Andoni et al. (2010), we can find all the leverage scores $\|U_{(i)}\|_2^2$ that are of magnitude at least $\tau\|U\|_F^2$ in small space and a single pass over the data. By increasing the space by a factor of $O(\varepsilon^{-2}\log n)$, we can also use the ℓ_2-samples to estimate the norms $\|U_{(i)}\|_2^2$ for the row indices i that we find.

Entropy. Given a distribution ρ, a statistic of ρ of interest is the entropy of this distribution, where the entropy is defined as $H(\rho) = \sum_i \rho(i)\log_2(1/\rho(i))$. This statistic can be approximated in a streaming setting. Indeed, it is known that estimating $H(\rho)$ up to an additive ε can be reduced to $(1+\tilde{\varepsilon})$-approximation of the ℓ_p-norm of the vector $(\rho(1),\ldots,\rho(n))$, for $O(\log 1/\varepsilon)$ different $p \in (0,1)$ (Harvey et al., 2008). Here $\tilde{\varepsilon} = \varepsilon/(\log^3 1/\varepsilon \cdot \log n)$. When applied to our setting, the distribution of interest is $\rho(i) = \frac{1}{d}\|U_{(i)}\|_2^2$. To compute the entropy of this distribution, there exist sketching matrices T for providing $(1+\varepsilon)$-approximations to the quantity $F_p(F_2)$ of an $n\times d$ matrix A, where $F_p(F_2)$ is defined as $\sum_{i=1}^n \|A_{(i)}\|_2^{2p}$, using $O(\varepsilon^{-4}\log^2 n\log 1/\varepsilon)$ bits of space (see Theorem 1 of Ganguly et al. (2008)). Thus, to compute the entropy of the leverage score distribution, we can do the following. First, maintain TA and ΠA in the first pass over the data, where T is a sketching matrix for $F_p(F_2)$, $p \in (0,1)$. At the end of the first pass, compute R^{-1}; and finally, compute $TAR^{-1}\Pi_2$, which effectively applies the $F_p(F_2)$-estimation matrix T to the rows of the matrix $AR^{-1}\Pi_2$. Therefore, by the results of Harvey et al. (2008) and Ganguly et al. (2008), we can compute an estimate ϕ which is within an additive ε of $H(\rho)$ using $O(d\varepsilon^{-4}\log^6 n\log^{14} 1/\varepsilon)$ bits of space and a single pass. We note that it is also possible to estimate $H(\rho)$ up to a multiplicative $1+\varepsilon$ factor using small, but more, space; see, for example, Harvey et al. (2008).

Sampling Row Identities. Another natural problem is that of obtaining samples of rows of A proportional to their leverage score importance sampling probabilities. To do so, we use ℓ_2-sampling (Monemizadeh and Woodruff, 2010; Andoni et al., 2010) as used above for finding the large leverage scores. First, compute TA and ΠA in the first pass over the data stream; then, compute R^{-1}; and finally, compute TAR^{-1}. Thus, by applying the procedures of Andoni et al. (2010) a total of s times independently, we obtain s samples $i_1,\ldots,i_s$, with replacement, of rows of A proportional

to $\|U_{(i_1)}\|_2^2, \ldots, \|U_{(i_s)}\|_2^2$, that is, to their leverage score. The algorithm requires $O(sd\varepsilon^{-2}\log^4 n)$ bits of space and runs in a single pass. To obtain more than just the row identities $i_1, \ldots, i_s$, for example, to obtain the actual samples, one can read off these rows from A in a second pass over the matrix.

References

D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.

N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 557–563, 2006.

N. Ailon and B. Chazelle. The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. *SIAM Journal on Computing*, 39(1):302–322, 2009.

N. Ailon and E. Liberty. Fast dimension reduction using Rademacher series on dual BCH codes. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–9, 2008.

A. Andoni, R. Krauthgamer, and K. Onak. Streaming algorithms from precision sampling. Technical report, 2010. Preprint: arXiv:1011.1263.

H. Avron, P. Maymounkov, and S. Toledo. Blendenpik: Supercharging LAPACK's least-squares solver. *SIAM Journal on Scientific Computing*, 32:1217–1236, 2010.

C. Bekas, E. Kokiopoulou, and Y. Saad. An estimator for the diagonal of a matrix. *Applied Numerical Mathematics*, 57:1214–1229, 2007.

C. Bekas, A. Curioni, and I. Fedulova. Low cost high performance uncertainty quantification. In *Proceedings of the 2nd Workshop on High Performance Computational Finance*, page Article No.: 8, 2009.

P. Bonacich. Power and centrality: A family of measures. *The American Journal of Sociology*, 92 (5):1170–1182, 1987.

C. Boutsidis, M.W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 968–977, 2009.

C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 305–314, 2011a.

C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. Technical report, 2011b. Preprint: arXiv:1103.0995.

E.J. Candes and B. Recht. Exact matrix completion via convex optimization. Technical report, 2008. Preprint: arXiv:0805.4471.

S. Chatterjee and A.S. Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statistical Science*, 1(3):379–393, 1986.

S. Chatterjee and A.S. Hadi. *Sensitivity Analysis in Linear Regression*. John Wiley & Sons, New York, 1988.

S. Chatterjee, A.S. Hadi, and B. Price. *Regression Analysis by Example*. John Wiley & Sons, New York, 2000.

A. Dasgupta, R. Kumar, and T. Sarlós. A sparse Johnson-Lindenstrauss transform. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 341–350, 2010.

P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36:132–157, 2006a.

P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1127–1136, 2006b.

P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30:844–881, 2008.

P. Drineas, J. Lewis, and P. Paschou. Inferring geographic coordinates of origin for Europeans using small panels of ancestry informative markers. *PLoS ONE*, 5(8):e11892, 2010a.

P. Drineas, M.W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2010b.

S. Ganguly, M. Bansal, and S. Dube. Estimating hybrid frequency moments of data streams. In *Proceedings of the 2nd Annual International Workshop on Frontiers in Algorithmics*, pages 55–66, 2008.

S. Georgiev and S. Mukherjee, 2011. Unpublished results. (2011).

A. Gittens and M. W. Mahoney, 2012. In preparation. (2012).

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1996.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

N.J.A. Harvey, J. Nelson, and K. Onak. Sketching and streaming entropy via approximation theory. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 489–498, 2008.

D.C. Hoaglin and R.E. Welsch. The hat matrix in regression and ANOVA. *The American Statistician*, 32(1):17–22, 1978.

E. A. Jonckheere, M. Lou, J. Hespanha, and P. Barooah. Effective resistance of Gromov-hyperbolic graphs: Application to asymptotic sensor network problems. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 1453–1458, 2007.

M. Magdon-Ismail. Row sampling for matrix algorithms via a non-commutative Bernstein bound. Technical report, 2010. Preprint: arXiv:1008.0587.

M. W. Mahoney. *Randomized Algorithms for Matrices and Data*. Foundations and trends in machine learning. NOW Publishers, Boston, 2011. Also available at: arXiv:1104.5557.

M.W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proc. Natl. Acad. Sci. USA*, 106:697–702, 2009.

X. Meng, M. A. Saunders, and M. W. Mahoney. LSRN: A parallel iterative solver for strongly over- or under-determined systems. Technical report, 2011. Preprint: arXiv:1109.5981.

M. Monemizadeh and D. P. Woodruff. 1-pass relative-error l_p-sampling with applications. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1143–1160, 2010.

S. Muthukrishnan. *Data Streams: Algorithms and Applications*. Foundations and Trends in Theoretical Computer Science. Now Publishers Inc, Boston, 2005.

M.Z. Nashed, editor. *Generalized Inverses and Applications*. Academic Press, New York, 1976.

M.E.J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27:39–54, 2005.

C.H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: a probabilistic analysis. *Journal of Computer and System Sciences*, 61(2):217–235, 2000.

P. Paschou, E. Ziv, E.G. Burchard, S. Choudhry, W. Rodriguez-Cintron, M.W. Mahoney, and P. Drineas. PCA-correlated SNPs for structure identification in worldwide human populations. *PLoS Genetics*, 3:1672–1686, 2007.

P. Paschou, J. Lewis, A. Javed, and P. Drineas. Ancestry informative markers for fine-scale individual assignment to worldwide populations. *Journal of Medical Genetics*, page doi:10.1136/jmg.2010.078212, 2010.

M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the 8th Annual ACM SIGKDD Conference*, pages 61–70, 2002.

V. Rokhlin and M. Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proc. Natl. Acad. Sci. USA*, 105(36):13212–13217, 2008.

V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 143–152, 2006.

A. Talwalkar and A. Rostamizadeh. Matrix coherence and the Nyström method. In *Proceedings of the 26th Conference in Uncertainty in Artificial Intelligence*, 2010.

P.F. Velleman and R.E. Welsch. Efficient computing of regression diagnostics. *The American Statistician*, 35(4):234–242, 1981.

Journal of Machine Learning Research 13 (2012) 3507-3531 Submitted 3/08; Revised 3/12; Published 12/12

PAC-Bayes Bounds with Data Dependent Priors

Emilio Parrado-Hernández EMIPAR@TSC.UC3M.ES
Department of Signal Processing and Communications
University Carlos III of Madrid
Leganés, 28911, Spain

Amiran Ambroladze A.AMBROLADZE@FREEUNI.EDU.GE
Department of Mathematics and Computer Science
Tbilisi Free University
Bedia Street
0182 Tbilisi, Georgia

John Shawe-Taylor J.SHAWE-TAYLOR@CS.UCL.AC.UK
Department of Computer Science
University College London
London, WC1E 6BT, UK

Shiliang Sun SHILIANGSUN@GMAIL.COM
Department of Computer Science and Technology
East China Normal University
500 Dongchuan Road
Shanghai 200241, China

Editor: Gabor Lugosi

Abstract

This paper presents the prior PAC-Bayes bound and explores its capabilities as a tool to provide tight predictions of SVMs' generalization. The computation of the bound involves estimating a prior of the distribution of classifiers from the available data, and then manipulating this prior in the usual PAC-Bayes generalization bound. We explore two alternatives: to learn the prior from a separate data set, or to consider an expectation prior that does not need this separate data set. The prior PAC-Bayes bound motivates two SVM-like classification algorithms, prior SVM and η-prior SVM, whose regularization term pushes towards the minimization of the prior PAC-Bayes bound. The experimental work illustrates that the new bounds can be significantly tighter than the original PAC-Bayes bound when applied to SVMs, and among them the combination of the prior PAC-Bayes bound and the prior SVM algorithm gives the tightest bound.

Keywords: PAC-Bayes bound, support vector machine, generalization capability prediction, classification

1. Introduction

Support vector machines (SVMs) (Boser et al., 1992; Vapnik, 1998; Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002) are accepted among practitioners as one of the most accurate automatic classification techniques. They implement linear classifiers in a high-dimensional feature space using the kernel trick to enable a dual representation and efficient computation. The danger of overfitting in such high-dimensional spaces is conquered by maximizing the margin of the classifier

on the training examples. For this reason there has been considerable interest in bounding the generalization in terms of the margin.

In fact, a main drawback that restrains engineers from using these advanced machine learning techniques is the lack of reliable predictions of generalization, especially in what concerns worst-case performance. In this sense, the widely used cross-validation generalization measures indicate little about the worst-case performance of the algorithms. The error of the classifier on a set of samples follows a binomial distribution whose mean is the true error of the classifier. Cross-validation is a sample mean estimation of the true error, and worst-case performance estimations concern the estimation of the tail of the error distribution. One could then employ statistical learning theory (SLT) tools to bound the tail of the distribution of errors. Early bounds have relied on covering number computations (Shawe-Taylor et al., 1998; Zhang, 2002), while later bounds have considered Rademacher complexity (Bartlett and Mendelson, 2002). The tightest bounds for practical applications appear to be the PAC-Bayes bound (McAllester, 1999; Langford and Shawe-Taylor, 2002; Catoni, 2007) and in particular the form given in Seeger (2002), Langford (2005) and Germain et al. (2009). However, there still exist a remarkable gap between SLT predictions and practitioners' experiences: SLT predictions are too pessimistic when compared to the actual results data analysts get when they apply machine learning algorithms to real-world problems.

Another issue affected by the ability to predict the generalization capability of a classifier is the selection of the hyperparameters that define the training. In the SVM case, these parameters are the trade-off between maximum margin and minimum training error, C, and the kernel parameters. Again, the more standard method of cross-validation has proved to be more reliable in most experiments, despite the fact that it is statistically poorly justified and relatively expensive.

The aim of this paper is to investigate whether the PAC-Bayes bound can be tightened towards less pessimistic predictions of generalization. Another objective is to study the implications of the bound in the training of the classifiers. We specifically address the use of the bound in the model selection stage and in the design of regularization terms other than the maximization of the margin.

The PAC-Bayes bound (retrospected in Section 2) uses a Gaussian prior centered at the origin in the weight space. The key to the new bounds introduced here is to use part of the training set to compute a more informative prior and then compute the bound on the remainder of the examples relative to this prior. This generalisation of the bound, called prior PAC-Bayes bound, is derived in Section 3. The prior PAC-Bayes bound was initially presented by Ambroladze et al. (2007). A slight nuisance of the prior PAC-Bayes bound is that a separate data set should be available in order to fix the prior. In Section 3.2, we further develop the expectation-prior PAC-Bayes bound as an interesting new approach which does not require the existence of the separate data set. We also derive a PAC-Bayes bound with a non-spherical Gaussian prior. To the best of our knowledge this is the first such application for SVMs.

The encouraging results of Ambroladze et al. (2007), motivate a further use of the prior PAC-Bayes bound. Section 4.1 introduces a new classification algorithm, the prior SVM, which replaces the margin maximization in the optimization problem by a regularization term that pushes towards the minimization of the PAC-Bayes bound. The optimization problem that produces the prior SVM is divided into three stages. The first one involves the learning of a prior formed by an ensemble of Gaussian distributions centered at different distances along the same direction. During the second stage, each component of the prior is mapped with a posterior that improves its classification accuracy while tightening the PAC-Bayes bound. In the last stage the prior component/posterior pair that achieves the lowest value of the PAC-Bayes bound is selected as prior SVM classifier. Section

4.2 presents a second algorithm, named η-prior SVM as a variant of prior SVMs where the position of component of the prior that goes into the overall classifier is optimised in a continuous range (not picked from a fixed set). Therefore, η-prior SVMs include a first optimization where the direction of the prior is learnt from a separate set of training patterns, and a second optimization that determines (i) the exact position of the prior along the already learnt direction and (ii) the position of the posterior. Furthermore we show that the performance of the algorithm can be bounded rigorously using PAC-Bayes techniques.

In Section 5 the new bounds and algorithms are evaluated on multiple classification tasks after a parameter selection. The experiments illustrate the capabilities of the prior PAC-Bayes bound to provide tighter predictions of the generalisation of an SVM. Moreover, the combination of the new bounds and the two prior SVM algorithms yields more dramatic tightenings of the bound. Besides, these classifiers achieve good accuracies, comparable to those obtained by an SVM with its parameters fixed with ten fold cross validation. We finish the experimental work showing that the use of a different value of C for the prior and the posterior that form the (η)prior SVM lead to a further tightening of the bound.

Finally, the main conclusions of this work and some related ongoing research are outlined in Section 6.

2. PAC-Bayes Bound for SVMs

This section is devoted to a brief review of the PAC-Bayes bound theorem of Langford (2005). Let us consider a distribution $\mathcal{D}$ of patterns $\mathbf{x}$ lying in a certain input space $\mathcal{X}$ and their corresponding output labels y ($y \in \{-1,1\}$). Suppose Q is a posterior distribution over the classifiers c. For every classifier c, the following two error measures are defined:

Definition 1 (True error) *The true error $c_{\mathcal{D}}$ of a classifier c is defined to be the probability of misclassifying a pattern-label pair $(\mathbf{x}, y)$ selected at random from $\mathcal{D}$*

$$c_{\mathcal{D}} \equiv Pr_{(\mathbf{x},y)\sim\mathcal{D}}(c(\mathbf{x}) \neq y).$$

Definition 2 (Empirical error) *The empirical error $\hat{c}_S$ of a classifier c on a sample S of size m is defined to be the error rate on S*

$$\hat{c}_S \equiv Pr_{(\mathbf{x},y)\sim S}(c(\mathbf{x}) \neq y) = \frac{1}{m}\sum_{i=1}^{m} I(c(\mathbf{x}_i) \neq y_i),$$

where $I(\cdot)$ is an indicator function equal to 1 *if the argument is true and equal to* 0 *if the argument is false.*

Now we define two error measures on the distribution of classifiers: the average true error, $Q_{\mathcal{D}} \equiv \mathbb{E}_{c\sim Q} c_{\mathcal{D}}$, as the probability of misclassifying an instance $\mathbf{x}$ chosen uniformly from $\mathcal{D}$ with a classifier c chosen according to Q; and the average empirical error $\hat{Q}_S \equiv \mathbb{E}_{c\sim Q}\hat{c}_S$, as the probability of classifier c chosen according to Q misclassifying an instance $\mathbf{x}$ chosen from a sample S.

For these two quantities we can derive the PAC-Bayes bound on the true error of the distribution of classifiers:

Theorem 3 (PAC-Bayes bound) *For all prior distributions $P(c)$ over the classifiers c, and for any $\delta \in (0,1]$,*

$$Pr_{S\sim\mathcal{D}^m}\left(\forall Q(c) : KL_+(\hat{Q}_S||Q_\mathcal{D}) \leq \frac{KL(Q(c)||P(c)) + \ln(\frac{m+1}{\delta})}{m}\right) \geq 1-\delta,$$

where $KL(Q(c)||P(c)) = \mathbb{E}_{c\sim Q}\ln\frac{Q(c)}{P(c)}$ is the Kullback-Leibler divergence, and $KL_+(q||p) = q\ln\frac{q}{p} + (1-q)\ln\frac{1-q}{1-p}$ for $p > q$ and 0 otherwise.

The proof of the theorem can be found in Langford (2005).

This bound can be specialized to the case of linear threshold classifiers. Suppose the m training examples define a linear classifier that can be represented by the following equation:

$$c_\mathbf{u}(\mathbf{x}) = \text{sign}(\mathbf{u}^T\phi(\mathbf{x})), \tag{1}$$

where $\phi(\mathbf{x})$ is a nonlinear projection to a certain feature space[1] where the linear classification actually takes place, and vector $\mathbf{u}$ in the feature space determines the separating hyperplane. Since we are considering only classifiers with threshold set to zero all the classifiers in the paper can be represented with unit vectors ($\|\mathbf{w}\| = 1$). However, as we will be considering distributions of classifiers we use the notation $\mathbf{u}$ to indicate weight vectors that can also be non-unit.

For any unit vector $\mathbf{w}$ we can define a stochastic classifier in the following way: we choose the distribution $Q(c_\mathbf{u}) = Q(c_\mathbf{u}|\mathbf{w},\mu)$, where $\mathbf{u} \sim \mathcal{N}(\mu\mathbf{w}, I)$ is drawn from a spherical Gaussian with identity covariance matrix centered along the direction pointed by $\mathbf{w}$ at a distance μ from the origin. Moreover, we can choose the prior $c_\mathbf{u} : \mathbf{u} \sim \mathcal{N}(\mathbf{0}, I)$ to be a spherical Gaussian with identity covariance matrix centered at the origin. Then, for classifiers of the form in Equation (1) the generalization performance can be bounded as

Corollary 4 (PAC-Bayes bound for SVMs (Langford, 2005)) *For all distributions $\mathcal{D}$, for all $\delta \in (0,1]$, we have*

$$Pr_{S\sim\mathcal{D}^m}\left(\forall \mathbf{w},\mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_\mathcal{D}(\mathbf{w},\mu)) \leq \frac{\frac{\mu^2}{2} + \ln(\frac{m+1}{\delta})}{m}\right) \geq 1-\delta.$$

It can be shown (see Langford, 2005) that

$$\hat{Q}_S(\mathbf{w},\mu) = \mathbb{E}_m[\tilde{F}(\mu\gamma(\mathbf{x},y))], \tag{2}$$

where $\mathbb{E}_m$ is the average over the m training examples, $\gamma(\mathbf{x},y)$ is the normalized margin of the training examples

$$\gamma(\mathbf{x},y) = \frac{y\mathbf{w}^T\phi(\mathbf{x})}{\|\phi(\mathbf{x})\|}, \tag{3}$$

and $\tilde{F} = 1 - F$ where F is the cumulative normal distribution

$$F(x) = \int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}}e^{-x^2/2}\mathrm{d}x. \tag{4}$$

1. This projection is induced by a kernel $\kappa(\cdot)$ satisfying $\kappa(\mathbf{x},\mathbf{y}) = \langle\phi(\mathbf{x}),\phi(\mathbf{y})\rangle$.

Note that the SVM expressed as (1) is computed with a single unit vector $\mathbf{w}$. The generalization error of such a classifier can be bounded by at most twice the average true error $Q_{\mathcal{D}}(\mathbf{w},\mu)$ of the corresponding stochastic classifier involved in Corollary 4 (Langford and Shawe-Taylor, 2002). That is, for all μ we have

$$\Pr_{(\mathbf{x},y)\sim\mathcal{D}}\left(\text{sign}(\mathbf{w}^T\phi(\mathbf{x})) \neq y\right) \leq 2Q_{\mathcal{D}}(\mathbf{w},\mu). \tag{5}$$

3. Data Dependent Prior PAC-Bayes Bounds for SVMs

This section presents some versions of the PAC-Bayes bound that aim at yielding a tighter prediction of the true generalization error of the classifier. These new bounds introduce more sophisticated designs for the prior distribution over the classifiers in order to reduce its divergence with the posterior distribution. The first set of bounds learns the prior distribution from a separate training data set that will not be used in the computation of the bound, whilst the second set learns the prior from mathematical expectations, avoiding to leave out a subset of patterns to calculate the bound.

3.1 Bounds Based on a Separate Set of Training Data

This section is a further extension of previous ideas presented by Ambroladze et al. (2007).

Our first contribution is motivated by the fact that the PAC-Bayes bound allows us to choose the prior distribution, $P(c)$. In the standard application of the bound $P(c)$ is chosen to be a spherical Gaussian centered at the origin. We now consider learning a different prior based on training an SVM on a subset T of the training set comprising r training patterns and labels. In the experiments this is taken as a random subset, but for simplicity of the presentation we will assume T comprises the last r examples $\{\mathbf{x}_k, y_k\}_{k=m-r+1}^{m}$.

With these r examples we can learn an (unit and biased) SVM classifier, $\mathbf{w}_r$, and form a prior $P(\mathbf{w}_r,\eta) \sim \mathcal{N}(\eta\mathbf{w}_r, I)$ consisting of a Gaussian distribution with identity covariance matrix centered along $\mathbf{w}_r$ at a distance η from the origin.

The introduction of this prior $P(\mathbf{w}_r,\eta)$ in Theorem 3 results in the following new bound.

Corollary 5 (Single-prior PAC-Bayes bound for SVMs) *Let us consider a prior on the distribution of classifiers consisting of a spherical Gaussian with identity covariance centered along the direction given by $\mathbf{w}_r$ at a distance η from the origin. Classifier $\mathbf{w}_r$ has been learnt from a subset T of r examples a priori separated from a training set S of m samples. Then, for all distributions $\mathcal{D}$, for all $\delta \in (0,1]$, we have*

$$Pr_{S\sim\mathcal{D}^m}\left(\forall \mathbf{w}_m,\mu : KL_+(\hat{Q}_{S\setminus T}||Q_D) \leq \frac{\frac{||\eta\mathbf{w}_r-\mu\mathbf{w}_m||^2}{2} + \ln(\frac{m-r+1}{\delta})}{m-r}\right) \geq 1-\delta,$$

where $\hat{Q}_{S\setminus T}$ is a stochastic measure of the empirical error of the classifier on the $m-r$ samples not used to learn the prior. This stochastic error is computed as in Equation (2) but averaged over $S\setminus T$.

Proof Since we separate r instances to learn the prior, the actual size of the training set to which we apply the bound is $m-r$. In addition, the stochastic error $\hat{Q}$ must be computed only on the instances not used to learn the prior, that is, the subset $S\setminus T$. Note also that the selection of T can not be optimised.

Using a standard expression for the KL divergence between two Gaussians in an N dimensional space,

$$\mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}_0,\Sigma_0)\|\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1)) = \\ \frac{1}{2}\left(\ln\left(\frac{\det\Sigma_1}{\det\Sigma_0}\right)+\mathrm{tr}(\Sigma_1^{-1}\Sigma_0)+(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)^T\Sigma_1^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)-N\right), \tag{6}$$

the KL divergence between prior and posterior is computed as follows:

$$\mathrm{KL}(Q(\mathbf{w},\mu)||P(\mathbf{w}_r,\eta)) = \mathrm{KL}(\mathcal{N}(\mu\mathbf{w},I)\|\mathcal{N}(\eta\mathbf{w}_r,I)) = \frac{1}{2}||\mu\mathbf{w}-\eta\mathbf{w}_r||^2.$$

■

Intuitively, if the selection of the prior is appropriate, the bound can be tighter than the one given in Corollary 4 when applied to the SVM weight vector on the whole training set. It is worth stressing that the bound holds for all $\mathbf{w}$ and so can be applied to the SVM trained on the whole set. This might at first appear to be 'cheating', but the critical point is that the bound is evaluated on the set $S\backslash T$ not involved in generating the prior. The experimental work illustrates how in fact this bound can be tighter than the standard PAC-Bayes bound.

Moreover, the structure of the prior may be further refined in exchange for a very small increase in the penalty term. This can be achieved with the application of the following result.

Theorem 6 (Mixture prior PAC-Bayes bound) *Let $\mathcal{P}(c)=\sum_{j=1}^{J}\pi_j P_j(c)$ be a prior distribution over classifiers consisting of a mixture of J components $\{P_j(c)\}_{j=1}^{J}$ combined with positive weights $\{\pi_j\}_{j=1}^{J}$ so that $\sum_{j=1}^{J}\pi_j=1$. Then, for all $\delta\in(0,1]$,*

$$Pr_{S\sim\mathcal{D}^m}\left(\forall Q(c): KL_+(\hat{Q}_S||Q_{\mathcal{D}})\leq\min_j\frac{KL(Q(c)||P_j(c))+\ln\frac{m+1}{\delta}+\ln\frac{1}{\pi_j}}{m}\right)\geq 1-\delta.$$

Proof

The bound in Theorem 3 can be instantiated for the ensemble prior $\mathcal{P}(c)$

$$\Pr_{S\sim\mathcal{D}^m}\left(\forall Q(c): \mathrm{KL}_+(\hat{Q}_S||Q_{\mathcal{D}})\leq\frac{\mathrm{KL}(Q(c)||\mathcal{P}(c))+\ln(\frac{m+1}{\delta})}{m}\right)\geq 1-\delta.$$

We now bound the KL divergence between the posterior $Q(c)$ and the ensemble prior $\mathcal{P}(c)$. For any $1\leq i\leq J$:

$$\mathrm{KL}\left(Q(c)\|\mathcal{P}(c)\right)=\int_{c\in C}Q(c)\left(\ln Q(c)-\ln(\sum_{j=1}^{J}\pi_j P_j(c))\right)dc$$

$$\leq\int_{c\in C}Q(c)(\ln Q(c)-\ln(\pi_i P_i(c)))dc=\mathrm{KL}(Q(c)\|P_i(c))-\ln(\pi_i),$$

where the inequality follows from the fact that we have reduced the value inside the $\ln(\cdot)$ term for all c. Finally, the particularisation for the term of minimal $\mathrm{KL}(Q(c)\|P_j(c))-\ln(\pi_j)$ completes the

proof. ■

Note that the inequality in the proof upper bounds the KL divergence to give a bound equivalent to performing a union bound. In particular applications it may be possible to obtain tighter bounds by estimating this KL divergence more closely.

This result can be also specialized for the case of SVM classifiers. The mixture prior is constructed by allocating Gaussian distributions with identity covariance matrix along the direction given by $\mathbf{w}_r$ at distances $\{\eta_j\}_{j=1}^J$ from the origin where $\{\eta_j\}_{j=1}^J$ are positive real numbers. In such a case, we obtain

Corollary 7 (Gaussian Mixture-prior PAC-Bayes bound for SVMs) *Let us consider a prior distribution of classifiers formed by an ensemble of equiprobable spherical Gaussian distributions* $\{P_j(c|\mathbf{w}_r,\eta_j)\}_{j=1}^J$ *with identity covariance and mean* $\eta_j\mathbf{w}_r$*, where* $\{\eta_j\}_{j=1}^J$ *are positive real numbers and* $\mathbf{w}_r$ *is a linear classifier trained using a subset* T *of* r *samples a priori separated from a training set* S *of* m *samples. Then, for all distributions* $\mathcal{D}$*, for all posteriors* $(\mathbf{w},\mu)$ *and for all* $\delta \in (0,1]$*, we have that with probability greater than* $1-\delta$ *over all the training sets* S *of size* m *sampled from* $\mathcal{D}$

$$KL_+(\hat{Q}_{S\setminus T}(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \min_j \frac{\frac{||\eta_j\mathbf{w}_r-\mu\mathbf{w}||^2}{2} + \ln(\frac{m-r+1}{\delta}) + \ln J}{m-r}.$$

Proof The proof is straightforward and can be completed by substituting $1/J$ for all π_j in Theorem 6 and computing the KL divergence between prior and posterior as in the proof of Corollary 5. ■

Note that the $\{\eta_j\}_{j=1}^J$ must be chosen before we actually compute the posterior. A linear search can be implemented for the value of μ that leads to the tightest bound for each particular prior. In the case of a mixture prior, the search is repeated for every member of the ensemble and the reported value of the bound is the tightest one found during the searches.

Moreover, the data distribution can also shape the covariance matrix of the Gaussian prior. Rather than take a spherically symmetric prior distribution we choose the variance in the direction of the prior vector to be $\tau > 1$. As with the prior PAC-Bayes bound the mean of the prior distribution is also shifted from the original in the direction $\mathbf{w}_r$. Seeger (2002) has previously considered non-spherical priors and (different) non-spherical posteriors in bounding Gaussian process classification. Our application to SVMs is not restricted to using specific priors and posteriors so that we have the flexibility to adapt our distributions in order to accommodate the prior derived from the last part of the data.

We introduce notation for the norms of projections for unit vector $\mathbf{u}$, $P_{\mathbf{u}}^{\|}(\mathbf{v}) = \langle \mathbf{u},\mathbf{v}\rangle$ and $P_{\mathbf{u}}^{\perp}(\mathbf{v})^2 = \|\mathbf{v}\|^2 - P_{\mathbf{u}}^{\|}(\mathbf{v})^2$.

Theorem 8 (τ-prior PAC-Bayes bound for linear classifiers) *Let us consider a prior* $P(c|\mathbf{w}_r,\tau,\eta)$ *distribution of classifiers consisting of a Gaussian distribution centred on* $\eta\mathbf{w}_r$*, with identity covariance matrix in all directions except* $\mathbf{w}_r$ *in which the variance is* τ^2*. Then, for all distributions* $\mathcal{D}$*, for all* $\delta \in (0,1]$*, we have that with probability at least* $1-\delta$ *over all the training samples of size* m *drawn from* $\mathcal{D}$*, for all posterior parameters (*$\mathbf{w}$*,* μ*),*

$$KL(\hat{Q}_{S\setminus T}(\mathbf{w},\mu)\|Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq$$

$$\frac{(\ln(\tau^2)+\tau^{-2}-1+P^{\|}_{\mathbf{w}_r}(\mu\mathbf{w}-\eta\mathbf{w}_r)^2/\tau^2+P^{\perp}_{\mathbf{w}_r}(\mu\mathbf{w})^2)+2\ln(\frac{m-r+1}{\delta})}{2(m-r)}.$$

Proof The application of the PAC-Bayes theorem follows that of Langford (2005) except that we must recompute the KL divergence. Using the expression for the KL divergence between two Gaussian distributions of (6) we obtain

$$\begin{aligned}&\mathrm{KL}(Q(\mathbf{w},\mu)\|P(\mathbf{w}_r,\tau,\eta))=\\&\frac{1}{2}\left(\ln(\tau^2)+\left(\frac{1}{\tau^2}-1\right)+\frac{P^{\|}_{\mathbf{w}_r}(\mu\mathbf{w}-\eta\mathbf{w}_r)^2}{\tau^2}+P^{\perp}_{\mathbf{w}_r}(\mu\mathbf{w})^2\right),\end{aligned}$$

and the result follows. ■

Note that the quantity

$$\hat{Q}_{S\setminus T}(\mathbf{w},\mu)=\mathbb{E}_{m-r}[\tilde{F}(\mu\gamma(\mathbf{x},y))]$$

remains unchanged as the posterior distribution is still a spherical Gaussian centred at $\mathbf{w}$.

3.2 Expectation-Prior PAC-Bayes Bound for SVMs

In this section, we attempt to start an interesting new approach on exploiting priors without the aid of a separate data set. The basic idea is to adopt the mathematical expectation of some quantity and then approximate this expectation by an empirical average computed on the available data.

An expectation that may result in reasonable priors is $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[y\phi(\mathbf{x})]$, which is used in the derivation of the bound below. Define $\mathbf{w}_p=\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[y\phi(\mathbf{x})]$ where $y\in\{+1,-1\}$. A special case of $\mathbf{w}_p$ is $\frac{1}{2}(\mathbf{w}^+-\mathbf{w}^-)$ with $\mathbf{w}^+=\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},y=+1}[\phi(\mathbf{x})]$, $\mathbf{w}^-=\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},y=-1}[\phi(\mathbf{x})]$ when each class has the same prior probability. We use its general form in deriving bounds.

Given a sample set S including m examples, the empirical estimate of $\mathbf{w}_p$ would be $\hat{\mathbf{w}}_p=\mathbb{E}_{(\mathbf{x},y)\sim S}[y\phi(\mathbf{x})]=\frac{1}{m}\sum_{i=1}^{m}[y_i\phi(\mathbf{x}_i)]$. We have the following bound.

Theorem 9 (Single-expectation-prior PAC-Bayes bound for SVMs) *For all $\mathcal{D}$, for all Gaussian prior $P\sim\mathcal{N}(\eta\mathbf{w}_p,I)$ over margin classifiers, for all $\delta\in(0,1]$:*

$$\begin{aligned}Pr_{S\sim\mathcal{D}^m}\quad&(\forall\mathbf{w},\mu: KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu))\leq\\&\frac{\frac{1}{2}(\|\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p\|+\eta\frac{R}{\sqrt{m}}(2+\sqrt{2\ln\frac{2}{\delta}}))^2+\ln(\frac{2(m+1)}{\delta})}{m})\geq 1-\delta,\end{aligned}$$

where the posterior is $Q\sim\mathcal{N}(\mu\mathbf{w},I)$ with $R=\sup_{\mathbf{x}}\|\phi(\mathbf{x})\|$.

Proof First, we try to bound $KL(Q||P)$. We have

$$\begin{aligned}KL(Q||P)&=\frac{1}{2}\|\mu\mathbf{w}-\eta\mathbf{w}_p\|^2\\&=\frac{1}{2}\|\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p+\eta\hat{\mathbf{w}}_p-\eta\mathbf{w}_p\|^2\\&=\frac{1}{2}\|\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p\|^2+\frac{1}{2}\|\eta\hat{\mathbf{w}}_p-\eta\mathbf{w}_p\|^2+(\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p)^{\top}(\eta\hat{\mathbf{w}}_p-\eta\mathbf{w}_p)\\&\leq\frac{1}{2}\|\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p\|^2+\frac{1}{2}\eta^2\|\hat{\mathbf{w}}_p-\mathbf{w}_p\|^2+\eta\|\mu\mathbf{w}-\eta\hat{\mathbf{w}}_p\|\|\hat{\mathbf{w}}_p-\mathbf{w}_p\|,\end{aligned}\tag{7}$$

where the last inequality uses Cauchy-Schwarz inequality. Now it suffices to bound $\|\hat{\mathbf{w}}_p - \mathbf{w}_p\|$.

Define $R = \sup_{\mathbf{x}} \|\phi(\mathbf{x})\|$. It is simple to show that $\sup_{(\mathbf{x},y)} \|y\phi(\mathbf{x})\| = \sup_{\mathbf{x}} \|\phi(\mathbf{x})\| = R$. With reference to a result on estimating the center of mass (Shawe-Taylor and Cristianini, 2004), we have

$$Pr\left(\|\hat{\mathbf{w}}_p - \mathbf{w}_p\| \geq \frac{2R}{\sqrt{m}} + \varepsilon\right) \leq \exp\left(-\frac{2m\varepsilon^2}{4R^2}\right).$$

Setting the right hand side equal to $\delta/2$, solving for ε shows that with probability at least $1-\delta/2$, we have

$$\|\hat{\mathbf{w}}_p - \mathbf{w}_p\| \leq \frac{R}{\sqrt{m}}\left(2 + \sqrt{2\ln\frac{2}{\delta}}\right). \tag{8}$$

Define $b = \frac{R}{\sqrt{m}}\left(2+\sqrt{2\ln\frac{2}{\delta}}\right)$, we have

$$Pr_{S\sim\mathcal{D}^m}\left(KL(Q||P) \leq \frac{1}{2}\|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|^2 + \frac{1}{2}\eta^2 b^2 + \eta b\|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|\right) \geq 1-\delta/2. \tag{9}$$

Then, according to Theorem 3, we have

$$Pr_{S\sim\mathcal{D}^m}\left(\forall Q(c) : KL_+(\hat{Q}_S||Q_{\mathcal{D}}) \leq \frac{KL(Q||P) + \ln(\frac{2(m+1)}{\delta})}{m}\right) \geq 1-\delta/2. \tag{10}$$

Define $a = \|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|$. Combining (9) and (10), we get

$$Pr_{S\sim\mathcal{D}^m}\left(\forall \mathbf{w},\mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \frac{\frac{1}{2}a^2 + \frac{1}{2}\eta^2 b^2 + \eta ab + \ln(\frac{2(m+1)}{\delta})}{m}\right) \geq 1-\delta,$$

where we used $(1-\delta/2)^2 > 1-\delta$. Rewriting the bound as

$$Pr_{S\sim\mathcal{D}^m}\left(\forall \mathbf{w},\mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \frac{\frac{1}{2}(a+\eta b)^2 + \ln(\frac{2(m+1)}{\delta})}{m}\right) \geq 1-\delta$$

completes the proof. ■

Considering at the same time Theorem 9 and the mixture-prior PAC-Bayes bound, it is not difficult to reach the following mixture-expectation-prior PAC-Bayes bound for SVMs.

Theorem 10 (Mixture-expectation-prior PAC-Bayes bound for SVMs) *For all $\mathcal{D}$, for all mixtures of Gaussian prior $\mathcal{P}(c) = \sum_{j=1}^{J}\pi_j P_j(c)$ where $P_j \sim \mathcal{N}(\eta_j\mathbf{w}_p, I)$ $(j=1,\ldots,J)$, $\pi_j \geq 0$ and $\sum_{j=1}^{J}\pi_j = 1$ over margin classifiers, for all $\delta \in (0,1]$:*

$$Pr_{S\sim\mathcal{D}^m}\quad\left(\forall \mathbf{w},\mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \min_j \frac{\frac{1}{2}(\|\mu\mathbf{w} - \eta_j\hat{\mathbf{w}}_p\| + \eta_j\frac{R}{\sqrt{m}}(2+\sqrt{2\ln\frac{2}{\delta}}))^2 + \ln(\frac{2(m+1)}{\delta}) + \ln\frac{1}{\pi_j}}{m}\right) \geq 1-\delta,$$

where the posterior is $Q \sim \mathcal{N}(\mu\mathbf{w}, I)$ with $R = \sup_{\mathbf{x}}\|\phi(\mathbf{x})\|$. If we consider equiprobable members in the mixture, then $\ln\frac{1}{\pi_j} = \ln J$.

Moreover, the expectation prior bound can also be extended to the case where the shape of the covariance matrix of the prior is also determined from the training data:

Theorem 11 (τ-Expectation-prior PAC-Bayes bound) *Consider a prior distribution $P \sim \mathcal{N}(\eta \mathbf{w}_p, I, \tau^2)$ of classifiers consisting of a Gaussian distribution centred on $\eta \mathbf{w}_p$, with identity covariance in all directions except $\mathbf{w}_p$ in which the variance is τ^2. Then, for all distributions $\mathcal{D}$, for all $\delta \in (0,1]$, we have*

$$Pr_{S\sim\mathcal{D}^m} \quad (\forall \mathbf{w}, \mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \frac{\frac{1}{2}(\ln(\tau^2) + \frac{(\|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\| + \eta\frac{R}{\sqrt{m}}(2+\sqrt{2\ln\frac{2}{\delta}}))^2 - \mu^2 + 1}{\tau^2} + \mu^2 - 1) + \ln(\frac{2(m+1)}{\delta})}{m}) \geq 1 - \delta,$$

where the posterior is $Q \sim \mathcal{N}(\mu\mathbf{w}, I)$ with $R = \sup_{\mathbf{x}} \|\phi(\mathbf{x})\|$. We can recover Theorem 9 by taking $\tau = 1$.

Proof According to Theorem 8,

$$KL(Q||P) = \frac{1}{2}\left(\ln(\tau^2) + \frac{1}{\tau^2} - 1 + \frac{P^{\|}_{\mathbf{w}_p^*}(\mu\mathbf{w} - \eta\mathbf{w}_p)^2}{\tau^2} + P^{\perp}_{\mathbf{w}_p^*}(\mu\mathbf{w})^2\right),$$

where $\mathbf{w}_p^* = \mathbf{w}_p / \|\mathbf{w}_p\|$. The last two quantities can be rewritten as

$$\begin{aligned}
\frac{P^{\|}_{\mathbf{w}_p^*}(\mu\mathbf{w} - \eta\mathbf{w}_p)^2}{\tau^2} + P^{\perp}_{\mathbf{w}_p^*}(\mu\mathbf{w})^2 &= \frac{1}{\tau^2}(\frac{\mathbf{w}_p^\top}{\|\mathbf{w}_p\|}(\mu\mathbf{w} - \eta\mathbf{w}_p))^2 + \|\mu\mathbf{w}\|^2 - (\frac{\mathbf{w}_p^\top}{\|\mathbf{w}_p\|}\mu\mathbf{w})^2 \\
&= \frac{1}{\tau^2}(\frac{\mathbf{w}_p^\top}{\|\mathbf{w}_p\|}\mu\mathbf{w} - \eta\|\mathbf{w}_p\|)^2 + \|\mu\mathbf{w}\|^2 - (\frac{\mathbf{w}_p^\top}{\|\mathbf{w}_p\|}\mu\mathbf{w})^2 \\
&= \frac{1}{\tau^2}(\eta^2\|\mathbf{w}_p\|^2 - 2\eta\mathbf{w}_p^\top\mu\mathbf{w}) + \|\mu\mathbf{w}\|^2 \\
&= \frac{1}{\tau^2}(\|\mu\mathbf{w} - \eta\mathbf{w}_p\|^2 - \|\mu\mathbf{w}\|^2) + \|\mu\mathbf{w}\|^2 \\
&= \frac{1}{\tau^2}(\|\mu\mathbf{w} - \eta\mathbf{w}_p\|^2 - \mu^2) + \mu^2 .
\end{aligned}$$

By Equation (7), we have

$$\|\mu\mathbf{w} - \eta\mathbf{w}_p\|^2 \leq \|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|^2 + \eta^2\|\hat{\mathbf{w}}_p - \mathbf{w}_p\|^2 + 2\eta\|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|\|\hat{\mathbf{w}}_p - \mathbf{w}_p\| .$$

By Equation (8), we have with probability at least $1 - \delta/2$

$$\|\hat{\mathbf{w}}_p - \mathbf{w}_p\| \leq \frac{R}{\sqrt{m}}\left(2 + \sqrt{2\ln\frac{2}{\delta}}\right).$$

With $a = \|\mu\mathbf{w} - \eta\hat{\mathbf{w}}_p\|$ and $b = \frac{R}{\sqrt{m}}\left(2 + \sqrt{2\ln\frac{2}{\delta}}\right)$, we have

$$Pr_{S\sim\mathcal{D}^m}\left(KL(Q||P) \leq \frac{1}{2}(\ln(\tau^2) + \frac{1}{\tau^2} - 1 + \frac{a^2 + \eta^2 b^2 + 2\eta ab - \mu^2}{\tau^2} + \mu^2)\right) \geq 1 - \delta/2. \qquad (11)$$

Then, according to Theorem 3, we have

$$Pr_{S\sim\mathcal{D}^m}(\forall Q(c) : KL_+(\hat{Q}_S||Q_{\mathcal{D}}) \leq \frac{KL(Q||P)+\ln(\frac{2(m+1)}{\delta})}{m}) \geq 1-\delta/2. \tag{12}$$

Combining (11) and (12) results in

$$\begin{aligned} Pr_{S\sim\mathcal{D}^m} \quad & (\forall \mathbf{w},\mu : KL_+(\hat{Q}_S(\mathbf{w},\mu)||Q_{\mathcal{D}}(\mathbf{w},\mu)) \leq \\ & \frac{\frac{1}{2}(\ln(\tau^2)+\frac{(a+\eta b)^2-\mu^2+1}{\tau^2}+\mu^2-1)+\ln(\frac{2(m+1)}{\delta})}{m}) \geq 1-\delta, \end{aligned}$$

which completes the proof. ■

4. Optimising the Prior PAC-Bayes Bound in the Design of the Classifier

Up to this point we have introduced the prior PAC-Bayes bounds as a means to tighten the original PAC-Bayes bound (this fact is illustrated in the experiments included in Section 5). The next contribution of this paper consists of the introduction of the optimisation of the prior PAC-Bayes bound into the design of the classifier. The intuition behind this use of the bounds is that classifiers reporting low values for the bound should yield a good generalization capability.

4.1 Prior SVM

The new philosophy is implemented in the prior SVM by replacing the maximization of the margin in the optimization problem defining the original SVM with a term that pushes towards the tightening of the prior PAC-Bayes bound. This subsection introduces the formulation of the new algorithm, a method to determine the classifier by means of off-the-shelf quadratic programming solvers, and a procedure to compute the prior PAC-Bayes bound for these new classifiers.

4.1.1 FORMULATION OF THE PRIOR SVMS

As stated before, the design criterion for the prior SVMs involves the minimization of the prior PAC-Bayes bound. Let us consider the simplest case of the bound, that is, a single prior centered on $\eta\mathbf{w}_r$, where $\mathbf{w}_r$ is the unit vector weight of the SVM constructed with r training samples and η is a scalar fixed a priori. For simplicity, we assume these r samples are the last ones in the training set $\{(\mathbf{x}_l, y_l)\}_{l=m-r+1}^{m}$. Therefore, $\mathbf{w}_r$ can be expressed in terms of these input patterns as:

$$\mathbf{w}_r = \frac{\sum_{l=m-r+1}^{m} y_l \alpha_l \phi(\mathbf{x}_l)}{\left\|\sum_{l=m-r+1}^{m} y_l \alpha_l \phi(\mathbf{x}_l)\right\|}.$$

In such a case, a small bound on the error of the classifier is the result of a small value of $\|\eta\mathbf{w}_r - \mu\mathbf{w}\|^2$, and a large value of the normalized margin of Equation (3) for the remaining training examples $\gamma(\mathbf{x}_i, y_i)$, $i = 1,\ldots,m-r$.

We start by addressing the separable case. Under perfect separability conditions, a good strategy to obtain a classifier of minimal bound is to solve the following optimization problem:

$$\min_{\mathbf{w}} \left[\frac{1}{2}\|\mathbf{w}-\eta\mathbf{w}_r\|^2\right] \tag{13}$$

subject to

$$y_i \mathbf{w}^T \phi(\mathbf{x}_i) \geq 1 \quad i = 1, \ldots, m-r. \tag{14}$$

Clearly, the objective function of (13) attempts to reduce the value of the right hand side of the bound, while the constraints in (14) that impose the separability of the classes lead to a small $\hat{Q}_S$.

Once $\mathbf{w}$ is found through the solution of (13) with constraints (14) the proper bound on the average true error of the prior SVM can be obtained by means of a further tuning of μ (that is, using $\mu\mathbf{w}$ instead of $\mathbf{w}$ as mean of the posterior distribution), where this last tuning will not change $\mathbf{w}$.

The extension of the prior SVM to the non-separable case is easily carried out through the introduction of positive slack variables $\{\xi_i\}_{i=1}^{m-r}$. Then the optimization problem becomes

$$\min_{\mathbf{w},\xi_i} \left[\frac{1}{2} \|\mathbf{w} - \mathbf{w}_r\|^2 + C \sum_{i=1}^{m-r} \xi_i \right] \tag{15}$$

subject to

$$y_i \mathbf{w}^T \phi(\mathbf{x}_i) \geq 1 - \xi_i \qquad i = 1, \ldots, m-r, \tag{16}$$

$$\xi_i \geq 0 \qquad i = 1, \ldots, m-r. \tag{17}$$

Note that the constraints in (16) also push towards the minimization of the stochastic error $\hat{Q}_S$. In this sense, for a sample $\mathbf{x}$ on the wrong side of the margin we have $\xi = 1 - y\mathbf{w}^T\phi(\mathbf{x}) > 1$, which leads to a margin $\gamma < 0$ and thus an increase in $\hat{Q}_S$ (see Equations (2) to (4)). Therefore, by penalizing ξ we enforce a small $\hat{Q}_S$.

Furthermore, Corollary 7 allows us to use a mixture of J distributions instead of one at the cheap cost of $\frac{\ln J}{m}$. This can be used to refine the selection of the weight vector of the prior SVMs through the following procedure:

1. First we determine a unit $\mathbf{w}_r$ with samples $\{(\mathbf{x}_l, y_l)\}_{l=m-r+1}^{m}$. Then we construct a mixture prior with J Gaussian components with identity covariance matrices centered at $\eta_j \mathbf{w}_r$, with η_j being J real positive constants.

2. For every element in the mixture we obtain a prior SVM classifier $\mathbf{w}^j$ solving

$$\min_{\mathbf{w}^j,\xi_i} \left[\frac{1}{2} \|\mathbf{w}^j - \eta_j \mathbf{w}_r\|^2 + C \sum_{i=1}^{m-r} \xi_i \right]$$

   subject to

$$y_i \phi(\mathbf{x}_i)^T w^j \geq 1 - \xi_i \quad i = 1, \ldots, m-r,$$

$$\xi_i \geq 0 \quad i = 1, \ldots, m-r.$$

   Afterwards, we obtain the bounds $Q_{\mathcal{D}}^j$ corresponding to the average true error of each one of the J prior SVMs by tuning μ (see Corollary 6).

3. We finally select as the prior SVM the $\mathbf{w}^j$ that reports the lowest bound $Q_{\mathcal{D}}^j$.

It should be pointed out that each prior scaling (η_j) that is tried increases the computational burden of the training of the prior SVMs by an amount corresponding to an SVM problem with $m-r$ data points.

Appendix A details a procedure to determine the solution $\mathbf{w}$ to the optimization problem given by (15) and constraints (16) and (17) based on the usual derivation of the SVM.

4.1.2 COMPUTING THE PAC-BAYES BOUND FOR THE PRIOR SVMS

The remainder of the section presents a method to compute the PAC-Bayes bound for a prior SVM obtained through the procedure described above. To simplify notation we have introduced the nonunit weight vector $\mathbf{w}_{m-r} = \mathbf{w} - \eta \mathbf{w}_r$, that includes the posterior part of the prior SVM. The bound is based on the relationship between two distributions of classifiers: the prior $P(\mathbf{w}_r, \eta) \sim \mathcal{N}(\eta \mathbf{w}_r, I)$ and the posterior $Q(\mathbf{w}, \mu) \sim \mathcal{N}(\mu \mathbf{w}, I)$.

The stochastic error $\hat{Q}_S$ in the left hand side of the bound can be straightforwardly obtained by using a unit $\mathbf{w}$ in (27) in Equations (2) to (4). For the right hand side of the bound, we need to compute $\mathrm{KL}(Q(\mathbf{w}, \mu)||P(\mathbf{w}_r, \eta)) = \frac{\|\eta \mathbf{w}_r - \mu \mathbf{w}\|^2}{2}$ which can be rewritten as

$$\mathrm{KL}(Q(\mathbf{w}, \mu)||P(\mathbf{w}_r, \eta)) = \frac{1}{2}\left(\mu^2 + \eta^2 - 2\mu\eta(\eta + \mathbf{w}_{m-r}^T \mathbf{w}_r)\right).$$

4.2 η-Prior SVM

When the prior SVM is learnt within a mixture priors setting, the last stage of the optimization is the selection of the best prior-component/posterior pair, among the J possibilities. These prior-component/posterior pairs are denoted by $(\eta_j, \mathbf{w}_j)$, where η_j is the jth scaling of the normalized prior $\mathbf{w}_r$. From the point of view of the prior, this selection process can be regarded as a search over the set of scalings using the mixture-prior PAC-Bayes bound as fitness function. Note that the evaluation of such a fitness function involves learning the posterior and the tuning of μ.

The idea presented in this section actually consists of two turns of the screw. First, the search in the discrete set of priors is cast as a linear search for the optimal scaling η in a continuous range of scalings $[\eta_1, \eta_J]$. Second, this linear search is introduced into the optimization of the posterior. Therefore, instead of optimizing a posterior for every scaling of the prior, the optimal scaling and posterior given a normalized prior are the output of the same optimization problem.

The sequel is devoted to the derivation of the resulting algorithm, called the η-prior SVMs, and to its analysis using the prior PAC-Bayes bound framework.

4.2.1 DERIVATION OF THE η-PRIOR SVMS

The η-prior SVM is designed to solve the following problem:

$$\min_{\mathbf{v}, \eta, \xi_i} \left[\frac{1}{2}\|\mathbf{v}\|^2 + C \sum_{i=1}^{m-r} \xi_i \right]$$

subject to

$$\begin{aligned} y_i(\mathbf{v} + \eta \mathbf{w}_r)^T \phi(\mathbf{x}_i) &\geq 1 - \xi_i \qquad i = 1, \ldots, m-r, \\ \xi_i &\geq 0 \qquad i = 1, \ldots, m-r. \end{aligned}$$

The final (unit vector) classifier will be

$$\mathbf{w} = (\mathbf{v} + \eta \mathbf{w}_r)/\|\mathbf{v} + \eta \mathbf{w}_r\|.$$

After a derivation analogous to that presented in Appendix A, we arrive at the following quadratic program

$$\max_{\alpha_i} \sum_{i=1}^{m-r} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m-r} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

subject to

$$\sum_{i=1}^{m-r}\sum_{k=m-r+1}^{m}\alpha_i y_i \tilde{\alpha}_k y_k \kappa(\mathbf{x}_i,\mathbf{x}_k) = \sum_{i=1}^{m-r} y_i \alpha_i g_i = 0 \qquad i = 1,\ldots,m-r,$$
$$0 \le \alpha_i \le C \qquad i = 1,\ldots,m-r,$$

where $g_i = \sum_{k=m-r+1}^{m} \tilde{\alpha}_k y_k \kappa(\mathbf{x}_i,\mathbf{x}_k)$ and $\tilde{\alpha}_k$ are the normalized dual variables for the prior learnt from the last r samples, $\{\mathbf{x}_k\}_{k=m-r+1}^{m}$. Once we have solved for α_i, we can compute η by considering some j such that $0 < \alpha_j < C$ and using the equation

$$y_j\left(\sum_{i=1}^{m-r}\alpha_i y_i \kappa(\mathbf{x}_i,\mathbf{x}_j) + \eta g_j\right) = 1.$$

4.2.2 BOUNDS FOR η-PRIOR SVMS

The statistical analysis of the η-prior SVMs can be performed using the τ-prior PAC-Bayes bound of Theorem 8, and τ-expectation prior PAC-Bayes bound. Rather than take a spherically symmetric prior distribution we choose the variance in the direction of the prior vector to be $\tau^2 > 1$. As with the prior SVM analysis the mean of the prior distribution is also shifted from the origin in the direction $\mathbf{w}_r$.

In order to apply the bound we need to consider the range of priors that are needed to cover the data in our application. The experiments conducted in the next section require a range of scalings of $\mathbf{w}_r$ from 1 to 100. For this we can choose $\eta = 50$, $\tau = 50$, and $\mu \le 100$ in all but one of our experiments, giving an increase in the bound over the factor $P_{\mathbf{w}_r}^{\perp}(\mu\mathbf{w})^2$ directly optimized in the algorithm of

$$\frac{\ln(\tau^2) + \tau^{-2} - 1 + P_{\mathbf{w}_r}^{\parallel}(\mu\mathbf{w} - \eta\mathbf{w}_r)^2/\tau^2}{2(m-r)} \le \frac{\ln(\tau) + 0.5\tau^{-2}}{m-r} \approx \frac{3.912}{m-r}. \tag{18}$$

We include Equation (18) to justify that our algorithm optimises a quantity that is very close to the expression in the bound. Note that the evaluation of the bounds presented in the experimental section are computed using the expression from Theorem 8 and not this approximate upper bound. One could envisage making a sequence of applications of the PAC-Bayes bound with spherical priors using the union bound and applying the result with the nearest prior. This strategy leads to a slightly worse bound as it fails to take into account the correlations between the different priors. This fact is illustrated in Section 5.

5. Experiments

This section is devoted to an experimental analysis of the bounds and algorithms introduced in the paper. The comparison of the algorithms is carried out on classification preceded by model selection tasks using some UCI (Blake and Merz, 1998) data sets (see their description in terms of number of instances, input dimensions and numbers of positive/negative examples in Table 1).

5.1 Experimental Setup

For every data set, we prepared 50 different training/test set partitions where 80% of the samples form the training set and the remaining 20% form the test set. From every training set we considered

Problem	# Examples	Input Dim.	Pos/Neg
Handwritten-digits (han)	5620	64	2791 / 2829
Waveform (wav)	5000	21	1647 / 3353
Pima (pim)	768	8	268 / 500
Ringnorm (rin)	7400	20	3664 / 3736
Spam (spa)	4601	57	1813 / 2788

Table 1: Description of data sets in terms of number of examples, number of input variables and number of positive/negative examples.

subsets with 20%, 30%, ..., 100% of the training patterns, in order to analyse the dependence of the bounds with the number of samples used to train the classifier. Note that all the training subsets from the same partition share the same test set.

With each of the training sets we learned a classifier with Gaussian RBF kernels preceded by a model selection. The model selection consists in the determination of an optimal pair of hyperparameters (C, σ). C is the SVM trade-off between the maximization of the margin and the minimization of the number of misclassified training samples; σ is the width of the Gaussian kernel, $\kappa(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|^2/(2\sigma^2))$. The best pair is sought in a 7×5 grid of parameters where $C \in \{0.01, 0.1, 1, 10, 100, 1000, 10000\}$ and $\sigma \in \{\frac{1}{4}\sqrt{d}, \frac{1}{2}\sqrt{d}, \sqrt{d}, 2\sqrt{d}, 4\sqrt{d}\}$, d being the input space dimension.

With respect to the parameters needed by the prior PAC-Bayes bounds, the number of priors J and the amount of patterns separated to learn the prior, the experiments reported by Ambroladze et al. (2007) suggest that $J = 10$ and $r = 50\%$ of the training set size lead to reasonable results.

The setup to calculate the bound values displayed in the next tables was as follows. We trained an instance of the corresponding classifier for each position of the grid of hyperparameters and compute the bound. We selected for that type of classifier the minimum value of the bound found through the whole grid. Then we averaged the 50 values of the bound corresponding to each of the training/testing partitions. We completed the average with the sample standard deviation. Note that proceeding this way we select a (possibly) different pair of hyperparameters for each of the 50 partitions. That is the reason why we name this task model selection plus classification.

The test error rates are computed after the following procedure. For each one of the training/test partitions we carried out the model selection described in the previous paragraph and selected the classifier of minimum bound. We classified the test set with this classifier and obtain the test error rate for those particular classifier and partition. Then we averaged the 50 test error rates to yield the test error rate for those particular data set, model selection method and type of classifier. Note again that the model selection has a significant impact on the reported test error rates.

Moreover, the reported values of the PAC-Bayes and the mixture-prior PAC-Bayes bounds correspond to the mean of the true error over the distribution of classifiers Q_D. The real true error c_D could then be bounded by twice this value (see Equation (5)). In all the experiments the bounds are obtained using a confidence of $\delta = 0.01$.

5.2 Results and Discussion

The section starts presenting an analysis of the performance of SVM with the prior PAC Bayes bounds introduced in this paper. We show how in most cases the use of an informative prior leads to a significant tightening of the bounds on the true error of the classifier. The analysis is then extended towards the new algorithms prior SVM and η-prior SVM. We show how their true error is predicted more accurately by the prior PAC Bayes bound. The observed test errors achieved by these algorithms are comparable to those obtained by SVMs with their hyperparameters fixed through ten fold cross validation. Finally, the prior SVM framework enables the use of a different value of parameter C for prior and posterior, that can be tuned using the prior PAC Bayes bound. The experiments show that the use of different values of C contributes to get even tighter lower bounds.

5.2.1 ANALYSIS OF THE SVM WITH THE PRIOR PAC BAYES BOUNDS

The first set of experiments is devoted to illustrate how tight can be the predictions about the generalisation capabilities of a regular SVM based upon the prior PAC-Bayes bounds. Thus, we have trained SVM using the hyperparameters that arrived at a minimum value of each of the following bounds:

PAC Bayes: the model selection is driven by the PAC Bayes bound of Langford (2005).

Prior PB: model selection driven by the mixture-prior PAC-Bayes bound of Corollary 7 with $J = 10$.

τ-prior PB: τ-prior PAC-Bayes bound of Theorem 8 with $J = 10$ and $\tau = 50$.

$\mathbb{E}$ prior PB: expectation-prior PAC-Bayes bound of Theorem 10.

τ-$\mathbb{E}$ prior PB: τ-expectation prior PAC-Bayes bound of Theorem 11.

Plots in Figure 1 show the performance of the different bounds as a function of the training set size. All the bounds achieve non trivial results even for training set sizes as small as 16% of the complete data set (20% of the training set). In most of the cases, the bounds with an informative prior are tighter than the original PAC Bayes bound with an spherical prior centred on the origin. The expectation prior is significantly better in data sets `wav` and `pim`, whilst the prior PAC Bayes and the τ-prior PAC Bayes are the tighter in problems `rin` and `spa`. Table 2 shows the values of the bounds when the SVM is determined using the 100% of the training set (80% of the data).

Moreover, an examination of the slopes of the plots corresponding to the bounds point out that those that learn the prior from a separate training set do converge faster than the original PAC Bayes and the expectation prior PAC Bayes bounds. Since the former present a $m - r$ in the denominator of the right hand side, one could a priori think that their convergence would be slower than that of the latter, with an m in the denominator. However, the experimental results show that it is better to devote those separate training patterns to acquire a more informative prior than to increase the weight of the denominator in the penalty term.

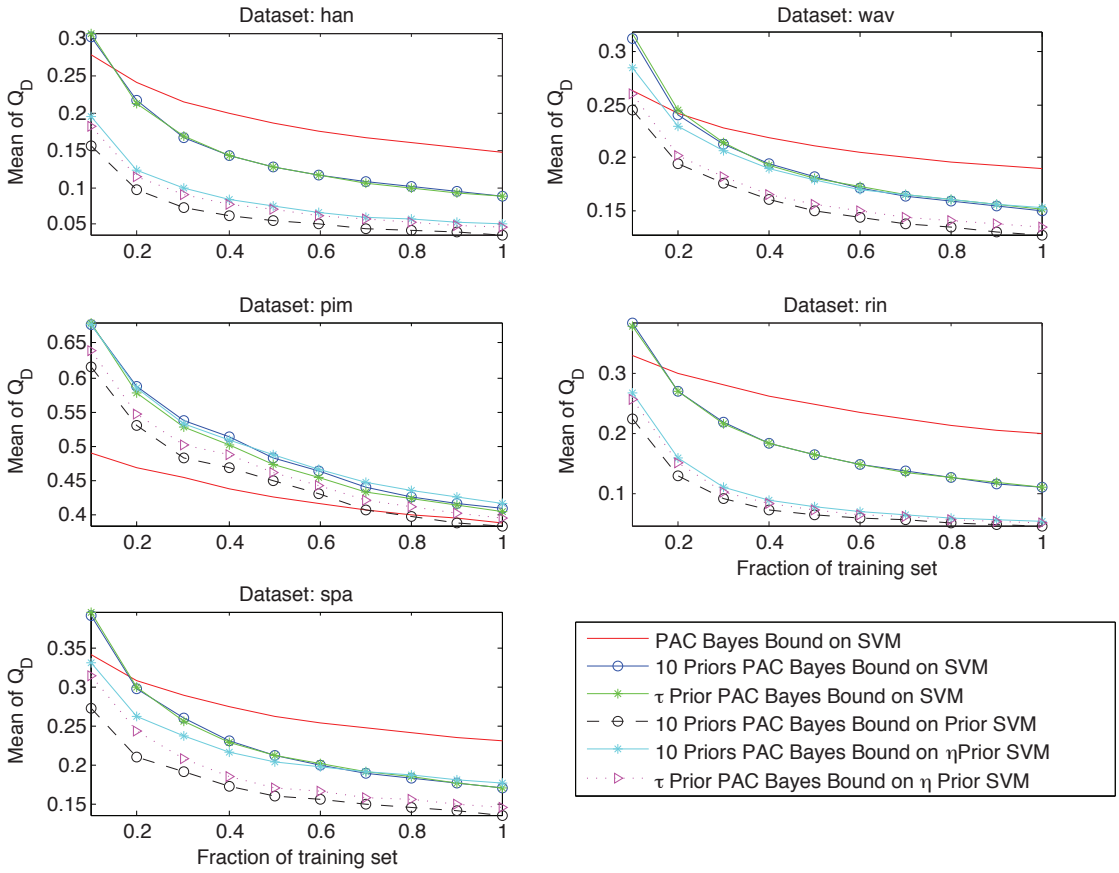


Figure 1: Analysis of SVM with data dependent prior PAC Bayes bounds.

	Data Set				
Bound	han	wav	pim	rin	spa
PAC Bayes	0.148 ± 0.000	0.190 ± 0.000	0.390 ± 0.001	0.198 ± 0.000	0.230 ± 0.000
Prior PB	$\underline{0.088 \pm 0.004}$	0.151 ± 0.004	0.411 ± 0.015	$\underline{0.110 \pm 0.004}$	$\underline{0.171 \pm 0.005}$
τ Prior PB	$\underline{0.088 \pm 0.004}$	0.152 ± 0.004	0.406 ± 0.013	$\underline{0.110 \pm 0.004}$	$\underline{0.172 \pm 0.006}$
$\mathbb{E}$ Prior PB	0.107 ± 0.001	$\underline{0.133 \pm 0.001}$	$\underline{0.352 \pm 0.004}$	0.194 ± 0.000	0.221 ± 0.001
τ$\mathbb{E}$ Prior PB	0.149 ± 0.000	0.191 ± 0.000	0.401 ± 0.001	0.199 ± 0.000	0.232 ± 0.000

Table 2: Values of the bounds for SVM.

5.2.2 ANALYSIS OF PRIOR SVM AND η-PRIOR SVM

We repeated the study on the new algorithms, prior SVM and η-prior SVM, which are designed to actually optimise prior PAC-Bayes bounds. The configurations classifier-bound considered for this study were the following:

prior SVM + Prior PB: prior SVM described in page 14 and mixture-prior PAC-Bayes bound of Corollary 7 with $J = 10$ priors .

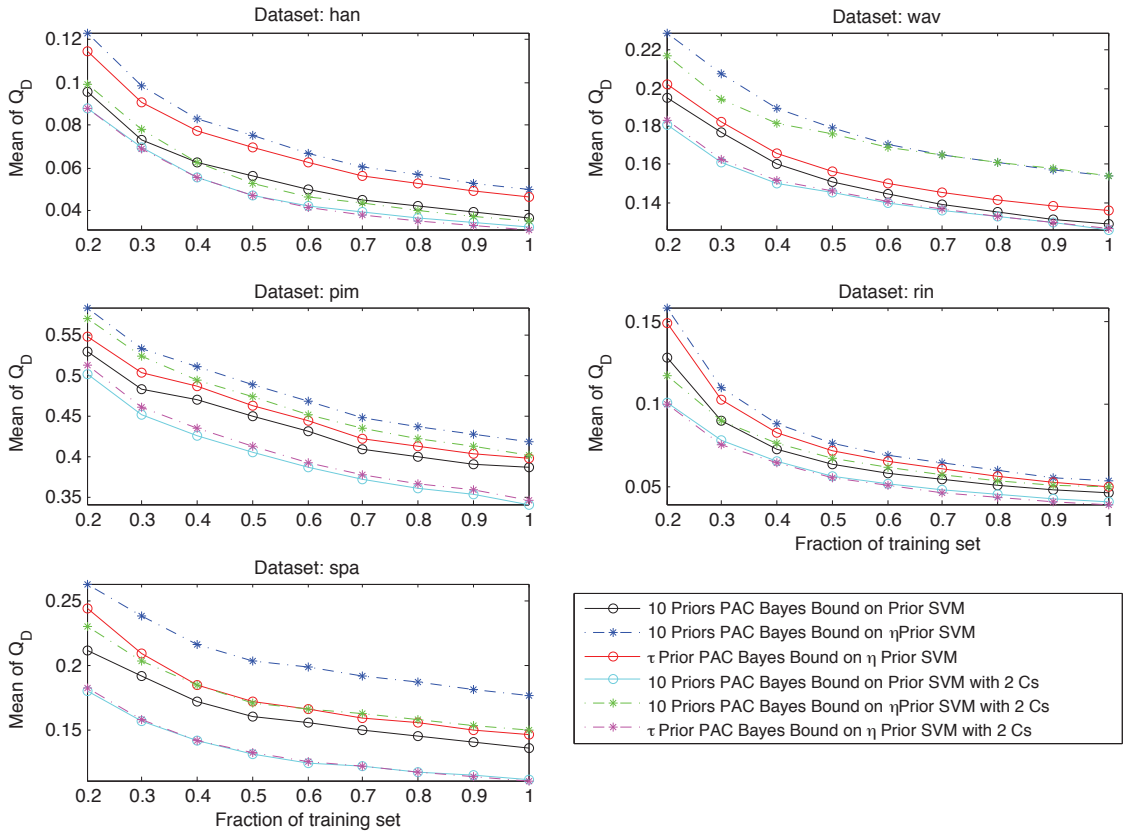


Figure 2: Bounds learning a prior classifier.

η-PSVM + Prior PB: η-prior SVM of Section 4.2.1 and mixture-prior PAC-Bayes bound of Corollary 7 considering η comes from a mixture prior setting of $J = 50$ components $\eta_j \mathbf{w}_r$ with the η_j equally spaced between $\eta_1 = 1$ and $\eta_{50} = 100$. This setting minimizes the penalty term in the prior PAC-Bayes bound as we are not actually using these components to learn the posterior.

η-PSVM + τ-Prior PB: η-prior SVM and the bound in Theorem 8.

As baseline results we include the better bounds found in the analysis of the SVM:

τ-Prior PB: τ prior PAC-Bayes bound of Theorem 8 with $J = 10$ and $\tau = 50$.

$\mathbb{E}$ **Prior PB:** expectation-prior PAC-Bayes bound of Theorem 10.

The plots in Figure 2 show the bounds on the true error, $Q_{\mathcal{D}}$, for the studied configurations bound/classifier as a function of the size of the training set. Table 3 shows these results for a training set of 80% of the complete data. In general, the bounds achieved on prior SVM and η-prior SVM are significantly tighter than the bounds on the SVM, being the mixture-prior PAC Bayes bound on prior SVM the tightest result.

	Data Set				
Bound	han	wav	pim	rin	spa
			Prior SVM		
Prior PB	$\underline{0.037 \pm 0.004}$	$\underline{0.128 \pm 0.004}$	0.386 ± 0.016	$\underline{0.046 \pm 0.003}$	$\underline{0.137 \pm 0.005}$
			η-Prior SVM		
Prior PB	0.050 ± 0.006	0.154 ± 0.004	0.419 ± 0.014	0.053 ± 0.004	0.177 ± 0.006
τ Prior PB	0.047 ± 0.005	0.135 ± 0.004	0.397 ± 0.014	0.050 ± 0.004	0.147 ± 0.006
			SVM		
τ Prior PB	0.088 ± 0.004	0.152 ± 0.004	0.406 ± 0.013	0.110 ± 0.004	0.172 ± 0.006
$\mathbb{E}$ Prior PB	0.107 ± 0.001	0.133 ± 0.001	$\underline{0.352 \pm 0.004}$	0.194 ± 0.000	0.221 ± 0.001

Table 3: Values of the bounds on the prior SVM and η-prior SVM classifiers.

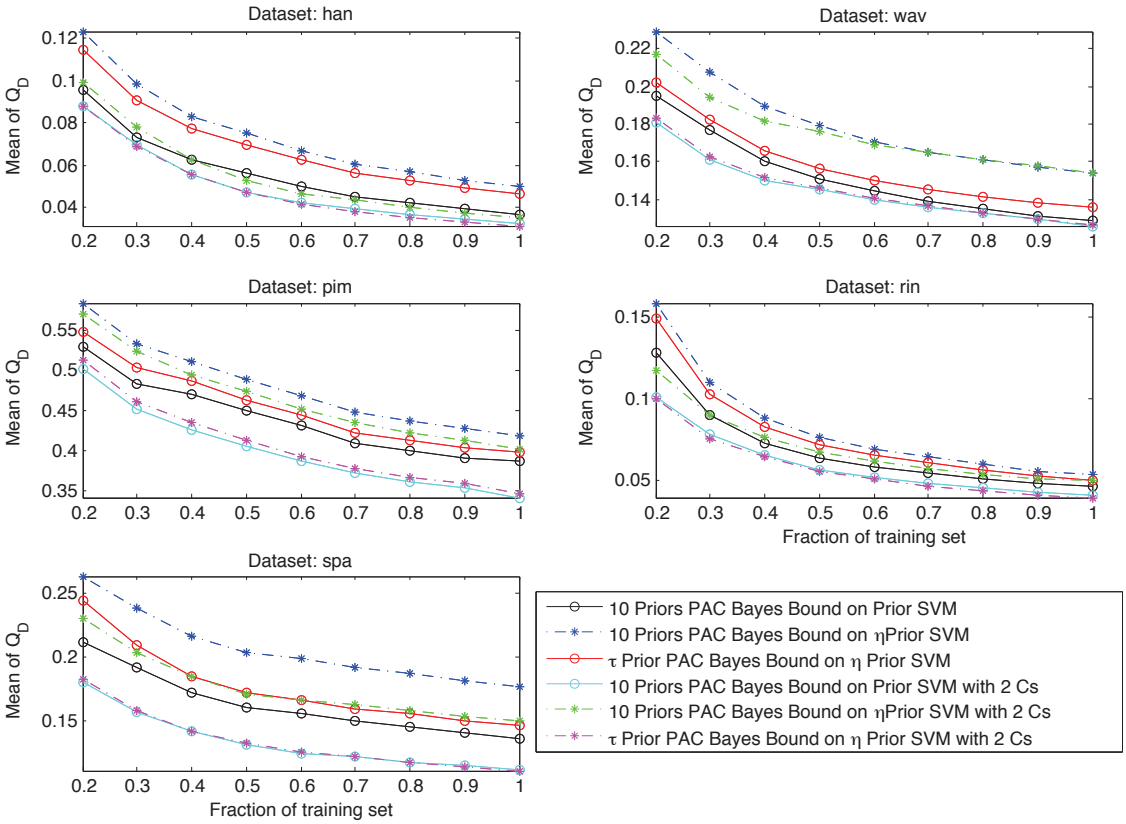


Figure 3: Bounds when prior and posterior have a different value of C.

Notice that in most of the configurations where the prior is learnt from a separate set the new bounds achieve a significant cut in the value of the PAC-Bayes bound, which indicates that learning an informative prior distribution helps to tighten the PAC-Bayes bound.

Furthermore, the two stages training of prior SVM and η-prior SVM enable the use of a different value of C for the prior and posterior classifiers. The intuition behind this proposal is that once the prior is fixed, the posterior could possibly accept a higher value C without overfitting.

	Data Set				
Bound	han	wav	pim	rin	spa
Prior SVM					
Prior PB	0.037 ± 0.004	0.128 ± 0.004	0.386 ± 0.016	0.046 ± 0.003	0.137 ± 0.005
Prior PB 2C	0.033 ± 0.002	0.126 ± 0.004	0.341 ± 0.019	0.041 ± 0.002	0.113 ± 0.004
η-Prior SVM					
Prior PB	0.050 ± 0.006	0.154 ± 0.004	0.419 ± 0.014	0.053 ± 0.004	0.177 ± 0.006
Prior PB 2C	0.035 ± 0.003	0.154 ± 0.004	0.401 ± 0.018	0.049 ± 0.003	0.150 ± 0.005
τ Prior PB	0.047 ± 0.005	0.135 ± 0.004	0.397 ± 0.014	0.050 ± 0.004	0.147 ± 0.006
τ Prior PB 2C	0.031 ± 0.002	0.126 ± 0.004	0.345 ± 0.019	0.039 ± 0.002	0.111 ± 0.005

Table 4: Values of the bounds on the prior SVM and η-prior SVM classifiers when different values of C are used for prior and posterior.

	Data Set				
Bound	han	wav	pim	rin	spa
Prior SVM					
Prior PB	0.010 ± 0.004	0.086 ± 0.007	0.246 ± 0.034	0.016 ± 0.003	0.082 ± 0.009
Prior PB 2C	0.011 ± 0.003	0.091 ± 0.009	0.251 ± 0.038	0.017 ± 0.003	0.069 ± 0.007
η-Prior SVM					
Prior PB	0.010 ± 0.005	0.086 ± 0.006	0.236 ± 0.028	0.016 ± 0.003	0.080 ± 0.009
Prior PB 2C	0.011 ± 0.003	0.087 ± 0.009	0.242 ± 0.039	0.018 ± 0.003	0.068 ± 0.008
τ Prior PB	0.010 ± 0.005	0.085 ± 0.006	0.238 ± 0.028	0.016 ± 0.003	0.080 ± 0.009
τ Prior PB 2C	0.011 ± 0.003	0.092 ± 0.010	0.248 ± 0.042	0.018 ± 0.003	0.070 ± 0.007
SVM					
10 FCV	0.008 ± 0.003	0.087 ± 0.007	0.251 ± 0.023	0.016 ± 0.003	0.067 ± 0.006

Table 5: Test error rates achieved by prior SVM and η-prior SVM classifiers when the hyperparameters are those that minimise a PAC Bayes bound. Prior and posterior are allowed to use a different value of the hyperparameter C.

To evaluate the goodness of this modification, we carried out again the experiments in this subsection but now allowing the prior and posterior to take different values of C from within the range proposed at the beginning of the section. The results displayed in Figure 3 and Table 4 show that the introduction of a different C significantly reduces the value of the bound.

Finally, Table 5 gives some insight about the performance of the new algorithms in terms of observed test error. The joint analysis of the bounds and the error rates on a separate test set shows that the prior PAC Bayes bounds are achieving predictions on the true error very close to the empirical estimations; as an example, for data set `wav` the bound on $Q_{\mathcal{D}}$ is around 13% and the empirical estimation is around 9%. Moreover, the combination of the new classifiers and bounds perform similarly to an SVM plus ten fold cross validation in terms of accuracy.

Figure 4 tries to illustrate qualitatively the discrepancies among the test error rate observed in crossvalidated SVM and that observed in the prior SVM. The figure shows the observed test error and the value of bounds on Q_D as functions of C for data sets `wav` and `pim`. The vertical pink line shows the crossvalidated C. The value of σ was fixed in both cases to the square root of the input data. In both cases, it is very noticeable the dramatic increase in the value of the bound as C increases, compared with a slight increase in the observed test error. A broadly accepted intuition

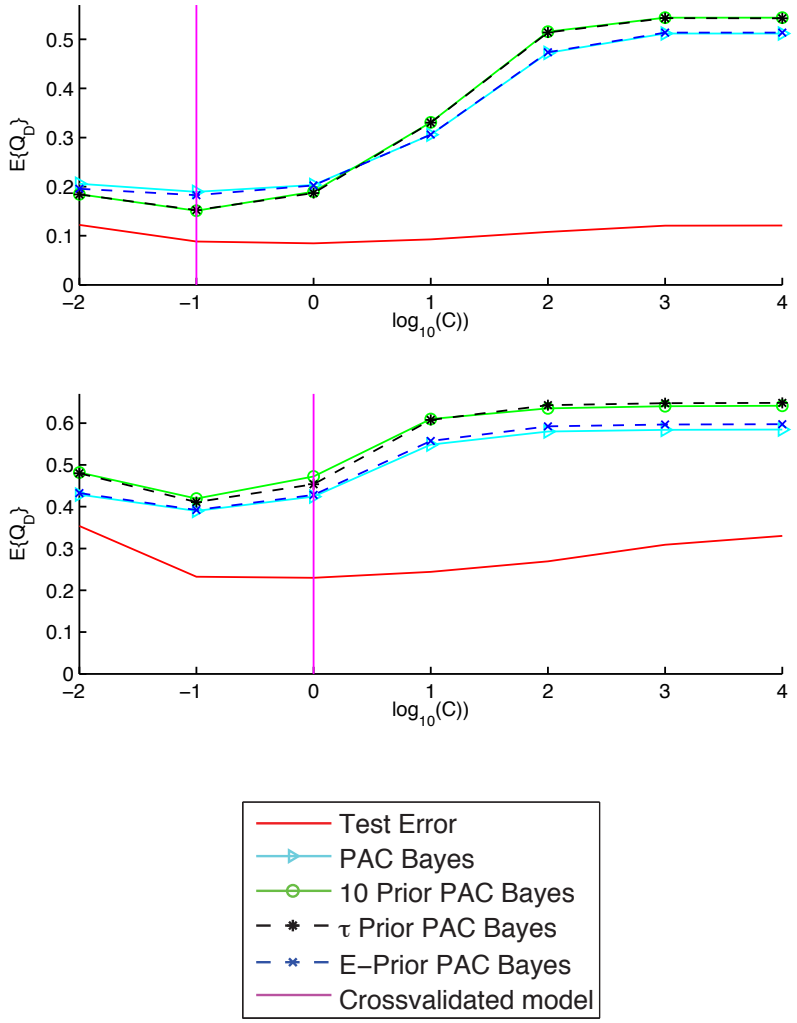


Figure 4: Values of bounds and observed test error rate as a function of C for data sets `wav` (top plot) and `pim` (bottom).

says that high values of C are likely to result in overfit, since the SVM is keener in reducing the training set error. However, our experiments seem to show that the bounds are overreacting to that behavior.

6. Conclusions

In this paper we have presented some strategies to tighten the already tight PAC-Bayes bound for binary classifiers by learning an informative prior distribution on classifiers. We have studied the SVM case, considering multivariate Gaussian priors and using some training data to infer their

mean and/or covariance matrix. The first strategy, named prior PAC Bayes bound, considers an identity covariance matrix. Then, an SVM learn on a separated subset of training samples serves as a direction along which to place the mean of the prior. This prior can be further refined in the τ-prior PAC Bayes bound case, where this direction is also used to stretch the covariance matrix. The second strategy, named expectation prior PAC-Bayes bound also considers identity covariances, but expresses the direction to place the prior as an statistic of the training data distribution and uses all the training samples to estimate such statistic. The expectation prior can also be refined stretching the covariance along the direction of the mean, yielding the τ-expectation prior PAC-Bayes bound.

The experimental work shows that these prior PAC-Bayes bounds achieve estimations of the expected true error of SVMs significantly tighter than those obtained with the original PAC-Bayes bound. It is remarkable that the prior PAC Bayes bounds improve the tightness of the PAC-Bayes bound even when the size of the training set experiences reductions of up to an 80% of its size.

The structure of the prior PAC-Bayes bound: learn a prior classifier using some data and then consider the SVM to be a posterior classifier inspired the design of new algorithms to train SVM-like classifiers. The prior SVM proposes a set of prior parts (fixed scalings along a prior direction learnt with separate data) and then fits a posterior part to each prior. The overall prior SVM classifier is the prior-posterior couple that yields a lower value of the bound. The η-prior SVM learns the scaling of the prior part and the posterior in the same quadratic program, thus significantly reducing the computational burden of the training. The analysis of these classifiers under the prior PAC-Bayes framework shows that the achieved bounds are dramatically tighter than those obtained for the original SVM under the same framework. Moreover, if the bound drives the selection of the hyperparameters of the classifiers, the observed empirical test error rate is similar to that observed in the SVM when the hyperparameters are tuned via ten fold cross validation.

Moreover, the prior SVM enables the use of different values of the regularisation constant C for both prior and posterior parts, which further tightens the bounds. The prior SVM classifiers with hyperparameters selected by minimising the τ-prior PAC Bayes bound achieve classification accuracies comparable to those obtained by an SVM with its parameters fixed by ten fold cross validation; with the great advantage that the theoretical bound on the expected true error provided by the τ-prior PAC Bayes bound is tightly close to the empirically observed.

All in all, the final message from this work is that the use informative priors can significantly improve the analysis and design of classifiers within the PAC-Bayes framework. We find the study of ways of extracting relevant prior domain knowledge from the available data and incorporating such knowledge in the form of the prior distribution to be a really promising line of research.

Acknowledgments

This work was partially supported by the IST Programme of the European Community under the PASCAL2 Network of Excellence IST-2007-216886. E. Parrado-Hernandez acknowledges support from Spain CICYT grant TIN2011-24533. Shiliang Sun is supported in part by the National Natural Science Foundation of China under Project 61075005, and the Fundamental Research Funds for the Central Universities. This publication only reflects the authors' views.

Appendix A.

The first step is to construct a Lagrangian functional to be optimized by the introduction of the constraints with multipliers α_i and ν_i, $i = 1,\ldots,m-r$,

$$L_P = \frac{1}{2}\|\mathbf{w} - \eta\mathbf{w}_r\|^2 + C\sum_{i=1}^{m-r}\xi_i - \sum_{i=1}^{m-r}\alpha_i\left(y_i\mathbf{w}^T\phi(\mathbf{x}_i) - 1 + \xi_i\right) - \sum_{i=1}^{m-r}\nu_i\xi_i\,, \quad \nu_i,\alpha_i \geq 0. \tag{19}$$

Taking the gradient of (19) with respect to $\mathbf{w}$ and derivatives with respect to ξ_i we obtain the optimality conditions:

$$\mathbf{w} - \eta\mathbf{w}_r = \sum_{j=1}^{m-r}\alpha_j y_j \phi(\mathbf{x}_j), \tag{20}$$

$$C - \alpha_i - \nu_i = 0 \Rightarrow 0 \leq \alpha_i \leq C \qquad i = 1,\ldots,m-r. \tag{21}$$

Plugging Equation (20) in functional (19) and applying the optimality condition (21) we arrive at the dual problem

$$\max_{\alpha_i} \frac{1}{2}\left\|\sum_{j=1}^{m-r}\alpha_j y_j \phi(\mathbf{x}_j)\right\|^2 - \sum_{i=1}^{m-r}\alpha_i\left(y_i\left(\eta\mathbf{w}_r^T + \sum_{j=1}^{m-r}\alpha_j y_j \phi^T(\mathbf{x}_j)\right)\phi(\mathbf{x}_i) - 1\right)$$

subject to

$$0 \leq \alpha_i \leq C \qquad i = 1,\ldots,m-r.$$

Now we can replace the prior $\mathbf{w}_r$ by its corresponding combination of mapped input vectors, $\mathbf{w}_r = \sum_{k=m-r+1}^{m} y_k\tilde{\alpha}_k\phi(\mathbf{x}_k)$ (with $\tilde{\alpha}_k$ being the scaled version of the Lagrange multipliers that yield a unit vector $\mathbf{w}_r$), and substitute kernel functions ($\kappa(\cdot,\cdot)$) for the inner products to arrive at

$$\max_{\alpha_i} \sum_{i=1}^{m-r}\alpha_i - \sum_{i=1}^{m-r}\eta\sum_{k=m-r+1}^{m}\alpha_i y_i\tilde{\alpha}_k y_k\kappa(\mathbf{x}_i,\mathbf{x}_k) - \frac{1}{2}\sum_{i,j=1}^{m-r}\alpha_i\alpha_j y_i y_j\kappa(\mathbf{x}_i,\mathbf{x}_j)$$

subject to

$$0 \leq \alpha_i \leq C \qquad i = 1,\ldots,m-r.$$

Grouping terms we have

$$\max_{\alpha_i} \sum_{i=1}^{m-r}\alpha_i\left(1 - y_i\eta\sum_{k=m-r+1}^{m}\tilde{\alpha}_k y_k\kappa(\mathbf{x}_i,\mathbf{x}_k)\right) - \frac{1}{2}\sum_{i,j=1}^{m-r}\alpha_i\alpha_j y_i y_j\kappa(\mathbf{x}_i,\mathbf{x}_j) \tag{22}$$

subject to

$$0 \leq \alpha_i \leq C \qquad i = 1,\ldots,m-r.$$

Now we can introduce the following matrix identifications to further compact Equation (22)

$$\mathbf{Y}_{(m-r),(m-r)} = \mathrm{diag}(\{y_i\}_{i=1}^{m-r}),$$

$$\mathbf{K}_{(m-r),(m-r)} = (\mathbf{K}_{(m-r),(m-r)})_{ij} = \kappa(\mathbf{x}_i,\mathbf{x}_j) \qquad i,j = 1,\ldots,m-r,$$

$$\mathbf{H}_{(m-r),(m-r)} = \mathbf{Y}_{(m-r),(m-r)}\mathbf{K}_{(m-r),(m-r)}\mathbf{Y}_{(m-r),(m-r)}, \tag{23}$$

$$\mathbf{v} = (\mathbf{v})_i = \left(1 - y_i \eta \sum_{k=m-r+1}^{m} \tilde{\alpha}_k y_k \kappa(\mathbf{x}_i, \mathbf{x}_k)\right) \qquad i = 1, \ldots, m-r, \tag{24}$$

$$\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{m-r}]^T. \tag{25}$$

Plugging (23), (24) and (25) in (22), we arrive at its final form that can be solved by off-the-shelf quadratic programming methods:

$$\max_{\alpha} \mathbf{v}^T \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H}_{(m-r),(m-r)} \boldsymbol{\alpha} \tag{26}$$

with box constraints

$$0 \leq \alpha_i \leq C \qquad i = 1, \ldots, m-r.$$

Once (26) is solved, the overall prior SVM classifier $\mathbf{w}$ can be retrieved from (20):

$$\mathbf{w} = \sum_{i=1}^{m-r} \alpha_i y_i \phi(\mathbf{x}_i) + \eta \sum_{k=m-r+1}^{m} \tilde{\alpha}_k y_k \phi(\mathbf{x}_k). \tag{27}$$

References

A. Ambroladze, E. Parrado-Hernández, and J. Shawe-Taylor. Tighter PAC-Bayes bounds. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 9–16. MIT Press, Cambridge, MA, 2007.

P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

C. L. Blake and C. J. Merz. *UCI Repository of Machine Learning Databases*. Department of Information and Computer Sciences, University of California, Irvine, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.

B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Conference on Computational Learning Theory*, COLT '92, pages 144–152, 1992.

O. Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*. Institute of Mathematical Statistics, Beachwood, Ohio, USA, 2007.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. PAC-Bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 353–360, 2009.

J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6(Mar):273–306, 2005.

J. Langford and J. Shawe-Taylor. PAC-Bayes and margins. In *Advances in Neural Information Processing Systems*, volume 14, Cambridge MA, 2002. MIT Press.

D. A. McAllester. PAC-Bayesian model averaging. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*, COLT '99, pages 164–170, 1999.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge (MA), 2002.

M. Seeger. PAC-Bayesian generalization error bounds for Gaussian process classification. *Journal of Machine Learning Research*, 3:233–269, 2002.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44:1926–1940, 1998.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

T. Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2:527–550, 2002.

Journal of Machine Learning Research 13 (2012) 3533-3537 Submitted 4/12; Revised 12/12; Published 12/12

DARWIN: A Framework for Machine Learning and Computer Vision Research and Development

Stephen Gould STEPHEN.GOULD@ANU.EDU.AU
Research School of Computer Science
The Australian National University
Canberra, ACT 0200, Australia

Editor: Antti Honkela

Abstract

We present an open-source platform-independent C++ framework for machine learning and computer vision research. The framework includes a wide range of standard machine learning and graphical models algorithms as well as reference implementations for many machine learning and computer vision applications. The framework contains Matlab wrappers for core components of the library and an experimental graphical user interface for developing and visualizing machine learning data flows.

Keywords: machine learning, graphical models, computer vision, open-source software

1. Introduction

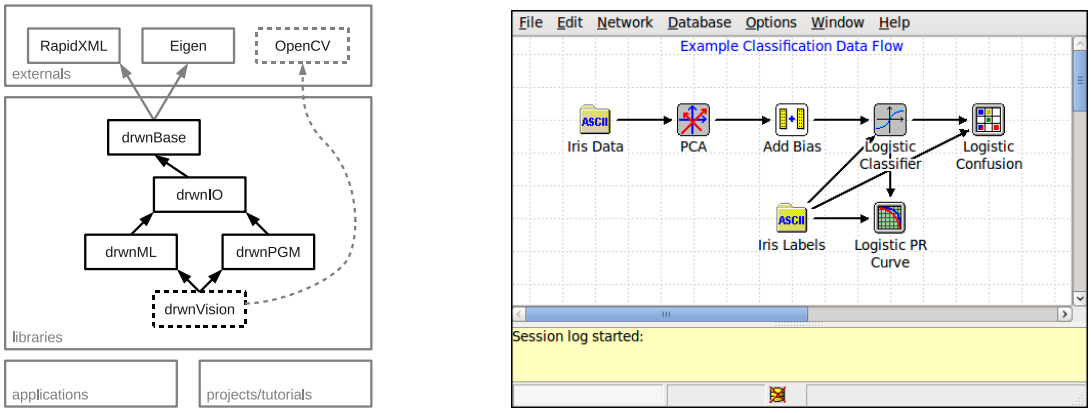
Machine learning and computer vision researchers have benefited over the last few years with the increased availability of numerous high quality open-source toolkits, (e.g., Gashler, 2011; King, 2009; Sonnenburg et al., 2010). These toolkits focus on delivering efficient implementations of mature algorithms but often omit infrastructure necessary for building end-to-end applications or experimenting with new algorithms. Furthermore, domain specific applications, such as those in computer vision, often require functionality from multiple packages complicating integration and maintenance. We have developed the DARWIN framework for machine learning and computer vision that aims to alleviate these limitations.

The goal of the DARWIN framework is two-fold: First, to provide a stable, robust and efficient library for machine learning practitioners, and second, to provide infrastructure for students and researchers to experiment with and extend state-of-the-art methods. To this end, we provide infrastructure for data management, logging and configuration; a consistent interface to standard machine learning and probabilistic graphical models algorithms; and well documented and easy to extend source code. The DARWIN framework is distributed under the BSD license. Importantly, it is free for both academic and commercial use.

2. Libraries

This section describes version 1.4 of the DARWIN framework. The code is divided into five main libraries, which we describe below. Sample applications wrap these libraries to provide out-of-the-box solutions, such classifier training and evaluation or probabilistic inference in graphical models. More sophisticated user applications can easily make use of these libraries.

The libraries have minimal dependence on external codebases, making DARWIN easy to install and maintain. Specifically, we require RapidXML[1] for XML parsing and Eigen[2] for linear algebra. OpenCV (Bradski and Kaehler, 2008) is required only for the (optional) computer vision component of the framework. Applications may link to other libraries to provide additional functionality, for example, GUI interfaces. An illustration of the library components and dependencies is shown in Figure 1(a).



(a) (b)

Figure 1: (a) Illustration of the library components and dependencies of the DARWIN framework. Dashed lines indicate optional components. (b) Screenshot of the experimental GUI for developing machine learning data flows.

Base. The `drwnBase` library provides core infrastructure and utility routines for the rest of the DARWIN framework, including (i) configuration management via command-line arguments and XML file processing, (ii) message, warning and error logging, (iii) runtime code profiling, (iv) basic multi-threading, and (v) miscellaneous routines for string manipulation, filesystem utilities, statistical functions, and XML processing.

IO. Common input/output functionality is provided by the `drwnIO` library. The main component of this library is a persistent data store for managing filesystem storage and in-memory caching of multiple data records.

Machine Learning. The machine learning library, `drwnML`, contains implementations of various algorithms for supervised and unsupervised machine learning. In particular, it includes algorithms for classification, regression and probability distribution modeling.

The library also contains utilities for dimensionality reduction (such as PCA and Fisher's LDA), unconstrained optimization, linear programming, and maintaining sufficient statistics and collating classification results. It also has a sparse vector implementation that is a drop-in replacement for the STL `vector` container. The container can be used to efficiently store sparse vectors with minimal changes to existing code. The concern here was memory usage rather than running time—a naive dot-product implementation, for example, will iterate over all entries in the sparse vector (including the zeros). More efficient dot-product member functions are provided by the sparse vector class.

1. RapidXML found at `http://rapidxml.sourceforge.net`.
2. Eigen found at `http://eigen.tuxfamily.org`.

Probabilistic Graphical Models. The probabilistic graphical models library, `drwnPGM`, provides functionality for (approximate) inference in structured probability spaces over discrete random variables. A factor graph data structure is used to specify the joint distribution in terms of functions, or factors, on overlapping subsets of the variables. The library provides classes for specifying factors and performing common factor operations, such as marginalization. This makes it easy to implement algorithms that manipulate factors. Reference implementations for many inference algorithms are also provided. These include:

- exact inference via the junction tree algorithm (Koller and Friedman, 2009)
- message passing inference algorithms such as synchronous and asynchronous sum-product and min-sum (max-product) (Koller and Friedman, 2009), and sequential tree-reweighted message passing (TRW-S) (Kolmogorov, 2006)
- move-making algorithms such as iterated conditional modes (ICM) (Besag, 1986), and α-expansion and $\alpha\beta$-swap (Boykov et al., 1999)
- linear programming relaxations, (e.g., Komodakis et al., 2007; Globerson and Jaakkola, 2007; Sontag et al., 2008; Meshi and Globerson, 2011)

Our implementations perform favorably against other graphical models inference packages, (e.g., Jaimovich et al., 2010; Mooij, 2010). For example, on the Rosetta Protein Design data set (Yanover et al., 2006), our implementation of the method of Sontag et al. (2008) runs faster than their code[3] on 86.6% of the problems.

Computer Vision. The `drwnVision` library builds on the OpenCV computer vision library (Bradski and Kaehler, 2008) to provide additional functionality for scene understanding. For example, the library provides infrastructure for multi-class semantic segmentation. Many approaches have been proposed to solve this problem. One approach is to construct a conditional Markov random field (CRF) over pixels in the image with learned unary and pairwise terms (He et al., 2004; Shotton et al., 2006). Another approach is to perform non-parametric label transfer by matching into a corpus of annotated images, (e.g., Liu et al., 2009), and does not rely on discriminatively trained classifiers. The vision library provides instances of both approaches and XML configurable applications make it easy to train and evaluate on new data sets. Moreover, documentation and scripts for generating results on standard data sets are distributed with the library.

Briefly, our CRF implementation is an enhanced variant of the baseline method described in Gould et al. (2009). It includes learned boosted decision tree classifiers for the unary terms and cross-validated contrast sensitive pairwise terms. On the standard 21-class MSRC data set (Criminisi, 2004) we achieve 78.4% accuracy using only unary terms and 82.3% with the full CRF model; on the Stanford Background data set (Gould et al., 2009) we achieve 77.1% (91.3%) accuracy and 79.9% (92.6%) accuracy on the semantic (geometric) labeling task for the unary and CRF model, respectively.

Our label transfer implementation makes use of the PatchMatch algorithm (Barnes et al., 2009) to construct a graph of dense patch correspondences between images. A full description of this approach is provided in Gould and Zhang (2012). Here we achieve 64.9% and 69.6% for the MSRC and Stanford Background data sets, respectively.

3. Code found at `http://cs.nyu.edu/~dsontag/code/mplp_ver1.tgz`.

3. Projects and Applications

The DARWIN framework is distributed with source code for a number of applications and projects that expose the functionality of the library, for example, training and evaluating classifiers or performing inference in probabilistic graphical models. Matlab `mex` wrappers are also provided for accessing core algorithms within the framework from the Matlab environment.

For non-expect machine learning practitioners it is often helpful to provide a graphical environment for designing machine learning pipelines. To this end, we have developed an experimental graphical user interface (GUI) for constructing machine learning data flows. A screenshot of the GUI is shown in Figure 1(b). The GUI is built using wxWidgets,[4] which must be installed separately. We intend to continue developing the GUI in future releases.

4. Documentation

Comprehensive documentation, including download and installation instructions, API reference, and short tutorial is available online at `http://drwn.anu.edu.au`. The documentation can also be generated locally from the source code using Doxygen.[5]

Acknowledgments

Some components of DARWIN were derived from the STAIR VISION LIBRARY[6] developed by the same author while at Stanford University. We thank the many contributors to that library. We also thank the early users of DARWIN for their suggestions and bug reports.

References

C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009.

J. E. Besag. On the statistical analysis of dirty pictures. *Royal Stat. Soc. Series B*, 48:259–302, 1986.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, 1999.

G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

A. Criminisi. Microsoft research cambridge (MSRC) object recognition pixel-wise labeled image database (version 2), 2004. Available for download under license from MSR at `http://research.microsoft.com/en-us/projects/ObjectClassRecognition/`.

M. S. Gashler. Waffles: A machine learning toolkit. *JMLR*, pages 2383–2387, July 2011.

4. wxWidgets found at `http://www.wxwidgets.org`.
5. Doxygen found at `http://www.doxygen.org`.
6. STAIR Vision Library found at `http://ai.stanford.edu/~sgould/svl`.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *NIPS*, 2007.

S. Gould and Y. Zhang. PatchMatchGraph: Building a graph of dense patch correspondences for label transfer. In *ECCV*, 2012.

S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

X. He, R. S. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

A. Jaimovich, O. Meshi, I. McGraw, and G. Elidan. FastInf: An efficient approximate inference library. *JMLR*, 11:1733–1736, 2010.

D. E. King. Dlib-ml: A machine learning toolkit. *JMLR*, 10:1755–1758, 2009.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 2006.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.

O. Meshi and A. Globerson. An alternating direction method for dual MAP LP relaxation. In *ECML*, 2011.

J. M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *JMLR*, 11:2169–2173, Aug 2010.

J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *JMLR*, 11:1799–1802, June 2010.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008.

C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation—an empirical study. *JMLR*, 2006.

Journal of Machine Learning Research 13 (2012) 3539-3583 Submitted 4/10; Revised 4/12; Published 12/12

Regularized Bundle Methods for Convex and Non-Convex Risks

Trinh-Minh-Tri Do[*] TRI.DO@IDIAP.CH
Idiap Research Institute
Rue Marconi 19
1920 Martigny, Switzerland

Thierry Artières THIERRY.ARTIERES@LIP6.FR
LIP6 - Université Pierre et Marie Curie
104 avenue du président Kennedy
75016 Paris, France

Editor: Tony Jebara

Abstract

Machine learning is most often cast as an optimization problem. Ideally, one expects a convex objective function to rely on efficient convex optimizers with nice guarantees such as no local optima. Yet, non-convexity is very frequent in practice and it may sometimes be inappropriate to look for convexity at any price. Alternatively one can decide not to limit *a priori* the modeling expressivity to models whose learning may be solved by convex optimization and rely on non-convex optimization algorithms. The main motivation of this work is to provide efficient and scalable algorithms for non-convex optimization. We focus on regularized unconstrained optimization problems which cover a large number of modern machine learning problems such as logistic regression, conditional random fields, large margin estimation, etc. We propose a novel algorithm for minimizing a regularized objective that is able to handle convex and non-convex, smooth and non-smooth risks. The algorithm is based on the cutting plane technique and on the idea of exploiting the regularization term in the objective function. It may be thought as a limited memory extension of convex regularized bundle methods for dealing with convex and non convex risks. In case the risk is convex the algorithm is proved to converge to a stationary solution with accuracy ε with a rate $O(1/\lambda\varepsilon)$ where λ is the regularization parameter of the objective function under the assumption of a Lipschitz empirical risk. In case the risk is not convex getting such a proof is more difficult and requires a stronger and more disputable assumption. Yet we provide experimental results on artificial test problems, and on five standard and difficult machine learning problems that are cast as convex and non-convex optimization problems that show how our algorithm compares well in practice with state of the art optimization algorithms.

Keywords: optimization, non-convex, non-smooth, cutting plane, bundle method, regularized risk

1. Introduction

Machine learning is most often cast as an optimization problem where one looks for the best model among a parameterized family of models. The best model is defined as the one with the set of parameters that minimizes an objective function (i.e. criterion). For some years now machine learning community aimed at designing new models in such a way that the resulting objective function is convex. Doing so brings the fundamental advantage that one can rely on efficient convex optimiza-

∗. Part of this work was done when TMT Do was at LIP6.

tion algorithms, with nice guarantees such as no local optima and easier theoretical analysis (e.g. for the convergence rate). For instance logistic regression, support vector machine, maximum margin Markov network, and conditional random fields have found widespread use in basic machine learning applications.

However, such a "simple convex modeling" may actually be outperformed by non-convex modeling in some important applications. For example on MNIST database, convex Gaussian-SVM reaches 1.4% error rate vs. 0.53% for non-convex convolutional nets (Jarrett et al., 2009).[1] Also non-convexity is much more frequent than convexity "in real life". A number of problems that machine learning researchers face today may not be easily cast as convex optimization problems without limiting *a priori* the expressivity of the models used and the potential of the models to learn (LeCun et al., 1998; Collobert et al., 2006; Bengio and Lecun, 2007). First, many real-world problems need complicated models whose learning requires solving non-convex optimization problems. For instance, models with non-convex discriminant function such as neural networks and hidden Markov models (HMMs) have become classical and reference models for many difficult tasks in vision and speech. Second, non-convexity of objective function naturally arises in learning paradigms such as unsupervised and semi-supervised learning as well as in transductive SVM, etc (Chapelle et al., 2006; Joachims, 1999).

Two strategies have been investigated to handle non-convexity in machine learning approaches. Few works attempted to use convex relaxation technique in order to transform an original non-convex problem into a convex one, this is a kind of "convexity at any price" strategy. Convex relaxation mechanics strongly depend on the application, there is no principled method for turning a non-convex problem to a convex one. It has been used in maximum margin clustering (Xu et al., 2004), transductive SVM (Xu et al., 2008), discriminative unsupervised structured predictors (Xu et al., 2006), large margin CDHMM (Sha and Saul, 2007). However, the robustness of this approach for complex problems is questionable since the use of strong assumptions may lead to poor approximation quality, thus provide poor performance in practice.

Since convex modeling does not cover all real-world problems and convex relaxation techniques are not always easy and robust, few researchers proposed to give up convexity and to focus on non-convex optimization techniques, for instance concave-convex procedure (CCCP) (Yuille and Rangarajan, 2003) and difference of convex (DC) programming (Horst and Thoai, 1999). These non-convex optimization techniques have been successfully applied for some tasks such as ramp loss SVM, non-convex TSVM (Collobert et al., 2006), kernel selection (Argyriou et al., 2006) or non-convex maximum margin clustering (Zhao et al., 2008). Note that these techniques cover only a limited class of problems and require an ad-hoc design for every machine learning problem. For instance, the CCCP can theoretically be applied to any continuous objective function since any such function can be decomposed into the difference of two convex functions, yet reformulating the original function to a concave-convex form may call for mathematical efforts. Furthermore not all decomposition are interesting.

We are concerned here with the development of generic optimization techniques able to deal with the general unconstrained optimization problem

$$\begin{array}{ll} \min_{\mathbf{w}} & f(\mathbf{w}) \\ with & f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + R(\mathbf{w}) \end{array} \tag{1}$$

1. A collection of evaluation results on MNIST data is available at: `http://yann.lecun.com/exdb/mnist`.

where $\mathbf{w} \in \mathbb{R}^D$ are the model parameters and $R(\mathbf{w})$ (the main objective) is a data-fitting measurement to be minimized which we consider to be not necessarily smooth everywhere nor convex. This unconstrained formulation covers many mentioned machine learning problems such as SVM, CRF, M3N, transductive SVM, ramp loss SVM, neural network (Do and Artières, 2010), Gaussian HMM (Do and Artières, 2009). Note that the formulation in Equation 1 does not apply easily to kernel methods which are based on an implicit data transformation (e.g. RBF kernel) and are preferably solved in the dual space. However, there are several methods that can enrich the model flexibility without considering an implicit data transformation. As an example, for low dimensional or sparse data, one could have an explicit and efficient transformation for polynomial kernel. Furthermore, instead of using a predefined implicit transformation one could also learn the explicit data transformation directly such as latent feature discovery based on Boltzmann machine (Hinton et al., 2006). At the end, while not covering kernel tricks, our general optimization problem can be used for learning many powerful non-linear models.

As the problem in Equation 1 is at the heart of many machine learning application, it is important to have an efficient non-convex optimization method for this class of minimization problem. Among candidate families of optimization algorithms, cutting plane methods and bundle based methods are very appealing for optimization problems such as the one in Equation 1 since, as opposed to many gradient descent based methods, it can naturally deal with its non-smooth everywhere feature (Kiwiel, 1985; Gaudioso and Monaco, 1992; Makela, 2002; Makela and Neittaanmaki, 1992; Schramm and Zowe, 1992). However the convergence of bundle methods for non-convex optimization is rather slow in practice. And theoretical results on convergence rate are indeed missing for non-convex objective functions. This explains in our opinion why the use of general non-convex bundle methods is still limited in machine learning. Another reason is the lack of easy-to-use implementation of non-convex bundle methods.

The recent success of convex regularized bundle methods (CRBMs) in machine learning (Smola et al., 2008; Weimer et al.; Joachims et al., 2009) motivated us to investigate extensions of bundle methods for proposing efficient algorithms able to deal with machine learning non-convex optimization problems, which is the core idea of this work. To design such an algorithm, we investigated new optimization algorithms that combines ideas from non-convex bundle methods (NBM) (Kiwiel, 1985) and from CRBMs (Smola et al., 2008). Our algorithm relies on two main contributions, a limited memory variant of bundle methods and the extension of CRBM to non-convex risks.

The limited memory variant may be used in CRBM as well as in our non-convex extension of CRBM. It allows limiting the algorithmic complexity of a single iteration in bundle methods while it is usually increasing (at least quadratically) with the number of iteration, which makes bundle methods not practical for difficult and large scale problems requiring thousands of iterations. We show that our limited memory variant, when included to CRBM, inherits its fast convergence rate in $O(1/\lambda\varepsilon)$ iteration to reach a gap below ε.

Our extension of CRBM to non-convex risks includes the limited memory variant and is designed to make bundle methods scalable for real life non-convex learning problems.[2] This is achieved by making the algorithm focus on the current best solution and by using a specific locality measure for regularized risks. Such a strategy allows fast convergence in practice on difficult and large scale machine learning problems that we investigated. Unfortunately this comes with only weak proof of convergence towards a stationary solution, relying on a moot assumption. In our

2. The MATLAB implementation of the proposed method is available at `https://forge.lip6.fr/projects/nrbm`.

opinion it is a kind of trade-off, a price to pay to achieve algorithmic efficiency in practice. As a consequence though we provide main theoretical results we do not include our convergence proofs here since these are weak, but these are available in an internal report (Do and Artieres, 2012).

First, in Section 2 we provide background on the cutting plane technique and on bundle methods, and we describe two main existing extensions, the convex regularized bundle method (CRBM) and the non-convex bundle method (NBM). Then, we present in Section 3 our two contributions yielding our algorithm, NRBM, which is a regularized bundle method for non-convex optimization. We propose few variants of our method in Section 3.3 and we discuss in Section 4 the convergence behavior of our method both for convex risks and for non-convex risks. Finally we provide in Section 5 a number of experimental results. We investigate first artificial test problems that show that our algorithm compares well to standard non-convex bundle methods while converging much faster, suggesting our algorithm may make large scale problems practical. Second we compare our algorithms to dedicated state of the arts optimization algorithms for a number of machine learning problems, including standard problems such as learning of transductive support vector machines learning, learning of maximum margin Markov networks, learning conditional random fields, as well as less standard but difficult optimization problems related to discriminative training of complex graphical models for handwriting and speech recognition.

2. Background on Cutting Plane and Bundle Methods

We provide now some background on the cutting plane principle and on optimization methods that have been built on this idea for convex and non-convex objective functions.

2.1 Cutting Plane Principle

Surely the most powerful method for non-smooth optimization is based on *polyhedral approximations*, whose basic element is the *cutting plane* (CP). For a given function $f(\mathbf{w})$, a cutting plane $c_{\mathbf{w}'}(\mathbf{w})$ is a first-order Taylor approximation computed at a particular point $\mathbf{w}'$:

$$f(\mathbf{w}) \approx c_{\mathbf{w}'}(\mathbf{w}) = f(\mathbf{w}') + \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w} - \mathbf{w}' \rangle$$

where $\mathbf{a}_{\mathbf{w}'} \in \partial f(\mathbf{w}')$ is a subgradient of f at $\mathbf{w}'$. For convex function, the subdifferential $\partial f(\mathbf{w}')$ is the set of vectors $\mathbf{a}$ that satisfies: $f(\mathbf{w}) \geq f(\mathbf{w}') + \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w} - \mathbf{w}' \rangle$. The concept of subdifferential is also generalized for non-convex functions, which is defined as the set of vectors $\mathbf{a}$ that satisfies: $f^{\circ}(\mathbf{w}';\mathbf{h}) \geq \langle \mathbf{a}, \mathbf{h} \rangle \ \forall \mathbf{h}$, where $f^{\circ}(\mathbf{w}', \mathbf{h})$ denotes the generalized directional derivative of f at $\mathbf{w}'$ in the direction $\mathbf{h}$.

Go back to the definition of the cutting plane approximation based on Taylor approximation, it may be rewritten as:

$$\begin{array}{ll} c_{\mathbf{w}'}(\mathbf{w}) & = \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w} \rangle + b_{\mathbf{w}'} \\ with & \mathbf{a}_{\mathbf{w}'} \in \partial f(\mathbf{w}') \\ & b_{\mathbf{w}'} = f(\mathbf{w}') - \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w}' \rangle. \end{array} \tag{2}$$

A cutting plane $c_{\mathbf{w}'}$ is an approximation of f which is accurate for $\mathbf{w}$ lying in the vicinity of $\mathbf{w}'$ where the CP is defined, i.e. where the subgradient is computed. The quality of such an approximation and the area where it is accurate depend on higher order information on f such as the Hessian matrix. Figure 1 illustrates the linear approximation implemented by a cutting plane for a one-dimensional function. Importantly, a cutting plane of a convex function f is an underestimator of f.

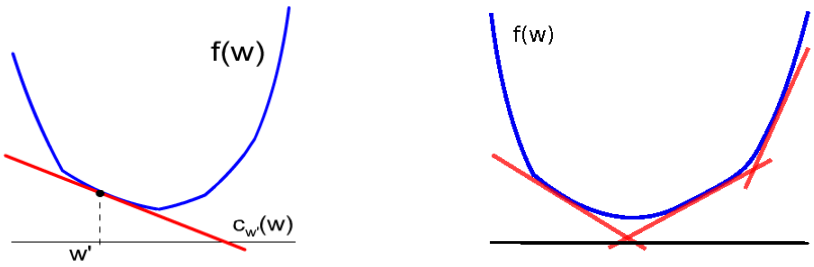


Figure 1: Basic approximation of a function f by a (underestimator) cutting plane at a point $\mathbf{w}'$ (left), and a more accurate approximation by taking the maximum over many cutting planes of f (right).

2.2 Cutting Plane Method for a Convex Objective

The *cutting plane method* has been proposed for the minimization of convex functions. In the case of a convex objective, any cutting plane of the objective f is an underestimator of f. The idea of the cutting plane method is that one can build an accurate approximation function (named g_t hereafter) of f, which is also an underestimator of f, as the maximum over many cutting plane approximation built at different points $\{\mathbf{w}_1, ..., \mathbf{w}_t\}$ as follows:

$$f(\mathbf{w}) \approx g_t(\mathbf{w}) = \max_{j=1..t} \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j}. \tag{3}$$

Of course $g_t(\mathbf{w})$ is an underestimator of $f(\mathbf{w})$. It is called the approximation function of f at iteration t.

The cutting plane method aims at iteratively building an increasingly accurate piecewise linear underestimator of the objective function by successively adding new cutting planes to the approximation g of f. If the approximation is good enough, one may hope that the minimum of f and of its approximation g will be very close or even equal. Every iteration, one adds a new cutting plane underestimator built at current solution, yielding a new piece-wise linear underestimator of f as in Equation 3. The minimization of this underestimator approximation is usually called the approximated problem (it is a linear program) and gives a new current solution, etc.

Note that the approximation function may not have a minimum, then artificial bounds may be placed on the points of $\mathbf{w}$, so that the minimization will be carried out over a compact set and consequently a exists.

The cutting plane method is described in Algorithm 1, it is proved to converge in a finite number of iterations to an ε-solution (Bertsekas et al., 2003).

2.3 Bundle Methods for a Convex Risk

Convex bundle method. One of the drawbacks of the cutting plane method is its instability. It may make large steps away from the optimum even when the current solution is close to it. Standard convex bundle method (CBM), also called *proximal cutting plane method* or *proximal bundle method*, tries to overcome this problem by adding to the polyhedral approximation function a regularization

Algorithm 1 Cutting Plane Method (for convex objective function)

Input: $\mathbf{w}_1$, f, ε
Output: $\mathbf{w}^*$
for $t = 1$ to ∞ **do**
  Compute $\mathbf{a}_{\mathbf{w}_t}$ and $b_{\mathbf{w}_t}$ according to Equation 2
  Minimize $g_t(\mathbf{w})$ (defined as in Equation 3) to get $\mathbf{w}_{t+1} \leftarrow \text{argmin}_{\mathbf{w}}\, g_t(\mathbf{w})$
  $gap = [\min_{j=1..t} f(\mathbf{w}_j)] - g_t(\mathbf{w}_{t+1})$
  if $gap < \varepsilon$ **then return** $\mathbf{w}_t$
end for

Algorithm 2 Convex Regularized Bundle Method (CRBM)

Input: $\mathbf{w}_1$, R, ε
Output: $\mathbf{w}^*$
for $t = 1$ to ∞ **do**
  Compute $\mathbf{a}_{\mathbf{w}_t}$ and $b_{\mathbf{w}_t}$ of R at $\mathbf{w}_t$
  $\mathbf{w}_t^* = \text{argmin}_{\mathbf{w} \in \{\mathbf{w}_1, \ldots \mathbf{w}_t\}} f(\mathbf{w})$
  $\tilde{\mathbf{w}}_t \leftarrow \text{argmin}_{\mathbf{w}}\, g_t(\mathbf{w})$ where $g_t(\mathbf{w})$ is defined as in Equation 6
  $gap_t = f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t)$
  $\mathbf{w}_{t+1} = \tilde{\mathbf{w}}_t$
  if $gap_t < \varepsilon$ **then return** $\mathbf{w}_t^*$
end for

term. The approximation function becomes:

$$f(\mathbf{w}) \approx g_t(\mathbf{w}) = (\mathbf{w} - \mathbf{w}_t)^\top H_t (\mathbf{w} - \mathbf{w}_t) + \max_{j=1..t} \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j} \tag{4}$$

where H_t is a positive definite symmetric matrix. The regularization term forces the new solution not to be too far from the current solution. In addition it makes the approximation function have a unique minimum (as long as the Hessian matrix of the regularization term is positive-definite as in our example) without adding artificial constraints. While the approximation function in Equation 4 can be used to generate new points, the standard bundle method also includes a line-search procedure which returns either a serious step (the objective at current solution has significantly decreased) or a null step (the decrease of f is too low and the approximation function should be improved).

Convex regularized bundle method. The convex regularized bundle method (CRBM) (Smola et al., 2008) is an instance of CBM algorithms for dealing with regularized (and convex) risks as in Equation 1. It relies on cutting planes that are built on the risk $R(\mathbf{w})$ only and does not use a line search procedure. Such a linear approximation of the risk $R(\mathbf{w})$ yields a quadratic approximation of the objective $f(\mathbf{w})$:

$$f(\mathbf{w}) \approx \frac{\lambda}{2}\|\mathbf{w}\|^2 + \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w} \rangle + b_{\mathbf{w}'}. \tag{5}$$

These two approximation functions on $R(\mathbf{w})$ and on $f(\mathbf{w})$ are illustrated in Figure 2. Note that this quadratic approximation of $f(\mathbf{w})$ is more accurate than a cutting plane approximation on $f(\mathbf{w})$. Furthermore, this trick avoids adding an artificial regularization term into the approximation problem as in standard bundle methods.

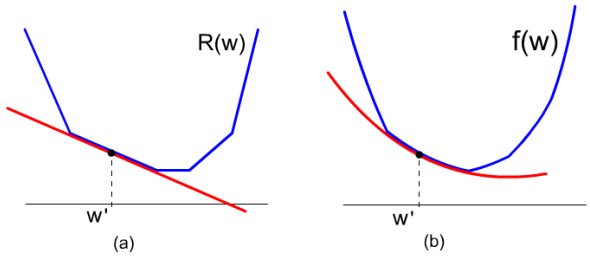


Figure 2: Cutting plane approximations in CRBM : A linear underestimator of $R(\mathbf{w})$ (a), and a quadratic underestimator of $f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + R(\mathbf{w})$ derived from this linear underestimator (b) (Cf. Equation 5)

CRBM is very similar to the cutting plane technique described before, where every iteration a new cutting plane approximation is built (at the current solution) and added to the current approximation function. The approximation of $f(\mathbf{w})$ at iteration t is then:

$$f(\mathbf{w}) \approx g_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max_{j=1..t} \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j} \tag{6}$$

and the approximation problem is

$$\tilde{\mathbf{w}}_t = \operatorname*{argmin}_{\mathbf{w}} g_t(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max_{j=1..t} \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j} \tag{7}$$

where $\langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j}$ is the approximation cutting plane of R built at $\mathbf{w}_j$, the solution at iteration j. Importantly, if $R(\mathbf{w})$ is convex then any cutting plane $\langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j}$ is an underestimator of $R(\mathbf{w})$, and its maximum, $\max_{j=1..t} \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j}$, is also an underestimator approximation of R. Hence, $g_t(\mathbf{w})$ are monotonically increasing quadratic underestimators of $f(\mathbf{w})$ which converge towards $f(\mathbf{w})$ as cutting planes are added.

Minimizing the approximation problem in CRBM. The approximation problem (Equation 7) at iteration t is an SVM-like optimization problem:

$$\begin{array}{rll} \tilde{\mathbf{w}}_t = & \operatorname{argmin}_{\mathbf{w}} \min_{\xi} & \frac{\lambda}{2}\|\mathbf{w}\|^2 + \xi \\ & s.t. & \langle \mathbf{a}_j, \mathbf{w} \rangle + b_j \leq \xi \;\; j = 1..t \end{array}$$

with $c_j(\mathbf{w}) = \langle \mathbf{a}_j, \mathbf{w} \rangle + b_j$. We can get its dual form easily through Lagrangian mechanics. The Lagrangian of the above optimization problem is:

$$L(\mathbf{w}, \xi, \boldsymbol{\alpha}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \xi + \sum_{j=1..t} \alpha_j (\langle \mathbf{a}_j, \mathbf{w} \rangle + b_j - \xi)$$

where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_t)$ are Lagrange multipliers. The solution is given by a saddle point of the Lagrangian, that must be minimized wrt. primal variables ($\mathbf{w}$,ξ) and maximized wrt. Lagrange multipliers. At a saddle point, the derivative of the Lagrangian wrt. ($\mathbf{w}$,ξ) must satisfy:

$$\begin{array}{ll} \frac{\partial L}{\partial \xi} = 0 & \iff \sum_{j=1..t} \alpha_j = 1, \\ \frac{\partial L}{\partial \mathbf{w}} = 0 & \iff \lambda \mathbf{w} = -(\sum_{j=1..t} \alpha_j \mathbf{a}_j). \end{array}$$

By substituting these results back into the Lagrangian, primal variables $\mathbf{w}$ and ξ disappear and we get the dual problem:

$$\begin{array}{lll} \boldsymbol{\alpha}_t = & \text{argmax}_{\boldsymbol{\alpha}\in\mathbb{R}^t} & -\frac{1}{2\lambda}\|\boldsymbol{\alpha}A_t\|^2 + \boldsymbol{\alpha}B_t \\ & s.t & \alpha_j \geq 0 \; \forall j = 1..t \\ & & \sum_{j=1..t} \alpha_j = 1 \end{array} \tag{8}$$

where $A_t = [\mathbf{a}_1; ...; \mathbf{a}_t]$ is a matrix (with $\mathbf{a}_j$ being row vectors), $B_t = [b_1; ...; b_t]$ is the vector of scalars and $\boldsymbol{\alpha}$ stands for the (row) vector of Lagrange multipliers (of length t at iteration t). Let $\boldsymbol{\alpha}_t$ be the solution of the above dual problem at iteration t, the solution of the primal problem is given by:

$$\begin{array}{ll} \tilde{\mathbf{w}}_t & = -\frac{\boldsymbol{\alpha}_t A_t}{\lambda}, \\ g_t(\tilde{\mathbf{w}}_t) & = -\frac{1}{2\lambda}\|\boldsymbol{\alpha}_t A_t\|^2 + \boldsymbol{\alpha}_t B_t. \end{array}$$

Convergence rate of CRBM. The convergence of CRBM is proved based on the fact that the gap between the best observed value $f(\mathbf{w}_t^*)$ and the minimum of the approximation function $g_t(\tilde{\mathbf{w}}_t)$ decreases every iteration. Since $g_t(\mathbf{w})$ is an underestimator of $f(\mathbf{w})$, the gap is greater than or equal to the difference between the best observed value $f(\mathbf{w}_t^*)$ and the minimum of $f(\mathbf{w})$. Therefore, if $gap_t \leq \varepsilon$ then $\mathbf{w}_t^*$ is an ε-solution of $f(\mathbf{w})$. By characterizing the decrease of the gap after each iteration, the authors of CRBM proved that the method require $O(1/\lambda\varepsilon)$ iterations to reach a gap below ε (Smola et al., 2008).

2.4 Non-Convex Bundle Methods (NBM)

Bundle methods have also been extended to deal with non-convex functions and have become a standard for minimizing non-smooth and non-convex function.

2.4.1 PRINCIPLE

There are many variants of non-convex bundle algorithm (NBM), with many parameters to tune. We present here a simple description of the method to better stress its main features. Basically NBM works similarly as standard bundle methods by building iteratively an approximation function via the cutting plane technique. However since the objective is no more convex, such an approximation function is not an underestimator of the objective anymore which makes things harder and requires a more complicated algorithm.

Every iteration the algorithm updates a number of quantities, whose set is usually called the *state* of the algorithm, based on the state in previous iteration. The state of the algorithm at iteration t, named $\mathbb{B}_t$, is a set of points, subgradients and locality measures to the current solution. At iteration t, the algorithm performs the following steps:

- ***Determine the search direction.*** This is done through minimizing the approximated problem defined by $\mathbb{B}_t$. The approximation problem is an instance of quadratic programming similar to the one in Equation 4, except that the raw cutting planes are adjusted to make sure that the approximation is a local underestimator of the objective function. The minimization of the approximation problem yields a new point $\tilde{\mathbf{w}}_t$.
- ***Perform a line search.*** The algorithm performs a special line search from the best current solution $\mathbf{w}_t^*$ to the minimum of the approximation problem $\tilde{\mathbf{w}}_t$.[3] The line search outputs a

3. Under some semi-smoothness assumptions it is proved that this line search algorithm terminates in a finite number of iterations (Luksan and Vlcek, 2000).

new solution $\mathbf{w}_{t+1}$. Two cases may arise. In a first case, this new solution does not lead to a significant improvement (i.e. decrease) in the objective function, we say the current iteration is a null step. In such a case, the best solution does not change (i.e. $\mathbf{w}^*_{t+1} \equiv \mathbf{w}^*_t$). Alternatively the new solution may bring a significant improvement in the objective (iteration is called a serious step). Then one defines the new best solution as $\mathbf{w}^*_{t+1} \equiv \mathbf{w}_{t+1}$. Note that in both cases, the approximation function is improved by adding a new cutting plane at $\mathbf{w}_{t+1}$. We do not present in details the line search procedure since it is both rather complicated and standard. Interested readers may find detailed description in the literature, e.g., (Luksan and Vlcek, 2000).

- ***Update the bundle and build a new approximation function.*** The set of cutting planes is expanded with the new cutting plane built at $\mathbf{w}_{t+1}$. Due to the non-convex feature of the objective function, the definition of approximation is not trivial, involving additional concepts such as locality measure, the strategy of NBM to deal with non-convexity will be detailed in the next subsection. Importantly, note that one gets more cutting planes in the bundle as the algorithm iterates, and such a ever increasing number of cutting planes may represent a potential problem wrt. computational and memory cost if many iterations are required. Usually to overcome such a problem, one uses an aggregated cutting plane in order to accumulate information of all cutting planes in previous iterations (Kiwiel, 1985). It allows discarding older cutting planes and helps limiting the algorithmic complexity. For instance, one may keep a fixed number of cutting planes in the bundle $\mathbb{B}_t$ by removing the oldest cutting plane. Then, the aggregated cutting plane allows preserving part of the information brought by removed cutting planes.

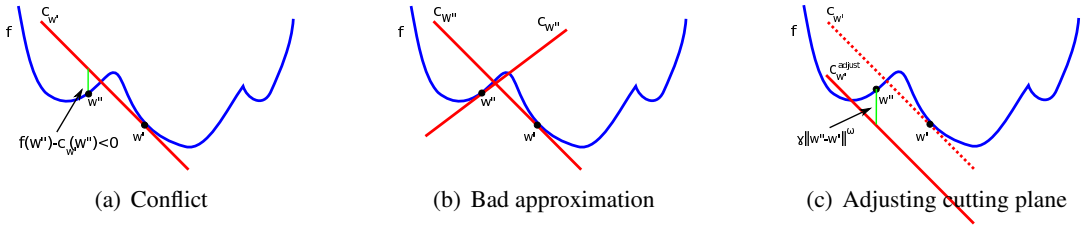
2.4.2 HANDLING NON-CONVEX OBJECTIVE FUNCTION

Bundle methods must be adapted to work for non-convex optimization since the core idea of using a first order Taylor approximation as an underestimator of the objective function does not hold anymore. Then, the standard approximation function, which is defined as the maximum over a set of cutting plane approximations, is not an underestimator of the non-convex objective function anymore. In addition although one may reasonably assume that a cutting plane built at a point $\mathbf{w}'$ is an accurate approximation of f in a small region around $\mathbf{w}'$, such an approximation may become very poor for $\mathbf{w}$ far from $\mathbf{w}'$. At the end, the maximum over cutting plane "approximations" may be a very poor approximation of the objective.

An example of poor approximation is shown in Figure 3(a). The linearization error $(f(\mathbf{w}) - c_{\mathbf{w}'}(\mathbf{w}))$ of a cutting plane $c_{\mathbf{w}'}$ at a point $\mathbf{w}''$ may be negative, meaning that the function is overestimated at that point. In the following we will say in such a case that there is a **conflict** between cutting plane $c_{\mathbf{w}'}$ and $\mathbf{w}''$. As can be seen, overestimation of a cutting plane at a local minimum will probably "remove" this minimum from the set of reachable solutions. Figure 3(b) shows that all three visible local minimums are "removed" by overestimation of the two cutting planes built at $\mathbf{w}'$ and $\mathbf{w}''$.

Non-convex bundle method strategy. In non-convex bundle methods (Kiwiel, 1985; Gaudioso and Monaco, 1992; Makela, 2002; Makela and Neittaanmaki, 1992; Schramm and Zowe, 1992) the solution to overcome conflicts between a cutting plane $c_{\mathbf{w}'}$ and a point $\mathbf{w}''$ is to lower the cutting plane $c_{\mathbf{w}'}$ by changing its offset while preserving the normal vector $\mathbf{a}_{\mathbf{w}'}$ (see Figure 3(c)). This leads to an adjusted cutting plane:

$$c^{adjust}_{\mathbf{w}'}(\mathbf{w}) = \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w} \rangle + b^{adjust}_{\mathbf{w}'}.$$



(a) Conflict (b) Bad approximation (c) Adjusting cutting plane

Figure 3: Cutting planes and linearization errors.

The offset $b_{\mathbf{w}'}$ is changed in $b^{adjust}_{\mathbf{w}'}$ so that that the linearization error of $c^{adjust}_{\mathbf{w}'}$ at $\mathbf{w}''$ is greater than or equal to both, the absolute value of the linearization error between $c_{\mathbf{w}'}$ and f at $\mathbf{w}''$, and a *locality measure* between $\mathbf{w}'$ and $\mathbf{w}''$:

$$f(\mathbf{w}'') - c^{adjust}_{\mathbf{w}'}(\mathbf{w}'') \geq |f(\mathbf{w}'') - c_{\mathbf{w}'}(\mathbf{w}'')|, \tag{9}$$

$$f(\mathbf{w}'') - c^{adjust}_{\mathbf{w}'}(\mathbf{w}'') \geq \gamma\|\mathbf{w}'' - \mathbf{w}'\|^{\omega} \tag{10}$$

where $\gamma \geq 0, \omega \geq 1$ are *locality measure parameters*. The condition (9) ensures that if the linearization error, $f(\mathbf{w}'') - c_{\mathbf{w}'}(\mathbf{w}'')$, is negative then the cutting plane has to be lowered at least twice the amount that is required to have linearization error zero. In other words, in the case of negative linearization error at $\mathbf{w}''$, the cutting plane is adjusted so that the new linearization error is positive, with at least the same magnitude as the "old" negative linearization error. The condition (10) defines another underestimator on the linearization error (of the adjusted cutting plane) which is based on the distance between two points $\mathbf{w}'$ and $\mathbf{w}''$. The further the two points are the greater the linearization error should be. The two conditions lead to the following offset change definition:

$$b^{adjust}_{\mathbf{w}'} = f(\mathbf{w}'') - \langle \mathbf{a}_{\mathbf{w}'}, \mathbf{w}'' \rangle - \max\left[|f(\mathbf{w}'') - c_{\mathbf{w}'}(\mathbf{w}'')|, \gamma\|\mathbf{w}'' - \mathbf{w}'\|^{\omega}\right].$$

This is the greatest offset (closest to $b_{\mathbf{w}'}$) that satisfies the two above conditions. Besides, one can easily check that if $c_{\mathbf{w}'}$ already satisfies both conditions (9) and (10) then $b^{adjust}_{\mathbf{w}'} = b_{\mathbf{w}'}$ and $c^{adjust}_{\mathbf{w}'}(\mathbf{w})$ and $c_{\mathbf{w}'}(\mathbf{w})$ coincide.

2.5 Conclusion

CRBM are a fast adaptation of bundle methods to convex and regularized risks. Every iteration a new cutting plane is added to the bundle so that the size of the bundle at iteration t is t. This makes tackling complex tasks, eventually requiring many iterations, difficult since the cost of solving the minimization of the approximated function is quadratic in the size of the bundle. To make CRBM more scalable we will provide a limited memory variant where the size of the bundle is limited to a given size (theoretically three CP are sufficient) whatever the iteration.

General non-convex bundle methods have been proved to have global convergence to cluster points which are stationary solutions. Note that a stationary solution is not necessarily a local minimum but may be a saddle point or even a local maximum. In practice, however, there are many hyper-parameters to tune (γ, ω, regularization term, and several hyper-parameters for the line search procedure) and convergence rate is not guaranteed, both drawbacks preventing using such algorithms for large scale applications. We will propose a variant of regularized bundle method that is adapted to non-convex risks and which is scalable in practice.

3. Non-Convex Regularized Bundle Method (NRBM)

The success of convex regularized bundle methods with improved convergence rate over bundle methods, both in theory and practice, motivated us to investigate their extension to non-convex optimization, leading to bundle methods for regularized non-convex risks (NRBM). To design such an algorithm, we propose two main contributions, the extension of CRBM to non-convex risks and a limited memory variant of bundle methods that allows limiting the algorithmic cost of a single iteration.

The extension of CRBM for non-convex function is not straightforward since, as we already observed when presenting NBM, the cutting plane approximation does not yield an underestimator of the objective function. Our proposal is to exploit some techniques of NBM for handling non-convex function while considering a special design of the algorithm in order to keep the fast convergence rate of CRBM. On one hand, we use standard techniques such as the introduction of locality measure and the adjustment of cutting planes in order to build local underestimator of the function at a given point. On the other hand, we propose novel techniques such as a particular definition of the locality measure for regularized risk and the introduction of constraints on CPs adjustment when dealing with conflicts, which guarantee a minimal improvement on the approximation gap within an iteration. At the end, we come up with a non-convex variant which inherits, in practice, the convergence rate of CRBM. Note however that we may only provide weak theoretical results on the convergence to a local minimum for the non-convex case. Convergence analysis is discussed in Section 4.

The ability of our method, NRBM, to deal with non-convex risk allows tackling a wide range of application and especially a number of everyday machine learning problems. Yet the algorithmic cost of a single iteration grows with the number of the iteration. Actually, the dual program of the approximation problem minimization in Equation 8 has a memory cost of $O(tD+t^2)$ for storing all the cutting planes and the dot product matrix between cutting planes' normal vectors (i.e. $\langle a_i, a_j \rangle$), where t is the number of cutting planes (it is equal to the iteration number in CRBM) and D is the dimensionality of $\mathbf{w}$. In addition, the computational cost for solving the dual program is usually quadratic or cubic in t. These costs may be prohibitive especially in situations where the objective is hard to optimize and the algorithm requires a large number of iterations to converge (e.g. weak regularization), where t may become very large. For instance, in experiments of training a linear SVM for adult data set (Teo et al., 2007), CRBM requires thousands of iterations for small values of λ. To overcome such an issue and to make our NRBM practical for large scale and difficult optimization problems we propose a limited memory mechanism. It is based on the use of a cutting plane aggregation method which allows drastically limiting the number of CPs in the working set at the price of a less accurate underestimator approximation. Note that such a limited memory variant may be used with convex and non-convex risks. Also, this limited memory variant applied to convex risks may be shown to inherit the convergence rate (w.r.t. the number of iterations) of CRBM, while the cost of every iteration does not depend on the iteration number anymore.

To ease the presentation, we will present in Section 3.1 the limited memory variant of bundle methods for the special case of convex risks. Then, we will consider in Section 3.2 our non-convex extension of CRBM for dealing with non-convex risks, named Non-convex Regularized Bundle Method, with includes as a particular feature the limited memory strategy.

Algorithm 3 Limited memory CRBM

Input: $\mathbf{w}_1$, R, λ, ε, M
Output: $\mathbf{w}^*$
Compute $\mathbf{a}_{\mathbf{w}_1}$ and $b_{\mathbf{w}_1}$ of R at $\mathbf{w}_1$
$\tilde{\mathbf{w}}_1 = -\mathbf{a}_1/\lambda$, $\tilde{\mathbf{a}}_1 = \mathbf{a}_{\mathbf{w}_1}$; $\tilde{b}_1 = b_{\mathbf{w}_1}$; $J_1 = \{1\}$
for $t = 2$ to ∞ **do**
  Compute new CP $(\mathbf{a}_{\mathbf{w}_t}, b_{\mathbf{w}_t})$ of R at $\mathbf{w}_t$
  $\mathbf{w}_t^* = \text{argmin}_{\mathbf{w} \in \{\mathbf{w}_1, \ldots \mathbf{w}_t\}} f(\mathbf{w})$
  $J_t \leftarrow \text{UpdateWorkingSet}(J_{t-1}, t, M)$
  $[\tilde{\mathbf{w}}_t, \tilde{c}_t] \leftarrow$ Minimize $g_t(\mathbf{w})$ in Equation 11
  $gap_t = f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t)$
  if $gap_t < \varepsilon$ **then return** $\mathbf{w}_t^*$
end for

3.1 Limited Memory for Convex Case

Our goal here is to limit the number of cutting planes used in the approximation function, which can be done by removing some of the previous cutting planes if the number of cutting planes reaches a given limit. However, the approximation gap is no more guaranteed to decrease after each iteration if one removes some of the CPs without care. The subgradient aggregation technique (Kiwiel, 1983) appears then to be an appealing solution since it can be used to accumulate information from multiple subgradients. Our proposal is to apply a similar technique to the set of cutting planes approximation of the risk function R, yielding an aggregated cutting plane.[4] Interestingly, we can show that if such an aggregated cutting plane is included in the approximation function, then one can remove any (or even all) previous cutting plane(s) while preserving the theoretical convergence rate $O(1/\lambda\varepsilon)$ iterations of CRBM.

Recall that the approximation function at iteration t is :

$$g_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max\left(\left(\max_{j \in J_t} c_j(\mathbf{w})\right), \tilde{c}_{t-1}(\mathbf{w})\right) \tag{11}$$

where $J_t \subset \{1, .., t\}$ stands for a working set of active cutting plane indexes that we keep at iteration t and $\tilde{c}_{t-1}(\mathbf{w}) = \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w} \rangle + \tilde{b}_{t-1}$ is the aggregated cutting plane which accumulates information from previous cutting planes, $c_1, ..., c_{t-1}$.

The limited memory CRBM is described in Algorithm 3. It takes as input an initial solution $\mathbf{w}_1$, the convex risk function R, the regularization parameter λ, the tolerance ε, and the maximum number of active CPs $M \geq 1$. It produces as output a solution of the optimization problem, $\mathbf{w}^*$. The principle of the algorithm is similar to CRBM except that one has to decide how to define J_t via the function UpdateWorkingSet(J_{t-1}, t, M) and how to define the aggregated cutting plane.

UpdateWorkingSet. At iteration t, a new cutting plane is added to the current set of cutting planes J_{t-1}, but if J_{t-1} is full (i.e., $|J_{t-1}| = M$) then we need to select a cutting plane in J_{t-1} to remove. A simple strategy is to replace the oldest cutting plane in J_{t-1} by the new one: $J_t = J_{t-1} \cup \{t\} \setminus \{t - M - 1\}$. Alternately, one may rely on a more sophisticated way for selecting which cutting plane to

4. We prefer this terminology to standard *aggregate subgradients* to stress that some cutting planes might be fully artificial and would not correspond to real subgradient of the risk in the non-convex case.

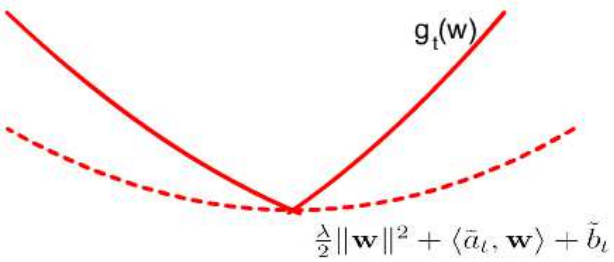


Figure 4: Quadratic underestimator of $g_t(\mathbf{w})$ (solid line) and corresponding aggregated cutting plane $\tilde{c}_t(\mathbf{w})$ (dash line)).

remove from J_{t-1}. In our implementation, we maintain a count for each CP which is the number of iterations in which the CP does not contribute to the aggregation CP (see below for details about the definition of the aggregation CP). Then the CP with highest count is selected to be removed.

Cutting plane aggregation. The use of an aggregated cutting plane is a key issue to limit storage requirements and computational effort per iteration. The technique is inspired by the subgradient aggregation idea of Kiwiel (1983), which can be viewed as building a low cost approximation of the piece-wise quadratic function in Equation 4. Basically, by considering a linear combination of subgradient of f computed in previous iterations, we can discard previous subgradients without losing all information. In our method, we also use aggregation technique for building a low cost approximation of the approximation function $g_t(\mathbf{w})$. Note that we use a slightly different terminology (CP aggregation instead of subgradient aggregation) since our goal is to build an approximation of f using cutting planes, rather than building an approximation of subdifferential as in standard bundle methods which aims at finding a solution with small sub-gradient. There are two key differences between our CP aggregation technique and the subgradient aggregation proposed originally by Kiwiel (1983). First, our method is specifically designed for quadratically regularized objective which makes possible to show that our limited memory variant using CP aggregation inherits the theoretical convergence rate of CRBM (as least for convex risks). Instead the standard subgradient aggregation technique can be applied to any objective function by using an additional regularization term in the search direction optimization problem. Second, while the original method focuses on aggregating subgradients, our algorithm applies the aggregation idea to both the direction, $\tilde{a}$, and to the offset, $\tilde{b}$ (and also to the locality measure in the non convex case, see later in Section 3.2.4).

At iteration t of Algorithm 3, the cutting plane aggregation $\tilde{c}_t(\mathbf{w})$ is derived from the minimization of $g_t(\mathbf{w})$. We use the cutting plane technique to build an underestimator of $g_t(\mathbf{w})$ at its minimum $\tilde{\mathbf{w}}_t = \operatorname{argmin}_{\mathbf{w}} g_t(\mathbf{w})$. Although any linear combination of previous cutting planes could yield an under estimator of $g_t(\mathbf{w})$, only one of them, that we note $\tilde{c}_t(\mathbf{w})$ hereafter, corresponds to a tight quadratic approximation $\frac{\lambda}{2}\|\mathbf{w}\|^2 + \tilde{c}_t(\mathbf{w})$ that reaches the same minimum as $g_t(\mathbf{w})$:

$$\tilde{\mathbf{w}}_t = \operatorname*{argmin}_{\mathbf{w}} g_t(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \tilde{c}_t(\mathbf{w}).$$

The particular property of $\tilde{c}_t(\mathbf{w})$ is important since it allows to guarantee that for the limited memory version of the algorithm, the gap between the best observed objective value and the minimum of the approximated function is unchanged even if one discards all previous cutting planes.

Figure 4 illustrates the quadratic function (in red dash line) derived from the aggregated cutting plane at iteration $t = 2$. The cutting plane $\tilde{c}_t(\mathbf{w})$ can be defined based on the dual solution of the approximation problem which may be characterized in primal and dual forms as follows:

$$\begin{array}{llll} & \textit{Primal} & & \textit{Dual} \\ \min_{\mathbf{w}} & \frac{\lambda}{2}\|\mathbf{w}\|^2+\xi & \max_{\boldsymbol{\alpha}} & -\frac{1}{2\lambda}\|\boldsymbol{\alpha}A_t\|^2+\boldsymbol{\alpha}B_t \\ s.t & \langle a_j,\mathbf{w}\rangle+b_j\leq\xi\ \forall j\in J_t & s.t & \alpha_j\geq 0\ \forall j\in J_t;\tilde{\alpha}\geq 0 \\ & \langle \tilde{a}_{t-1},\mathbf{w}\rangle+\tilde{b}_{t-1}\leq\xi & & (\sum_{j\in J_t}\alpha_j)+\tilde{\alpha}=1 \end{array}$$

where $A_t = [...;\mathbf{a}_j;...,\tilde{\mathbf{a}}_{t-1}]$ is a matrix (with $\mathbf{a}_j$ and $\tilde{\mathbf{a}}_{t-1}$ being row vectors), $B_t = [...;b_j;...;\tilde{b}_{t-1}]$ is the vector of scalars and $\boldsymbol{\alpha}$ stands for the (row) vector of Lagrange multipliers (of length $|J_t|+1$ at iteration t). We denote α_j as the Lagrange multiplier associated with the CP c_j and we denote $\tilde{\alpha}$ as the Lagrange multiplier associated with the aggregated CP $\tilde{c}_{j-1}$. Let $\boldsymbol{\alpha}_t$ be the solution of the above dual program then the minimizer of the primal can be expressed as:

$$\tilde{\mathbf{w}}_t = -\frac{\boldsymbol{\alpha}_t A_t}{\lambda} = -\frac{\sum_{j\in J_t}\alpha_j\mathbf{a}_j+\tilde{\alpha}\tilde{\mathbf{a}}_{t-1}}{\lambda}.$$

The following proposition show how to use $\boldsymbol{\alpha}_t$ for defining a tight underestimator of $g_t(\mathbf{w})$.

Proposition 1 *Let $\tilde{c}_t(\mathbf{w}) = \langle\tilde{\mathbf{a}}_t,\mathbf{w}\rangle+\tilde{b}_t$ be the aggregated CP defined by:*

$$\begin{array}{ll} \tilde{\mathbf{a}}_t & = \boldsymbol{\alpha}_t A_t = \sum_{j\in J_t}\alpha_j\mathbf{a}_j+\tilde{\alpha}\tilde{\mathbf{a}}_{t-1}, \\ \tilde{b}_t & = \boldsymbol{\alpha}_t B_t = \sum_{j\in J_t}\alpha_j b_j+\tilde{\alpha}\tilde{b}_{t-1} \end{array}$$

then the quadratic function $\frac{\lambda}{2}\|\mathbf{w}\|^2+\tilde{c}_t(\mathbf{w})$ is an underestimator of $g_t(\mathbf{w})$, which reaches the same minimum value as $g_t(\mathbf{w})$ at the same point, $\tilde{\mathbf{w}}_t$.

Proof First, by construction we have $\tilde{\mathbf{w}}_t = -\frac{\tilde{\mathbf{a}}_t}{\lambda}$ which implies that the derivative of $\frac{\lambda}{2}\|\mathbf{w}\|^2+\tilde{c}_t(\mathbf{w})$ is null at $\tilde{\mathbf{w}}_t$. Second, we can show that $\frac{\lambda}{2}\|\tilde{\mathbf{w}}_t\|^2+\tilde{c}_t(\tilde{\mathbf{w}}_t) = g_t(\tilde{\mathbf{w}}_t)$. Actually:

$$\begin{array}{lll} g_t(\tilde{\mathbf{w}}_t) & = -\frac{1}{2\lambda}\|\boldsymbol{\alpha}_t A_t\|^2+\boldsymbol{\alpha}_t B_t & = -\frac{\lambda}{2}\|\frac{\tilde{\mathbf{a}}_t}{\lambda}\|^2+\tilde{b}_t \\ & = \frac{\lambda}{2}\|\frac{\tilde{\mathbf{a}}_t}{\lambda}\|^2-\lambda\|\frac{\tilde{\mathbf{a}}_t}{\lambda}\|^2+\tilde{b}_t & = \frac{\lambda}{2}\|\tilde{\mathbf{w}}_t\|^2-\langle\tilde{\mathbf{a}}_t,\frac{\tilde{\mathbf{a}}_t}{\lambda}\rangle+\tilde{b}_t \\ & = \frac{\lambda}{2}\|\tilde{\mathbf{w}}_t\|^2+\langle\tilde{\mathbf{a}}_t,\tilde{\mathbf{w}}_t\rangle+\tilde{b}_t. & \end{array} \quad (12)$$

In other words, the quadratic function $\frac{\lambda}{2}\|\mathbf{w}\|^2+\tilde{c}_t(\mathbf{w})$ and the approximation function $g_t(\mathbf{w})$ reach the same minimum value $g_t(\tilde{\mathbf{w}})$ at the same point $\tilde{\mathbf{w}}_t$.

Finally, we show that $\frac{\lambda}{2}\|\mathbf{w}\|^2+\tilde{c}_t(\mathbf{w})$ is an underestimator of $g_t(\mathbf{w})$. Let

$$h_t(\mathbf{w}) = \max\left[\max_{j=\in J_t}\langle\mathbf{a}_j,\mathbf{w}\rangle+b_j,\langle\tilde{\mathbf{a}}_{t-1},\mathbf{w}\rangle+\tilde{b}_{t-1}\right]$$

be the piecewise linear approximation of $R(\mathbf{w})$ at iteration t, we have:

$$0\in\partial g_t(\tilde{\mathbf{w}}_t)\equiv\lambda\tilde{\mathbf{w}}_t+\partial h_t(\tilde{\mathbf{w}}_t)$$

since $\tilde{\mathbf{w}}_t$ is the optimum solution of minimizing $g_t(\mathbf{w})$. Note that $\tilde{\mathbf{a}}_t = -\lambda\tilde{\mathbf{w}}_t$, the above equation implies that $\tilde{\mathbf{a}}_t\in\partial h_t(\tilde{\mathbf{w}}_t)$. In other words, $\tilde{\mathbf{a}}_t$ is a subgradient of $h_t(\mathbf{w})$ at $\tilde{\mathbf{w}}_t$. Furthermore, since $g_t(\tilde{\mathbf{w}}_t) = \frac{\lambda}{2}\|\tilde{\mathbf{w}}_t\|^2+h_t(\tilde{\mathbf{w}}_t)$, Equation 12 gives:

$$\langle\tilde{\mathbf{a}}_t,\tilde{\mathbf{w}}_t\rangle+\tilde{b}_t = h_t(\tilde{\mathbf{w}}_t).$$

The cutting plane $\tilde{c}_t(\mathbf{w})$ is then an underestimator of $h_t(\mathbf{w})$ built at $\tilde{\mathbf{w}}_t$ (recall that $h_t(\mathbf{w})$ is convex), and thus $\frac{\lambda}{2}\|\mathbf{w}\|^2 + \tilde{c}_t(\mathbf{w})$ is a quadratic underestimator of $g_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + h_t(\mathbf{w})$. Note that since $\frac{\lambda}{2}\|\mathbf{w}\|^2 + \tilde{c}_t(\mathbf{w})$ is an underestimator of $g_t(\mathbf{w})$ and $g_t(\mathbf{w})$ is an underestimator of $f(\mathbf{w})$ at $\mathbf{w}_t^*$, the quadratic function $\frac{\lambda}{2}\|\mathbf{w}\|^2 + \tilde{c}_t(\mathbf{w})$ is also an underestimator of $f(\mathbf{w})$ at $\mathbf{w}_t^*$.

■

3.2 Regularized Bundle Method for Non-Convex Risks

To handle non-convex objective function, we introduce some new notations in addition to the notation used in Algorithm 3. In the following, we recall useful notations from previous section, and we introduce additional notations that will be useful hereafter.

Notations from limited memory CRBM. At iteration t, $\mathbf{w}_t$ is the current solution and $\mathbf{w}_t^*$ is the best observed solution. J_t corresponds to the working set of cutting plane, which is involved in the definition of the approximation $g_t(\mathbf{w})$. $\tilde{\mathbf{w}}_t$ is the solution of the minimization of $g_t(\mathbf{w})$, it is also considered as the solution in the next iteration.

Raw and modified cutting planes. We have to distinguish between a raw linear cutting plane of the risk $c_{\mathbf{w}_j}$ (with $c_{\mathbf{w}_j}(\mathbf{w}) = \langle \mathbf{a}_{\mathbf{w}_j}, \mathbf{w} \rangle + b_{\mathbf{w}_j}$) that is built at a particular iteration j of the algorithm and the eventually modified versions of this cutting plane that might be used in posterior iterations. Indeed a cutting plane may be modified multiple times for solving conflicts as in standard NBM method. At iteration t we note c_j^t (with $c_j^t(\mathbf{w}) = \langle \mathbf{a}_j, \mathbf{w} \rangle + b_j^t$) the cutting plane which is derived from $c_{\mathbf{w}_j}$, the raw CP originally built at iteration j. Unlike NBM, the normal vector $\mathbf{a}_j$ in our algorithm might be different than the subgradient $\mathbf{a}_{\mathbf{w}_j}$ computed at $\mathbf{w}_j$, due to our particular solving conflict method. However, once defined at iteration j, the normal vector $\mathbf{a}_j$ remains fixed over iterations. On the contrary, the offset might be modified multiple times for solving conflicts occurring after iteration j, and we use a superscript t indicating the iteration number for the cutting plane's offset b_j^t.

Bundle. The bundle $\mathbb{B}_t$ denotes the state of the algorithm at iteration t. It consists in a set of cutting planes which were built at previous solutions, c_j^t for $j \in J_t$. Similarly to non-convex bundle methods, we define a locality measure which is associated to any active cutting plane. It is related to the locality measure between the cutting plane (actually the point where the cutting plane was built) and the best current observed solution. We note s_j^t the locality measure between cutting plane c_j^t and the best observed solution up to iteration t, $\mathbf{w}_t^*$. The full bundle information is:

$$\mathbb{B}_t = \{c_j^t, s_j^t\}_{j \in J_t} \cup \{\tilde{c}_{t-1}^t, \tilde{s}_{t-1}^t\}$$

where $\tilde{c}_{t-1}^t$ is an aggregated cutting plane and $\tilde{s}_{t-1}^t$ is its locality measure to the best observed solution $\mathbf{w}_t^*$. Similar to the aggregation technique presented in Section 3.1, the aggregated CP $\tilde{c}_{t-1}^t$ can be viewed as a convex combination of CPs in previous iterations. For non-convex objective function, each CP in the bundle is associated with a locality measure, including the aggregated CPs whose locality measure is a convex combinations of locality measures of other CPs.

3.2.1 SKETCH OF ALGORITHM

The main algorithm is described in Algorithm 4, for which the input is similar to the case of Algorithm 3 except the fact that the risk R can be non-convex. To deal with non-convexity, the key idea to use CPs in the bundle to build a local underestimator of f around the best observed solution.

Algorithm 4 NRBM

Input: $\mathbf{w}_1$,R, λ, ε,M
Output: $\mathbf{w}^*$
Initialization:
  Compute cutting plane $c_{\mathbf{w}_1}$ of R
  $[c_1^1, s_1^1] = [\tilde{c}_1^1, \tilde{s}_1^1] = [c_{\mathbf{w}_1}, 0]$
  $\tilde{\mathbf{w}}_1 = -\mathbf{a}_1/\lambda$
  $\mathbb{B}_1 = \{c_1^1, s_1^1, \tilde{c}_1^1, \tilde{s}_1^1\}$
for $t = 2$ to ∞ **do**
  $\mathbf{w}_t \leftarrow \tilde{\mathbf{w}}_{t-1}$
  Compute cutting plane $c_{\mathbf{w}_t}$ of R
  $\mathbf{w}_t^* = \text{argmin}_{\mathbf{w} \in \{\mathbf{w}_1, \ldots \mathbf{w}_t\}} f(\mathbf{w})$
  $\mathbb{B}_t$ = UpdateBundle($\mathbb{B}_{t-1}, \mathbf{w}_{t-1}^*, \mathbf{w}_t^*, c_{\mathbf{w}_t}, \mathbf{w}_t$,M)
  $(\tilde{\mathbf{w}}_t, \tilde{c}_t^t, \tilde{s}_t^t)$ = MinimizeApproximationProblem($\mathbb{B}_t$,λ)
  $gap_t = f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t)$
  if $gap_t < \varepsilon$ **then return** $\mathbf{w}_t^*$
end for

Similar to CRBM and limited memory CRBM, the approximation problem is designed in such a way that one can use the minimum of the approximation problem as the new current solution. In other words, NRBM does not require a dedicated line search procedure to ensure convergence as in the standard NBM (Kiwiel, 1985). Such a line search is not required for convergence matters in our method but it may be still used for improving convergence rate in practice (see Section 3.3.2).

Initialization

Initialization consists in providing a first bundle $\mathbb{B}_1$. Starting with an initial solution $\mathbf{w}_1$, we build the first cutting plane $c_1^1 = c_{\mathbf{w}_1} = \langle \mathbf{a}_{\mathbf{w}_1}, \mathbf{w} \rangle + b_{\mathbf{w}_1}$. Note that at iteration $t = 1$, there is only one cutting plane c_1^1 and the aggregated cutting plane is also c_1^1: $[\tilde{c}_1^1, \tilde{s}_1^1] = [c_1^1, s_1^1]$. The *approximation function* is then:

$$g_1(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \langle \mathbf{a}_1, \mathbf{w} \rangle + b_1^1$$

which reaches its minimum at $\tilde{\mathbf{w}}_1 = -\mathbf{a}_1/\lambda$. The state of algorithm $\mathbb{B}_1$ is set to c_1^1 and $\tilde{c}_1^1$ (which coincide) with their corresponding locality mesures to the best solution $\mathbf{w}_1$ ($\tilde{s}_1^1 = s_1^1 = 0$).

Iteration **t**

Every iteration the algorithm determine a new bundle $\mathbb{B}_t$, the best observed solution up to iteration t, $\mathbf{w}_t^*$, and the new current (and temporary) solution $\mathbf{w}_t$. At iteration $t > 1$, few steps are successively performed:

- Build a new cutting plane at $\tilde{\mathbf{w}}_{t-1}$ the minimizer of approximation function in previous iteration ($g_{t-1}(\mathbf{w})$).
- Update the best observed solution $\mathbf{w}_t^*$.
- Solve any conflict between the best observed solution,$\mathbf{w}_t^*$, and all cutting planes in the bundle. This is done through a call to UpdateBundle function which we detail later. This yields a piece-wise quadratic function g_t which is a local underestimator approximation of f. As said before, in addition to cutting planes built at previous solutions (e.g. at $\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}$), we use a

special aggregated cutting plane, $\tilde{c}^t_{t-1}$ for gathering information of previous cutting planes up to iteration $t-1$. The approximation function at iteration t is then:

$$g_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max\left[\max_{j\in J_t} c^t_j(\mathbf{w}), \tilde{c}^t_{t-1}(\mathbf{w})\right] \tag{13}$$

where, as in Section 3.1, J_t stands for a subset of cutting planes defined in previous iterations if one wishes to use a limited memory variant.

- Minimize g_t. This gives a solution named $\tilde{\mathbf{w}}_t$ which will be used in next iteration. Note that a side effect of this minimization is the definition of a new aggregated cutting plane and its locality measure to the best observed solutions.

This procedure is repeated until the gap (i.e. the difference between the best observed value of objective function and the minimum of the approximation function) is less than a desired accuracy ε. We say that an ε-solution has been reached.

We detail in the following sections how the approximation is built and procedure for solving conflict in the update of the bundle. Then we provide details on our definition of the aggregated cutting plane.

3.2.2 LOCALITY MEASURE AND CONDITIONS ON CPS

Given a set of cutting plane approximation of R, one could build a local underestimator of f in the vicinity of $\mathbf{w}$ by descending CPs that yields non positive linearization error of f at $\mathbf{w}$. Our algorithm focus on solving conflicts between CPs in the bundle and the best observed solution $\mathbf{w}^*_t$. While sharing some concepts with NBM such as locality measure, null step and descent step our method is based on a new greedy strategy for solving conflicts which guarantee a minimum improvement of the approximation gap after each iteration which is similar to CRBM.[5]

Locality measure definition. We propose to define the locality measure between a cutting plane previously built at iteration j and the current best solution $\mathbf{w}^*_t$ based on the trajectory from $\mathbf{w}_j$ to $\mathbf{w}^*_t$. We exploit the same shape of our regularization term (L2 norm) to define our locality measure.[6] At iteration t, we define the locality measure between CP c^t_j built at $\mathbf{w}_j$ and $\mathbf{w}^*_t$ as:

$$s^t_j = s(\mathbf{w}_j, \mathbf{w}^*_t) = \frac{\lambda}{2}\left(\|\mathbf{w}_j - \mathbf{w}^*_j\|^2 + \sum_{k=j+1}^{t} \|\mathbf{w}^*_k - \mathbf{w}^*_{k-1}\|^2\right)$$

which yields a natural recursive formulate:

$$s^t_j = s^{t-1}_j + \frac{\lambda}{2}\|\mathbf{w}^*_t - \mathbf{w}^*_{t-1}\|^2, \forall j < t.$$

Lower bound and upper bound on offset adjustment. As in NBM, raw CP cannot always be used to build an underestimator of $f(\mathbf{w})$, which is non-convex so that CP need adjustments. We discuss two conditions that define an upper and an underestimator on a CP's offset modification when solving a conflict with respect to $\mathbf{w}^*_t$.

5. Note that we use the terminology descent step instead of serious steps since descent step here is not fully similar to serious step in standard non convex bundle methods.
6. Standard bundle methods use γd^ω where d is the Euclidean distance and $\gamma > 0$ and ω are hyper parameters (Cf. Equation 10).

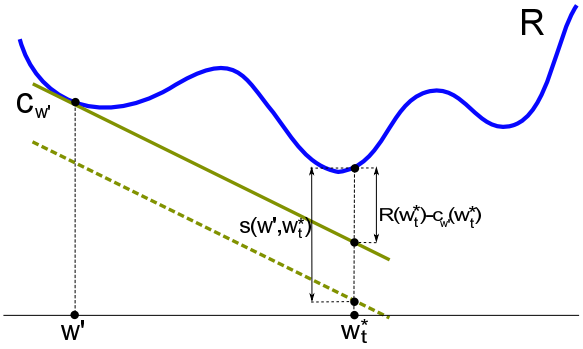


Figure 5: Conflict between $\mathbf{w}_t^*$ and a cutting plane $c_{\mathbf{w}'}$.

First, as in standard NBM (recall Equation 10), we consider the following first *condition* requiring that a CP built at $\mathbf{w}'$, $c_{\mathbf{w}'}$, gives a positive linearization error at $\mathbf{w}_t^*$, which must grow with the locality measure of the CP to $\mathbf{w}_t^*$:

$$R(\mathbf{w}_t^*) - c(\mathbf{w}_t^*) \geq s(\mathbf{w}', \mathbf{w}_t^*) \tag{14}$$

where $s(.,.)$ is our non-negative locality measure between the two points. The positive value of $s(\mathbf{w}', \mathbf{w}_t^*)$ ensures that the linear approximation $c_{\mathbf{w}'}(\mathbf{w})$ is an underestimator of $R(\mathbf{w})$ at least within a small region around $\mathbf{w}_t^*$. Figure 5 illustrates this case. The cutting plane $c_{\mathbf{w}'}$ which was built at $\mathbf{w}'$ does not satisfy condition 14. This conflict between cutting plane $c_{\mathbf{w}'}$ and $\mathbf{w}_t^*$ is solved in NBM by lowering $c_{\mathbf{w}'}$ (by tuning its offset b') so that the linearization error at $\mathbf{w}_t^*$, $R(\mathbf{w}_t^*) - c_{\mathbf{w}'}(\mathbf{w}_t^*)$, becomes at least $s(\mathbf{w}', \mathbf{w}_t^*)$. This yield an *upper bound* on the new offset b':

$$b' \leq R(\mathbf{w}_t^*) - \langle \mathbf{a}', \mathbf{w}_t^* \rangle - s(\mathbf{w}', \mathbf{w}_t^*). \tag{15}$$

Unfortunately if a cutting plane is lowered too much, the minimum of the approximation function is not guaranteed to improve every iteration anymore. For instance it may happen that the minimum of the approximated function is not changed once the new cutting plane has been lowered, yielding a infinite loop without any improvement on the solution. Standard non-convex bundle methods handle this problem with a special line search procedure (between the current best observed solution and the minimum of the approximation problem) with stopping conditions that ensure some minimal changes of the approximation problem.

We found instead that there is a simple sufficient condition that guarantees an improvement of the minimum of the approximation function every iteration (required by Lemma 4). It concerns the new added cutting plane only and writes: $\frac{\lambda}{2}\|\mathbf{w}_t\|^2 + \langle \mathbf{a}_t, \mathbf{w}_t \rangle + b_t^t \geq f(\mathbf{w}_t^*)$. In other words, we need to ensure that the approximation at $\mathbf{w}_t$ using the new added cutting plane is greater or equal to the best observed function value. Note that $\mathbf{w}_t$ is the minimizer of the approximation in the previous iteration, $g_{t-1}(\mathbf{w})$, this condition influences directly the gap between the best observed function value and the minimum of the approximation. The condition can be seen as a lower bound on the modified offset:

$$b_t^t \geq f(\mathbf{w}_t^*) - \frac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \mathbf{a}_t, \mathbf{w}_t \rangle. \tag{16}$$

Algorithm 5 UpdateBundle

Input: $\mathbb{B}_{t-1} = \{c_j^{t-1}, s_j^{t-1}\}_{j \in J_{t-1}} \cup \{\tilde{c}_{t-1}^{t-1}, \tilde{s}_{t-1}^{t-1}\}, \mathbf{w}_{t-1}^*, \mathbf{w}_t^*, \mathbf{w}_t, c_{\mathbf{w}_t}, M$
Output: $\mathbb{B}_t = \{c_j^t, s_j^t\}_{j \in J_t} \cup \{\tilde{c}_{t-1}^t, \tilde{s}_{t-1}^t\}$
if $\mathbf{w}_t^* \neq \mathbf{w}_{t-1}^*$ **then** Descent Step
  for $j \in J_{t-1}$
    $s_j^t = s_j^{t-1} + \frac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_{t-1}^*\|^2$
    $b_j^t = \min[b_j^{t-1}, R(\mathbf{w}_t^*) - \langle \mathbf{a}_j, \mathbf{w}_t^* \rangle - s_j^t]$
  end
  $\tilde{s}_{t-1}^t = \tilde{s}_{t-1}^{t-1} + \frac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_{t-1}^*\|^2$
  $\tilde{b}_{t-1}^t = \min[\tilde{b}_{t-1}^{t-1}, R(\mathbf{w}_t^*) - \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w}_t^* \rangle - \tilde{s}_{t-1}^t]$
  $\tilde{c}_{t-1}^t(\mathbf{w}) := \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w} \rangle + \tilde{b}_{t-1}^t$
  $[c_t^t, s_t^t] = [c_{\mathbf{w}_t}, 0]$
else Null Step
  for $j \in J_{t-1}$
    $c_j^t = c_j^{t-1}$; $s_j^t = s_j^{t-1}$;
  end
  $\tilde{c}_{t-1}^t = \tilde{c}_{t-1}^{t-1}$; $\tilde{s}_{t-1}^t = \tilde{s}_{t-1}^{t-1}$;
  if condition (15) is not satisfied for $c_{\mathbf{w}_t}$ **then**
    $[c_t^t, s_t^t]$ = SolveConflictNullStep($\mathbf{w}_t^*, \mathbf{w}_t, c_{\mathbf{w}_t}$)
  else $[c_t^t, s_t^t] = [c_{\mathbf{w}_t}, \frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}_t^*\|^2]$
end
J_t =UpdateWorkingSet(J_{t-1}, t, M)
return $\mathbb{B}_t = \{c_j^t, s_j^t\}_{j \in J_t}\} \cup \{\tilde{c}_{t-1}^t, \tilde{s}_{t-1}^t\}$

3.2.3 BUNDLE UPDATE

The approximation function, g_t, is refined every iteration, Algorithm 5 describes the *UpdateBundle* process. It takes as input:

- The bundle at previous iteration
- The best observed solutions at previous iteration $\mathbf{w}_{t-1}^*$
- The best observed solutions at current iteration $\mathbf{w}_t^*$
- The current solution $\mathbf{w}_t$ and its corresponding raw cutting plane, $c_{\mathbf{w}_t}$.

The algorithm is designed so that at the end of iteration t, all $(|J_t| + 1)$ cutting planes in the bundle (i.e. the $|J_t|$ "normal" cutting planes and the aggregated cutting plane) satisfy condition in Equation 15 while the new added cutting plane c_t^t also satisfies condition in Equation 16. Note that $c_{\mathbf{w}_t}$ always satisfies (16) by definition of $\mathbf{w}_t^*$, so that c_t^t also satisfies (16) in case there is no conflict ($c_t^t \equiv c_{\mathbf{w}_t}$).

As the two conditions (15) and (16) involve the best observed solution, we distinguish two cases when solving conflict. Either the current solution is the best solution up to now (hence $\mathbf{w}_t^* \neq \mathbf{w}_{t-1}^*$), in which case we call the iteration a descent step. Or the current solution is not the best solution (i.e. $\mathbf{w}_t^* \equiv \mathbf{w}_{t-1}^*$), then the iteration is said to be a null step. We detail these two cases now.

Descent Step. In the case of a descent step, condition (16) is trivially satisfied for the new added cutting plane since $c_t^t \equiv c_{\mathbf{w}_t}$. Hence solving an eventual conflict is rather simple in this case. It is done by setting:

$$b_j^t = \min[b_j^{t-1}, R(\mathbf{w}_t^*) - \langle \mathbf{a}_j, \mathbf{w}_t^* \rangle - s_j^t]$$

Algorithm 6 SolveConflictNullStep

1: **Input:** $\mathbf{w}_t^*, \mathbf{w}_t, c_{\mathbf{w}_t}$ with parameters $(\mathbf{a}_{\mathbf{w}_t}, b_{\mathbf{w}_t})$
2: **Output:** c_t^t with parameters $(\mathbf{a}_t, b_t^t)$ and s_t^t
3: $s_t^t = \frac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_t\|^2$
4: Compute L, U according to Equation 17
5: **if** $L \leq U$ **then** $[\mathbf{a}_t, b_t^t] = [\mathbf{a}_{\mathbf{w}_t}, L]$ **else**
6: $\quad \mathbf{a}_t = -\lambda \mathbf{w}_t^*$ NullStep2 case
7: $\quad b_t^t = f(\mathbf{w}_t^*) - \frac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \mathbf{a}_t, \mathbf{w}_t \rangle$

for all j in the working set. A similar modification may be applied to the aggregated cutting plane:

$$\tilde{b}_{t-1}^t = \min[\tilde{b}_{t-1}^{t-1}, R(\mathbf{w}_t^*) - \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w}_t^* \rangle - \tilde{s}_{t-1}^t]$$

where $\tilde{s}_{t-1}^t = \tilde{s}_{t-1} + \frac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_{t-1}^*\|^2$. At the end, the adjusted aggregated CP (in the working set of iteration t) is:

$$\tilde{c}_{t-1}^t(\mathbf{w}) = \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w} \rangle + \tilde{b}_{t-1}^t.$$

Null Step. In the case of a null step, the best observed solution did not change, so that $s_j^t = s_j^{t-1}, \forall j = 1, ..., (t-1)$ and $\tilde{s}_{t-1}^t = \tilde{s}_{t-1}^{t-1}$. Since all cutting planes in $\mathbb{B}_{t-1}$ were already adjusted to satisfy positive linearization error condition wrt. the best solution at previous iteration, a conflict (if any) may only arise between the new cutting plane $c_{\mathbf{w}_t}$ and the best observed solution $\mathbf{w}_t^*$. So that all CPs (including aggregated CP) remain unchanged (see Algorithm 5 line 13) except the new added CP which must be checked for conflict.

In the null step case, solving conflict is not as simple as in a descent step case since as we said before, for convergence proof matters, we need the new cutting plane to satisfy both conditions (15) and (16). Algorithm 6 modifies c_t^t in such a way that it guarantees that the new cutting plane c_t^t with parameters $\mathbf{a}_t$ and b_t^t satisfies conditions (15) and (16). In a first attempt it tries to solve the conflict by tuning b_t^t alone while fixing $\mathbf{a}_t = \mathbf{a}_{\mathbf{w}_t}$. Indeed conditions (15) and (16) may be rewritten as:

$$\begin{aligned} b_t^t &\leq R(\mathbf{w}_t^*) - \langle \mathbf{a}_{\mathbf{w}_t}, \mathbf{w}_t^* \rangle - s_t^t = U, \\ b_t^t &\geq f(\mathbf{w}_t^*) - \tfrac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \mathbf{a}_{\mathbf{w}_t}, \mathbf{w}_t \rangle = L \end{aligned} \tag{17}$$

which define an upper bound U and a lower bound L for b_t^t. If $L \leq U$ any value in (L, U) works (in our implementation we set $b_t^t = L$).

However it may happen that $L > U$, then tuning b_t^t is not enough (this is what we call a NullStep2 case in Algorithm 6). Both b_t^t and the normal vector $\mathbf{a}_t$ need to be adjusted to make sure that the conflict is solved (see Line 6 in Algorithm 6).

Figure 6(top-left) illustrates an example of NullStep2 where the gradient information given at $\mathbf{w}_t$ is not helpful for building a local underestimator approximation at $\mathbf{w}_t^*$. The quadratic approximation corresponding to cutting plane $c_{\mathbf{w}_t}$ is plotted in orange, which is not a local underestimator of $f(\mathbf{w})$ at $\mathbf{w}_t^*$. The conflict is so severe that it cannot be solved by just lowering the cutting plane. It should be lowered too much with respect to condition in Equation 15 (Figure 6 (top-right)), meaning that the approximation function would be unchanged and the algorithm would loop without finding a good solution.

In a NullStep2 case, we propose to ignore the gradient information at $\mathbf{w}_t$ and to rather focus on the region around the best observed solution $\mathbf{w}_t^*$ by adding a particular CP (leading to a quadratic

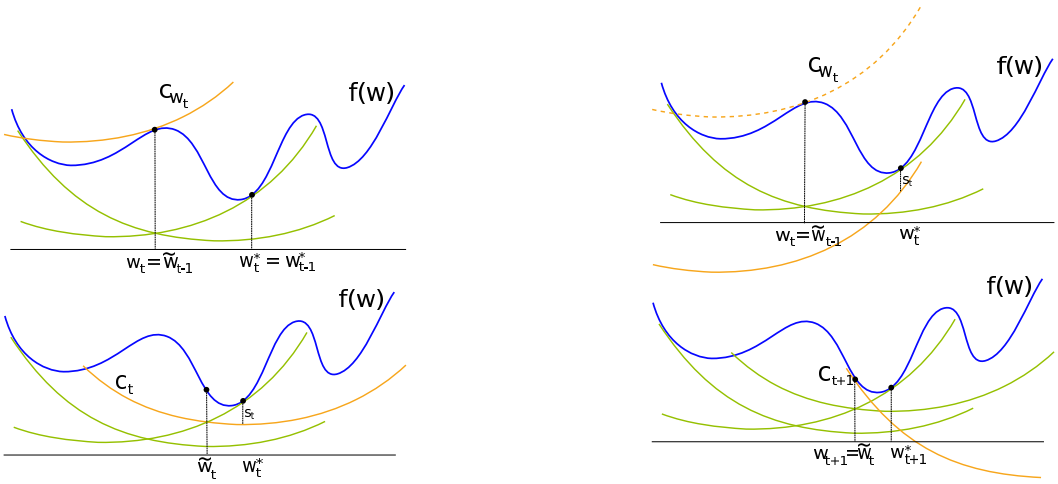


Figure 6: Illustration of NullStep2. Top-left: conflict arise at iteration t. Top-right: can not solve conflict by descend the cutting plane. Bottom-left: Nullstep2, modifying the cutting plane to solve the conflict at iteration t. Bottom-right: There is no conflict at iteration $t+1$.

local underestimator, $\frac{\lambda}{2}\|\mathbf{w}\|^2 + \langle \mathbf{a}_t, \mathbf{w}\rangle + b_t^t$) satisfying both conditions in Equation 15 and 16). This quadratic function is defined so that it reaches its minimum at $\mathbf{w}_t^*$ and the linearization error of the cutting plane $\langle \mathbf{a}_t, \mathbf{w}\rangle + b_t^t$ at $\mathbf{w}_t^*$ is $\frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}_t^*\|^2$ (see the orange quadratic curve in Figure 6 (bottom-left)). The new cutting plane is defined as:

$$\begin{aligned} c_t^t(\mathbf{w}) &= \langle \mathbf{a}_t, \mathbf{w}\rangle + b_t^t, \\ \mathbf{a}_t &= -\lambda \mathbf{w}_t^*, \\ b_t^t &= f(\mathbf{w}_t^*) - \tfrac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \mathbf{a}_t, \mathbf{w}_t\rangle, \\ s_t^t &= \tfrac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}_t^*\|^2. \end{aligned}$$

This CP satisfies condition (16) by construction. It also satisfies condition (15) as we show now:

$$\begin{aligned} \langle \mathbf{a}_t, \mathbf{w}_t^*\rangle + b_t^t &= \langle \mathbf{a}_t, \mathbf{w}_t^*\rangle + f(\mathbf{w}_t^*) - \tfrac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \mathbf{a}_t, \mathbf{w}_t\rangle \\ &= R(\mathbf{w}_t^*) + \langle \mathbf{a}_t, \mathbf{w}_t^* - \mathbf{w}_t\rangle + \tfrac{\lambda}{2}(\|\mathbf{w}_t^*\|^2 - \|\mathbf{w}_t\|^2) \\ &= R(\mathbf{w}_t^*) + \langle \mathbf{a}_t + \tfrac{\lambda}{2}(\mathbf{w}_t^* + \mathbf{w}_t), \mathbf{w}_t^* - \mathbf{w}_t\rangle \end{aligned}$$

where we used the definition of the objective function $f(\mathbf{w}_t^*) = \frac{\lambda}{2}\|\mathbf{w}_t^*\|^2 + R(\mathbf{w}_t^*)$. Then, substituting $-\lambda \mathbf{w}_t^*$ for $\mathbf{a}_t$ (Cf. Line 6) we obtain:

$$\begin{aligned} & \langle \mathbf{a}_t, \mathbf{w}_t^*\rangle + b_t^t &&= R(\mathbf{w}_t^*) - \tfrac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_t\|^2 \\ \iff\ & \langle \mathbf{a}_t, \mathbf{w}_t^*\rangle + b_t^t &&= R(\mathbf{w}_t^*) - s_t^t \\ \iff\ & b_t^t &&= R(\mathbf{w}_t^*) - \langle \mathbf{a}_t, \mathbf{w}_t^*\rangle - \tfrac{\lambda}{2}\|\mathbf{w}_t^* - \mathbf{w}_t\|^2 \end{aligned}$$

and condition in Equation 15 is satisfied.

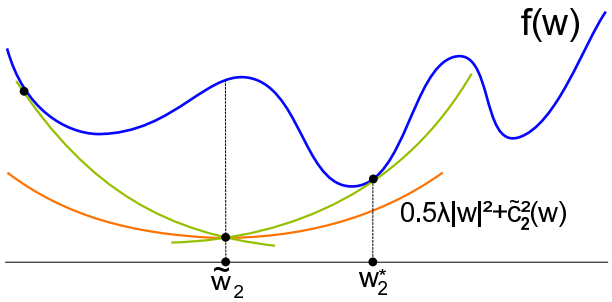


Figure 7: Quadratic underestimator of $g_t(\mathbf{w})$ derived from the aggregated cutting plane $\tilde{c}_t^t(\mathbf{w})$.

3.2.4 APPROXIMATED PROBLEM AND AGGREGATED CUTTING PLANE

In the non-convex case the aggregated CP is still an underestimator of approximation problem. Figure 7 illustrates the quadratic function (in orange) derived from the aggregated cutting plane at iteration $t = 2$.

Solving the approximated problem and definition of the aggregated cutting plane are completely similar to the case of limited memory CRBM, with the only difference that we use here at iteration t the bundle at iteration t that may include cutting planes that have been modified during previous iterations.The minimization of the approximation function ($g_t(\mathbf{w})$ in Equation 13) can be solved in the dual space as:

$$\begin{array}{llll} & \textit{Primal} & & \textit{Dual} \\ \min_{\mathbf{w}} & \frac{\lambda}{2}\|\mathbf{w}\|^2+\xi & \max_{\boldsymbol{\alpha}} & -\frac{1}{2\lambda}\|\boldsymbol{\alpha}A_t\|^2+\boldsymbol{\alpha}B_t \\ s.t & \langle a_j^t,\mathbf{w}\rangle + b_j^t \leq \xi \; \forall j \in J_t & s.t & \alpha_j \geq 0 \; \forall j \in J_t; \tilde{\alpha} \geq 0 \\ & \langle \tilde{a}_{t-1}^t,\mathbf{w}\rangle + \tilde{b}_{t-1}^t \leq \xi & & (\sum_{j\in tJ_t}\alpha_j)+\tilde{\alpha}=1 \end{array}$$

where $A_t = [...;\mathbf{a}_j^t;...;\tilde{\mathbf{a}}_{t-1}^t]$ is a matrix (with $\mathbf{a}_j^t$ and $\tilde{\mathbf{a}}_{t-1}^t$ being row vectors), $B_t = [...;b_j^t;...;\tilde{b}_{t-1}^t]$ is the vector of scalars and $\boldsymbol{\alpha}$ stands for the (row) vector of Lagrange multipliers (of length $|J_t|+1$ at iteration t). We denote α_j as the Lagrange multiplier associated with the CP c_j^t and we denote $\tilde{\alpha}$ as the Lagrange multiplier associated with the aggregated CP $\tilde{c}_{j-1}^t$. Let $\boldsymbol{\alpha}_t$ be the solution of the above dual program then the minimizer of the primal can be expressed as:

$$\tilde{\mathbf{w}}_t = -\frac{\boldsymbol{\alpha}_t A_t}{\lambda}.$$

Hence the definition of the aggregated cutting plane follows:

$$\begin{aligned} \tilde{\mathbf{a}}_t &= \boldsymbol{\alpha}_t A_t, \\ \tilde{b}_t &= \boldsymbol{\alpha}_t B_t. \end{aligned}$$

*Locality measure associated to the aggregated cutting plane.*The aggregated CP $\tilde{c}_t^t$ accumulates information from many cutting planes built at different points so that one cannot immediately define a locality measure $\tilde{s}_t^t$ between $\tilde{c}_t^t$ and the current best observed solution $\mathbf{w}_t^*$. However, $\tilde{c}_t^t$ being a convex combination of cutting planes, we chose to define $\tilde{s}_t^t$ as the corresponding convex combination of locality measures associated to cutting planes:

$$\tilde{s}_t^t = \sum_{j\in J_t}\alpha_j s_j^t + \tilde{\alpha}\tilde{s}_{t-1}^t.$$

Interestingly using this aggregated locality measure, one can show that there is no conflict between $\tilde{c}_t^t$ and $\mathbf{w}_t^*$ since $R(\mathbf{w}_t^*) - \tilde{c}_t^t(\mathbf{w}_t^*) \geq \tilde{s}_t^t$. Indeed, we have:

$$\begin{aligned} R(\mathbf{w}_t^*) - c_j^t(\mathbf{w}_t^*) &\geq s_j^t \; \forall j \in J_t, \\ R(\mathbf{w}_t^*) - \tilde{c}_{t-1}^t(\mathbf{w}_t^*) &\geq \tilde{s}_{t-1}^t. \end{aligned}$$

Multiplying these equations by α_j's and $\tilde{\alpha}$ then taking the sum gives the result:

$$R(\mathbf{w}_t^*) - \tilde{c}_t^t(\mathbf{w}_t^*) \geq \tilde{s}_t^t.$$

3.3 Variants

In this section we discuss two variants (and their implementations issues) that allow speeding up convergence in practice.

3.3.1 REGULARIZATION

In previous section we presented our method with a standard L2 regularization term $\frac{\lambda}{2}\|\mathbf{w}\|^2$. Yet this choice is not always a good one for non-convex optimization problems where convergence to a poor local optima is a severe problem. Alternatively one may prefer to regularize around a first reasonable solution $\mathbf{w}^{reg}$ and use a regularization term such as $\|(\mathbf{w} - \mathbf{w}^{reg})\|^2$. For instance to learn Hidden Markov Models with a large margin criterion using a variant of NRBM, we used a model learned with Maximum Likelihood as $\mathbf{w}^{reg}$ (Do and Artières, 2009). Furthermore, if all parameters in $\mathbf{w}$ do not have the same nature (magnitude) then using only one weight-cost (λ) for all parameters is not wise. So one may prefer the following regularization term:

$$\frac{\lambda}{2}\|(\mathbf{w} - \mathbf{w}^{reg}) \otimes \theta\|^2$$

where θ is a positive vector of regularization weights and $\otimes$ stands for element-wise product. The use of different θ values depending on the parameters allows introducing some prior information. Again, taking our example of learning Hidden Markov Models, we used different θ values for regularizing transition probabilities and emission probabilities parameters.

3.3.2 FAST VARIANT WITH LINE SEARCH

In Algorithm 4, the minimum point of the approximation function is not guaranteed to be a better solution than the current best observed solution, which may result in null steps. Few works showed that one can speed up cutting plane based methods with a linesearch procedure (Franc and Sonnenburg, 2008; Do and Artières, 2008), which may be efficient to compute in some cases (e.g. primal objective of linear SVM).

The idea is that a line search ensures that we get a better solution every iteration, assuming that the search direction is a descent direction. If the search direction is not a descent direction then the line search returns the best solution along the search direction (should be close to the current solution), which will be used to build a new cutting plane in the next iteration. In our case, without specific knowledge of $f(\mathbf{w})$ we use a general line search technique.

Since the line search may require considerable more function/subgradient evaluations, one can initialize the step size based on the step size reached in previous iteration. In our implementation (a line search with Wolfe conditions), initial step size is computed so that the step length is the same as

the final stepsize in previous iteration. This simple implementation works well and most of the time we need only one function/subgradient evaluation (when initial step size satisfies Wolfe conditions).

We investigated two strategies. In the *full line search strategy* , every iteration we add two cutting planes to the approximation problem, one at the minimum point of the current approximated problem and one at the solution of the line search. In this case, the role of the line search is to improve the quality of the approximated problem every iteration. In the *greedy line search strategy* we consider adding only one cutting plane at the solution of the line search in order to limit the number of function/subgradient evaluation at each iteration. This strategy also works well in practice as we will see in experiment section.

4. Convergence Analysis

In this section, we provide theoretical results for our algorithm. For a convex objective function, when disabling locality measure (putting these to 0), our algorithm can be viewed as a limited memory variant of CRBM, and we provide a proof on the convergence rate of the algorithm under a standard assumption. For non-convex objective function, the convergence analysis is much more complicated and requires a disputable assumption. For these reasons, we only present main results for the non-convex case in this paper, while the corresponding proofs can be found in an internal report (Do and Artieres, 2012).

4.1 Convergence Analysis for NRBM: Convex Case

We provide in this section theoretical results on the convergence behavior of our algorithm applied to convex risks. First we present a theorem in Section 4.1.2 which characterizes its convergence rate and shows that our algorithm inherits the fast convergence rate of CRBM from which it is inspired (note that we consider here the particular case of quadratic regularization with non-smooth objective function).

In the case of a convex risk one can either use the convex version of our algorithm which remains to using Algorithm 3 or the non-convex version (Algorithm 4) while disabling all locality measure (i.e. putting these to 0, Algorithm 4 will become Algorithm 3 since conflicts will not occur for convex risk). We prove in the following the main results for the convex version.

4.1.1 ASSUMPTIONS

The necessary assumption for proving our main results are the following:

- H1 : The empirical risk is Lipschitz continuous with a constant G.

H1 is a rather standard assumption, which was used for proving convergence results in previous works (Smola et al., 2008; Shalev-Shwartz et al., 2007; Joachims, 2006). It is in particular a reasonable assumption in case of smooth almost everywhere risks such as those one gets using hinge loss and maximum margin criterion (SVM, structured output prediction, etc).

4.1.2 MAIN RESULTS

We provide here an upper bound on the convergence rate of our variant of limited memory CRBM, by studying the decrease of the gap, defined as the difference between the minimum observed value

of the objective and the minimum of the current approximated problem, with iteration number. Indeed, this gap can be used for bounding from above the accuracy of the current solution (in terms of the objective value).

We begin with some preliminary results. Lemmas 1 and 2 are general results that are needed for Lemma 3 which establishes a lower bound on the improvement of the approximation gap at each iteration.

Lemma 1 *Teo et al., 2007 The minimum of* $\frac{1}{2}qx^2 - lx$ *with* $l, q > 0$ *and* $x \in [0,1]$ *is bounded from above by* $-\frac{l}{2}\min(1, l/q)$

Lemma 2 *Function* $h(x) = x - \frac{x}{2}\min(1, x/q)$ *is monotonically increasing for all* $q > 0$.

Proof We have :

$$h(x) = \begin{cases} x - x^2/2q & if x < q \\ x/2 & if x \geq q \end{cases}$$

where $x/2$ is always monotonically increasing, then h is for monotonically increasing for $x \geq q$. For $x \in (-\infty, q)$, $h'(x) = 1 - x/q > 0$ because $x < q$ and $q > 0$. Moreover, h is continuous (at $x = q$), thus h is monotonically increasing whatever x. ■

Lemma 3 *The approximation gap decreases according to:*

$$gap_{t-1} - gap_t \geq \min\left(\frac{gap_{t-1}}{2}, \frac{(gap_{t-1})^2\lambda}{8G^2}\right) \tag{18}$$

where the approximation gap is defined as $gap_t = f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t)$.

Proof We focus on deriving an underestimator on the minimum value of $g_t(\mathbf{w})$ based solely on this aggregated cutting plane and on the new added cutting plane at iteration t. This is simpler than exploiting the complete approximation function. Note that this is possible since the aggregated cutting plane accumulates information about the approximation problem at previous iterations. We have:

$$g_t(\mathbf{w}) \geq \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max\left[\langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w}\rangle + \tilde{b}_{t-1}, \langle \mathbf{a}_t, \mathbf{w}\rangle + b_t\right]. \tag{19}$$

Let find the minimum of the right side. The dual program of this minimization problem is:

$$\begin{array}{ll} \max_{\tilde{\alpha}_{t-1}, \alpha_t} & -\frac{\lambda}{2}\|\frac{\tilde{\alpha}_{t-1}\tilde{\mathbf{a}}_{t-1} + \alpha_t \mathbf{a}_t}{\lambda}\|^2 + \tilde{\alpha}_{t-1}\tilde{b}_{t-1} + \alpha_t b_t \\ s.t & 0 \leq \tilde{\alpha}_{t-1}, \alpha_t \leq 1 \\ & \tilde{\alpha}_{t-1} + \alpha_t = 1 \end{array}$$

where $\tilde{\alpha}_{t-1}, \alpha_t \in \mathbb{R}$ are Lagrange multipliers. This quadratic program has 2 variables and can be further simplified as:

$$\begin{array}{rll} & \max\limits_{\alpha_t \in [0,1]} & -\frac{1}{2\lambda}\|\tilde{\mathbf{a}}_{t-1} + \alpha_t(a_t - \tilde{\mathbf{a}}_{t-1})\|^2 + \alpha_t(b_t - \tilde{b}_{t-1}) + \tilde{b}_{t-1} \\ = & \max\limits_{\alpha_t \in [0,1]} & -\frac{1}{2\lambda}\|\mathbf{a}_t - \tilde{\mathbf{a}}_{t-1}\|^2(\alpha_t)^2 + (\frac{\|\tilde{\mathbf{a}}_{t-1}\|^2}{\lambda} - \frac{\langle \mathbf{a}_t, \tilde{\mathbf{a}}_{t-1}\rangle}{\lambda} + b_t - \tilde{b}_{t-1})\alpha_t - \frac{\|\tilde{\mathbf{a}}_{t-1}\|^2}{2\lambda} + \tilde{b}_{t-1} \\ = & -\min\limits_{\alpha_t \in [0,1]} & \frac{1}{2}q(\alpha_t)^2 - l\alpha_t - g_{t-1}(\mathbf{w}_t) \end{array} \tag{20}$$

where $q = \frac{\|\mathbf{a}_t - \tilde{\mathbf{a}}_{t-1}\|^2}{\lambda}$ and $l = \frac{\|\tilde{\mathbf{a}}_{t-1}\|^2}{\lambda} - \frac{\langle \mathbf{a}_t, \tilde{\mathbf{a}}_{t-1}\rangle}{\lambda} + b_t - \tilde{b}_{t-1}$.

Note that $\mathbf{w}_t = \tilde{\mathbf{w}}_{t-1} = -\frac{\tilde{\mathbf{a}}_{t-1}}{\lambda}$. Hence the linear factor may be rewritten as:

$$\begin{aligned} l &= \tfrac{\|\tilde{\mathbf{a}}_{t-1}\|^2}{\lambda} - \tfrac{\langle \mathbf{a}_t, \tilde{\mathbf{a}}_{t-1}\rangle}{\lambda} + b_t - \tilde{b}_{t-1} \\ &= \langle \mathbf{a}_t, \mathbf{w}_t\rangle + b_t - \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w}_t\rangle - \tilde{b}_{t-1} \\ &= \tfrac{\lambda}{2}\|\mathbf{w}_t\|^2 + \langle \mathbf{a}_t, \mathbf{w}_t\rangle + b_t - \tfrac{\lambda}{2}\|\mathbf{w}_t\|^2 - \langle \tilde{\mathbf{a}}_{t-1}, \mathbf{w}_t\rangle - \tilde{b}_{t-1} \\ &= f(\mathbf{w}_t) - g_{t-1}(\mathbf{w}_t). \end{aligned}$$

Using Lemma 1 the maximum value in Equation 20 is greater or equal than $\frac{l}{2}\min(1, l/q) + g_{t-1}(\mathbf{w}_t) = \frac{f(\mathbf{w}_t)-g_{t-1}(\mathbf{w}_t)}{2}\min\left(1, \frac{f(\mathbf{w}_t)-g_{t-1}(\mathbf{w}_t)}{q}\right) + g_{t-1}(\mathbf{w}_t)$. This latter quantity is then a lower bound of the minimum of the right side in Equation 19, thus:

$$\begin{aligned} & g_t(\mathbf{w}_{t+1}) && \geq \min\left(\tfrac{f(\mathbf{w}_t)-g_{t-1}(\mathbf{w}_t)}{2}, \tfrac{(f(\mathbf{w}_t)-g_{t-1}(\mathbf{w}_t))^2}{2q}\right) + g_{t-1}(\mathbf{w}_t) \\ \Rightarrow\ & g_t(\mathbf{w}_{t+1}) && \geq \min\left(\tfrac{f(\mathbf{w}_t^*)-g_{t-1}(\mathbf{w}_t)}{2}, \tfrac{(f(\mathbf{w}_t^*)-g_{t-1}(\mathbf{w}_t))^2}{2q}\right) + g_{t-1}(\mathbf{w}_t) \\ \Rightarrow\ & f(\mathbf{w}_t^*) - g_t(\mathbf{w}_{t+1}) && \leq f(\mathbf{w}_t^*) - g_{t-1}(\mathbf{w}_t) - \min\left(\tfrac{f(\mathbf{w}_t^*)-g_{t-1}(\mathbf{w}_t)}{2}, \tfrac{(f(\mathbf{w}_t^*)-g_{t-1}(\mathbf{w}_t))^2}{2q}\right). \end{aligned}$$

Note that $f(\mathbf{w}_t^*) \leq f(\mathbf{w}_{t-1}^*)$. Replacing $f(\mathbf{w}_t^*)$ by $f(\mathbf{w}_{t-1}^*)$ in the right side of previous equation and using Lemma 2 one gets:

$$\begin{aligned} & gap_t && \leq f(\mathbf{w}_{t-1}^*) - g_{t-1}(\mathbf{w}_t) - \min\left(\tfrac{f(\mathbf{w}_{t-1}^*)-g_{t-1}(\mathbf{w}_t)}{2}, \tfrac{(f(\mathbf{w}_{t-1}^*)-g_{t-1}(\mathbf{w}_t))^2}{2q}\right) \\ \Leftrightarrow\ & gap_t && \leq gap_{t-1} - \min\left(\tfrac{gap_{t-1}}{2}, \tfrac{gap_{t-1}^2}{2q}\right). \end{aligned}$$

Finally since $q = \frac{1}{\lambda}\|\mathbf{a}_t - \tilde{\mathbf{a}}_{t-1}\|^2 \leq 4G^2/\lambda$, and substituting this back in previous formula gives the result. ■

Theorem 1 *Algorithm 3 produces an approximation gap below* ε *in* $O(1/\lambda\varepsilon)$ *iterations. More precisely it reaches a approximation gap below* ε *after* T *steps with:*

$$\begin{aligned} & T \leq T_0 + 8G^2/\lambda\varepsilon - 2 \\ & with\ T_0 = 2log_2 \tfrac{\lambda\|\mathbf{w}_1 + \mathbf{a}_1/\lambda\|}{G} - 2. \end{aligned}$$

Proof Let consider the two quantities occurring in Equation 18, $gap_{t-1}/2$ and $\lambda gap_{t-1}^2/8G^2$.

We first show that the situation where $gap_{t-1}/2 > \lambda gap_{t-1}^2/8G^2$ (i.e. $gap_{t-1} > 4G^2/\lambda$) may only happen a finite number of iterations, T_0. Actually if $gap_{t-1} > 4G^2/\lambda$ Lemma 3 shows that $gap_t \leq gap_{t-1}/2$ and the gap is at least divided by two every iteration. Then $gap_{t-1} > 4G^2/\lambda$ may arise for at most $T_0 = log_2(\lambda gap_1/4G^2) + 1$. Since $gap_1 = \frac{\lambda}{2}\|\mathbf{w}_1 + \mathbf{a}_1/\lambda\|^2$ (it may be obtained analytically since the approximation function in the first iteration is quadratic), $T_0 = 2log_2 \frac{\lambda\|\mathbf{w}_1 + \mathbf{a}_1/\lambda\|}{G} - 2$.

Hence after at most T_0 iterations the decrease of the gap obeys $gap_t - gap_{t-1} \leq -gap_{t-1}^2/8G^2 \leq 0$. To estimate the number of iterations required to reach $gap_t \leq \varepsilon$ we introduce a function $u(t)$ which is an upper bound of gap_t (Teo et al., 2007). Solving differential equation $u'(t) = -\frac{\lambda}{8G^2}u^2(t)$ with

boundary condition $u(T_0) = 4G^2/\lambda$ gives $u(t) = \frac{8G^2}{\lambda(t+2-T_0)} \geq gap_t / \forall t \geq T_0$. Solving $u(t) \leq \varepsilon \iff t \geq 8G^2/\lambda\varepsilon + T_0 - 2$, the solution is reached with accuracy ε within $\left[T_0 + 8G^2/\lambda\varepsilon - 2\right]$ iterations. ■

Next we show that if the algorithm reaches a null gap then it has found the global minimum.

Theorem 2 *If $gap_t = 0$ at iteration t of Algorithm 3, then $\mathbf{w}_t^* = \tilde{\mathbf{w}}_t$ and $\mathbf{w}_t^*$ is the global minimum of f.*

Proof

We have $g_t(\mathbf{w}_t^*) = f(\mathbf{w}_t^*)$ since the approximation errors are zero at points where cutting plane were built. Hence, the null gap between $f(\mathbf{w}_t^*)$ and the minimum of g_t, $g_t(\tilde{\mathbf{w}}_t)$, indicates that $g_t(\mathbf{w}_t^*) = g_t(\tilde{\mathbf{w}}_t)$, i.e., $\mathbf{w}_t^* \equiv \tilde{\mathbf{w}}_t$. Since $g_t(\mathbf{w})$ is an underestimator of $f(\mathbf{w})$, it's minimum value,$g_t(\tilde{\mathbf{w}}_t)$,is less than or equal to the minimum of $f(\mathbf{w})$. Therefore, $\mathbf{w}_t^*$ is the minimum of $f(\mathbf{w})$. ■

Note that in the case the algorithm can not reach null gap after a finite iterations, both $f(\mathbf{w}_t^*)$ and $g_t(\tilde{\mathbf{w}}_t)$ converge to the minimum of f, $f(\mathbf{w}^*)$, since $f(\mathbf{w}_t^*) \geq f(\mathbf{w}^*)$, $g_t(\tilde{\mathbf{w}}_t) \leq f(\mathbf{w}^*)$ and $f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t) \to 0$.

4.1.3 EXTENSION OF RESULTS TO VARIANTS WITH LINE SEARCH

As our proof is based on the cutting plane built at the minimum of approximated problem $\langle \mathbf{a}_t, \mathbf{w} \rangle + b_t$, and the aggregated cutting plane, $\langle \tilde{\mathbf{a}}_t, \mathbf{w} \rangle + \tilde{b}_t$, all theoretical results hold for the *full line search variant* for which the two CPs are present in the approximation problem.

However, the things are more complicated for the *greedy line search strategy* and the proofs do not hold anymore in their actual shape. Yet, such a strategy is less expensive than the *full* one and it is efficient in practice. All results of the line search variant in the experiment section have been gained using this implementation.

4.2 Convergence Analysis for NRBM : Non-Convex Case

We provide in this section theoretical results on the convergence behavior of our algorithm. First we present a theorem in Section 4.2.2 which characterizes its convergence rate and shows that our algorithm inherits fast convergence rate of CRBM from which it is inspired (note that we consider here the particular case of quadratic regularization with non-smooth objective function). Next we provide theorem that characterizes the solution the algorithm converges to. First of all we detail and discuss the necessary assumptions used for proving these results, then we present our main results. See Do and Artieres (2012) for detailed proofs.

4.2.1 ASSUMPTIONS

The necessary assumptions for proving our main results are the following:

- H1 : The empirical risk is Lipschitz continuous with a constant G.
- H2 : The number of iterations where a conflict is solved by modifying the normal vector $\mathbf{a}_t$ (NullStep2 case in Algorithm 4) is finite.

Under the H1 assumption, we could get the same theoretical results on convergence rate as previous section for non-convex objective function. In other words, we can prove (with a more complicated proofs) that the approximation gap_t decrease towards zeros with a rate $O(1/\lambda\varepsilon)$ and that the algorithm has found a stationary solution if a null gap is reached. However, these results do not imply the convergence to a global minimum for the non-convex case, and we need to provide additional results for proving that the algorithm generates stationary solutions.

Our proof on the convergence towards stationary solution require an additional assumption H2, which states that the number of NullStep2 in Algorithm 4 is finite. Recall that there is a NullStep2 at iteration t if and only if the raw cutting plane built at current solution $\mathbf{w}_t$ is not compatible with the best observed solution $\mathbf{w}_t^*$. Hence, since the current solution and the best observed solution get closer as the iteration number increases we may hope that NullStep2 do not arise after a finite number of iterations. Furthermore, it is very likely that if the algorithm gets close enough to a stationary solution $\mathbf{w}^*$ lying within a smooth area then it should converge towards this stationary solution without conflicts anymore, as it would do in case of a convex and smooth objective. This is particularly expected for our algorithm (compared to standard non-convex bundle methods) since it focuses on maintaining a good approximation function around the best current solution. Another important point is that we did not observe any case of infinite number of conflicts in our experiments (on both academic optimization problems and machine learning problems) where NullStep2 mainly occurred in a few early iterations.

At the end these claims are still not proved so that the convergence of NRBM to a stationary solution is not fully proved here, but we believe that our convergence analysis establishes some important elements towards a fast and fully proved bundle method for minimizing non-convex regularized function.

4.2.2 MAIN RESULTS

Similar to the case of convex risk, we can prove that the approximation decreases as the algorithm iterates, under the hypothesis H1 only.

Lemma 4 *The approximation gap of Algorithm 4 decreases according to:*

$$gap_{t-1} - gap_t \geq \min\left(\frac{gap_{t-1}}{2}, \frac{(gap_{t-1})^2\lambda}{8G^2}\right)$$

where the approximation gap is defined as $gap_t = f(\mathbf{w}_t^*) - g_t(\tilde{\mathbf{w}}_t)$.

Proof The proof is provided in supplementary material. ■

This lemma implies a first theorem that provides a theoretical lower bound on convergence speed.

Theorem 3 *Algorithm 4 reaches a gap below* ε *with a number of iterations* $O(1/\lambda\varepsilon)$.

Next we show that if the algorithm reaches a null gap then it has found a stationary solution.

Theorem 4 *If* $gap_t = 0$ *at iteration* t *of Algorithm 4, then* $\mathbf{w}_t^* = \tilde{\mathbf{w}}_t$ *and* $\mathbf{w}_t^*$ *is a stationary point of objective function* f*, i.e.* $\mathbf{0} \in \partial f(\mathbf{w}_t^*)$.

Optimizer	Non-Convex	Non-Smooth	line-search
Non-convex Bunlde Method (NBM)	yes	yes	yes
LBFGS	no	no	yes
Stochastic Gradient Descend (SGD)	yes	yes	no
Subgradient Descend (SG)	yes	yes	no
Concave-Convex Procedure (CCCP)	dedicated solver for difference of convex functions		
SVMstruct	dedicated solver for a particular convex problem		
UniverSVM	dedicated solver for a particular non-convex problem		

Table 1: List of solvers that are considered in the experimental comparison.

The two following theorems say that, under Hypothesis H1 and H2, if the sequence $(\mathbf{w}_t)$ and $(\mathbf{w}_t^*)$ generated by NRBM Algorithm are infinite they have cluster points, and these cluster points are stationary solutions.[7]

Theorem 5 *If Algorithm 4 does not reach a stationary solution in a limited number of iterations, the two infinite sequences* $(\mathbf{w}_t)$ *and* $(\mathbf{w}_t^*)$ *generated by Algorithm 4 have cluster points.*

Theorem 6 *Let* $\mathbf{w}^*$ *be a cluster point of the sequence* $(\mathbf{w}_t^*)$*. Then under assumptions H1 and H2,* $\mathbf{w}^*$ *is a stationary solution of* $f(\mathbf{w})$*.*

4.2.3 EXTENSION OF RESULTS TO VARIANTS

Similarly to the convex case, our results on convergence rate and vonvergence to a stationary solution hold for the *full line search strategy* but not for the *greedy line search strategy*, the latter being less expensive and equally accurate in practice.

5. Experiments

In this section, we compare our optimization method NRBM to standard and non-standard (i.e., methods designed for solving a particular machine learning problem) optimizers listed in Table 1. We also implemented the sped-up version of NRBM with soft line search procedure, this latter is called NRBMLS. The implementation of our two algorithms NRBM and NRBMLS are in Matlab (available at `https://forge.lip6.fr/projects/nrbm`), and the implementation of NBM is in Fortran (available at http://napsu.karmitsa.fi/lmbm/).

First a series of experiments has been performed on artificial problems where we tested optimization algorithms for optimizing a manually defined non-convex and/or non smooth objective function. This allows deep understanding of the behavior of our approach. Then we consider machine learning problems of increasing optimization difficulty. We first consider a convex and smooth optimization problem (learning a CRF). Next we consider a convex and non smooth optimization problem (learning a M3N). Next we consider a non-convex and non smooth optimization problem, learning a transductive SVM. Finally we study two larger scale non-convex optimization problems for learning graphical models for speech and handwriting recognition: learning Hidden Markov Models with a large margin criterion (non smooth) (Do and Artières, 2009); and learning a model

7. Let $\{x_n\}$ be a sequence of real vectors, then x is a cluster point of $\{x_n\}$ if for every $\varepsilon > 0$, there are infinitely many points x_n such that $\|x - x_n\| < \varepsilon$.

mixing a deep neural network feature extractor and conditional random fields (smooth) (Do and Artières, 2010). For each of these optimization problems we compare our methods with state of the art dedicated optimization methods. Note that although many optimizer are implemented in Matlab, dynamic programming for structured problems (CRF, M3N, CDHMM, NeuroCRF) are written in C mex-files.

5.1 Artificial Test Problems

Experiments were carried out on two academic non-convex test problem problem (Haarala et al., December 2004). We followed here experimental settings with a few modifications. Actually, we add a regularization term to the initial solution of each problem. We did not use the origin as regularization point, which may lead to trivial optimum solution $\mathbf{0}$ and optimum value 0, since this may cause numeric problems when using relative tolerance on objective value. In the following, we note w_i the i^{th} coordinate of vector $\mathbf{w} \in \mathbb{R}^D$ in the search space. We note $\mathbf{w}^{(0)}$ the initial solution.

The two objective functions that we seek to optimize are named and defined as follows. Note that both problems may be instantiated with a varying number of dimensions (i.e. parameters), D, this allows investigating small to larger scale problems.
Chained Mifflin 2

$$f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}-\mathbf{w}^{(0)}\|^2 + \sum_{i=1}^{D-1}\left(-w_i + 2(w_i^2 + w_{i+1}^2 - 1) + 1.75|w_i^2 + w_{i+1}^2 - 1|\right)$$

with $w_i^{(0)} = -1$ for all $i = 1, ..., D$.
Chained Cessent 2

$$f(\mathbf{w}) = \tfrac{\lambda}{2}\|\mathbf{w}-\mathbf{w}^{(0)}\|^2 + \textstyle\sum_{i=1}^{D-1} \max \quad \begin{matrix} \left[w_i^2 + (w_{i+1}-1)^2 + w_{i+1} - 1\,,\right. \\ \left. -w_i^2 - (w_{i+1}-1)^2 + w_{i+1} + 1\right] \end{matrix}$$

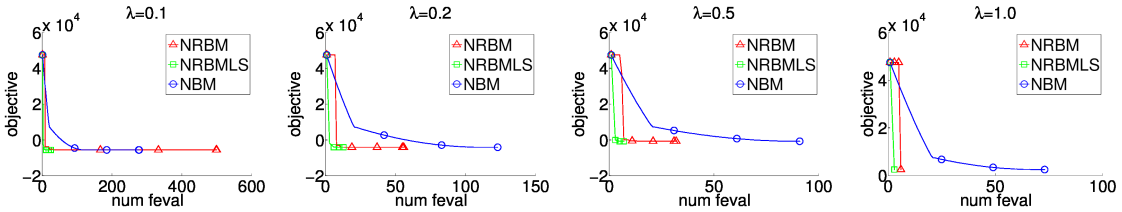
with

$$w_i^{(0)} = \begin{cases} -1.5 & when \quad mod(i,2) = 1 \\ 2.0 & when \quad mod(i,2) = 0 \end{cases}.$$

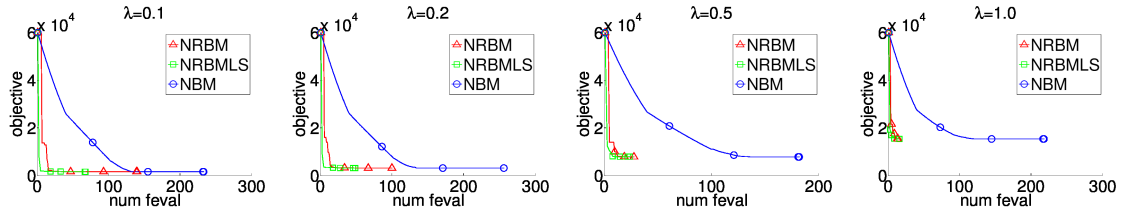
We compare our algorithms, NRBM and NRBMLS (NRBM with linesearch), and standard Non-Convex Bundle Method (NBM). In order to do so we conducted extensive experiments to investigate the respective convergence behaviour of the methods (convergence rate and quality of the solution found). Tables 2 and 3 report results gained for both data sets Chained Mifflin 2 and Chained 2 Cessent for NBM, NRBM and NRBMLS for various experimental settings: Data dimensionality D ranges from 10^2 up to 10^5, and regularization parameter λ ranges from 0.1 to 1.0. We compare the three algorithms with respect to:

- The value of the objective at the solution found.
- The number of objective evaluations needed.
- The cpu time of the optimization (indicative).

For NRBM and NRBMLS, optimization is performed until the approximation gap becomes less than 0.1% of the absolute value of objective function (relative tolerance 10^{-3}), and we also set the relative tolerance of NBM to 10^{-3}. Note however that the two stopping criteria do not coincide, which may lead to different final accuracy (we will come back on this point later in sections 5.5 and



(a) Chained Mifflin 2 D=10000



(b) Chained Cessent 2 D=10000

Figure 8: Comparison of convergence behavior for NRBM, NRBMLS, and NBM. Figures show the value of the objective function as a function of the number of objective function evaluations. There is one row of plots per data set, with a plot for every value of λ.

5.6). Note also that the CPU time of the optimization is not indicative and should be taken carefully since implementations are not equally optimized.

We may draw some comments from these tables (note that we observed similar results, not included here, on few other artificial data sets than the two studied here). First we focus on the linesearch efficiency by comparing NRBM and NRBMLS in terms of convergence rate (measured by the number of evaluations of the objective) and of accuracy of the solution reached. One can easily observe that in some cases NRBMLS performs similarly as NRBM but in most cases it significantly improves convergence rate and leads to a better solution, whatever the dimensionality D, and whatever the amount of regularization λ.

From the point of view of convergence accuracy NBM often outperforms both of our methods, NRBM and NRBMLS, and converges to a better solution with a very slightly lower objective value. This is reasonable since NBM uses an additional stopping condition which is similar to the one used in proximal bundle method. However, the results in Figure 8 suggest that NRBM and NRBMLS are faster than NBM to reach a reasonable solution.

From the convergence rate point of view, NBM is faster than NRBM for low dimensionality and low λ only, but NRBM is faster than NBM when λ increases for low dimensionality and whatever λ for high dimensional problems. NRBMLS is always faster than NRBM and NBM, whatever λ and whatever the problem dimensionality. Depending on the settings, NRBM and NRBMLS may be up to 50 times faster than NBM, and this is particularly true for high dimensional optimization problems. Finally, as λ gets bigger, both NRBM and NRBMLS converge faster, as expected by the theoretical proven convergence rate $O(\frac{1}{\lambda\varepsilon})$. On the contrary, NBM cannot always benefit from the large value of λ (see Table 3).

At the end, our algorithms are shown to converge towards solutions that compare well to the ones found by standard non-convex bundle methods but they do converge much faster and the benefit seems to increase with the dimensionality of the problem. This suggests that our methods are better candidates for large scale machine learning problems involving non-convex optimization.

$D = 10^2$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
	obj.	eval	time	obj.	eval	time	obj.	eval	time	obj.	eval	time
NBM	-55.68	254	0.03s	-41.32	29	0.00s	-8.167	12	0.00s	24.92	10	0.00s
NRBM	-54.19	501	1.52s	-41.31	280	0.44s	-8.163	20	0.02s	24.93	8	0.01s
NRBMLS	-55.66	45	0.03s	-41.32	18	0.01s	-8.165	7	0.01s	24.93	5	0.01s
$D = 10^3$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	-560.9	157	0.16s	-416.3	48	0.04s	-83.17	30	0.03s	249.9	23	0.02s
NRBM	-556.2	501	1.96s	-416.3	163	0.39s	-83.16	19	0.04s	250.0	7	0.02s
NRBMLS	-560.7	43	0.08s	-416.3	19	0.04s	-83.16	7	0.02s	250.0	4	0.01s
$D = 10^4$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	-5613	431	4.77s	-4166	123	2.20s	-833.2	91	1.44s	2500	73	0.99s
NRBM	-5609	501	13.77s	-4166	67	2.13s	-832.8	34	1.06s	2500	6	0.19s
NRBMLS	-5611	24	0.54s	-4166	13	0.36s	-833.2	7	0.18s	2500	3	0.09s
$D = 10^5$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	-56138	407	96.64s	-41666	363	94.75s	-8333	284	71.54s	25000	214	50.87s
NRBM	-56119	172	57.57s	-41661	33	12.84s	-8332	32	9.2s	25000	6	1.73s
NRBMLS	-56097	30	7.19s	-41664	17	6.14s	-8333	7	1.74s	25000	3	0.85s

Table 2: Chained Mifflin 2 data set. Comparative results of convergence quality (objective value, column *obj*) and convergence rate (number of evaluations of the objective function, column *eval* and cputime, column *time*) for the three optimization methods NBM, NRBM and NRBMLS, for various values of regularization level (λ) and for various data dimensionality (D).

5.2 CRF Training: Smooth and Convex Objective Function

To begin with, we perform experiments with a smooth and convex objective function for learning a conditional random field (CRF), and compare NRBMLS (convex setting) with NBM, Stochastic Gradient Descent (SGD), and LBFGS, a popular choice of optimizer for CRF.[8] Note that both NRBM and LBFGS use an approximation of the objective function (non-smooth piece-wise quadratic function in NRBM and a second order approximation in LBFGS). While the smooth approximation in LBFGS is suitable for smooth objective function, we would like to know how the non-smooth approximation works for smooth objective function. For LBFGS, the algorithm stop once the ratio between the norm of the gradient and the norm of the solution is smaller than a small tolerance. In our experiment, we find a tolerance parameter which yields comparative accuracies between LBFGS and NRBMLS for several values of λ, but we could not guarantee fair stopping criterion in all cases. The SGD setting is based on Leon Bottou implementation (Bottou, 2008).

We use a fixed tolerance on the norm of gradient for LBFGS which yields comparable accuracies to NRBMLS for several values of λ.

8. We used a Matlab implementation of LBFGS provided by Fei Sha.

$D = 10^2$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
	obj.	eval	time	obj.	eval	time	obj.	eval	time	obj.	eval	time
NBM	15.75	58	0.00s	31.23	105	0.01s	77.84	124	0.02s	152.2	145	0.02s
NRBM	15.64	334	81.71s	31.23	132	1.91s	77.93	66	1.14s	152.3	21	0.02s
NRBMLS	15.63	111	8.12s	31.21	103	4.66s	79.15	49	0.38s	152.2	29	0.02s
$D = 10^3$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	156.3	180	0.21s	312.5	194	0.35s	780.9	146	0.17s	1531	163	0.18s
NRBM	157.1	501	11.44s	312.8	104	4.34s	781.6	41	0.32s	1532	15	0.03s
NRBMLS	156.3	105	11.75s	313.0	82	1.27s	781.5	36	0.18s	1531	17	0.03s
$D = 10^4$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	1563	276	5.76s	3125	311	6.49s	7812	194	3.06s	15316	289	4.02s
NRBM	1564	218	25.31s	3128	72	2.74s	7818	29	0.78s	15319	14	0.32s
NRBMLS	1564	75	6.9s	3127	42	2.64s	7829	22	0.40s	15327	16	0.21s
$D = 10^5$	$\lambda = 0.1$			$\lambda = 0.2$			$\lambda = 0.5$			$\lambda = 1$		
NBM	15625	628	154.32s	31251	660	178.43s	78126	636	143.97s	1.562e+5	628	151s
NRBM	15635	106	49.97s	31272	48	20.38s	78146	21	8.38s	1.532e+5	14	3.86s
NRBMLS	15638	62	30.12s	31269	48	16.29s	78247	21	6.24s	1.532e+5	16	2.51s

Table 3: Chained Cessent 2. Comparative results of convergence quality (objective value, column *eval*) and convergence rate (number of evaluations of the objective function, *eval* and cputime, column *time*) for the three optimization methods NBM, NRBM and NRBMLS, for various values of regularization level (λ) and for various data dimensionality (D).

When working with convex function, we can simply disable the use of the locality measure and the conflict handling procedure so that the method, NRBMLS-convex, becomes much similar to CRBM. Note that in NRBMLS-convex, we use the soft linesearch in order to speed up the convergence rate.

We conducted experiments on an OCR data set used by Taskar et al. (2004) for evaluating Maximum Margin Markov Networks. This OCR data set consists of 6877 words which correspond to roughly 52,000 characters (Kassel, 1995). OCR data are sequences of isolated characters (each represented as a binary vector of dimension 128) belonging to 26 classes. The data set was divided in 10 folds for cross validation. We use here the 8 first folds for training and the two last fold for testing.

Table 4 shows results of the four optimization algorithms for training CRFs with various regularization values (λ). As can be seen, the three batch optimization methods (NRBMLS, NBM, and LBFGS) significantly outperform SGD for this task, which is in contradiction to Bottou's result where SGD clearly outperformed LBFGS for the CONLL2000 data (Bottou, 2008). We believe that the difference between these two contradictory results may come from the fact that the OCR problem is maybe more complex, meaning that the OCR problem would require more training data (the predictive error rates on the two data sets are 14% for OCR and 6% for CONLL2000). This suggests that the OCR data is less redundant than the CONLL2000 data, which would explain why SGD, whose efficiency depends on data redundancy, is less effective for OCR data.

Now looking at batch optimization methods, NRBMLS requires less iterations than NBM to reach similar objective value, it also slightly outperforms LBFGS. While these results might be biased by unequal stopping criteria, they suggest that the non-smooth approximation technique also

	NRBMLS				NBM				LBFGS			
λ	time	eval	obj.	err.	time	eval	obj.	err.	time	eval	obj.	err.
1e-04	306s	192	2.71	0.145	370s	244	2.71	0.145	340s	225	2.71	0.146
5e-05	405s	258	2.53	0.145	520s	353	2.53	0.144	458s	309	2.53	0.144
2e-05	408s	266	2.37	0.143	671s	471	2.37	0.142	583s	406	2.37	0.144
1e-05	419s	279	2.29	0.143	881s	636	2.29	0.142	583s	414	2.29	0.142
5e-06	556s	376	2.24	0.143	1199s	881	2.24	0.144	815s	593	2.24	0.143
2e-06	700s	483	2.21	0.145	1136s	834	2.21	0.145	956s	710	2.21	0.145
1e-06	753s	515	2.20	0.145	1026s	757	2.20	0.145	741s	554	2.20	0.146

	SGD											
	100 epochs			500 epochs			1000 epochs			2000 epochs		
λ	time	obj.	err.	time	obj.	err.	time	obj.	err.	time	obj.	err.
1e-04	161s	2.83	0.156	805s	2.75	0.146	1610s	2.72	0.145	3221s	2.71	0.146
5e-05	157s	2.71	0.156	784s	2.59	0.148	1567s	2.56	0.147	3134s	2.54	0.144
2e-05	151s	2.63	0.161	755s	2.48	0.153	1509s	2.44	0.151	3019s	2.41	0.147
1e-05	147s	2.60	0.165	733s	2.47	0.158	1466s	2.41	0.154	2933s	2.36	0.150
5e-06	143s	2.59	0.166	714s	2.48	0.161	1429s	2.42	0.158	2857s	2.36	0.154
2e-06	139s	2.58	0.168	696s	2.51	0.163	1392s	2.47	0.163	2784s	2.41	0.159
1e-06	137s	2.58	0.167	687s	2.52	0.167	1374s	2.50	0.164	2747s	2.45	0.163

Table 4: Comparative results of learning a CRF on OCR data set (Kassel, 1995). The maximum number of gradient stored is set to 200 for NRBMLS, NBM and LBFGS. Each row corresponds to a particular value of λ and provides for both methods the total cputime (time), the number of objective evaluation (eval), the value of the objective at convergence (obj), the classification error rate on the test set (err).

works well for smooth functions. Moreover, NRBM can exploit efficiently the regularization term in order to outperform NBM.

5.3 M3N Training: Non-Smooth and Convex Optimization Problem

In a second series of experiments we compare the efficiency of different optimization methods on learning a M3N, which yields a convex and non-smooth optimization problem. In addition to NBM and SGD, we also report the results of SVMstruct, the state of the art solver for M3N. We used the latest version SVMstruct 3.0 with 1-slack formulation, which is several orders of magnitude faster than previous methods.[9]

Here again we use NRBM-convex and a soft linesearch for speeding up convergence rate (noted NRBMLS). Note that SVMstruct 1-slack can also be viewed as a special case of CRBM, and the implementation of SVMstruct includes also some speed up techniques in the dual space.

Table 5 reports experimental results on the OCR data set (with the same setting as before). Similar to CRF experiments, we observe that SGD is not very effective. Looking at the results

9. The implementation of SVMstruct was downloaded from: `http://svmlight.joachims.org/svm_struct.html`.

	NRBMLS				NBM				SVMstruct			
λ	time	eval	obj.	err.	time	eval	obj.	err.	time	eval	obj.	err.
1e-03	53s	158	2.92	0.143	61s	196	2.92	0.143	104s	1359	2.93	0.152
5e-04	69s	208	2.73	0.142	83s	269	2.73	0.141	137s	1849	2.74	0.148
2e-04	105s	314	2.52	0.135	151s	490	2.52	0.140	204s	2633	2.54	0.140
1e-04	161s	476	2.41	0.139	285s	926	2.41	0.138	311s	3644	2.43	0.140
5e-05	225s	644	2.33	0.137	383s	1241	2.33	0.138	468s	4975	2.35	0.138
2e-05	484s	1224	2.26	0.137	605s	1963	2.26	0.137	1432s	9292	2.28	0.139
1e-05	763s	1636	2.23	0.136	749s	2435	2.23	0.139	4834s	18369	2.25	0.143

	SGD											
	100 epochs			500 epochs			1000 epochs			2000 epochs		
λ	time	obj.	err.	time	obj.	err.	time	obj.	err.	time	obj.	err.
1e-03	41s	3.05	0.150	206s	2.96	0.146	411s	2.93	0.145	823s	2.92	0.144
5e-04	41s	2.91	0.150	207s	2.79	0.145	414s	2.76	0.144	827s	2.74	0.142
2e-04	41s	2.78	0.151	207s	2.66	0.145	415s	2.60	0.142	829s	2.56	0.140
1e-04	41s	2.71	0.149	206s	2.58	0.146	413s	2.53	0.144	825s	2.49	0.140
5e-05	41s	2.65	0.148	206s	2.52	0.145	412s	2.48	0.144	824s	2.43	0.142
2e-05	41s	2.63	0.147	205s	2.51	0.146	410s	2.47	0.145	821s	2.42	0.144
1e-05	41s	2.65	0.152	205s	2.49	0.147	409s	2.46	0.144	819s	2.42	0.144

Table 5: Comparative results of NRBM, NBM, SVMstruct and SGD for learning a M3N on OCR data set (Kassel, 1995). Each row corresponds to a particular value of λ and provides for both methods the total cputime (time), the number of objective evaluation (eval), the value of the objective at convergence (obj), the classification error rate on the test set (err).

of NRBMLS and SGD-2000 epochs, we see that NRBMLS outputs better objective values and test error rates, while requiring significantly less number of iteration. Furthermore, NRBMLS is constantly faster than NBM of the same family, and significantly outperforms SVMstruct too.

5.4 Transductive SVM Training: Non-Smooth and Non-Convex Optimization Problem

Finally, we consider the problem of binary classification in a semi supervised setting with a training set of n labeled examples $\{(x_1,y_1),..,(x_n,y_n)\}$ and m unlabeled examples $\{x_{n+1},..,x_{n+m}\}$. The unconstrained primal formulation of TSVM is then:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}\max(0,1-y_i\langle x_i,\mathbf{w}\rangle) + \gamma\frac{1}{m}\sum_{i=n+1}^{n+m}\max(0,1-|\langle x_i,\mathbf{w}\rangle|)$$

where γ is a trade-off parameter between the labeled loss term and the unlabeled loss term. This objective function belongs to the regularized function family that can be solved by NRBM. Furthermore it is also an instance of difference of convex functions and can be solved by CCCP.

We trained TSVM on a subset of the MNIST data set consisting of samples of digit 3 and 8, the data was preprocessed with Principal Components Analysis (PCA) as is usually done. We split

	NRBMLS				NBM				CCCP Primal			
γ	time	eval	obj.	err.	time	eval	obj.	err.	time	eval	obj.	err.
1	0.2s	24	0.26	0.057	0.8s	113	0.26	0.058	1.4s	181	0.26	0.057
2	0.1s	21	0.35	0.055	0.4s	59	0.35	0.054	1.6s	206	0.36	0.054
5	0.2s	25	0.55	0.051	1.1s	197	0.55	0.051	2.7s	375	0.57	0.057
10	0.3s	39	0.81	0.049	0.3s	57	0.81	0.051	4.8s	680	0.86	0.056
20	0.7s	90	1.20	0.048	0.5s	104	1.20	0.051	7.5s	1083	1.72	0.122
50	0.3s	41	2.19	0.084	1.0s	193	2.00	0.054	1.1s	151	9.38	0.548

	UniverSVM			SG 100 epochs			SG 500 epochs			SG 5000 epochs		
γ	time	obj.	err.	time	obj.	err.	time	obj.	err.	time	obj.	err.
1	64s	n.a.	0.057	1s	0.28	0.058	3s	0.26	0.057	33s	0.26	0.058
2	65s	n.a.	0.057	1s	0.37	0.055	3s	0.35	0.054	32s	0.35	0.053
5	53s	n.a.	0.049	1s	1.18	0.329	3s	0.61	0.051	32s	0.55	0.048
10	59s	n.a.	0.075	1s	1.81	0.455	3s	1.62	0.401	30s	1.60	0.390
20	63s	n.a.	0.055	1s	2.49	0.489	3s	2.37	0.476	27s	2.35	0.475
50	69s	n.a.	0.143	1s	4.09	0.501	3s	3.60	0.505	26s	3.48	0.503

Table 6: Comparative results of transductive SVM training. Each row corresponds to a particular value of γ. For each optimization method, we report the total cputime (time), the number of objective evaluation (eval), the value of the objective (obj), the classification error rate on the test set (err). The error rate of the initial solution (a SVM trained on labeled samples only, with $\lambda = 0.01$) is 0.0776.

the training data into 200 labeled samples and 11782 unlabeled samples, the test set contains 1984 samples. We set $\lambda = 0.01$ since it gives best results on the test data.

We initialized the TSVM with the SVM solution on the labeled set. Then we optimized the non-convex objective with NRBMLS, CCCP in primal (Yuille and Rangarajan, 2003), CCCP in dual with UniverSVM, and Gradient Descent.[10] For CCCP-Primal, we used our NRBM-convex solver (disabling the use of locality measures) for every iteration, then we stopped CCCP once the improvement of the objective function was not better than 1% of its current value. Note that UniverSVM solve the TSVM problem in dual space and there is no equivalent concept of the number of primal objective evaluations. We also perform experiments with Gradient Descend method since the implementation of SGD for TSVM is not trivial.

The comparative results of the 5 optimization methods are shown in Table 6. We observe that all methods achieve similar best error rates ($0.048 - 0.049$) but primal optimization methods outperform the only one dual optimization method (UniverSVM) in term of speed. Looking at each row (for the same γ value), we see that NRBMLS always reach a good solution which make it the most robust optimizer in this experiment.

The behavior of Subgradient Method (SG) is quite complex. On one hand, one observe that it seems to converge to the same solution of NRBMLS for small values of γ. On other hand, for

10. The implementation of UniverSVM was downloaded from: `http://www.kyb.tuebingen.mpg.de/bs/people/fabee/universvm.html`.

large γ (i.e. the non-convex term is more important), the solution found by SG is not good in term of objective value and test error rate. Although SG and NRBMLS reach the same best error rate (0.048), SG requires a larger number of iterations to converge. This suggests the potential of using SG for the TSVM problem, but we need a very careful design (learning rate, update rule, etc.) to overcome the slow convergence rate.

Comparing NRBMLS and CCCP-Primal, we see that NRBMLS clearly outperforms CCCP, it is faster and converges to a better solution. This can be explained by the fact that CCCP spends too much effort to minimize the convex approximation function (whose evaluation requires the same computing cost as the original non-convex function) every CCCP iteration. While convex approximation of non-convex function is often accurate only on a neighborhood of the coordinate where its was constructed, CCCP may perform useless computations to reach the minimum of the convex approximation. Instead, NRBMLS optimizes more directly the non-convex objective by updating iteratively the approximation. At the end, it converges faster to a good solution. Furthermore, CCCP may fail to get a good solution when the value of γ is too big (i.e. when the non-convex term becomes more important) while NRBM seems to be more robust in this hard setting.

At last it must be noticed that using CCCP may require a lot of effort to mathematically reformulate the objective function into a concave and a convex term. The extension of TSVM for multiclass classification and for structured prediction is for instance not straightforward with CCCP while it is with NRBM.

5.5 Large Margin Training for Continuous Density Hidden Markov Models

Hidden Markov Models (HMMs) have been widely used for automatic speech recognition (Rabiner, 1990) and handwriting recognition (Hu et al., 2000). Continuous Density HMMs (CDHMMs) are particularly suited for dealing with sequences of real-valued feature vectors that one gets after typical front-end processing in signal processing tasks. CDHMMs usually exploit Gaussian mixture models to describe the variability of observations in a state. HMM parameters are learnt on a partially labeled data set (since state sequences of training sequences are unknown) with the Expectation-Maximization algorithm (EM) to maximize the joint likelihood of observation sequences and of hidden state sequences.

Recently, few approaches have been proposed for large margin learning of HMMs, especially in the speech recognition community (Sha and Saul, 2007; Jiang and Li, 2007) (see Yu and Deng, 2007 for a review). However none of these works actually handle the whole problem of max-margin learning for HMM parameters in the standard partially labeled setting. For instance Sha and Saul (2007) and Jiang and Li (2007) tackle a simplified convex optimization problem. Indeed, the main difficulty one encounters when formulating the maximum margin learning of CDHMM lies in the non-convexity of the optimization problem which comes from the presence of hidden state variables (the sequence of states and of Gaussian components) and from the discriminative function which is quadratic with respect to some parameter (e.g., covariance matrix). Instead of relying on a convex relaxation technique, we proposed to directly optimize the non-smooth non-convex objective function (Do and Artières, 2009):

$$\min_{\mathbf{w}} \quad \tfrac{\lambda}{2}\|\mathbf{w}-\mathbf{w}_0\|^2 + \textstyle\sum_i \max_{\mathbf{y}} \left(F(\mathbf{x}^i,\mathbf{y},\mathbf{w}) + \Delta(\mathbf{y}^i,\mathbf{y}) - F(\mathbf{x}^i,\mathbf{y}^i,\mathbf{w})\right) \tag{21}$$

where $(\mathbf{x}^i,\mathbf{y}^i)$ are input and output sequences, $\mathbf{w}$ are model parameters, and discriminative functions $F(\mathbf{x}^i,\mathbf{y},\mathbf{w}) = \max_{\mathbf{s}\in S(\mathbf{y}),\mathbf{m}} \log p(\mathbf{x}^i,\mathbf{y},\mathbf{s},\mathbf{m}|\mathbf{w})$ are Viterbi-approximation of log likelihood ($\mathbf{s}$ and $\mathbf{m}$

	NRBMLS			SGD^+			SGD		
λ	eval	obj.	err.	eval	obj.	err.	eval	obj.	err.
0.010	72	0.583	0.281	72	0.598	0.286	439	0.594	0.285
0.005	104	0.542	0.276	104	0.612	0.281	502	0.552	0.280
0.003	132	0.507	0.274	132	0.653	0.278	594	0.517	0.274
0.002	396	0.474	0.277	396	0.765	0.277	667	0.487	0.275
	NBM^+			NBM^*			NBM		
0.010	72	0.584	0.282	83	0.583	0.282	416	0.580	0.283
0.005	104	0.545	0.278	139	0.542	0.279	629	0.539	0.277
0.003	132	0.512	0.276	242	0.507	0.275	916	0.504	0.275
0.002	396	0.475	0.278	541	0.474	0.277	1181	0.472	0.278

Table 7: Large margin training for Continuous Density Hidden Markov Model. SGD and NBM results were reported with different various criteria. The $^+$ superscript corresponds to setting the maximum number of function evaluation to the one of NRBM. The * superscript corresponds to running optimization until the same objective function as NRBM solution was found.

are hidden state segmentation and mixture component assignments respectively). Note that the objective function is regularized with an initial solution $\mathbf{w}_0$ (MLE solution in practice) which yields better results than considering the standard regularization term $\frac{\lambda}{2}\|\mathbf{w}\|^2$.

Experiments were conducted on the TIMIT data set, which is a well-known benchmark data set for speech recognition. The training set consists of 3696 utterances, corresponding to 1,100,000 frames. We consider here the best CDHMM topology with 3 states per phoneme and 4 Gaussians per state, it corresponds to about 925,000 model parameters (Do and Artières, 2009).

We compare the optimization results of NRBMLS with SGD and NBM. Note that CCCP is not applicable since there is no concave-convex reformulation of the objective function in Equation 21 in the literature. Table 7 reports the number of function evaluation, the objective function value, and the phone recognition error rate for the three optimization methods (execution time is not reported since the experiments were launch on different machines). Here again, NRBM is stopped once the gap is below 1% of the objective value. For comparison, we reported SGD and NBM results with various stoping criteria: same number as NRBMLS (SGD^+ and NBM^+), reach NRBMLS's objective value (NBM^*) and the native stop criteria of the optimizers (SGD stops once the (online estimated) objective value is not improved after 5 iterations or the maximum number of function evaluations (2000) is reached, NBM stops once the norm of the search direction is smaller than 10^{-4}).

Looking at final results of the 3 optimizers with their native stoping criteria, we found that NRBMLS and NBM seems to converge to the same local minima in all cases while SGD converges to different solutions. For example, in the case of $\lambda = 0.003$ (the optimum value), NRBMLS and NBM reach objective values 0.507 and 0.504 respectively, i.e, the relative difference is 0.6% (which is less than the stoping condition of NRBMLS). Also, while NBM reaches a more accurate solution it does not outperform NRBM form the recognition rate point of view (0.275 vs. 0.274). Based on the final objective values of SGD results, one could say that SGD is not a good optimizer, yet it also allows achieving good recognition rate. This observation confirms again that stochastic learning

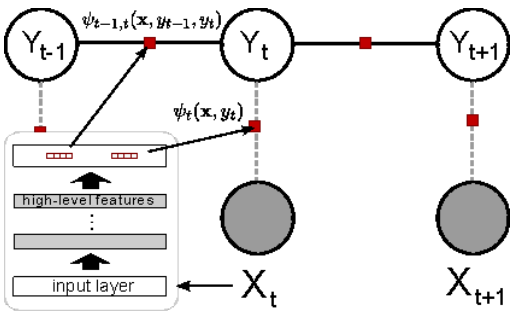


Figure 9: A chain-structured NeuroCRF.

may not always be good as an optimization tool but it may lead to solutions with good generalization properties.

At the end, NRBMLS converges significantly faster than the two other optimizers as it reaches a better solution (in term of objective value and error rate) when using a fixed number of iterations (columns SGD$^+$ and NBM$^+$), and it requires less iterations to reach a given value of the objective function (column NBM*, there is no column SGD* since SGD does not reach a similar objective value as NRBMLS).

5.6 Learning a Non Linear CRF

Finally, we consider the discriminative training of a NeuroCRF, this may be viewed as an extension of deep neural networks for structured output prediction (Do and Artières, 2010). Alternatively it may be thought as a hybrid model for labeling sequences that consists in a conditional random field exploiting features extracted with a deep neural network. Figure 9 illustrates such a linear chain NeuroCRF which combines a standard CRF with a deep neural network for feature extraction. Such a model implements a posterior distribution following

$$p(\mathbf{y}|\mathbf{x}) \propto \prod_{c \in C} e^{\langle \mathbf{w}_c, \Phi(\mathbf{x}, \mathbf{y}_c, \mathbf{w}_{NN}) \rangle} \tag{22}$$

where C denote the set of cliques of the CRF, $\mathbf{w}_c$ the weights associated to clique c, $\mathbf{w}_{NN}$ the weights of the deep NN, and Φ a function that extract features from $\mathbf{x}$.

Learning involves optimizing jointly the NN weights $\mathbf{w}_{NN}$ and the CRF parameters $\mathbf{w}_c$. As it has been suggested in previous works (Hinton et al., 2006), the deep neural network needs to be pretrained through unsupervised learning (we used Restricted Boltzmann Machine (RBM) in our experiments). This initial solution is then fine-tuned by optimizing a regularized version of the conditional likelihood in Equation 22. Our experiments compare optimization algorithms for the fine tuning step. Indeed, the pretraining of RBMs cannot be done by greedy optimizer such as NRBM or LBFGS since the computation of the gradient is intractable. Hence in our experiments, we pretrained the deep NNs by cascading RBMs trained with the original Contrastive Divergence algorithm (Hinton et al., 2006). The pretrained deep neural network is used to build the initial NeuroCRF model, called $\mathbf{w}_0$ where the initial CRF weights are drawn at random. Then, the NeuroCRF is fine tuned on a labeled data set of n input-output sequence $\{\mathbf{x}^i, \mathbf{y}^i\}_{i=1..n}$ using the following criteria:

$$\min_{\mathbf{w}} \quad \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}_0\|^2 + \frac{1}{m}\sum_{i=1..n} log(p(\mathbf{y}^i|\mathbf{x}^i; \mathbf{w})) \tag{23}$$

	NRBMLS			SGD$^+$			SGD		
λ	eval	obj.	err.	eval	obj.	err.	eval	obj.	err.
0.0010	64	1.847	0.050	64	1.604	0.059	2000	1.528	0.058
0.0005	81	1.203	0.047	81	1.190	0.056	2000	1.076	0.054
0.0003	116	0.842	0.046	116	0.941	0.053	2000	0.840	0.055
0.0002	116	0.616	0.045	116	0.775	0.054	2000	0.691	0.056
0.0001	130	0.365	0.050	130	0.557	0.055	2000	0.514	0.056
	NBM$^+$			NBM*			NBM		
0.0010	64	1.853	0.052	66	1.844	0.052	318	1.645	0.050
0.0005	81	1.233	0.046	94	1.202	0.048	305	1.075	0.048
0.0003	116	0.854	0.048	128	0.842	0.046	323	0.764	0.046
0.0002	116	0.660	0.047	175	0.616	0.046	238	0.594	0.045
0.0001	130	0.405	0.046	-	-	-	203	0.371	0.047
	LBFGS$^+$			LBFGS*			LBFGS		
0.0010	64	1.802	0.051	54	1.845	0.052	2095	1.591	0.048
0.0005	81	1.196	0.047	79	1.201	0.047	2104	1.010	0.047
0.0003	116	0.805	0.045	88	0.841	0.046	2119	0.696	0.049
0.0002	116	0.617	0.045	118	0.616	0.045	2131	0.511	0.052
0.0001	130	0.375	0.047	152	0.365	0.049	2135	0.300	0.053

Table 8: Optimization results for the discriminative training of NeuroCRFs. SGD, NBM and LBFGS results were reported with different various criteria. The $^+$ superscript corresponds to setting the maximum number of function evaluation to the one of NRBM. The * superscript corresponds to running optimization until the same objective function as NRBM solution was found.

where m is total the number of tokens in the data set and $p(\mathbf{y}^i|\mathbf{x}^i;\mathbf{w})$ is the conditional likelihood of the correct labeling, whose gradient can be computed by back propagation. Since we use sigmoid activation in hidden units of the deep neural network, the final objective function is non-convex with many local minima and plateaus. This is the reason why neural network optimization is difficult and the final solution is sensitive to initialization.

We evaluated the discriminative training of NeuroCRF on OCR data. Table 8 reports optimization results of NRBMLS and the other applicable state-of-the-art optimizers: SGD, NBM, and LBFGS. We use similar stopping condition setting as in previous sections. Looking at final error rates, we found that the final solutions of NRBMLS and NBM are better than the solutions obtained by other optimizers. The final objective values of these two algorithms are also quite close, suggesting that the two methods converge to similar local optimum. Note that NRBMLS is faster than NBM in all cases as NBM requires more iterations to reach the final objective value of NRBMLS (see column NBM*). Note also that the result of NBM* for $\lambda = 0.0001$ is not available, since NBM converged to a local minima with higher objective value than that of NRBMLS. Importantly, while NRBMLS can exploit the regularization term to converge quickly, a strong regularization term does not help NBM reducing the number of iterations. NRBMLS was also faster and reached better solution than SGD for small values of λ (including the best one $\lambda = 0.0002$). Unlike reported in

previous section on training CDHMMs, SGD solutions have relative high error rates, indicating that the problem of local minima is maybe more severe in the case of deep neural networks.

Finally, the results of LBFGS$^+$ and LBFGS* show that NRBM and LBFGS have comparable convergence speed until the stoping criteria of NRBM were reached (at least for the studied range of λ). Actually, LBFGS is slightly faster than NRBMLS for large values of λ but it is slower than NRBMLS for small values of λ. While NRBM stopped with acceptable solutions (in term of recognition rate) after about one hundred iterations, LBFGS continued looking for better solution and did not converge after 2000 iterations (corresponding to more than 2000 function evaluation). The long run of LBFGS also leads to a significantly better objective function than that of NRBMLS. This reflects the fact that the final solutions of NRBMLS are local minima or belong to plateaus in which gradients is very close to zero. Interestingly as already reported in previous sections, a better objective value does not always mean a better recognition rate: the long run of LBFGS does not improve the recognition performance but leads to over fitting problems. At the end, LBFGS can be viewed as a good/greedy optimizer for NeuroCRF (in term of objective value) but it lacks of early stopping condition. While having comparable convergence speed as LBFGS, NRBMLS has an intuitive built-in stoping criteria (based on the gap) that appears to be relatively robust for many applications.

5.7 Summary of Results and Discussion

We conducted extensive objective evaluation of NRBM and NRBMLS on few artificial problems and on five standard machine learning problems. Our preliminary results on artificial optimization problems show the potential of our approach and highlight the benefit of the line-search procedure to the proposed method, which improves both the convergence speed and the quality of the solution. We then evaluated our method on machine learning problems with real data sets, showing that it can be applied to a large variety of machine learning frameworks including convex and non-convex problems, smooth and non-smooth objectives, to learn linear and non-linear model (e.g. artificial neural network). While being generic to regularized machine learning problem, NRBMLS also showed great performance compared to dedicated solvers such as SVMStruct (for M3N) or UniverSVM (for transductive SVM).

Comparison with competitive solvers. Since NRBMLS may be viewed as a variant of NBM (both use a linesearch), their differences in performance is much interesting. We found that NRBMLS and NBM often converge to the same solution (i.e., having similar objective value) but NBM generally requires more objective function evaluations than NRBMLS to reach a solution with similar objective value (e.g. NRBMLS 132 vs. NBM* 242 in Table 7). We also compared NRBMLS to LBFGS, a popular solver for smooth functions, and we showed that NRBMLS is slightly faster than LBFGS. While NRBMLS and LBFGS have competitive performances the main advantage of NRBMLS is its generic feature, it can be applied to non-smooth function as well. Finally, the experimental comparison with SGD shows that the implemented version (which has been used for training CRFs) is not efficient for the five considered problems. While SGD may be very fast for large data sets, its main weakness remains the need of carefully tuning of parameters (e.g., step size) for every problem. Note that all these generic optimizers have gradient-based stopping criterion which work well if one needs an accurate solution (when subgradient is close to null vector). However, in the case where one does not need a too accurate solution (for example, to avoid over fitting), the subgradient information is not a strong measure of the quality of the solution. Alternatively NRBM uses

a gap-based stopping criterion which measures directly the quality of the solution in terms of the objective value. This criterion of NRBMs seems to be efficient for machine learning problems since NRBMLS systematically reaches the best error rates in all five machine learning experiments (using only one threshold of 0.1% of the objective value).

6. Conclusion

We proposed a new bundle optimization method called Non-convex Regularized Bundle Method (NRBM) able to deal with the minimization of regularized non-convex functions. We built on ideas from Convex Regularized Bundle Methods and on ideas from Non-Convex Bundle Methods, exploiting the regularization term of the objective and using a particular design of the aggregated cutting plane to build limited memory variants. We also discussed variants of the method and showed that integrating a line search may increase convergence rate in practice.

Experimental results on artificial problems show that our method is significantly faster than standard Non-convex Bundle Method, which is a state of the art method for non-smooth and non-convex optimization. We also presented experimental results on various convex and non-convex difficult machine learning problems, which demonstrate the potential and the wide application range of our algorithms. On one hand, our variant of bundle method got positive results on five different machine learning problems compared to state-of-the-art optimizers. On the other hand, our method is rather easy to use as the stopping condition is intuitive and efficient for various machine learning applications.

At the end, though the limited memory variant can be proved to inherit the fast convergence rate of CRBM in case of convex risks, we did not provide an analog satisfactory proof for the non-convex extension which is then more an algorithmic proposition that we validate experimentally on various machine learning optimization problems. Theoretical convergence analysis could be one direction for future work. We are also interested in considering more sophisticated approximation techniques such as second order approximation (similar to LBFGS) or non-convex approximation. While the approximation function could be more complex, one could expect that a more accurate approximation technique could improve the convergence speed.

References

A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil. A dc-programming algorithm for kernel selection. In *Proceedings of the Twenty-third International Conference on Machine Learning*, pages 41–48, 2006.

Y. Bengio and Y. Lecun. *Scaling Learning Algorithms Towards AI*. MIT Press, 2007.

D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, Belmont, MA, 2003.

L. Bottou. Stochastic gradient descent examples on toy problems, 2008. URL `http://leon.bottou.org/projects/sgd`.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006. URL `http://www.kyb.tuebingen.mpg.de/ssl-book`.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *Proceedings of the Twenty-third International Conference on Machine Learning*, pages 201–208. ACM Press, 2006.

T. M. T. Do and T. Artières. A fast method for training linear svm in the primal. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, pages 272–287, Berlin, Heidelberg, 2008. Springer-Verlag.

T. M. T. Do and T. Artières. Large margin training for hidden Markov models with partially observed states. In Léon Bottou and Michael Littman, editors, *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pages 265–272, Montreal, June 2009. Omnipress.

T. M. T. Do and T. Artières. Neural conditional random fields. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

T. M. T. Do and T. Artieres. Regularized bundle methods for convex and non convex risks: additional material to the jmlr paper. Technical report, LIP6, Pierre and Marie Curie University, Paris, 2012. URL `http://www.lip6.fr/lip6/reports/2012/lip6-2012-001.pdf`.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for support vector machines. In *Proceedings of the Twenty-fifth International Conference on Machine learning*, pages 320–327, New York, NY, USA, 2008. ACM.

M. Gaudioso and M.F. Monaco. Variants to the cutting plane approach for convex nondifferentiable optimization. *Optimization 25*, pages 65–75, 1992.

M. Haarala, K. Miettinen, and M. M. Makela. New limited memory bundle method for large-scale nonsmooth optimization. *Optimization Methods and Software*, 19:673–692(20), December 2004.

G. E. Hinton, S. Osindero, and Y-W Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

R. Horst and N. V. Thoai. Dc programming: overview. *Optimization Theory and Applications*, 103 (1):1–43, 1999. ISSN 0022-3239.

J. Hu, S. Gek Lim, and M.K. Brown. Writer independent on-line handwriting recognition using an hmm approach. *Pattern Recognition*, 33(1):133–147, 2000.

K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *Proceedings of the Twelfth International Conference on Computer Vision*. IEEE, 2009.

H. Jiang and X. Li. Incorporating training errors for large margin HMMs under semi-definite programming framework. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages IV–629. IEEE, 2007.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

T. Joachims. Training linear SVMs in linear time. In *Proceedings of the Twelfth International Conference On Knowledge Discovery and Data Mining*, pages 217 – 226, 2006.

T. Joachims, T. Finley, and Chun-Nam Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.

R. H. Kassel. *A Comparison of Approaches to On-line Handwritten Character Recognition*. PhD thesis, Cambridge, MA, USA, 1995.

K. Kiwiel. An aggregate subgradient method for nonsmooth convex minimization. *Mathematical Programming*, 27:320–341, 1983.

K. C. Kiwiel. *Methods of Descent for Nondifferentiable Optimization*. Springer-Verlag, 1985.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

L. Luksan and J. Vlcek. Introduction to nonsmooth analysis. theory and algorithms. Technical report, Universita degli Studi di Bergamo, 2000.

M. Makela. Survey of bundle methods for nonsmooth optimization. *Optimization Methods and Software*, 17:1–29, 2002.

M. Makela and P. Neittaanmaki. *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*. World Scientific Publishing Co., 1992.

L. R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2: 121–152, 1992.

F. Sha and L.K. Saul. Large margin hidden markov models for automatic speech recognition. *Advances in Neural Information Processing Systems*, 19:1249, 2007.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the Twenty-fourth International Conference on Machine Learning*, pages 807–814, New York, USA, 2007. ACM Press.

A. Smola, S. V. N. Vishwanathan, and Q. V. Le. Bundle methods for machine learning. *Advances in Neural Information Processing Systems*, 20:1377–1384, 2008.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. *Advances in Neural Information Processing Systems*, 16, 2004.

C. H. Teo, Q. V. Le, A. Smola, and S. V.N. Vishwanathan. A scalable modular convex solver for regularized risk minimization. In *Proceedings of the Thirteenth International Conference On Knowledge Discovery and Data Mining*, pages 727–736, New York, USA, 2007. ACM.

M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola. *Advances in Neural Information Processing Systems*, 20.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *Advances in Neural Information Processing Systems*, 17:1537–1544, 2004.

L. Xu, D. Wilkinson, F. Southey, and D. Schuurmans. Discriminative unsupervised learning of structured predictors. In *Proceedings of the Twenty-third international conference on Machine learning*, pages 1057–1064. ACM, 2006.

Z. Xu, R. Jin, J. Zhu, I. King, and M. Lyu. Efficient convex relaxation for transductive support vector machine. *Advances in Neural Information Processing Systems*, 20:1641–1648, 2008.

D. Yu and L. Deng. Large-margin discriminative training of hidden markov models for speech recognition. In *Proceedings of the International Conference on Semantic Computing*, pages 429–438, Washington, DC, USA, 2007. IEEE Computer Society.

A. L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proceeding of the Twenty-fifth International Conference on Machine Learning*, pages 1248–1255, Helsinki, Finland, 2008.

Journal of Machine Learning Research 13 (2012) 3585-3618 Submitted 10/11; Revised 10/12; Published 12/12

Learning Symbolic Representations of Hybrid Dynamical Systems

Daniel L. Ly DLL73@CORNELL.EDU
Hod Lipson* HOD.LIPSON@CORNELL.EDU
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, NY 14853, USA

Editor: Yoshua Bengio

Abstract

A hybrid dynamical system is a mathematical model suitable for describing an extensive spectrum of multi-modal, time-series behaviors, ranging from bouncing balls to air traffic controllers. This paper describes multi-modal symbolic regression (MMSR): a learning algorithm to construct non-linear symbolic representations of discrete dynamical systems with continuous mappings from unlabeled, time-series data. MMSR consists of two subalgorithms—clustered symbolic regression, a method to simultaneously identify distinct behaviors while formulating their mathematical expressions, and transition modeling, an algorithm to infer symbolic inequalities that describe binary classification boundaries. These subalgorithms are combined to infer hybrid dynamical systems as a collection of apt, mathematical expressions. MMSR is evaluated on a collection of four synthetic data sets and outperforms other multi-modal machine learning approaches in both accuracy and interpretability, even in the presence of noise. Furthermore, the versatility of MMSR is demonstrated by identifying and inferring classical expressions of transistor modes from recorded measurements.

Keywords: hybrid dynamical systems, evolutionary computation, symbolic piecewise functions, symbolic binary classification

1. Introduction

The problem of creating meaningful models of dynamical systems is a fundamental challenge in all branches of science and engineering. This rudimentary process of formalizing empirical data into parsimonious theorems and principles is essential to knowledge discovery as it provides two integral features: first, the abstraction of knowledge into insightful concepts, and second, the numerical prediction of behavior. While many parametric machine learning techniques, such as neural networks and support vector machines, are numerically accurate, they shed little light on the internal structure of a system or its governing principles. In contrast, symbolic and analytical models, such as those derived from first principles, provide such insight in addition to producing accurate predictions. Therefore, the automated the search for symbolic models is an important challenge for machine learning research.

Traditionally, dynamical systems are modeled exclusively as either a continuous evolution, such as differential equations, or as a series of discrete events, such as finite state machines. However, systems of interest are becoming increasingly complex and exhibit a non-trivial interaction of both continuous and discrete elements, which cannot be modeled exclusively in either domain (Lunze,

∗. Also in the Faculty of Computing and Information Science.

2002). As a result, hybrid automata, mathematical models which incorporate both continuous and discrete components, have become a popular method of describing a comprehensive range of real-world systems as this modeling technique is perfectly suited for systems that transition between distinct qualitative behaviors. Hybrid dynamical models have been successfully applied to succinctly describe systems in a variety of fields, ranging from the growth of cancerous tumors (Anderson, 2005) to air traffic control systems (Tomlin et al., 1998).

Although it is plausible to construct hybrid models from inspection and first principles, this process is laborious and requires significant intelligence and insight since each subcomponent is itself a traditional modeling problem. Furthermore, the relationships between every permutation of the subcomponents must be captured, further adding to the challenge. Thus, the ability to automate the modeling of hybrid dynamical systems from time-series data will have a profound affect on the growth and automation of science and engineering.

Despite the variety of approaches for inferring models of time-series data, none are particularly well-suited for building apt descriptions of hybrid dynamical systems. Traditional approaches assume an underlying form and regress model parameters; some approaches conform the data using prior knowledge (Ferrari-Trecate et al., 2003; Vidal et al., 2003; Paoletti et al., 2007), while others are composed of generalized, parametric, numerical models (Chen et al., 1992; Bengio and Frasconi, 1994; Kosmatopouous et al., 1995; Le et al., 2011). Although numeric approaches may be capable of predicting behavior with sufficient accuracy, models of arbitrary systems often require vast numbers of parameters, which obfuscates the interpretability of the inferred model (Breiman, 2001). This trade-off between accuracy and complexity for parametric models is in direct opposition to a fundamental aspect of scientific modeling—abstracting relationships that promote the formulation of new theorems and principles. Thus, constructing symbolic models of hybrid dynamical systems which can be easily and naturally interpreted by scientists and engineers is a key challenge.

The primary contribution of this paper is a novel algorithm, called multi-modal symbolic regression (MMSR), to learn symbolic models of discrete dynamical systems with continuous mappings, as an initial step towards learning hybrid automata. It is a data-driven algorithm that formulates symbolic expressions to describe both the continuous behavior and discrete dynamics of an arbitrary system. Two general learning processes are presented: the first algorithm, clustered symbolic regression (CSR), generates symbolic models of piecewise functions and the second algorithm, transition Modeling (TM), searches for symbolic inequalities to model transition conditions. These processes are then applied to a hybrid dynamical systems framework and are used to reliably model a variety of classical problems. MMSR is also applied to infer the modes of operation of a field-effect transistor, similar to those derived from first principles, from measured observations.

The remainder of this paper is organized as follows: Section 2 provides a brief introduction to hybrid dynamical systems, as well as a description of the relevant work in related fields. Section 3 introduces the theoretical background and implementation details of MMSR, with CSR and SR described in Section 3.2 and 3.3, respectively. Section 4 compares MMSR to traditional machine learning algorithms on four synthetic data sets and presents the inferred transistor model. The paper is concluded in Section 5.

2. Background

This section begins with a brief introduction to the mathematical background of hybrid automata, an inclusive model that describes a variety of hybrid systems. A subset of this general model is

described and formulated as the inference target. This is followed by a discussion of the related work in learning hybrid dynamical systems.

2.1 Hybrid Automata

Due to the its inherent complexity, hybrid dynamical systems have only recently emerged as an area of formal research. Consequently, there is a lack of a common framework, terminology and definition that is universally adopted (Henzinger, 1996; van der Schaft and Schumacher, 2000; Branicky, 2005). Our work uses a popular model called the hybrid automata, which extends the finite automata model to include continuous dynamics. The evolution of the system is deterministic. Each automata, $\mathcal{H}$, is defined as a 5-tuple, $\mathcal{H} = (\mathcal{W}, \mathcal{X}, \mathcal{M}, \mathcal{F}, \mathcal{T})$, with the following definitions:

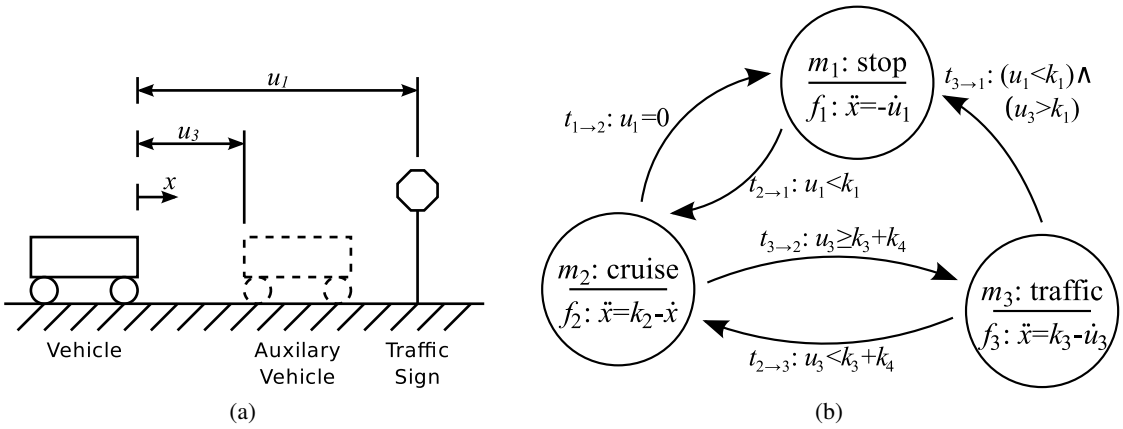


Figure 1: An example of a hybrid automata model for a simple, 1D driverless car. A schematic of the system is shown in a) and the system diagram represented as a directed graph is shown in b). $\mathcal{W}$ consists of two inputs, u_1 and u_3, which corresponds to the distance to the nearest sign and vehicle, respectively; while $\mathcal{X}$ consists of the state variables x and $\dot{x}$, which describe the vehicle's position and velocity. $\mathcal{M}$ consists of three modes, $\{m_1, m_2, m_3\}$, which represent distinct behaviors corresponding to whether the vehicle is approaching a traffic sign, cruising or driving in traffic; and the behaviors for each mode is described by $\{f_1, f_2, f_3\}$. There are five transitions events, each represented by a Boolean condition.

- $\mathcal{W}$ defines a communication space for which *external variables*, w, can take their values. The external variables can be further subdivided into *input variables*, $u \in \mathbb{R}^a$, and *output variables*, $y \in \mathbb{R}^b$, where $\mathcal{W} = \{u, y\}$.
- $\mathcal{X}$ defines a continuous space for which continuous *state variables*, $x \in \mathbb{R}^c$, can take their values.
- $\mathcal{M}$ defines a countable, discrete set of *modes* in which only a single mode, $m \in \{1, 2, \ldots, K\}$, is occupied at a given time. Each mode is represented as a vertex in the system diagram and may have an associated label for ease of reference.

- $\mathcal{F}$ defines a countable, discrete set of first-order, coupled, differential-algebraic equations (DAE). Each equation defines relationship between the state variables, their first-order time derivatives and the inputs:

$$f_k(x, \dot{x}, w) = 0$$

and the solution to these DAE are called the *activities* or *behaviors* of the mode. Each mode, m_k, is defined by its corresponding behavior, f_k, and the solution to the DAE defines the continuous evolution of the system when it is in that mode.

- $\mathcal{T}$ defines a countable, discrete set of *transitions* or *events*, where $t_{k \to k'}$ denotes a Boolean expression that represents the condition to transfer from mode k to mode k'. If none of the transition conditions are satisfied, then the hybrid automata remains in the current mode and the state variables evolves according to the specified behavior. These are represented as directed edges in the system diagram. For K modes, there are must be at least $K-1$ transitions and at most K^2 transitions. The transitions defines the discrete evolution of the system by describing how the mode is updated over time.

The challenge in modeling hybrid automata arises from the property that the latent "state" of a hybrid automata depends on both the discrete mode, m_k, as well as the continuous state space vector, x. As with all dynamical systems, the evolution of the system depends on the initial condition of the latent modes as well as the input variables. An example of a hybrid automata model for a simple, 1D driverless car is illustrated in Figure 1.

2.2 Discrete Dynamical System with Continuous Mappings

Hybrid automata are complex models which are capable of describing multi-modal behavior and latent continuous and discrete variables. To restrict the scope of the general problem, a number of assumptions are applied:

1. Each behavior is unique—No two behaviors are the same for any combination of modes: $f_i(\cdot) \neq f_j(\cdot), \forall m_i \neq m_j$.

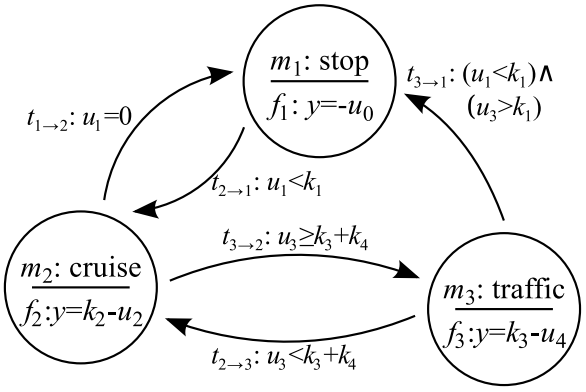
2. There are no continuous state space variables—All continuous states are directly observable and, thus, the behaviors are defined as strictly input-output relationships, $y = f(u)$, as opposed to DAEs, $y = f(x, \dot{x}, u)$.

3. The number of discrete modes is known—The cardinality of the modes, $|\mathcal{M}| = K$, is provided.

These assumptions describe a continuous-discrete hybrid system that evolves with discrete dynamics but contains continuous input-output relationships. This formulation makes the symbolic inference tractable while also providing a first step to the general solution of inferring hybrid automata. With the exception of assumption 3, each assumption defines a subset of models. If assumption 1 is relaxed, then the model becomes a continuous input-output hidden Markov model (Bengio and Frasconi, 1994). If assumption 2 is further relaxed, then the model becomes the standard hybrid automata described in Section 2.1.

The resulting discrete dynamical system with continuous mappings are defined as a 4-tuple, $\mathcal{H} = (\mathcal{W}, \mathcal{M}, \mathcal{F}, \mathcal{T})$, which are time-series models in which the output is dependent on both the

observed input as well as the latent mode variable. Furthermore, the evolution of the latent mode variable is dependent on the input via the transition conditions. To continue with the driverless car example, a hybrid automata is transformed into the desired model by converting the mode and differentiated inputs as explicit inputs and outputs (Figure 2).



Hybrid Automata	IOHMM
$\dot{u}_1$	u_0
$\dot{u}_3$	u_4
$\dot{x}$	u_2
$\ddot{x}$	y

Figure 2: A conversion of the 1D driverless car hybrid automata as a discrete dynamical model with continuous mappings. The system diagram and variable conversion are shown.

2.3 Related Work

Although there is little work in the automated, data-driven construction of models of hybrid dynamical systems whose components are expressed in symbolic mathematics, there are many machine learning approaches that are capable of describing and predicting the behavior of multi-modal, time-series data via alternate approaches.

Interest in hybrid dynamical systems is primarily spurred by the control systems community and consequently, they have proposed a variety of approaches to infer dynamical systems. One approach addresses the problem of modeling discrete-time hybrid dynamical systems by reducing the problem to switched, piecewise affine models (Ferrari-Trecate et al., 2003; Vidal et al., 2003), and procedures using algebraic, clustering-based, Bayesian, and bounded-error methods have been proposed (Paoletti et al., 2007). This modeling technique imposes a linear form to the system's dynamics, which substantially simplifies the modeling but limits the explanatory range of such models by enforcing linear approximations to non-linear systems.

Another approach uses non-linear, generalized, parametric models to represent hybrid dynamical systems. Chen et al. (1992) modeled hybrid systems using radial basis function networks, while Le et al. (2011) used support vector machines to approximate non-linear systems. The primary limitation with this approach is its reliance on parametric modelling in the form of neural networks or support vector machines. While parametric models can produce arbitrarily accurate predictions, they often require a vast quantity of weight parameters to achieve such accuracies in non-trivial systems. This tradeoff between accuracy and complexity often results in uninterpretable models which makes it difficult to extract meaningful relationships from the data (Breiman, 2001).

Recurrent neural networks (RNNs) have been a popular approach for modeling time-series data. The input-output relationship of a general, continuous dynamical system has been modeled with RNNs (Kosmatopouous et al., 1995) and it has also been shown that recurrent neural networks are capable of modeling finite state machines (Horne and Hush, 1996). However, there has been no reported work on specifically modeling discrete dynamical systems with continuous inputs and outputs. RNNs are also restricted by their parametric nature, often resulting in dense and uninterpretable models.

Consequently, there has been significant interest in extracting rules from parametric, recurrent neural networks to build a formal, symbolic model and providing an important layer of knowledge abstraction (Omlin and Giles, 1993, 1996a,b; Vahed and Omlin, 2004). However, a recent review of this work suggests that there is limited progress in handling RNNs with non-trivial complexity (Jacobsson, 2006).

The learning of input-output hidden Markov machines has been previously studied by Bengio and Frasconi (1994), which uses architecture based on neural networks to predict both the output of each mode as well as the transition conditions. The generalized expectation-maximization algorithm is used to optimize the parameters of in each of neural networks. Although the original work was implemented on grammatical inference with discrete inputs and outputs, the framework has since been adapted to several applications in the continuous domain (Marcel et al., 2000; Gonzalez et al., 2005). However, these approaches also rely on complex parametric neural network models.

Our technique attempts to resolve these various challenges by building models of hybrid dynamical systems that uses non-linear symbolic expressions for both the behaviors as well as the transitions. Rather than imposing a linear structure or using parametric models, symbolic expressions are inferred to provide a description that is in the natural, mathematical representation used by scientists and engineers.

3. Multi-Modal Symbolic Regression Learning Algorithm

This section begins with a formalization of the learning problem. This is followed by the description of two, general algorithms: clustered symbolic regression and transition modelling. The section is concluded by combining both subalgorithms within a hybrid dynamics framework to form the multi-modal symbolic regression algorithm.

3.1 Problem Formalization

The goal of the algorithm is to infer a symbolic discrete dynamics model with continuous mappings from time-series data, where symbolic mathematical expressions are learned for both the behaviors as well as the transition events. Consider a dynamical system that is described by:

$$
\begin{aligned}
m_n &= T(m_{n-1}, u_n), \\
y_n &= F(m_n, u_n) \quad = \begin{cases} F_1(u_n) & , \quad \text{if } m_n = 1 \\ \vdots & , \quad \vdots \\ F_K(u_n) & , \quad \text{if } m_n = K \end{cases}
\end{aligned}
$$

where $u_n \in \mathbb{R}^p$ is the input vector at time n, $y_n \in \mathbb{R}^r$ is the output vector, and $m_n \in M = \{1, 2, \ldots, K\}$ is the mode state.

The goal is to infer a multi-modal, input-output model that minimizes the normalized, negative log probability of generating the desired output vector under a mixture of Laplacians model, E, over the time series:

$$E = \frac{1}{N}\sum_{n=1}^{N} -\ln\left(\sum_{k=1}^{K} \gamma_{k,n} e^{-\frac{||y_n - \hat{y}_{k,n}||}{\sigma_y}}\right) \tag{1}$$

where $\gamma_{k,n} = p(m_n = k)$ or the probability that the system is in mode k, $\hat{y}_{k,n}$ is the output of function $F_k(u_n)$, and σ_y is the standard deviation of the output data.

This error metric is adapted from related work by Jacobs et al. (1991) on mixtures of local experts, but with the assumption of Laplacian, as opposed to Gaussian, distributions. The Laplacian distribution was chosen due to its relationship to absolute error, rather than squared error for Gaussian distributions. Note that for true mode probabilities or uni-modal models, this error metric indeed reduces to normalized, mean absolute error. Mean absolute error was preferred over squared error as is it more robust to outlier errors that occur due to misclassification.

To learn symbolic models of discrete dynamical systems with continuous mappings, the multi-modal symbolic regression (MMSR) algorithm is composed of two general algorithms: clustered symbolic regression (CSR) and transition modelling (TM). CSR is used to cluster the data into symbolic subfunctions, providing a method to determine the modal data membership while also inferring meaningful expressions for each subfunction. After CSR determines the modal membership, TM is then applied to find symbolic expressions for the transition conditions.

This algorithm varies from traditional learning approaches for hidden Markov model; conventional Baum-Welch or forward-backward algorithms are insufficient for dealing with the input-output relationships and transition conditions. Bengio and Frasconi (1994) approached the learning challenge by introducing the generalized expectation-maximization (GEM) to find the optimum parameters for the input-output functions and transition conditions simultaneously. However, for non-trivial, continuous systems, the GEM approach is likely to settle on local optima due to the inability of transition modelling to discriminate distinct modes. By dividing the problem into the CSR and TM subdomains, our approach leverages the property that each behavior is unique to infer accurate and consistent hybrid dynamical systems.

3.2 Clustered Symbolic Regression

The first algorithm is clustered symbolic regression (CSR), which involves using unsupervised learning techniques to solve the challenging issue of distinguishing individual functions while simultaneously infers a symbolic model for each of them. This novel algorithm is presented as a generalized solution to learning piecewise functions, distinct from the hybrid dynamics framework.

This subsection begins with a formal definition of the problem, followed by a brief overview of two learning approaches: symbolic regression (SR) and expectation-maximization (EM), respectively. These approaches are then unified as clustered symbolic regression.

3.2.1 PROBLEM DEFINITION

Consider the following, generalized piecewise function:

$$y_n = f(u_n) = \begin{cases} f_1(u_n) & , \quad \text{if } d_n \in D_1 \\ \vdots & , \quad \vdots \\ f_K(u_n) & , \quad \text{if } d_n \in D_K \end{cases}$$

where $u_n \in \mathbb{R}^p$ is the observable input vector at index n, $y_n \in \mathbb{R}^r$ is the output vector, $d_n \in \mathbb{R}^q$ is the domain input vector and D is a set of mutually exclusive membership subdomains. The domain input vector can be composed of both the observable variables, u_n, as well as latent variables, allowing for latent subdomains definitions. Given the number of subdomains, K, infer a model that minimizes the within-domain, absolute error, E:

$$E_{CSR} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{k,n} ||y_n - \hat{y}_n|| \tag{2}$$

where $\gamma_{k,n}$ is the probability that the input-output pair belongs to the subdomain D_k and $\hat{y}_n$ are model predictions of the output at time n.

In essence, this formulation is an unsupervised clustering problem. However, unlike traditional clustering problems, each cluster is represented by a symbolic expression and there is no prior knowledge regarding the structure of these submodels. There has been no reported work on mixture models where each component model is dependent on an arbitrary functional of the input; conventional mixture models assume that each cluster belongs to the same fixed-structure, parametric family of distributions (Bishop, 2006).

3.2.2 SYMBOLIC REGRESSION

The first component of CSR is symbolic regression (SR): an genetic programming algorithm that is capable of reliably inferring symbolic expressions that models a given data set (Cramer, 1985; Dickmanns et al., 1987; Koza, 1992; Olsson, 1995). Provided with a collection of building blocks and a fitness metric, SR attempts to find the combination of primitives that best maximizes the stated fitness function.

SR is a population-based, heuristic search algorithm which uses biologically-inspired evolutionary techniques to efficiently explore a boundless search space. The process begins with a random population of candidate expressions. A fitness metric, such as squared or absolute error, is used to rank each candidate based on its ability to model the data set. The best candidates are selected to produce the next generation of expressions, through evolution-inspired techniques such as stochastic recombination and mutation. This process is summarized in Figure 3.

In particular, SR uses a microprogram or tree structure to represent symbolic expressions. In a tree hierarchy, branches are mathematical operations while leaves are constants and input variables. This tree structure provides a terse representation of symbolic functions that is capable of representing a wide range of plausible expressions—for example, complex expressions, including non-linear and rational equations, can be easily represented. Generating random trees for initialization generally involves randomly filling the tree with nodes, sampling random variables and constants where necessary, allowing for arbitrarily sized expressions. Evaluating an expression requires simply substituting values from the data set, traversing the tree and computing well-defined

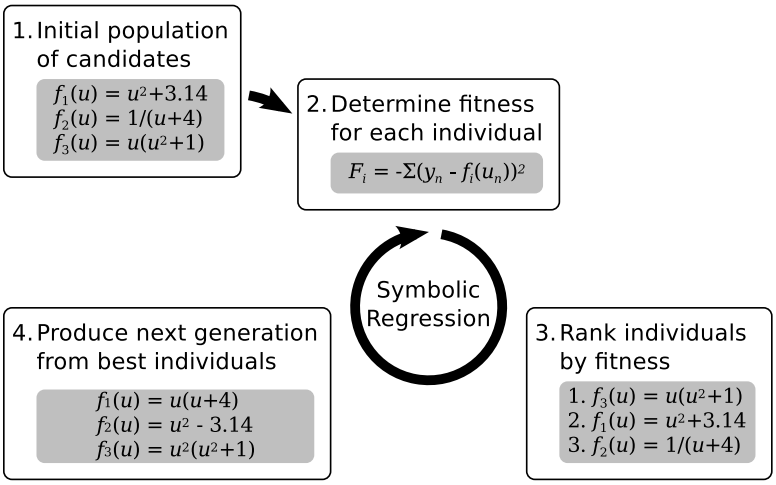


Figure 3: Flowchart describing the symbolic regression algorithm.

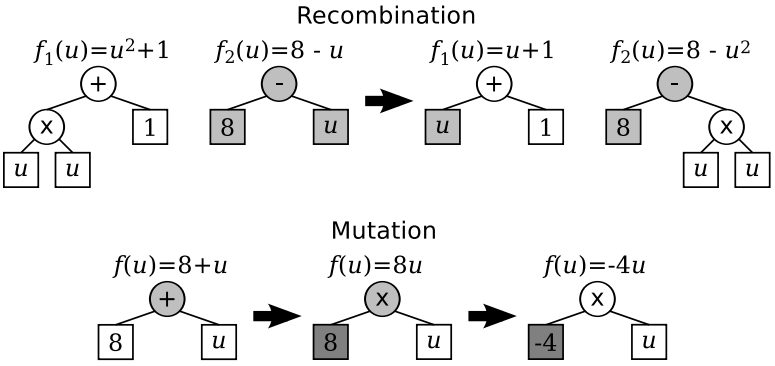


Figure 4: Symbolic expressions represented as tree structures, with examples of recombination and mutation operations.

operations. Furthermore, the tree structure is extremely amenable to the evolutionary processes of recombination and mutation through tree manipulations (Figure 4).

SR was chosen as the modeling algorithm because it provides three unique advantages. First, SR includes form and structure as part of the inference problem. Free form expressions are generated by rearranging primitives in a boundless tree structure, resulting in a rich range of possible expressions. In contrast, parametric models constrict their solution space to sums of basis and transfer functions.

Next, SR produces solutions that are easily interpreted. Unlike other machine learning algorithms which tweak a vast collection of intangible numerical parameters, symbolic expressions are the foundation of mathematical notation and often provide key insight into the fundamental relationships of such models.

Finally, an important quality of SR is its natural method to deal with overfitting, where the inferred model captures the peculiarities of the data set rather than the underlying truth. Overfitting is a major issue in machine learning and, to an even greater extent, multi-modal systems where one overfit behavior can cripple the progress in the remaining behavior. In symbolic expressions,

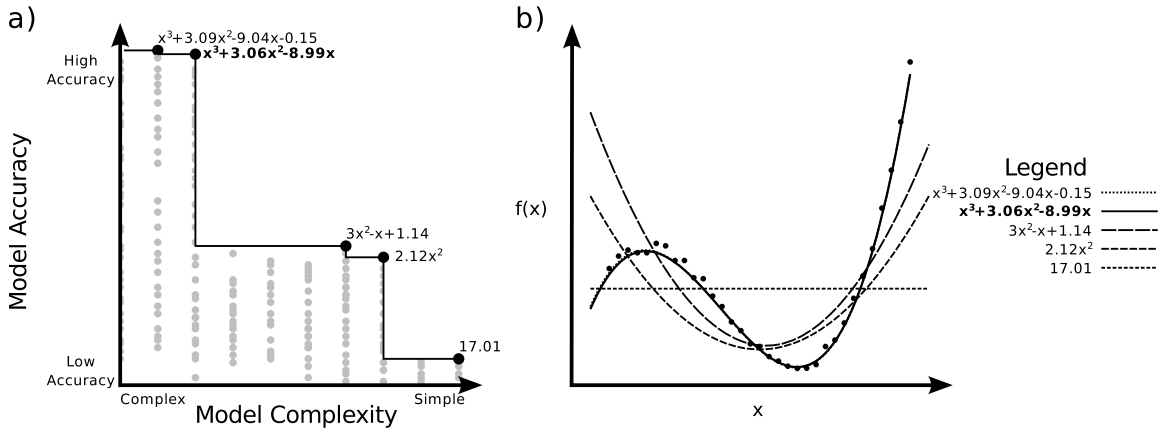


Figure 5: An example of the set of solutions provided by symbolic regression. a) The Pareto optimal solutions (in black) are expressions that have the best accuracy for all models of equal or lower complexity. Suboptimal solutions generated by SR are in grey. b) A plot of the non-dominated solutions with the corresponding the data set. The true model is in bold text, while the remaining solutions are either under- or overfit.

overfitting occurs by inferring a model with greater complexity than the ground truth—for example, a cubic model is used to fit a quadratic function. Thus, there is a fundamental trade-off between the accuracy and complexity of a candidate model, where overfitting incurs additional complexity to accurately model the noise contributions in the data.

Instead of simply reporting the most accurate model found by SR, which is susceptible to overfitting, the inherent population dynamics is leveraged to provide a multi-objective approach to dealing with overfitting. By design, SR generates numerous candidate models with varying degrees of complexity and accuracy. Rather than considering every considering every generated expression as a candidate model, individual expressions are compared against a continuously updated, multi-objective record. This approach, called Pareto optimization, forms a set of non-dominated solutions which provide the best fitness for a given complexity (Figure 5). This method reformulates the problem of overfitting as model selection along the accuracy and complexity trade-off, a property later exploited by CSR to reliably find solutions. As Pareto optimization is a post-processing technique that analyzes expressions only after they have been generated by SR, it does not interfere with the underlying search process.

Recent advances in SR implementations have made it a powerful search tool—even for difficult search spaces, it can often find good, if not globally optimal, solutions. It capable of non-linear regression and has even been shown to find differential equations (Iba, 2008), implicit equations (Schmidt and Lipson, 2009a) and even conservation laws (Schmidt and Lipson, 2009b). Despite the range of successful applications, SR is limited to individual functions. Currently, there are no algorithms that are capable of symbolically regressing unlabeled data generated by multi-modal systems, such as data from a hybrid dynamical system or piecewise function.

Algorithm 1 Generalized EM

```
input  → observed data - X
output → model parameters - θ
```

```
initialize model parameters - θ_old
while convergence is not achieved :
    # expectation step
        compute probability of observing latent variables - p(Z|X,θ_old)
    # maximization step
        compute new model parameters - θ_new = arg max_θ Σ_Z p(Z|X,θ_old) ln p(X|Z,θ)
        update model parameters - θ_old = θ_new
return model parameters θ_old
```

3.2.3 EXPECTATION-MAXIMIZATION

The second component of CSR is the expectation-maximization (EM) algorithm: a machine learning algorithm that searches for the maximum likelihood estimates of model parameters and latent variables. Formally, the EM solves the following problem: given a joint distribution $p(X,Z|\theta)$ over observed variables X and latent variables Z, governed by the model parameters θ, determine the parameter values that maximize the likelihood function $p(X|\theta)$ (Dempster et al., 1977).

The EM algorithm is an iterative two-step process, which begins with initially random model parameters. In the expectation step, the expected value of the latent variables is determined by calculating the log-likelihood function given the current model parameters. This is followed by the maximization step, where the model parameters are chosen in order to maximize the expected value given the latent variables. Each cycle of EM increases the incomplete-data log-likelihood, unless it is already at a local optimum. The implementation details of EM are summarized in Algorithm 1.

The EM algorithm is a popular framework for a variety of mixture models, including mixture of Gaussians, mixture of Bernoulli distributions and even Bayesian linear regression (Bishop, 2006). Although evolutionary computation has been applied to the EM framework (Martinez and Virtia, 2000; Pernkopf and Bouchaffra, 2005) , the focus has been on exploring different optimization approaches in the maximization step for mixture of Gaussians, as opposed to investigating different types of models found through evolutionary computation.

3.2.4 CLUSTERED SYMBOLIC REGRESSION

The clustered symbolic regression (CSR) is a novel algorithm that is capable of finding symbolic expressions for piecewise functions. By applying an EM framework to SR, this algorithm determines both the model parameters, mathematical expressions and the corresponding variances for each subfunction, as well as the latent variables, the membership of each data point, for a piecewise function.

To aid in the formulation of the algorithm, the SR optimization is interpreted in a statistical framework where the output of each subfunction defines the expected value conditional on the input and state, $f_k(u_n) = E[y_n|m_k,u_n]$, where m_k is 1 if $d_k \in D_k$ and 0 otherwise. Assuming that the noise follows a Gaussian distribution, then the following definition is obtained:

$$p_k(y_n|u_n) = \mathcal{N}\left(y_n|f_k(u_n),\sigma_k^2\right) \tag{3}$$

where $\mathcal{N}\left(x|\mu,\sigma^2\right)$ defines a Gaussian distribution over x with a mean μ and variance σ^2.

The expectation step consists of evaluating the expected membership values using the current model. Using the probabilistic framework for defining functions (Equation 3), the probability of membership, $\gamma_{k,n}$, of an input-output pair to a function f_k is:

$$\gamma_{k,n} = \frac{\mathcal{N}\left(y_n|f_k(u_n),\sigma_k^2\right)}{\sum_{k=1}^{K}\mathcal{N}\left(y_n|f_k(u_n),\sigma_k^2\right)}. \tag{4}$$

Note that the membership probability reinforces exclusivity—given two subfunctions with the same expression, the model with lower variance has stronger membership values over the same data. This property is advantageous given the assumption that each behavior is unique; as one subfunction becomes increasingly certain as a result of a decreased variance, the other subfunctions are forced to model the remaining data.

Next, the Maximization Step consists of finding the expressions for each behavior and variances that best explain the data points given the current membership distribution. The variance of each behavior is updated by computing the unbiased, weighted sample variance using the functions obtained by SR (Equation 5)

$$\sigma_k^2 = \frac{\sum_{n=1}^{N}\gamma_{k,n}}{(\sum_{n=1}^{N}\gamma_{k,n})^2 - \sum_{n=1}^{N}\gamma_{k,n}^2}\sum_{n=1}^{N}\gamma_{k,n}||y_n - f_k(u_n)||^2. \tag{5}$$

To find the behavior for each mode, SR is used to efficiently find the most suitable expression for the subfunction relationship:

$$y_n = f_k(u_n). \tag{6}$$

Although the CSR is designed to optimize the weighted absolute error (Equation 2), that fitness metric is not ideal for each local SR search. The individual data sets, described by membership probabilities, contain both measurement noise as well as classification error. Data from erroneous classifications often produces heavy-tail outliers. Thus, each local search requires a robust metric that does not assume exponentially bound likelihoods—the weighted, mean logarithmic error was selected as the fitness metric:

$$F_{\text{local},k} = -\frac{\sum_{n=1}^{N}\gamma_{k,n}\log(1+||y_n - f_k(u_n)||)}{\sum_{n=1}^{N}\gamma_{k,n}}. \tag{7}$$

However, applying the logarithmic fitness naively tended to bias the SR search to find the same expression for every behavior in the initial iteration, resulting in a symmetrical, local optimum for the EM algorithm. A greedy implementation of the EM algorithm that updates each behavior sequentially was used to resolve this issue. This approach enforces a natural priority to the learning algorithm allowing each behavior to model the data of its choice, forcing the remaining functions to the model the remaining data.

A primary issue with the EM algorithm is that it is sensitive to initial conditions and likely to settle on local optima. Local optima occurs when some solutions are overfit, which results in underfit solutions for the remaining behaviors. By exploiting the set of solutions provided by Pareto optimization, CSR is significantly more robust to initial conditions and able to find the global optima with greater consistency. Assuming the Pareto optimal set is exhaustive, if an overfit solution exists

in the Pareto optimal set, then the true and less complex solution also exists in the set. Thus, the challenge of avoiding local optima due to overfitting is reduced to selecting the most appropriate solution from this set.

Each solution in the Pareto optimal set is selected, temporary membership, $\gamma'_{k,n}$, and variances, σ'^2_k, are calculated and the global error is determined (Equation 2). The global error is used to compute the Akaike Information Criterion (Akaike, 1974) score, a metric that rewards models that best explain the data with the least number of parameters:

$$AIC = 2c + N \log |E_{CSR}| \tag{8}$$

where c is the number of nodes in the tree expression and N is the number of data points. The solution with the lowest global AIC score is deemed to have the most information content, and thus, is the most appropriate solution. Although other information based methods are available (Solomonoff, 1964; Wallace and Dowe, 1999), AIC was used because of its ease of application.

Note that while the AIC score is used for model selection, using this metric directly in the SR as a fitness function often leads to inferior results as it biases the search space to look for simple solutions which can lead to underfit models. Instead, this approach of focusing solely on model accuracy, populating a set of candidate solutions ranging in complexity and and then using the AIC to select from this list proved to produce the most consistent and reliable models.

The complete CSR algorithm is summarized in Algorithm 2.

3.3 Transition Modeling

The second algorithm is transition Modeling (TM), which is a supervised learning technique that determines symbolic, discriminant inequalities for transition events by restating a classification problem as a regression problem using function composition. This algorithm is presented as a generalized solution to classification with symbolic expressions, separate from the hybrid dynamics framework. This subsection begins with a formal definition of the problem, followed by a discussion of related work and description of the algorithm.

3.3.1 PROBLEM DEFINITION

Consider the general, binary classification problem:

$$\zeta_n = \begin{cases} 1 & , \text{ if } u_n \in Z \\ 0 & , \text{ if } u_n \notin Z \end{cases} \tag{9}$$

where $u_n \in \mathbb{R}^p$ is the input vector at index n, $\zeta_n \in \mathbb{B}$ is the corresponding label and Z is the characteristic domain. Infer the discriminant function which describes the characteristic domain that minimizes the classification error:

$$E_{TM} = \sum_{n=1}^{N} ||\zeta_n - \hat{\zeta}_n||. \tag{10}$$

Despite the variety of approaches to binary classification, the explicit requirements for a nonlinear, symbolic model of the discriminant function makes this problem challenging. While multiclass classification may be more appropriate, it results in a significantly more challenging problem for symbolic models and is left for future work.

Algorithm 2 Clustered Symbolic Regression

```
input  → unclustered input-output data - u_n, y_n
       → the number of subfunctions - K
output → behavior for each mode - f_k(u_n)
       → variance for each mode - σ_k^2
```

```
function symbolic_regression(search_relationship, fitness_function) :
    initialize population with random expressions defined by search_relationship
    for predefined computational effort :
        generate new expressions from existing population (Figure 4)
        calculate fitness of all expressions according to fitness_function
        remove unsuitable expressions from the population
        for each pop_expr in the population :
            for each pareto_expr in the pareto_set :
                if ((pop_expr.fitness > pareto_expr.fitness) and
                    (pop_expr.complexity <= pareto_expr.complexity)) :
                    add pop_expression to pareto_set
                    remove pareto_expression from pareto set
    return pareto_set
```

```
initialize random membership values
for each behavior in K modes :
    sr_solutions = symbolic_regression(Equation 6, Equation 7)
    set behavior f_k to solution with lowest local AIC score in sr_solutions
set variance for each behavior - σ_k^2 (Equation 5)
while convergence is not achieved :
    for each behavior in K modes :
        # expectation step
            for all the N data points :
                compute membership values - γ_{k,n} (Equation 4)
        # maximization step
            sr_solutions = symbolic_regression(Equation 6, Equation 7)
            for each solution in sr_solutions :
                compute temporary membership values - γ̌_{k,n} (Equation 4)
                compute temporary variance - σ̌^2_k (Equation 5)
                compute global fitness using temporary values - E_CSR (Equation 2)
                compute AIC score using global fitness (Equation 8)
            set behavior f_k to solution with lowest AIC score in sr_solutions
            set variance to corresponding value - σ_k^2 (Equation 5)
return behaviors f_k and variances σ_k^2
```

3.3.2 RELATED WORK

Although using evolutionary computation for classification has been previously investigated, this algorithm is novel due to its reformulation of the classification problem as symbolic regression, providing an assortment of benefits.

The majority of classifying evolutionary algorithms impose a fuzzy logic structure with triangular or trapezoidal membership domains (Jagielska et al., 1999; Arslan and Kaya, 2001; Mendes et al., 2001). A genetic algorithm is then used to optimize the parameters of these fixed-structure discriminant functions. This technique is difficult to scale to non-linear, multi-inputs domains as it

only searches for the model parameters using a fixed model structure. Furthermore, the solutions may be difficult to interpret or express succinctly as the number of domains increases.

Muni et al. (2004) designed an evolutionary program that is capable of generating symbolic expressions for discriminant functions. This program was limited to a classification framework, resulting in application-specific algorithms, fitness metrics and implementations. Our approach is novel as it adapts the well-developed framework of SR, allowing for a unified approach to both domains.

3.3.3 TRANSITION MODELING ALGORITHM

The Transition Modeling (TM) algorithm builds on the infrastructure of SR. The discriminant functions are expressed symbolically as an inequality, where the data has membership if the inequality evaluates to true. For example, the inequality $Z(u) : u \geq 0$ denotes the membership for positive values of u, while $Z(u_1, u_2) : u_1^2 + u_2^2 \leq r^2$ describes membership for an inclusive circle of radius r. The key insight in reforming the classification problem into a regression problem is that function composition with a Heavyside step function is equivalent to searching for inequalities:

$$\zeta = \text{step}(x) = \begin{cases} 1 & , x \geq 0 \\ 0 & , x < 0 \end{cases} .$$

Using the step function and function composition, the classification problem (Equation 9) is reformatted as a standard symbolic regression problem using the search relationship:

$$\zeta_n = \text{step}(Z(u_n)).$$

This reformulation allows a symbolic regression framework to find for symbolic, classification expressions, $Z(\cdot)$, that define membership domains. The expression is readily transformed into an inequality, $Z(\cdot) \geq 0$, allowing for natural interpretation.

Although the step function illustrates the relationship between TM and SR, it is actually difficult to use in practice due to the lack of gradient in the fitness landscape. Small perturbations in the expression are likely to have no effect on the fitness, which removes any meaningful incremental contributions from gradient dependent techniques, such as hill climbing. Thus, searching with step functions requires that the exact expression is found through the stochastic processes of recombination and mutation, which may lead to inconsistent results and inefficient computational effort. Instead, a function composition with the sigmoid (Equation 11) was found to be more practical as a 'soft' version of the step function, leading to the search expression in Equation 12 while still using the fitness metric (Equation 10).

$$\text{sig}(x) = \frac{1}{1 + e^{-x}}, \tag{11}$$

$$\zeta_n = \text{sig}(Z(u_n)). \tag{12}$$

The sigmoid function provides three important benefits. First, it provides a quantified measure of degree of belief. In the limit of $|x| \to \infty$, the sigmoid function approaches the step function. Thus, the magnitude of the scaling factor in $Z(\cdot)$ provides a numerical measure of the certainty of the classifier; confident classifiers have expressions with large scaling factors. Furthermore, for ease of interpretability, the scaling factor is easily removed via by algebraic simplifications. The second

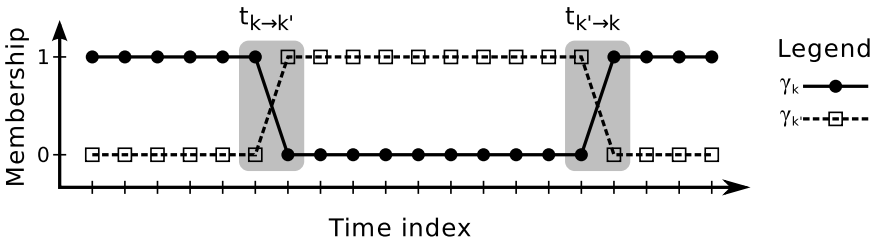


Figure 6: An example of the time-series data of membership signals. Transitions are highlighted in grey.

benefit is that sigmoid TM provides an elegant method to deal with uncertain or fuzzy memberships. Since the sigmoid is a continuous function ranging from 0 to 1, it is able to represent all degrees of membership as opposed to purely Boolean classification. The final benefit is inherited from SR: a range of solutions is provided via Pareto optimality, balancing model complexity and model accuracy, and model selection is used to prevent overfit solutions.

3.4 Modeling Hybrid Dynamical Systems

To infer symbolic models of hybrid dynamical systems, two general CSR and TM algorithms are applied to form the multi-modal symbolic regression algorithm (MMSR). CSR is first used to cluster the data into distinct modes while simultaneously inferring symbolic expressions for each subfunction. Using the modal membership from CSR, TM is subsequently applied to find symbolic expressions for the transition conditions. Of the 4-tuple description of in Section 2.2, $\mathcal{H} = (\mathcal{W}, \mathcal{M}, \mathcal{F}, \mathcal{T})$, the communication space, $\mathcal{W}$, is provided by the time-series data and it is the goal of MMSR to determine the modes, $\mathcal{M}$, behaviors, $\mathcal{F}$, and transitions, $\mathcal{T}$.

Using the unlabeled time-series data, the first step is to apply CSR. CSR determines the modes of the hybrid system, $\mathcal{M}$, by calculating the membership of an input-output pair (expectation step of Algorithm 2). Simultaneously, CSR also infers a non-linear, symbolic expression for each of the behaviors, $\mathcal{F}$, through weighted symbolic regression (maximization step of Algorithm 2).

Using the modal memberships from CSR, TM searches for symbolic expressions of the transition events, $\mathcal{T}$. To find the transitions, the data must be appropriately pre-processed within the hybrid system framework. Transition events are defined as the conditions for which the system moves from one mode to another. Using the membership values from CSR to determine the mode at every data point, searching for transition events is rephrased as a classification problem: a transition from mode k to mode k' occurs at index n if and only if $\gamma_{k,n} = 1$ and $\gamma_{k',n+1} = 1$ (Figure 6). Thus, the classification problem is applied to membership levels of the origin and destination modes. For finding all transition events from mode k to mode k', the search relationship and fitness metric are respectively:

$$\gamma_{k',n+1} = \text{sig}(t_{k\to k'}(u_n)),$$
$$F_{\text{transition}} = -\sum_{n=1}^{N-1} \gamma_{k,n} ||\gamma_{k',n+1} - \text{sig}(t_{k\to k'}(u_n))||^2.$$

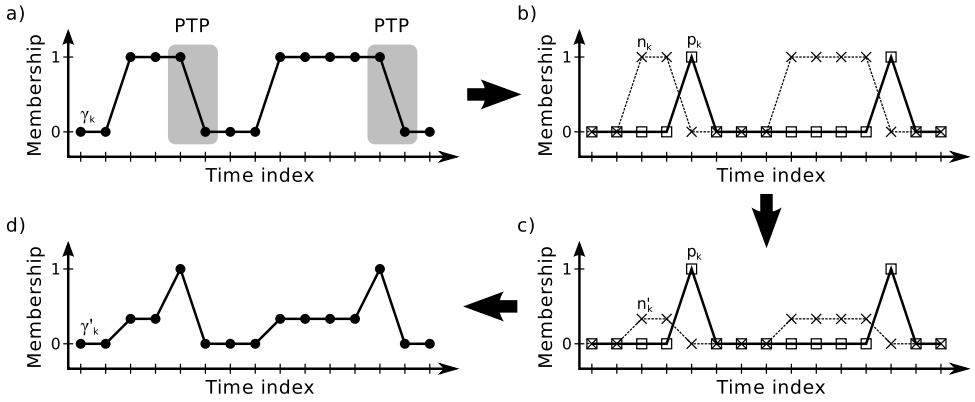


Figure 7: An example of PTP-NTP weight balance. a) Original weight data ($\gamma_{k,n}$). b) Weight data decomposed into $p_{k,n}$ and $n_{k,n}$ signals. c) Scaled $\tilde{n}_{k,n}$ signal. d) $p_{k,n}$ and $\tilde{n}_{k,n}$ recombined to form balanced ($\tilde{\gamma}_{k,n}$).

It is important to realize that most data sets are heavily biased against observing transitions—the frequency at which a transition event occurs, or a positive transition point (PTP), is relatively rare compared to the frequency of staying in the same node, or negative transition point (NTP). A PTP is defined mathematically for mode k at index n if $\gamma_{k,n} = 1$ and $\gamma_{k,n+1} = 0$; all other binary combinations of values are considered NTPs. This definition is advantageous since PTPs are identified by only using the membership information of only the current mode, $\gamma_{k,n}$, and no other membership information from the other modes are required.

The relative frequencies of PTP and NTP affects the TM algorithm since the data set is imbalanced: the sum of the weights associated with NTPs is significantly larger than the respective sum for PTPs. As a consequence, expressions which predict that no transitions ever occur result in a high fitness. Instead, equal emphasis on PTPs and NTPs via a simple pre-processor heuristic was found to provide much better learning for TM.

The first step in this weight rebalance pre-processing is to generate two new time-series signals, $p_{k,n}$ and $n_{k,n}$, which decomposes the membership data into PTP and NTP components, respectively (Equation 13-14). The $n_{k,n}$ signal is then scaled down by the ratio of the sum of the two components (Equation 15), which ensures that the $\tilde{n}_{k,n}$ signal has equal influence on TM as the $p_{k,n}$ signal. Finally, the components are recombined to produced the new weights, $\tilde{\gamma}_{k,n}$ (Equation 16). This process is illustrated in Figure 7.

$$p_{k,n} = \gamma_{k,n}(1 - \gamma_{k,n+1}), \tag{13}$$

$$n_{k,n} = \gamma_{k,n} - p_{k,n}, \tag{14}$$

$$\tilde{n}_{k,n} = n_{k,n} \frac{\sum_{n=1}^{N-1} p_{k,n}}{\sum_{n=1}^{N-1} n_{k,n}}, \tag{15}$$

$$\tilde{\gamma}_{k,n} = p_{k,n} + \tilde{n}_{k,n}. \tag{16}$$

A benefit of this formulation is that it can be applied for uncertain or fuzzy membership values. To summarize, after the pre-processing for PTP-NTP weight rebalance described in Equation 13-16, the search relationship in Equation 17 and fitness metric in Equation 18 is applied to TM for finding

all transition events from mode k to mode k'. The best expression is selected using the AIC ranking based on the transition fitness.

$$\gamma_{k',n+1} = \text{sig}(t_{k\to k'}(u_n)), \tag{17}$$

$$F_{\text{transition}} = -\frac{\sum_{n=1}^{N-1} \tilde{\gamma}_{k,n} ||\gamma_{k',n+1} - \text{sig}(t_{k\to k'}(u_n))||^2}{\sum_{n=1}^{N-1} \tilde{\gamma}_{k,n}}. \tag{18}$$

The complete MMSR algorithm to learn analytic models of hybrid dynamical systems is summarized in Algorithm 3.

4. Results

This section begins with a description of the experimental setup for both the synthetic and real data experiments. Next is a discussion of the synthetic experiments, starting with an overview of alternative approaches, a list of the performance metrics, a summary of four data sets and finally, a discussion of MMSR performance in comparison to the baseline approaches. MMSR is then used to identify and characterize field-effect transistor modes, similar to those derived from first principles, based on real data. This section concludes with a brief discussion of the scalability of MMSR.

4.1 Experimental Details

In these experiments, the publicly available Eureqa API (Schmidt and Lipson, 2012) was used as a backend for the symbolic regression computation in both the CSR and TM. To illustrate the robustness of MMSR, the same learning parameters were applied across all the data sets, indicating that task-specific tuning of these parameters was not required:

- The SR for CSR was initially executed for 10000 generations and this upper limit was increased by 200 generations every iteration, until the global error produced less than 2% change for five EM iterations. Once CSR was complete, the SR for TM was a single 20000 generation search for each transition.

- The CSR algorithm was provided all the continuous inputs, while the TM algorithm was also provided with the one-hot encoding of binary signals, according to the data.

- The default settings in Eureqa, the SR backend, were used:

  - Population size = 64
  - Mutation probability = 3%
  - Crossover probability = 70%

- The basic algebraic building blocks were used for both algorithms: $\{\text{constants}, +, -, \times, /\}$. These building blocks were chosen as they form a fundamental set of basis operations that are capable of constructing more complex expressions. Additional building blocks such as trigonometric or transcendental functions could be included, but in their absence, numerical approximations, such as Taylor expansions, are inferred.

Algorithm 3 Multi-modal Symbolic Regression

```
input  → unclustered input-output data - u_n, y_n
       → the number of subfunctions - K
output → behavior for each mode - f_k(u_n)
       → variance for each mode - σ_k^2
       → transitions between each mode - t_{k→k'}(u_n)
```

```
function symbolic_regression(search_relationship, fitness_function) :
    initialize population with random expressions defined by search_relationship
    for predefined computational effort :
        generate new expressions from existing population (Figure 4)
        calculate fitness of all expressions according to fitness_function
        remove unsuitable expressions from the population
        for each pop_expr in the population :
            for each pareto_expr in the pareto_set :
                if ((pop_expr.fitness > pareto_expr.fitness) and
                    (pop_expr.complexity <= pareto_expr.complexity)) :
                    add pop_expression to pareto_set
                    remove pareto_expression from pareto set
    return pareto_set
```

```
# Clustered symbolic regression
initialize random membership values
for each behavior in K modes :
    sr_solutions = symbolic_regression(Equation 6, Equation 7)
    set behavior f_k to solution with lowest local AIC score in sr_solutions
set variance for each behavior - σ_k^2 (Equation 5)
while convergence is not achieved :
    for each behavior in K modes :
        # expectation step
            for all the N data points :
                compute membership values - γ_{k,n} (Equation 4)
        # maximization step
            sr_solutions = symbolic_regression(Equation 6, Equation 7)
            for each solution in sr_solutions :
                compute temporary membership values - γ̌_{k,n} (Equation 4)
                compute temporary variance - σ̌^2_k (Equation 5)
                compute global fitness using temporary values - E_CSR (Equation 2)
                compute AIC score using global fitness (Equation 8)
            set behavior f_k to solution with lowest AIC score in sr_solutions
            set variance to corresponding value - σ_k^2 (Equation 5)
# Transition modelling
for each mode k in K modes :
    for each different mode k' in K − 1 modes :
        rebalance the PTP and NTP weights (Equation 13-16)
        tm_solutions = symbolic_regression(Equation 17, Equation 18)
        for each solution in tm_solutions :
            compute AIC score using transition fitness (Equation 8)
        set transition t_{k→k'}(u_n) to solution with lowest AIC score in tm_solutions

return behaviors f_k, variances σ_k^2 and transitions t_{k→k'}(u_n)
```

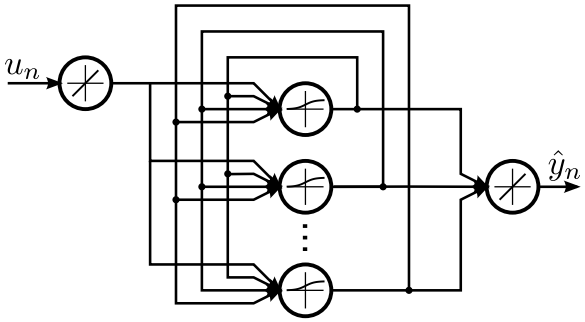


Figure 8: Schematic diagram of the fully recurrent neural network.

4.2 Synthetic Data Experiments

This section discusses a collection of experiments on hybrid systems generated by computer simulation. It begins with an introduction of alternative multi-modal model inference approaches, followed by an outline of the metrics used to measure their performance and a description of the data sets used for model comparison. This section concludes with a summary and discussion of the experimental results.

4.2.1 ALTERNATIVE MODELS

This subsection describes two traditional machine learning approaches to modeling multi-modal time-series data: fully recurrent neural networks and neural network based, input-output hidden Markov machines.

Fully Recurrent Neural Network—A fully recurrent neural network (RNN) is a neural network where connections form a directed cycle (Figure 8). This baseline recurrent network was composed of an input layer of nodes with linear transfer functions, a single hidden layer of nodes with sigmoidal transfer functions and a linear transfer function as the output node. The output of the hidden layer was consequently fed back as an input with a one cycle delay, allowing the network to store memory and making it capable of modelling multi-modal behavior.

The network was implemented using the open source, machine learning library PyBrain (Schaul et al., 2010) and was trained via backpropagation through time (Rumelhart et al., 1986). The training data was split into a training and validation subset, where the training subset consists of the initial contiguous 75% portion of the data. The training was terminated either via early stopping or when the training error decreased by less than 0.01% for 10 iterations. The size of the hidden layers, h, ranged from 10, 25, 50 to 100 nodes based on complexity of the data set. The weights were initialized by sampling a zero mean Gaussian random variable with a standard deviation of 1. The learning rate was $\varepsilon = 0.0005/h$ and used a momentum of 0.1. The learning rate was sufficiently small that the gradients never grew exponentially.

Neural Network Based IOHMM—The neural network based IOHMM (NNHMM) architecture by Bengio and Frasconi (1994) is a Markov model that also captures input-output relationships. Provided with the number of modes, NNHMM uses two collections of neural networks: one to predict the input-output mapping of each mode and another to predict the distribution of states. As no prior information was provided, the networks are designed to be as general as possible: a multilayer perception and one layer of hidden nodes with sigmoidal transfer functions. The input-

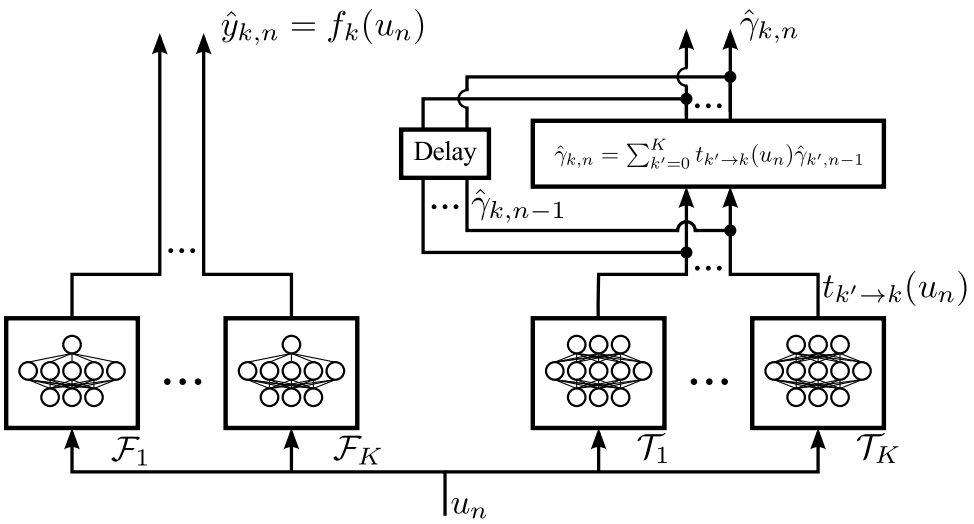


Figure 9: Schematic diagram of the neural network based IOHMM architecture (Bengio and Frasconi, 1994). $\mathcal{F}_k$ are regression networks with a single sigmoidal hidden layer and $\mathcal{T}_k$ are softmax networks with a single sigmoidal hidden layer.

output networks used linear input and output layers while the state prediction network used a linear input layer and a softmax output layer.

The generalized EM algorithm (GEM) was applied for training was terminated either via early stopping or when the validation error produced less than a 0.01% decrease for 50 EM iterations. The size of the hidden layers, h, are identical for every network in the architecture and ranged from 5, 10, and 20 nodes. The weights were initialized by sampling a zero mean Gaussian random variable with a standard deviation of 1. The learning rate was $\varepsilon = 0.0002/h$ with no momentum. The learning rate was sufficiently small that the gradients never grew exponentially.

4.2.2 PERFORMANCE METRICS

There are three performance metrics of interest: model accuracy, complexity and fidelity.

Model accuracy is a measure of ability of a learning algorithm to predict outputs given inputs. The trained model is used to predict the evolution of the time series data and the error is the negative log probability of a mixture of Laplacians (Equation 1).

For time series prediction, the accumulation of the state error over time, also known as drift, becomes a significant factor. Repeated iterations of accurate but not-perfect transitions over a prolonged period of time will result in a significant accumulated error. Drift is managed with a closed-loop system (Figure 10), where the output of the previous time step is also provided. With the MMSR algorithm, a closed-loop model is trivially constructed by setting the previous state probabilities according to the clustering component in CSR. However, the neural network based algorithms cannot be reformed into a closed-loop model without retraining the network or adapting the framework to execute some form of clustering.

Model complexity is a measure of the total number of free parameters required for the model. For MMSR, the complexity is dynamic and is measured as the sum of nodes in the expression trees.

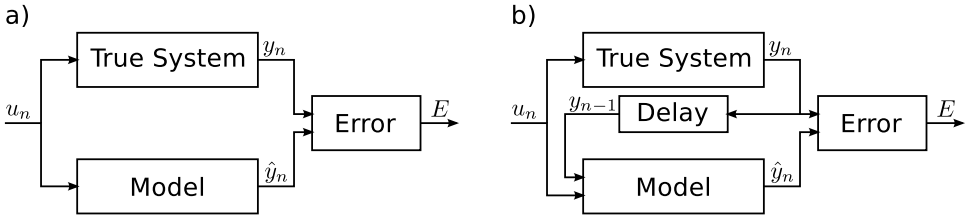


Figure 10: Schematic diagram of an open-loop and closed-loop system in a) and b), respectively.

Data Set	Mode (m_k)	Behavior (f_k)	No. of Points	Destination Mode ($m_{k'}$)	Transition ($t_{k\to k'}$)	No. of Transitions
Hysteresis Relay	1	$y=1$	2037	2	$u>0.5$	65
	2	$y=-1$	2059	1	$u<-0.5$	65
Continuous Hysteresis	1	$y=0.5u^2+u-0.5$	2051	2	$u>0.98$	40
	2	$y=-0.5u^2+u+0.5$	2045	1	$u<-0.98$	40
Phototaxic Robot	1	$y=u_2-u_1$	1568	2	$u_4=1$	36
				3	$u_5=1$	34
	2	$y=1/(u_1-u_2)$	1257	1	$u_3=1$	31
				3	$u_5=1$	40
	3	$y=0$	1271	1	$u_3=1$	38
				2	$u_4=1$	35
Non-linear System	1	$y=u_1u_2$	1302	3	$u_1^2+u_2^2<9$	331
	2	$y=6u_1/(6+u_2)$	1535	1	$u_1^2+u_2^2>25$	332
	3	$y=(u_1+u_2)/(u_1-u_2)$	1259	2	$u_1u_2>0$	332

Table 1: Summary of test data sets

The neural network based algorithms have a static complexity, which is the number of hidden nodes in all of subnetworks. Although the node count does not account for the complexity of operations and more comprehensive measures exist (Vladislavleva et al., 2009), it does provide a simple and coarse measure of complexity and acts as a first approximation to human interpretability.

Model fidelity is a measure of the MMSR's ability to reproduce the form or mathematical structure of original system. This metric is important as it integral to the primary goal of knowledge extraction—predictive accuracy is insufficient as the models must reproduce the expressions and not an approximation.

In symbolic representations, expressions are considered equivalent if and only if each subtree differs by at most scalar multiplicatives. For example, the expression $y=u^2/(1+u)$ is considered to be equivalent to $y=1.1u^2/(0.9+u)$, but the Taylor series approximation about $u=1\to y=-0.125+0.5u+0.125u^2$ is considered dissimilar regardless of its numeric accuracy. The fidelity is measured as the percentage of correctly inferred expression forms. In comparison, all neural network based systems are function approximations by design and thus, are immeasurable with respect to model fidelity.

4.2.3 DATA SETS

Since there are no standardized data sets for the inference of hybrid dynamical systems, the MMSR algorithm was evaluated on a collection of four data sets based on classical hybrid systems (Henzinger, 1996; van der Schaft and Schumacher, 2000) and intelligent robotics (Reger et al., 2000).

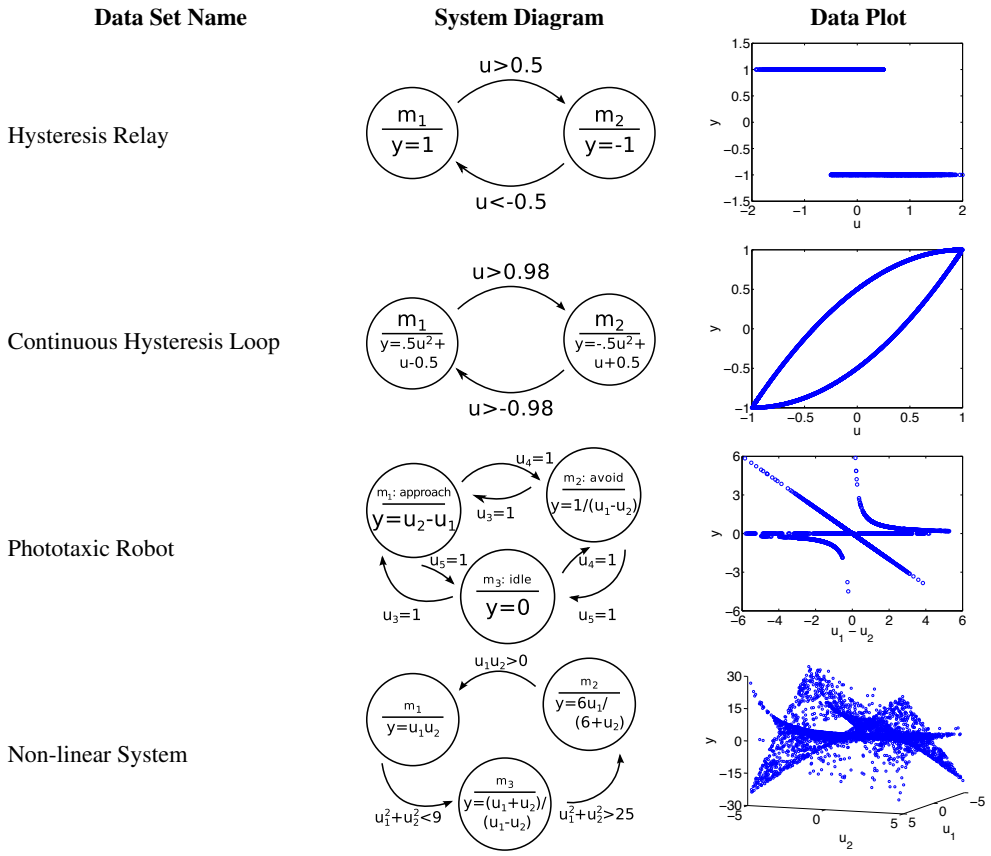


Figure 11: The system diagram and plots of the noiseless test data sets.

These data sets range in complexity in both the discrete and continuous domains. Furthermore, these data sets contain non-trivial transitions and behaviors, and thus, present more challenging inference problems than the simple switching systems often used to evaluate parametric models of hybrid systems (Le et al., 2011). Simple switching systems have trivial discrete dynamics where the transition to any mode does not depend on the current mode.

Training and test sets were generated; the training sets were corrupted with varying levels of additive Gaussian noise, while the test sets remained noiseless. The level of noise was defined as the ratio of the Gaussian standard deviation to the standard deviation of the data set (Equation 19). The noise was varied from 0% to 10% in 2% increments.

$$N_p = \frac{\sigma_{\text{noise}}}{\sigma_y}. \tag{19}$$

The statistics of all four data sets are summarized in Table 1, while the system diagrams and test data set are shown in Figure 11.

Hysteresis Relay—The first data set is a hysteresis relay: a classical hybrid system (Visintin, 1995; van der Schaft and Schumacher, 2000). It is the fundamental component of hysteresis models and consists of two modes: 'switched-on' and 'switched-off'. Each mode has a constant output and transitions occur at a threshold of the input. Although it is a simple hybrid dynamical system with

linear behaviors, it does not exhibit simple switching as the transitions depend on the mode since both behaviors are defined for $u \in [-0.5, 0.5]$.

Continuous Hysteresis Loop—The second data set is a continuous hysteresis loop: a non-linear extension of the classical hybrid system (Visintin, 1995). The Preisach model of hysteresis is used, where numerous hysteresis relays are connected in parallel and summed. As the number of hysteresis relays approaches infinity, a continuous loop is achieved. The data set is generated by repeatedly completing a single pass in the loop. Although there are still two modes, this data set is significantly more complex due to the symmetry of error functions about the line $y = u$, as well as the fact that transition depend on the mode and occur at a continuity in the output domain.

Phototaxic Robot—The third data set is a light-interacting robot (Reger et al., 2000). The robot has phototaxic movement: it either approaches, avoids, or remains stationary depending on the color of light. The output y is velocity of the robot. There are five inputs: u_1 and u_2 are the absolute positions of robot and light, respectively, while $\{u_3, u_4, u_5\}$ is a binary, one-hot encoding of the light color, where 0 indicates the light is off and 1 indicates the light is on. This modeling problem is challenging due to the variety of inputs and non-uniform distribution of data. However, it does exhibit simple modal switching behavior that only depends on the light input.

Non-linear System—The fourth and final data set is a system without any physical counterpart, but the motivation for this system was to evaluate the capabilities of the learning algorithms for finding non-linear, symbolic expressions. The system consists of three modes, where all of the behaviors and transition conditions consist of non-linear equations which cannot be modeled via parametric regression without incorporating prior knowledge. All the expressions are a function of the variables u_1 and u_2, the discriminant functions are not linearly separable and the transitions are modally dependent.

4.2.4 EXPERIMENTAL RESULTS

MMSR, along with the two parametric baselines, was evaluated on all four data sets and the performance metrics are summarized in Figure 12. This section begins with overview of the algorithms' general performance, followed by case study analysis of each data set in the following subsections.

First, MMSR was able to reliably reconstruct the original model from the unlabeled, time-series data. The process of converting the program output into a hybrid automata model is summarized in Figure 13, from a run obtained on the light-interacting robot training data with 10% noise. Provided with the number of modes, the algorithm searched for distinct behaviors and their subsequent transitions, returning a single symbolic expression for each of the inferred components. The expressions were algebraically simplified as necessary, and a hybrid dynamical model was constructed.

Comparing the algorithms on predictive accuracy, the closed-loop MMSR model outperformed the neural network baselines on every data set across all the noise conditions. The open-loop MMSR model was able to achieve similar performance to its closed-loop counterpart for most systems, with the exception of the noisy continuous hysteresis loop. For low noise conditions, MMSR achieves almost perfect predictions, even in open-loop configurations.

In comparison, the RNN approach had difficulty modeling the time-series data sets while NNHMM performed marginally better. As the model accuracy is normalized by the standard deviation of the data set, these neural network baselines was able to capture some characteristics of the data set and performed much better than predicting the mean of the data, which would achieve an error of 1.

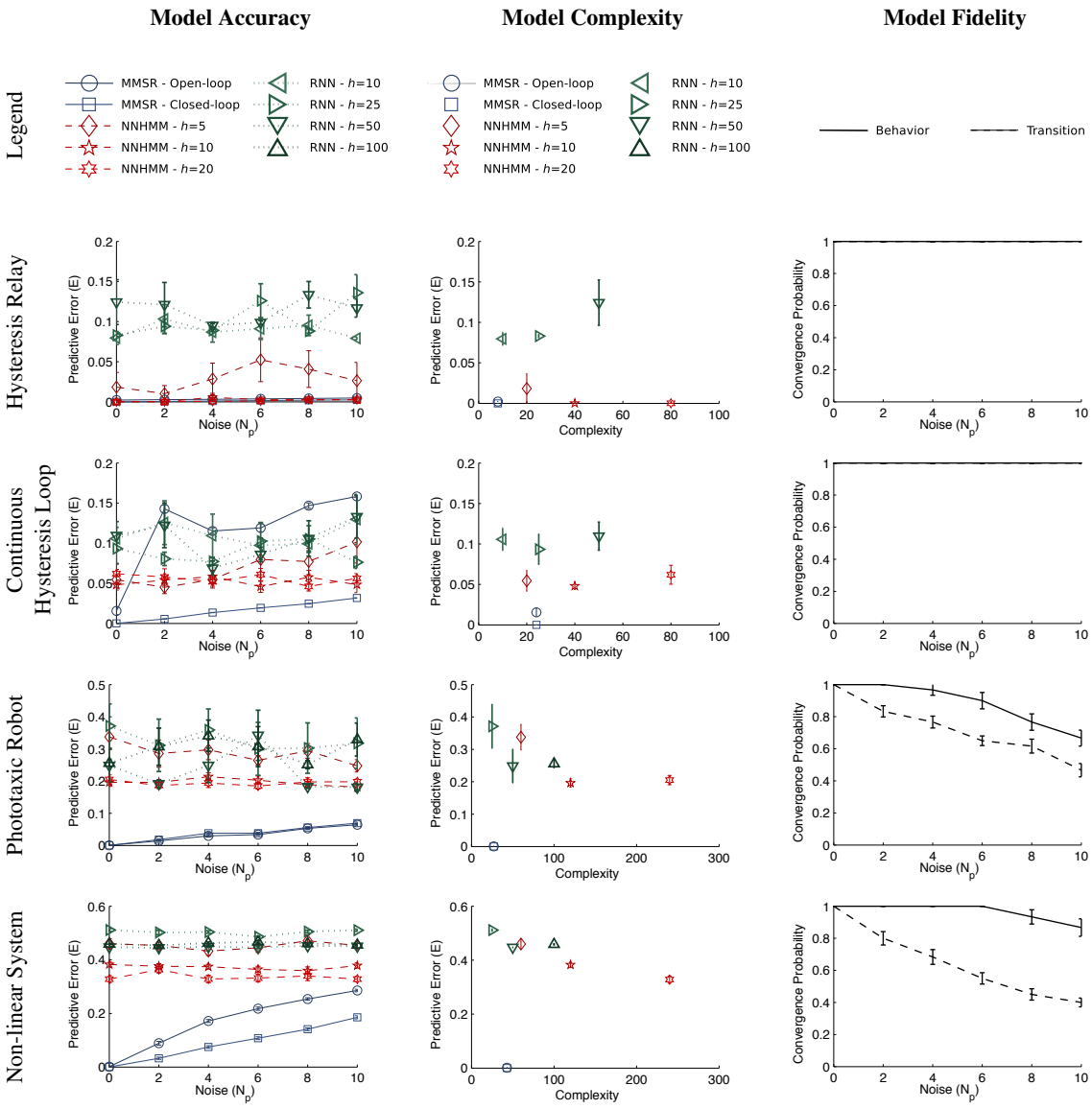


Figure 12: The performance metrics on four systems. Error bars indicate standard error ($n = 10$).

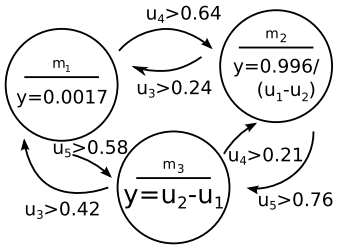
However, other than the simplest data set, none of the parametric approaches were able to converge on an accurate representation, even with noiseless training data.

There is an inverse relationship between the generality of the algorithm and its performance at inferring hybrid dynamical systems. Although RNNs are capable of representing a wide variety of phenomena, the learning algorithm often settles on a poor local optimum while NNHMM leverages a structural composition to achieve marginally better performance. MMSR, however, is tailored to inferring hybrid dynamical systems from unlabeled, time-series data and consequently infers a superior model for numerical predictions.

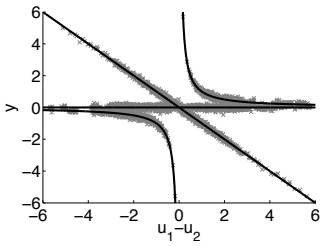
```
//Behaviors
f1(u)=0.0017
f2(u)=0.996/(u1-u2)
f3(u)=u2-u1

//Transitions
f1->f2=sig(11.87*u4-7.60)
f1->f3=sig(128.0*u1^2*u3-30.72*u1^2)
f2->f1=sig(16.11*u2^2*u5-12.35*u2^2)
f2->f3=sig(45.09*u2^2*u3-9.47*u2^2)
f3->f1=sig(174.9*u4-73.49)
f3->f2=sig(9.28*u1^2*u5-5.38*u1^2)
```

(a) Program Output



(b) Inferred System Diagram



(c) Behaviors with Training Data

Figure 13: Conversion from program output to hybrid dynamical model for the phototaxic robot with 10% noise. Algebraic simplifications were required to convert program output (a) to inequalities in canonical form (c).

Furthermore, not only was MMSR a superior predictor, the numerical accuracy was achieved with less free parameters than the neural network baselines. Even though the measure of counting nodes provides only a coarse measure of complexity, the neural network approaches have significantly more error despite having up to five times the number of free parameters on noiseless training data. This suggests that the symbolic approach is better suited for the primary goal of knowledge extraction, by providing accurate as well as parsimonious models.

In addition, for the neural network approaches, increasing the model complexity does not necessarily result in greater accuracy. In fact, for most data sets, once the number of hidden nodes reached a threshold, the trained models generally become less accurate despite having additional modelling capabilities. For multi-model problems, the parameter space is non-convex and contains local optima—as the number of hidden nodes increases, the probability of finding a local optima increases as well. Thus, for parametric models, the number of hidden nodes must be tuned to account for the complexity of the data set, presenting another challenge to the arbitrary application of parametric models.

Finally, MMSR was able to achieve reliable model fidelity. In the noiseless training sets, the correct expressions for both the behaviors and events were inferred with perfect reliability. As the signal to noise ratio was increased, the probability of convergence varied significantly depending on the characteristics of the data set. Generally, the algorithm was able to repeatedly find the correct form for the behaviors for the majority of the data sets. In contrast, the transition expressions were more difficult to infer since the model fidelity deteriorates at lower noise levels. This is a result of TM's dependence on accurate membership values from CSR—noisy data leads to larger classification errors, amplifying the challenge of modeling transitions.

Despite the model fidelity's sensitivity to noise, the algorithm was nonetheless able to accurately predict outputs for a wide range of noise conditions. The inferred expressions, regardless of the expression fidelity, were still accurate numerical approximations for both open- and closed-loop models.

Hysteresis Relay—This simple data set was modeled by accurately by both MMSR and NNHMM, while RNN had relative difficulties. This was the only data set that NNHMM was able to achieve

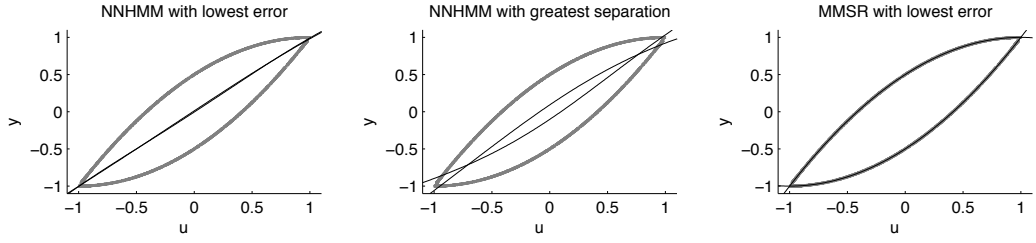


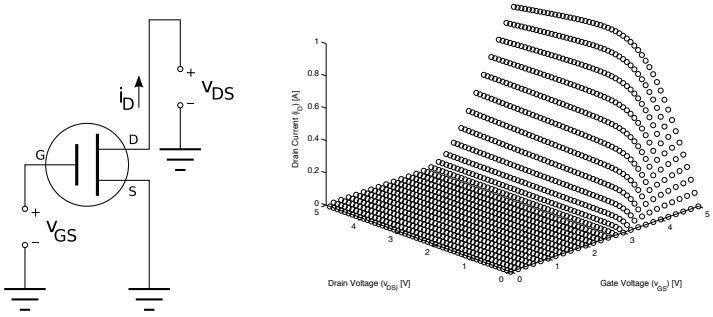
Figure 14: The input-output relationship of the regression networks of NNHMM and symbolic expressions of MMSR (black) overlaid on the Continuous Hysteresis Loop data (grey).

near perfect accuracy with ten or more hidden nodes per network, but failed when provided with only five hidden nodes per network. In terms of model fidelity, MMSR was able to achieve perfect expressions with respect to all the noise conditions.

Continuous Hysteresis Loop—This data set was ideal as it was sufficiently difficult to model, but simple enough to analyze and provide insight into how the algorithms perform on hybrid dynamical systems. The closed-loop MMSR was able to significantly outperform NNHMM and RNN under all noise conditions, but the open-loop MMSR fared worse than the parametric baselines in the presence of noise. This result was particularly interesting, since perfect model fidelity was achieved for all noise conditions. The predictive error in the open-loop MMSR occurred as a result of the continuous transition condition—under noisy conditions, the model can fail to predict a transition even with a correct model. As a result, a missed transition accumulates significant error for open-loop models. A closed-loop model is able to account for missed transitions, resulting in consistently accurate models.

Next, NNHMM outputs were analyzed to understand the discrepancy in predictive accuracy. Figure 14 shows the input-output relationships of NNHMM's best performing model, NNHMM's model that obtains the greatest separation and MMSR's best performing model, respectively. NNHMM had significant difficulties breaking the symmetry in the data set as the best model captured only the symmetry, while the locally-optimal asymmetrical model was both inferior in predictive accuracy and was significantly far from the ground truth. In comparison, MMSR was able to deal with the symmetrical data and infer unique representations. Such analysis could not be applied to RNNs as it is impossible to decouple the input-output relationships from the model transition components.

Phototaxic Robot—The phototaxic robot provided a challenging problem with an increased number of modes and asymptotic behaviors. Also, the distribution of the data was non-uniform and deceptive as it was sparse around the non-linear features. However, MMSR was able to achieve perfect model fidelity for low noise systems, which slowly degraded with respect to noise. Compared to the neural network approaches, both the open-loop and closed-loop produced significantly more accurate predictions under every noise condition. Note the simple switching behavior resulted in open-loop model accuracy that is comparable to the closed-loop counterpart, suggesting that closed-loop models are not necessary for simple switching systems.



(a) Circuit Diagram (b) Measured Data

Figure 15: A circuit diagram indicating the two input voltages, v_{GS} and v_{DS}, and the output current i_D, and the measured 3D data plot from the ZVNL4206AV nMOSFET.

This data set provides an example of how symbolic expressions aid in knowledge abstraction as it is easy to infer that the relative distance between the robot and the light position, $u_1 - u_2$, is an integral component of the system as it is a repeated motif in the each of the behaviors. It is significantly more difficult to extract the same information from parametric approaches like neural networks.

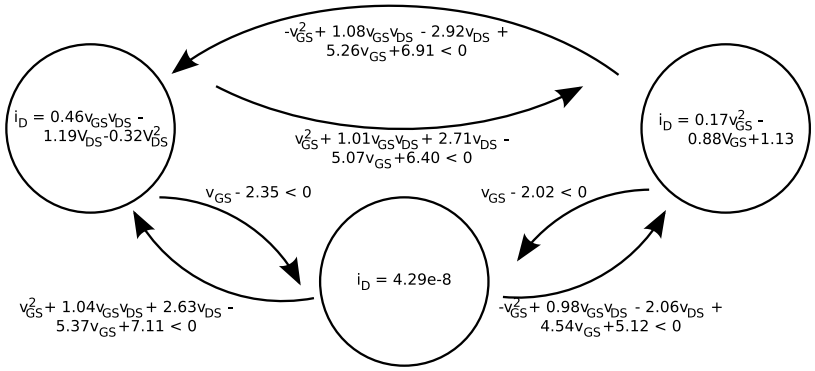
Non-linear System—The final data set provided a difficult modelling challenge that included non-linear behaviors which cannot be modeled by by parametric regression. Yet, MMSR reliably inferred the correct model for low noise systems and produced accurate predictions in all noise levels despite the noise sensitivity of model fidelity. The neural network approaches were significantly less accurate while using more free parameters.

4.3 Real Data Experiment

This section provides a case study of MMSR on real-world data while also exemplifying the benefits of symbolic model inference. This case study involves the inference of an n-channel metal-oxide semiconductor field-effect transistor (nMOSFET), a popular type of transistor ubiquitous in digital and analog circuits. nMOSFETs exhibit three distinct characteristics, which are governed by the physical layout and the underlying physics (Sedra and Smith, 2004), making them an ideal candidate for hybrid system analysis.

The transistor was placed in a standard configuration to measure the current-voltage characteristics, where the drain current is set as a function of the gate and drain voltages (Figure 15a). The transistor was a Diodes Inc. ZVNL4206AV nMOSFET and the data was recorded with a Keithley 2400 general purpose sourcemeter. The data was collected via random voltage sweeps from 0-5V, and the subsequent current was measured (Figure 15b).

The three discrete modes as well as the two-dimensional, non-linear input-output mapping makes this a non-trivial modelling problem. Furthermore, the regions are non-overlapping and continuous, which add another challenge in discerning the discrete modes. After applying MMSR with the setup described in Section 4.1, a hybrid dynamical system was inferred (Figure 16a). MMSR



(a) Inferred system diagram

$$i_D = \begin{cases} 4.29e\text{-}8 & , \quad \text{if } v_{GS} \leq 2.02 \\ 0.46\left((v_{GS}-2.59)v_{DS} - 0.71v_{DS}^2\right) & , \quad \text{if } v_{GS} > 2.68 \text{ and } (v_{GS} - 1.01v_{DS}) > 2.39 \\ 0.17(v_{GS}-2.76)(V_{GS}-2.40) & , \quad \text{if } v_{GS} > 2.11 \text{ and } (v_{GS} - 0.98v_{DS}) \leq 2.43 \end{cases}$$

(b) Inferred mode expressions

$$i_D = \begin{cases} 0 & , \quad \text{if } v_{GS} \leq k_1 \\ k_2\left((v_{GS}-k_1)v_{DS} - \frac{1}{2}v_{DS}^2\right) & , \quad \text{if } v_{GS} > k_1 \text{ and } (v_{GS} - v_{DS}) > k_1 \\ \frac{1}{2}k_2(v_{GS}-k_1)^2 & , \quad \text{if } v_{GS} > k_1 \text{ and } (v_{GS} - v_{DS}) \leq k_1 \end{cases}$$

(c) Classically derived mode expressions

Figure 16: The inferred hybrid model compared to the derived expressions.

was applied for ten independent runs and the median performing model was reported. As the transitions events were consistent between modes, which is indicative of the simple switching behavior exhibited by transistors, the system diagram was simplified to a piecewise representation with additional symbolic manipulations (Figure 16b).

When the inferred expressions are compared to classical equations (Sedra and Smith, 2004), the results are remarkably similar (Figure 16c). This suggests that MMSR is capable of inferring the ground truth of non-trivial systems from real-world data. While the model is sufficiently numerically accurate, the more impressive and relevant consequence is that MMSR was able to find the same expressions as engineer would derive from first principles, but inferred the results from unlabeled data. For an engineer or scientist presented with an unknown device with multi-modal behavior, beginning with apt, mathematical descriptions of a system might provide essential insight and understanding to determining the governing principles of that system. This capability provides an important advantage over traditional parametric machine learning models.

4.4 Scalability

Given that extracting dynamical equations from experimental data is an NP hard problem (Cubitt et al., 2012), determining the optimal model for hybrid dynamical systems is intractable. While evolutionary computational approaches are heuristic, exploratory methods that unable to guarantee optimality of a candidate model, in practice, they often find good and meaningful solutions. Rather

than a traditional lower-bound analysis, analyzing the computational complexity is used to provide insight to the scope of problems that are well suited for MMSR inference.

To assess the performance scalability of MMSR, the computational complexity of SR must first be analyzed as it is the primary computational kernel. As convergence on the global solution is not guaranteed, in the worst-case analysis, the complete search space is exhausted in a stochastic manner. For b building blocks and a tree depth size of c nodes, the search space grows exponentially with a complexity of $O(b^c)$. However, on average, SR performs significantly better than the worst case, although the performance is highly case dependent. Furthermore, evolutionary algorithms are naturally parallel, providing scalability with respect to the number of processors.

For the MMSR learning algorithm, two components are analyzed independently. With the worst-case SR complexity $O(b^c)$ and k modes, CSR has a compounded linear complexity with respect to the number of modes, $O(kb^c)$, while TM has a quadratic complexity of $O(k^2b^c)$, since transitions for every combination of modes must be considered. In terms of worst-case computational effort, this suggests that this algorithm would scale better for systems with numerous simple modes than it would for systems with fewer modes of higher complexity. For the data sets described in this section, the algorithm required an average of 10 and 45 minutes for the bi- and tri-modal systems, respectively, on a single core of a 2.8GHz Intel processor.

5. Discussion and Future Work

A novel algorithm, multi-modal symbolic regression (MMSR), was presented to infer non-linear, symbolic models of hybrid dynamical systems. MMSR is composed of two general subalgorithms. The first subalgorithm is clustered symbolic regression (CSR), designed to construct expressions for piecewise functions of unlabeled data. By combining symbolic regression (SR) with expectation-maximization (EM), CSR is able to separate the data into distinct clusters, and then subsequently find mathematical expressions for each subfunction. CSR exploits the Pareto front of SR to consistently avoid locally optimal solutions, a common challenge in EM mixture models. The second subalgorithm is transition modeling (TM), which searches for binary classification boundaries and expresses them as a symbolic inequality. TM uniquely capitalizes on the pre-existing SR infrastructure through function composition. These two subalgorithms are combined and used to infer symbolic models of hybrid dynamical systems.

MMSR is applied to four synthetic data sets, which span a range of classical hybrid automata and intelligent robotics. The training data was also corrupted with various levels of noise. The inferred models were compared via three performance metrics: model accuracy, complexity, and fidelity. MMSR inferred reliable models for noiseless data sets and outperformed its neural network counterparts in both model accuracy as well as model complexity. Furthermore, MMSR was used to identify and characterize field-effect transistor modes, similar to those derived from first principles, demonstrating a possible real-world application unique to this algorithm.

Symbolic modelling provides numerous benefits over parametric numerical models with the primary advantage of operating in symbolic expressions, the standard language of mathematics and science. Symbolic modelling provides the potential for knowledge abstraction and deeper understanding, as compared to the alternative of numeric,parametric approaches. In addition, there is a wealth of theory in symbolic mathematics, including approximation and equivalence theories such as Taylor expansions, which may aid understanding inferred models. Even having symbolic expres-

sions to identify reoccurring motifs and subexpressions may provide insight in the inner workings of the system.

A primary concern for symbolic modeling is how well it extends as the complexity increases and whether an easily interpretable model exists. However, the alternatives struggle equally in such cases. Deriving models from first principles is often similarly challenging while parametric approaches, such as RNN and NNHMM, are likely to settle on local optima and have difficulty achieve even numerically accurate models, even for relatively simple hybrid dynamical systems.

This work is the first step towards the generalized problem of modeling complex, multi-modal dynamical systems. While symbolic expressions may not exist for complex systems, it does present a viable alternative approach that may have the additional benefit of insight and interpretability. Future work includes extending the model to infer differential equations and investigating higher dimensional systems.

Acknowledgments

This research is supported in part by the U.S. National Institute of Health (NIH) National Institute of Arthritis and Musculoskeletal and Skin Diseases (NIAMS) grant AR-R01-052345, NIH National Institute of Drug Abuse (NIDA) grant RC2 DA028981 and Defense Threat Reduction Agency (DTRA) grant HDTRA1-09-0013. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NIH or DTRA. D.L. Ly would also like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for their support through the PGS program. We also acknowledge Michael D. Schmidt, J. Aaron Lenfestey, Hadas Kress-Gazit, Robert MacCurdy and Jonathan Shu for their insightful discussions and feedback.

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

A. P. A. Anderson. A hybrid mathematical model of solid tumour invasion: the importance of cell adhesion. *Mathematical Medicine and Biology*, 22:163–186, 2005.

A. Arslan and M. Kaya. Determination of fuzzy logic membership functions using genetic algorithms. *Fuzzy Sets and Systems*, 118:297–306, 2001.

Y. Bengio and P. Frasconi. An input-output HMM architecture. *Advances in Neural Information Processing Systems*, 7:427–434, 1994.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

M. S. Branicky. *Handbook of Networked and Embedded Control Systems*, chapter "Introduction to hybrid systems", pages 91–116. Birkhauser, 2005.

L. Breiman. Statistical modeling: the two cultures. *Statistical Science*, 16(3):199–231, 2001.

S. Chen, S.A. Billings, and B.M Grant. Recursive hybrid algorithm for non-linear system identification using radial basis function networks. *International Journal of Control*, 55:1051–1070, 1992.

N.L. Cramer. A representation for the adaptive generation of simple sequential programs. *International Conference on Genetic Algorithms*, pages 183–187, 1985.

T.S. Cubitt, J. Eisert, and M.M. Wolf. Extracting dynamic equations from experimental data is NP hard. *Physical Review Letters*, 108, 2012.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, 1977.

D. Dickmanns, J. Schmidhuber, and A. Winklhofer. Der genetische algorithmus: eine implementierung in prolog. Fortgeschrittenenpraktikum, Institut für Informatik, Technische Universität Mänchen, 1987.

G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39:205–217, 2003.

A. M. Gonzalez, A. M. S. Roque, and J. Garcia-Gonzalez. Modeling and forecasting electricity prices with input/output hidden Markov models. *IEEE Transactions on Power Systems*, 20(1): 13–24, 2005.

T. A. Henzinger. The theory of hybrid automata. *Symposium on Logic in Computer Science*, pages 324–335, 1996.

B. G. Horne and D. R. Hush. Bounds on the complexity of recurrent neural network implementations of finite state machines. *Neural Networks*, 9(2):243–252, 1996.

H. Iba. Inference of differential equation models by genetic programming. *Information Sciences*, 178:4453–4468, 2008.

R. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

H. Jacobsson. Rule extraction from recurrent neural networks: a taxonomy and review. *Neural Computation*, 17(6):1223–1263, 2006.

I. Jagielska, C. Matthews, and T. Whitfort. An investigation into the application of neural networks, fuzzy logic, genetic algorithms and rough sets to automated knowledge acquisition for classification problems. *Neurocomputing*, 24:37–54, 1999.

E. B. Kosmatopouous, M. M. Polycarpou, M. A. Christodoulou, and P. A. Ioannou. High-order neural network structures for identification of dynamical systems. *IEEE Transactions on Neural Networks*, 9(2):422–431, 1995.

J. R. Koza. *Genetic Programming: On The Programming of Computers by Means of Natural Selection.* MIT Press, 1992.

V.L. Le, G. Bloch, and F. Lauer. Reduced-size kernel models for nonlinear hybrid system identification. *IEEE Transactions on Neural Networks*, 22(12):2398–2405, 2011.

J. Lunze. *Modelling, Analysis and Design of Hybrid Systems*, chapter "What is a hybrid system?", pages 3–14. Springer, 2002.

S. Marcel, O. Bernier, J. E. Viallet, and D. Collobert. Hand gesture recognition using input-output hidden Markov models. *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 456–461, 2000.

A.M. Martinez and J. Virtia. Learning mixture models using a genetic version of the EM algorithm. *Pattern Recognition Letters*, 21:759–769, 2000.

R. R. F. Mendes, F. B. Voznika, A. A. Freitas, and J. C. Nievola. Discovering fuzzy classification rules with genetic programming and co-evolution. *ECML Principles and Practice of Knowledge Discover in Databases*, 2168:314–325, 2001.

D. P. Muni, N. R. Pal, and J. Das. A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation*, 8(2):183–196, 2004.

J.R. Olsson. Inductive functional programming using incremental program transformation. *Artificial Intelligence*, 74(1):55–81, 1995.

C.W. Omlin and C.L. Giles. Extraction, insertion and refinement of symbolic rules in dynamically driven recurrent neural networks. *Connection Science*, 5(3-4):307–337, 1993.

C.W. Omlin and C.L. Giles. Constructing deterministic finite-state automata in recurrent neural networks. *Journal of the ACM*, 43(6):937–972, 1996a.

C.W. Omlin and C.L. Giles. Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, 9(1):41–52, 1996b.

S. Paoletti, A.L. Juloski, G. Ferrari-Trecate, and R. Vidal. Identification of hybrid systems: a tutorial. *European Journal of Control*, 13:242–260, 2007.

R. Pernkopf and D. Bouchaffra. Genetic-base EM algorithm for learning Gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1344–1348, 2005.

B. D. Reger, K. M. Fleming, V. Sanguineti, S. Alford, and F. A Mussa-Ivaldi. Connecting brains to robots: the development of a hybrid system for the study of learning in neural tissues. *International Conference on Artificial Life*, pages 263–272, 2000.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. PyBrain. *Journal of Machine Learning Research*, 11:743–746, 2010.

M. D. Schmidt and H. Lipson. Symbolic regression of implicit equations. *Genetic Programming Theory and Practice*, 7:73–85, 2009a.

M. D. Schmidt and H. Lipson. Distilling free-from natural laws from experimental data. *Science*, 324(5923):81–85, 2009b.

M. D. Schmidt and H. Lipson. Eureqa, 2012. `http://creativemachines.cornell.edu/eureqa`.

A. S. Sedra and K. C. Smith. *Microelectronic Circuits*. Oxford University Press, 2004.

R.J. Solomonoff. A formal theory of inductive inference I, II. *Information and Control*, 7:1–22, 224–254, 1964.

C. Tomlin, G. J. Pappas, and S. S. Sastry. Conflict resolution for air traffic management: a study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, 1998.

A. Vahed and C.W. Omlin. A machine learning method for extracting symbolic knowledge from recurrent neural networks. *Neural Computation*, 16:59–71, 2004.

A. van der Schaft and H. Schumacher. *An Introduction To Hybrid Dynamical Systems*. Springer, 2000.

R. Vidal, S. Soatto, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. *Conference on Decision and Control*, pages 167–172, 2003.

A. Visintin. *Differential Models of Hysteresis*. Springer, 1995.

E. Vladislavleva, G. Smits, and D. den Hertog. Order of nonlinearity as a complexity measure for models generated by symbolic regression via Pareto genetic programming. *IEEE Transactions on Evolutionary Computation*, 13(2):333–349, 2009.

C.S. Wallace and D.L. Dowe. Minimum message length and Kolmogorov complexity. *The Computer Journal*, 42(4):270–283, 1999.

Journal of Machine Learning Research 13 (2012) 3619-3622 Submitted 10/11; Published 12/12

SVDFeature: A Toolkit for Feature-based Collaborative Filtering

Tianqi Chen TQCHEN@APEX.SJTU.EDU.CN
Weinan Zhang WNZHANG@APEX.SJTU.EDU.CN
Qiuxia Lu LUQIUXIA@APEX.SJTU.EDU.CN
Kailong Chen CHENKL@APEX.SJTU.EDU.CN
Zhao Zheng ZHENGZHAO@APEX.SJTU.EDU.CN
Yong Yu YYU@APEX.SJTU.EDU.CN
Apex Data and Knowledge Management Lab
Shanghai Jiao Tong University
800 Dongchuan Road
Shanghai 200240 China

Editor: Mikio Braun

Abstract

In this paper we introduce SVDFeature, a machine learning toolkit for feature-based collaborative filtering. SVDFeature is designed to efficiently solve the feature-based matrix factorization. The feature-based setting allows us to build factorization models incorporating side information such as temporal dynamics, neighborhood relationship, and hierarchical information. The toolkit is capable of both rate prediction and collaborative ranking, and is carefully designed for efficient training on large-scale data set. Using this toolkit, we built solutions to win KDD Cup for two consecutive years.

Keywords: large-scale collaborative filtering, context-aware recommendation, ranking

1. Introduction

Recommender system, which recommends items based on users' interests, has become more and more popular in many real-world situations. Collaborative filtering (CF) techniques, as the main thrust behind recommender systems, have been developed for many years and keep to be a hot area in both academia and industry. In this paper, we focus on building collaborative filtering based recommendation toolkit which can effectively leverage the rich information of data collected and naturally scale up to very large data set.

Matrix factorization (MF) is one of the most popular CF methods, and variants of it have been proposed in specific settings. However, traditional approaches design specific models for each problem, demanding great efforts in engineering. Fortunately the majority of factorization models share many common patterns, which enables us to summarize them into a single model and implement a unified toolkit, called SVDFeature.[1] SVDFeature enables incorporating auxiliary information via feature engineering. This helps save the efforts of engineering for specific models and allows users to focus on model design. Meanwhile, SVDFeature is specifically designed to handle large data sets. With SVDFeature, we have won the two most recent KDD Cups. Moreover, it costs less than

1. Project page can be found at `http://svdfeature.apexlab.org`.

2GB memory to get an excellent performance of 22.16 on RMSE on Yahoo! Music data set with 200 million ratings records. This has manifested SVDFeature great modeling power and scalability.

2. Model of SVDFeature

There are three important factors in most CF problems: users' interests, items' properties and other factors that directly affect users' preferences over items. Various kinds of information can be used to model these factors. For example, users' browsing history over movie reviews may correlate with users' taste over movies; the information of the director and actors of a movie can be used to predict its properties; the rating history over similar movies directly affect whether a user will favor the current one. Our model summarizes the three factors as feature vectors (denoted by $\alpha \in \mathbb{R}^m, \beta \in \mathbb{R}^n, \gamma \in \mathbb{R}^s$) and predicts the preference score $\hat{y}$ as

$$\hat{y}(\alpha,\beta,\gamma) = \left(\sum_{j=1}^{s} \gamma_j b_j^{(g)} + \sum_{j=1}^{n} \alpha_j b_j^{(u)} + \sum_{j=1}^{m} \beta_j b_j^{(i)}\right) + \left(\sum_{j=1}^{n} \alpha_j \mathbf{p}_j\right)^T \left(\sum_{j=1}^{m} \beta_j \mathbf{q}_j\right).$$

Here the model parameter set is defined as $\Theta = \{b^{(g)}, b^{(u)}, b^{(i)}, \mathbf{p}, \mathbf{q}\}$. $\mathbf{p}_j \in \mathbb{R}^d$ and $\mathbf{q}_j \in \mathbb{R}^d$ are d dimensional latent factors associated with each feature. $b_j^{(u)}$, $b_j^{(i)}$ and $b_j^{(g)}$ are bias parameters that directly influence the prediction. We call α user feature, β item feature, and γ global feature. This model is described as part of solution to KDD Cup (Chen et al., 2011, 2012). We call it feature-based matrix factorization. Intuitively, we use a linear model to construct user and item latent factors from features. The parameters are trained efficiently by minimizing a loss function using stochastic gradient descent. The supported loss functions include square-loss, negative logistic log-likelihood loss and smoothed hinge loss.

Many state-of-the-art collaborative filtering algorithms can be implemented using SVDFeature. Let us suppose we want to recommend music tracks for users using the rating history, and we have known the albums of tracks and the timestamps of the ratings as auxiliary information. As we know, the album information can be used to better represent the content of tracks, and the temporal information can be used to detect the changes of item popularity. All these auxiliary information can help improve the performance of a recommender system. Taking them into consideration, we can represent a user u's preference towards track i at time t as follows

$$\hat{y}(u,i,t) = b_{i,bin(t)} + b_i + b_{p(i)} + b_u + \mathbf{p}_u^T(\mathbf{q}_i + \mathbf{q}_{al(i)})$$

where $al(i)$ is the album of a track i. We segment time into consecutive bins and define $bin(t)$ to map timestamp to corresponding bin index. We use time localized bias $b_{i,bin(t)}$ to model items' popularity change over time. To incorporate the taxonomical information, we introduce $\mathbf{q}_{al(i)}$ to make the prediction towards track i depend on the latent factor of corresponding album. To implement this model using SVDFeature, we can simply define the features as follows

$$\alpha_h = \begin{cases} 1 & h = u \\ 0 & h \neq u \end{cases}, \quad \beta_h = \begin{cases} 1 & h = i \text{ or } h = al(i) \\ 0 & otherwise \end{cases}, \quad \gamma_h = \begin{cases} 1 & h = \#bins \times i + bin(t) \\ 0 & otherwise \end{cases}.$$

The item feature β is defined to be indicator of whether the record is related to the track and its corresponding album. We also define a global feature γ_h to encode the temporally varying item bias $b_{i,bin(t)}$. We can further incorporate more information by defining more features.

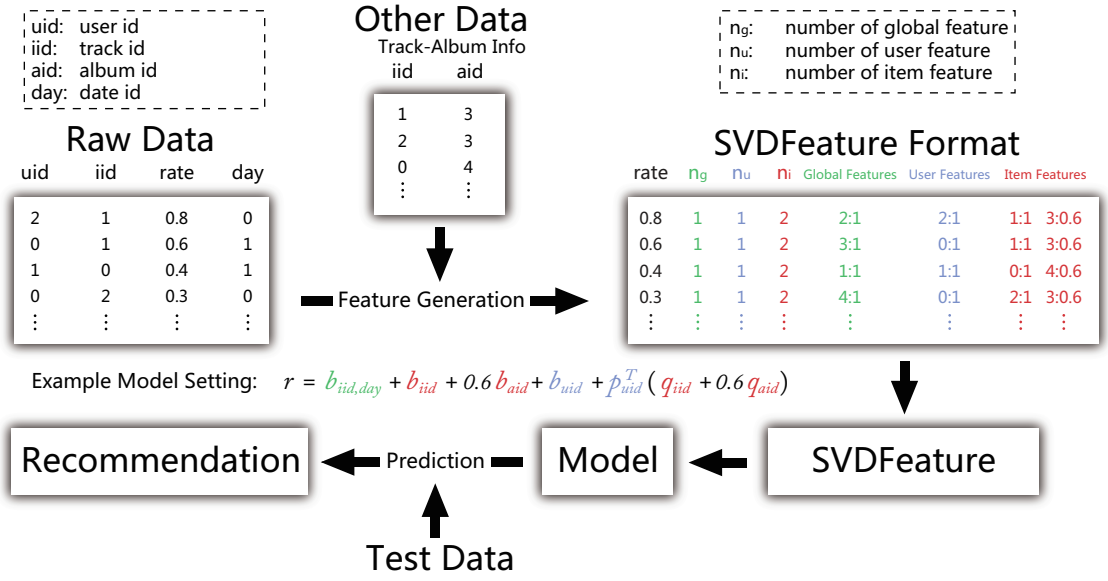


Figure 1: Usage flow example of SVDFeature

3. Using SVDFeature

Figure 1 gives a usage example of SVDFeature to implement the model introduced in previous section. We encode the album information as item features and item day biases as global features. The number 0.6 associated with album id is an empirical parameter chosen by the user to control the influence of album information in prediction. Assuming there are only two days' records in the data, the global feature index is defined as $gid = 2 \times iid + day$,where the number of item day biases is twice the number of items. SVDFeature will learn a feature-based matrix factorization model with the given training data and make predictions on supplied test feature files. We provide a manual to give more details about the usage of SVDFeature.

4. Handling Big Data

Recommendation algorithms often have to deal with problems with large scale data set in real world, which has been taken into consideration in designing SVDFeature. In our approach, we store the data into a buffer file in hard disk. The data is shuffled before storing, and then the training program linearly iterates over the buffer and updates the model with each training sample. This approach allows us to do training as long as the model fits into memory. To reduce the additional cost of I/O introduced by data loading, we use a pre-fetch strategy. An independent thread is created to fetch the data from hard disk into a memory queue. At the same time, the training thread reads the data from memory queue and updates the model. This pipeline style of execution releases the burden of I/O from training thread. As long as I/O speed is similar to (or faster than) training speed, the cost of I/O is negligible. Figure 2 shows the procedure of pipeline execution.

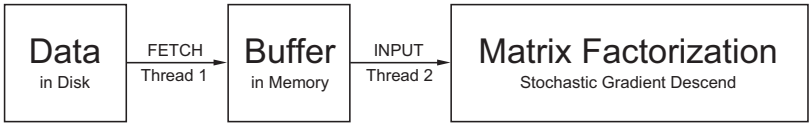


Figure 2: Pipeline training

5. Extra Features

SVDFeature also provides several other extra features for better modeling capability. We list notable features here: (1) Efficient speedup training for user feedback information; (2) Supporting collaborative ranking training; (3) Different kinds of regularization options; (4) Separated feature table extension for user and item features. The details are described in the project page and manual.

6. Availability and Documentation

The source code of SVDFeature is implemented in C++ and can be compiled under both Linux and Windows. The project is self-contained and only depends on standard libraries. SVDFeature is released under Apache License, Version 2.0. We provide a technical document introducing the algorithm and a user manual describing the usage details of the toolkit. To help users get started, we also provide a demo folder with example shell scripts that show the procedures from feature generation to training and prediction.

Acknowledgments

Yong Yu is supported by grants from NSFC-RGC joint research project 60931160445. We thank the reviewers and editor for their helpful comments. We greatly appreciate the discussions with Zhengdong Lu, Diyi Yang and Li'ang Yin.

References

T. Chen, Z. Zheng, Q. Lu, X. Jiang, Y. Chen, W. Zhang, K. Chen, Y. Yu, N. Liu, B. Cao, L. He, and Q. Yang. Informative ensemble of multi-resolution dynamic factorization models. In *KDD-Cup Workshop*, 2011.

T. Chen, L. Tang, Q. Liu, D. Yang, S. Xie, X. Cao, C. Wu, E. Yao, Z. Liu, Z. Jiang, C. Chen, W. Kong, and Y. Yu. Combining factorization model and additive forest for collaborative followee recommendation. In *KDD-Cup Workshop*, 2012.

Journal of Machine Learning Research 13 (2012) 3623-3680 Submitted 11/11; Revised 9/12; Published 12/12

Smoothing Multivariate Performance Measures

Xinhua Zhang[*] XINHUA2@CS.UALBERTA.CA
Department of Computing Science
University of Alberta
Alberta Innovates Center for Machine Learning
Edmonton, Alberta T6G 2E8, Canada

Ankan Saha ANKANS@CS.UCHICAGO.EDU
Department of Computer Science
University of Chicago
Chicago, IL 60637, USA

S.V. N. Vishwanathan VISHY@STAT.PURDUE.EDU
Departments of Statistics and Computer Science
Purdue University
West Lafayette, IN 47907-2066, USA

Editor: Sathiya Keerthi

Abstract

Optimizing multivariate performance measure is an important task in Machine Learning. Joachims (2005) introduced a Support Vector Method whose underlying optimization problem is commonly solved by cutting plane methods (CPMs) such as SVM-Perf and BMRM. It can be shown that CPMs converge to an ε accurate solution in $O\left(\frac{1}{\lambda\varepsilon}\right)$ iterations, where λ is the trade-off parameter between the regularizer and the loss function. Motivated by the impressive convergence rate of CPM on a number of practical problems, it was conjectured that these rates can be further improved. We disprove this conjecture in this paper by constructing counter examples. However, surprisingly, we further discover that these problems are not inherently hard, and we develop a novel smoothing strategy, which in conjunction with Nesterov's accelerated gradient method, can find an ε accurate solution in $O^*\left(\min\left\{\frac{1}{\varepsilon}, \frac{1}{\sqrt{\lambda\varepsilon}}\right\}\right)$ iterations. Computationally, our smoothing technique is also particularly advantageous for optimizing multivariate performance scores such as precision/recall break-even point and ROCArea; the cost per iteration remains the same as that of CPMs. Empirical evaluation on some of the largest publicly available data sets shows that our method converges significantly faster than CPMs without sacrificing generalization ability.

Keywords: non-smooth optimization, max-margin methods, multivariate performance measures, Support Vector Machines, smoothing

1. Introduction

Recently there has been an explosion of interest in applying machine learning techniques to a number of application domains, much of which has been fueled by the phenomenal success of binary Support Vector Machines (SVMs). At the heart of SVMs is the following regularized risk mini-

∗. Xinhua Zhang is now working at the Machine Learning Group of NICTA, Canberra, Australia.

mization problem:

$$\min_{\mathbf{w}} J(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + R_{\text{emp}}(\mathbf{w}). \tag{1}$$

Here R_{emp} is the so-called empirical risk (see below), $\frac{1}{2}\|\mathbf{w}\|^2$ is the regularizer, $\lambda > 0$ is a scalar which trades off the importance of the empirical risk and the regularizer. Given a training set with n examples $\mathcal{X} := \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{\pm 1\}$, the empirical risk R_{emp} minimized by SVMs is the average of the hinge loss:

$$R_{\text{emp}}(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^n \max(0, 1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle), \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean dot product.

Binary SVMs optimize classification accuracy, while many application areas such as Natural Language Processing (NLP) frequently use more involved multivariate performance measures such as precision/recall break-even point (PRBEP) and area under the Receiver Operating Characteristic curve (ROCArea). Joachims (2005) proposed an elegant SVM based formulation for *directly* optimizing these performance metrics. In his formulation, the empirical risk R_{emp} in (1) is replaced by

$$R_{\text{emp}}(\mathbf{w}) := \max_{\mathbf{z} \in \{-1,1\}^n} \left[\Delta(\mathbf{z}, \mathbf{y}) + \frac{1}{n}\sum_{i=1}^n \langle \mathbf{w}, \mathbf{x}_i \rangle (z_i - y_i) \right]. \tag{3}$$

Here, $\Delta(\mathbf{z}, \mathbf{y})$ is the multivariate discrepancy between the correct labels $\mathbf{y} := (y_1, \ldots, y_n)^\top$ and a candidate labeling $\mathbf{z} := (z_1, \ldots, z_n)^\top \in \{\pm 1\}^n$. In order to compute the multivariate discrepancy for PRBEP we need the false negative and false positive rates, which are defined as

$$b = \sum_{i \in \mathcal{P}} \delta(z_i = -1) \quad \text{and} \quad c = \sum_{j \in \mathcal{N}} \delta(z_j = 1), \text{ respectively.}$$

Here $\delta(x) = 1$ if x is true and 0 otherwise, while $\mathcal{P}$ and $\mathcal{N}$ denote the set of indices with positive ($y_i = +1$) and negative ($y_i = -1$) labels respectively. Furthermore, let $n_+ = |\mathcal{P}|$, $n_- = \left|\mathcal{N}\right|$. With this notation in place, $\Delta(\boldsymbol{z}, \boldsymbol{y})$ for PRBEP is defined as

$$\Delta(\boldsymbol{z}, \boldsymbol{y}) = \begin{cases} b/n_+ & \text{if } b = c \\ -\infty & \text{otherwise} \end{cases}. \tag{4}$$

ROCArea, on the other hand, measures how many pairs of examples are mis-ordered. Denote $m = n_+ n_-$. Joachims (2005) proposed using the following empirical risk, R_{emp}, to directly optimize the ROCArea:[1]

$$R_{\text{emp}}(\mathbf{w}) = \frac{1}{m} \max_{\mathbf{z} \in \{0,1\}^m} \left[\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} z_{ij} \left[1 - \mathbf{w}^\top (\mathbf{x}_i - \mathbf{x}_j) \right] \right]. \tag{5}$$

We will call the regularized risk minimization problem with the PRBEP loss (3) as the PRBEP-problem and with the ROCArea loss (5) as the ROCArea-problem respectively. As is obvious,

1. The original formulation uses $1 - 2\mathbf{w}^\top(\mathbf{x}_i - \mathbf{x}_j)$, and we rescaled it to simplify our presentation in the sequel.

both these problems entail minimizing a non-smooth objective function which is *not* additive over the data points. Fortunately, cutting plane methods (CPMs) such as SVM-Perf (Joachims, 2006) and BMRM (Teo et al., 2010) can be used for this task. At each iteration these algorithms only require a sub-gradient of R_{emp}, which can be efficiently computed by a *separation* algorithm with $O(n \log n)$ effort for both (3) and (5) (Joachims, 2005). In this paper, we will work with BMRM as our prototypical CPM, which, as Teo et al. (2010) point out, includes SVM-Perf as a special case.

CPMs are popular in Machine Learning because they come with strong convergence guarantees. The tightest upper bound on the convergence speed of BMRM, which we are aware of, is due to Teo et al. (2010). Theorem 5 of Teo et al. (2010) asserts that BMRM can find an ε-accurate solution of the regularized risk minimization problem (1) with non-smooth empirical risk functions such as (3) and (5) after computing $O(\frac{1}{\lambda\varepsilon})$ subgradients. This upper bound is built upon the following standard assumption:

Assumption 1 (A1) *The subgradient of R_{emp} is bounded, that is, at any point $\mathbf{w}$, there exists a subgradient $\boldsymbol{g} \in \partial R_{\text{emp}}(\mathbf{w})$ such that $\|\boldsymbol{g}\| \leq G < \infty$.*

In practice the observed rate of convergence of BMRM is significantly faster than predicted by theory. Therefore, it was conjectured that perhaps a more refined analysis might yield a tighter upper bound (Teo et al., 2010). Our first contribution in this paper is to disprove this conjecture under assumption **A1**. We carefully construct PRBEP and ROCArea problems on which BMRM requires at least $\Omega(1/\varepsilon)$ iterations to converge for a fixed λ. It is worthwhile emphasizing here that due to the specialized form of the objective function in (1), lower bounds for *general* convex function classes such as those studied by Nesterov (2003) and Nemirovski and Yudin (1983) do not apply. In addition, our examples stick to binary classification and are substantially different from the one in Joachims et al. (2009) which requires an increasing number of classes.

This result leads to the following natural question: do the lower bounds hold because the PRBEP and ROCArea problems are fundamentally hard, or is it an inherent limitation of CPM? In other words, does there exist a solver which solves the PRBEP and ROCArea problems by invoking the separation algorithm for fewer than $O(\frac{1}{\lambda\varepsilon})$ times?[2] We provide partial answers. To understand our results one needs to understand another standard assumption that is often made when proving convergence rates:

Assumption 2 (A2) *Each $\mathbf{x}_i$ lies inside an L_2 (Euclidean) ball of radius R, that is, $\|\mathbf{x}_i\| \leq R$.*

Clearly assumption **A2** is more restrictive than **A1**, because $\|\boldsymbol{x}_i\| \leq R$ implies that for all $\boldsymbol{w}$, any subgradient of R_{emp} has L_2 norm at most $2R$ in both (3) and (5). Our second contribution in this paper is to show that the $O(\frac{1}{\lambda\varepsilon})$ barrier can be broken under assumption **A2**, while making a similar claim just under assumption **A1** remains an open problem. In a nutshell, our algorithm approximates (1) by a smooth function, which in turn can be efficiently minimized by using either accelerated gradient descent (Nesterov, 1983, 2005, 2007) or a quasi-Newton method (Nocedal and Wright, 2006). This technique for non-smooth optimization was pioneered by Nesterov (2005). However, applying it to multivariate performance measures requires special care. We now describe some relevant details and point out technical difficulties that one encounters along the way.

2. We want the rates to be better in both λ *and* in ε.

1.1 Nesterov's Formulation

Let A be a linear transform and assume that we can find a smooth function $g^\star_\mu(A^\top \mathbf{w})$ with a Lipschitz continuous gradient such that $\left|R_{\text{emp}}(\mathbf{w}) - g^\star_\mu(A^\top \mathbf{w})\right| \leq \mu$ for all $\mathbf{w}$. It is easy to see that

$$J_\mu(\mathbf{w}) := \frac{\lambda}{2}\|\mathbf{w}\|^2 + g^\star_\mu(A^\top \mathbf{w}) \tag{6}$$

satisfies $\left|J_\mu(\mathbf{w}) - J(\mathbf{w})\right| \leq \mu$ for all $\mathbf{w}$. In particular, if we set $\mu = \varepsilon/2$ and find a $\mathbf{w}'$ such that $J_\mu(\mathbf{w}') \leq \min_{\mathbf{w}} J_\mu(\mathbf{w}) + \varepsilon/2$, then it follows that $J(\mathbf{w}') \leq \min_{\mathbf{w}} J(\mathbf{w}) + \varepsilon$. In other words, $\mathbf{w}'$ is an ε accurate solution for (1).

By applying Nesterov's accelerated gradient method to $J_\mu(\mathbf{w})$ one can find an ε-accurate solution of the original nonsmooth problem (1) after querying the gradient of $g^\star_\mu(A^\top \mathbf{w})$ at most

$$O^*\left(\sqrt{D}\,\|A\| \min\left\{\frac{1}{\varepsilon}, \frac{1}{\sqrt{\lambda\varepsilon}}\right\}\right) \tag{7}$$

number of times (Nesterov, 1983, 2005).[3] Here $\|A\|$ is the matrix norm of A, which is defined as $\|A\| = \max_{\boldsymbol{u},\boldsymbol{v}:\|\boldsymbol{u}\|=\|\boldsymbol{v}\|=1} \boldsymbol{u}^\top A \boldsymbol{v} = \sqrt{\lambda_{\max}(A^\top A)}$, with $\lambda_{\max}$ denoting the maximum eigenvalue of A, when the norm considered is the Euclidean norm. Furthermore, D is a geometric constant that depends solely on g_μ and is independent of ε, λ, or A. Compared with the $O(\frac{1}{\lambda\varepsilon})$ rates of CPMs, the $\frac{1}{\sqrt{\lambda\varepsilon}}$ part in (7) is already superior. Furthermore, many applications require $\lambda \ll \varepsilon$ and in this case the $\frac{1}{\varepsilon}$ part of the rate is even better. Note that CPMs rely on $\frac{\lambda}{2}\|\mathbf{w}\|^2$ to stabilize each update, and it has been empirically observed that they converge slowly when λ is small (see, e.g., Do et al., 2009).

Although the above scheme is conceptually simple, the smoothing of the objective function in (1) has to be performed very carefully in order to avoid dependence on n, the size of the training set. The main difficulties are two-fold. First, one needs to obtain a smooth approximation $g^\star_\mu(A^\top \mathbf{w})$ to $R_{\text{emp}}(\mathbf{w})$ such that $\sqrt{D}\,\|A\|$ is small (ideally a constant). Second, we need to show that computing the gradient of $g^\star_\mu(A^\top \mathbf{w})$ is no harder than computing a subgradient of $R_{\text{emp}}(\mathbf{w})$. In the sequel we will demonstrate how both the above difficulties can be overcome. Before describing our scheme in detail we would like to place our work in context by discussing some relevant related work.

1.2 Related Work

Training large SV models by using variants of stochastic gradient descent has recently become increasingly popular (Bottou, 2008; Shalev-Shwartz et al., 2007). While the dependence on ε and λ is still $\Omega(\frac{1}{\lambda\varepsilon})$ or worse (Agarwal et al., 2009), one gets bounds on the computational cost independent of n. However, stochastic gradient descent can only be applied when the empirical risk is *additively decomposable*, that is, it can be written as the average loss over individual data points like in (2). Since the non-linear multivariate scores such as the ones that we consider in this paper are not additively decomposable, this rules out the application of online algorithms to these problems.

Traditionally, batch optimizers such as the popular Sequential Minimal Optimization (SMO) worked in the dual (Platt, 1998). However, as List and Simon (2009) show, for the empirical risk (2) based on hinge loss, if the matrix which contains the dot products of all training instances

3. For completeness we reproduce relevant technical details from Nesterov (1983) and Nesterov (2005) in Appendix A.

(also known as the kernel matrix) is not strictly positive definite, then SMO requires $O(n/\varepsilon)$ iterations with each iteration costing $O(np)$ effort. However, when the kernel matrix is strictly positive definite, then one can obtain an $O(n^2 \log(1/\varepsilon))$ bound on the number of iterations, which has better dependence on ε, but is prohibitively expensive for large n. Even better dependence on ε can be achieved by using interior point methods (Ferris and Munson, 2002) which require only $O(\log(\log(1/\varepsilon))$ iterations, but the time complexity per iteration is $O(\min\{n^2 p, p^2 n\})$.

Recently, there has been significant research interest in optimizers which directly optimize (1) because there are some distinct advantages (Teo et al., 2010). Chapelle (2007) observed that to find a $\mathbf{w}$ which generalizes well, one only needs to solve the primal problem to very low accuracy (e.g., $\varepsilon \approx 0.001$). In fact, to the best of our knowledge, Chapelle (2007) introduced the idea of smoothing the objective function to the Machine Learning community. Specifically, he proposed to approximate the binary hinge loss by a smooth Huber's loss and used the Newton's method to solve this smoothed problem. This approach yielded the best overall performance in the Wild Competition Track of Sonnenburg et al. (2008) for training binary linear SVMs on large data sets. A similar smoothing approach is proposed by Zhou et al. (2010), but it is also only for hinge loss.

However, the smoothing proposed by Chapelle (2007) for the binary hinge loss is rather ad-hoc, and does not easily generalize to (3) and (5). Moreover, a function can be smoothed in many different ways and Chapelle (2007) did not explicitly relate the influence of smoothing on the rates of convergence of the solver. In contrast, we propose principled approaches to overcome these problems.

Of course, other smoothing techniques have also been explored in the literature. A popular approach is to replace the nonsmooth max term by a smooth log-sum-exp approximation (Boyd and Vandenberghe, 2004). In the case of binary classification this approximation is closely related to logistic regression (Bartlett et al., 2006; Zhang, 2004), and is equivalent to using an entropy regularizer in the dual. However, as we discuss in Section 3.1.2 this technique yields exorbitantly large $\sqrt{D}\,\|A\|$ in the bound (7) when applied to optimize multivariate performance measures.

1.3 Notation And Paper Outline

We assume a standard setup as in Nesterov (2005). Lower bold case letters (e.g., $\mathbf{w}$, $\boldsymbol{\mu}$) denote vectors, w_i denotes the i-th component of $\mathbf{w}$, $\mathbf{0}$ refers to the vector with all zero components, $\boldsymbol{e}_i$ is the i-th coordinate vector (all 0's except 1 at the i-th coordinate) and Δ_k refers to the k dimensional simplex. Unless specified otherwise, $\|\cdot\|$ refers to the Euclidean norm $\|\mathbf{w}\| := \sqrt{\langle \boldsymbol{w}, \boldsymbol{w} \rangle}$. We denote $\overline{\mathbb{R}} := \mathbb{R} \cup \{\infty\}$, and $[t] := \{1, \ldots, t\}$. $g^\star$ stands for the Fenchel dual of a function g, and $\partial g(\boldsymbol{w})$ is the subdifferential of g at $\boldsymbol{w}$ (the set of all subgradients of g at $\boldsymbol{w}$).

In Section 2, we first establish the lower bounds on the rates of convergence of CPMs when used to optimize the regularized risk of PRBEP and ROCArea. To break this lower bound, our proposed strategy constructs a smoothed surrogate function $g_\mu^\star(A^\top \mathbf{w})$, and the details are given in Section 3. We will focus on efficiently computing the gradient of the smooth objective function in Section 4. Empirical evaluation is presented in Section 5, and the paper concludes with a discussion in Section 6.

1.4 Improvement On Earlier Versions Of This Paper

This paper is based on two prior publications of the same authors: Zhang et al. (2011a) at NIPS 2010 and Zhang et al. (2011b) at UAI 2011, together with a technical report (Zhang et al., 2010). Major improvements have been made on them, which include:

- New lower bounds for PRBEP and ROCArea losses (see Sections 2.2 and 2.3 respectively).
- Significantly simplified algorithms for computing the gradient of smoothed objectives for both PRBEP and ROCArea losses (See Sections 4.1 and 4.2 respectively).
- Empirical evaluation on some of the largest publicly available data sets including those from the Pascal Large Scale Learning Workshop in Section 5 (Sonnenburg et al., 2008)
- Open source code based on PETSc (Balay et al., 2011) and TAO (Benson et al., 2010) packages made available for download from Zhang et al. (2012).

2. Lower Bounds For Cutting Plane Methods

BMRM (Teo et al., 2010) is a state-of-the-art cutting plane method for optimizing multivariate performance scores which directly minimizes the non-smooth objective function $J(\boldsymbol{w})$. At every iteration, BMRM replaces R_{emp} by a piecewise linear lower bound R_k^{cp} and obtains the next iterate $\mathbf{w}_k$ by optimizing

$$\mathbf{w}_k := \operatorname*{argmin}_{\boldsymbol{w}} J_k(\mathbf{w}) := \frac{\lambda}{2}\|\mathbf{w}\|^2 + R_k^{\text{cp}}(\mathbf{w}), \quad \text{where } R_k^{\text{cp}}(\mathbf{w}) := \max_{i\in[k]} \langle \mathbf{w}, \mathbf{a}_i \rangle + b_i. \tag{8}$$

Here $\mathbf{a}_i \in \partial R_{\text{emp}}(\mathbf{w}_{i-1})$ denotes an *arbitrary* subgradient of R_{emp} at $\mathbf{w}_{i-1}$ and $b_i = R_{\text{emp}}(\mathbf{w}_{i-1}) - \langle \mathbf{w}_{i-1}, \mathbf{a}_i \rangle$. The piecewise linear lower bound is successively tightened until the gap

$$\varepsilon_k := \min_{0\le t\le k} J(\mathbf{w}_t) - J_k(\mathbf{w}_k) \tag{9}$$

falls below a predefined tolerance ε.

Since J_k in (8) is a convex objective function, the optimal $\boldsymbol{w}_k$ can be obtained by solving its dual problem, which is a quadratic programming problem with simplex constraints. We refer the reader to Teo et al. (2010) for more details. The overall procedure of BMRM is summarized in Algorithm 1. Under Assumption **A1**, Teo et al. (2010) showed that BMRM converges at $O(\frac{1}{\lambda\varepsilon})$ rates:

Theorem 3 *Suppose Assumption 1 holds. Then for any* $\varepsilon < 4G^2/\lambda$, BMRM *converges to an* ε *accurate solution of* (1) *as measured by* (9) *after at most the following number of steps:*

$$\log_2 \frac{\lambda J(\mathbf{0})}{G^2} + \frac{8G^2}{\lambda\varepsilon} - 1.$$

2.1 Lower Bound Preliminaries

Since most rates of convergence found in the literature are upper bounds, it is important to rigorously define the meaning of a lower bound with respect to ε, and to study its relationship with the upper bounds. At this juncture it is also important to clarify an important technical point. Instead of

Algorithm 1: BMRM.

Input: A tolerance level of function value ε.

Initialize: $\boldsymbol{w}_0$ to arbitrary value, for example, $\mathbf{0}$. Let $k = 1$.

while TRUE do

  Pick arbitrary subgradient of R_{emp} at $\mathbf{w}_{k-1}$: $\mathbf{a}_k \in \partial R_{\text{emp}}(\mathbf{w}_{k-1})$.

  Let $b_k = R_{\text{emp}}(\mathbf{w}_{k-1}) - \langle \mathbf{w}_{k-1}, \mathbf{a}_k \rangle$.

  Solve $\mathbf{w}_k = \operatorname{argmin}_{\mathbf{w}} J_k(\mathbf{w}) = \operatorname{argmin}_{\mathbf{w}} \left\{ \frac{\lambda}{2}\|\mathbf{w}\|^2 + \max_{i\in[k]} \{\langle \mathbf{w}, \mathbf{a}_i \rangle + b_i\} \right\}$.

  if $\min_{0\le t\le k} J(\mathbf{w}_t) - J_k(\mathbf{w}_k) < \varepsilon$ **then return** $\boldsymbol{w}_k$. **else** $k = k+1$.

minimizing the objective function $J(\mathbf{w})$ defined in (1), if we minimize a scaled version $cJ(\mathbf{w})$ this scales the approximation gap (9) by c. Assumption **A1** fixes this degree of freedom by bounding the scale of the objective function.

Given a function $f \in \mathcal{F}$ and an optimization algorithm A, suppose $\{\mathbf{w}_k\}$ are the iterates produced by some algorithm A when minimizing some function f. Define $T(\varepsilon; f, A)$ as the first step index k when $\mathbf{w}_k$ becomes an ε accurate solution:[4]

$$T(\varepsilon; f, A) = \min \{k : f(\mathbf{w}_k) - \min_{\boldsymbol{w}} f(\boldsymbol{w}) \le \varepsilon\}.$$

Upper and lower bounds are both properties for a pair of $\mathcal{F}$ and A. A function $\overline{\kappa}(\varepsilon)$ is called *an* upper bound of $(\mathcal{F}, A)$ if for all functions $f \in \mathcal{F}$ and all $\varepsilon > 0$, it takes *at most* order $\overline{\kappa}(\varepsilon)$ steps for A to reduce the gap to less than ε, that is,

$$\textbf{(UB)} \qquad \forall\, \varepsilon > 0, \forall\, f \in \mathcal{F},\ T(\varepsilon; f, A) \le \overline{\kappa}(\varepsilon).$$

On the other hand, we define lower bounds as follows. $\underline{\kappa}(\varepsilon)$ is called *a* lower bound of $(\mathcal{F}, A)$ if for any $\varepsilon > 0$, there exists a function $f_\varepsilon \in \mathcal{F}$ depending on ε, such that it takes *at least* $\underline{\kappa}(\varepsilon)$ steps for A to find an ε accurate solution of f_ε:

$$\textbf{(LB)} \qquad \forall\, \varepsilon > 0, \exists\, f_\varepsilon \in \mathcal{F},\ s.t.\ T(\varepsilon; f_\varepsilon, A) \ge \underline{\kappa}(\varepsilon).$$

Clearly, this lower bound is sufficient to refute upper bounds or to establish their tightness. The size of the function class $\mathcal{F}$ affects the upper and lower bounds in opposite ways. Suppose $\mathcal{F}' \subset \mathcal{F}$. Proving upper (resp. lower) bounds on $(\mathcal{F}', A)$ is usually easier (resp. harder) than proving upper (resp. lower) bounds for $(\mathcal{F}, A)$. Since the lower bounds studied by Nesterov (2003) and Nemirovski and Yudin (1983) are constructed by using *general* convex functions, they do not apply here to our specialized regularized risk objectives.

We are interested in bounding the *primal gap* of the iterates $\mathbf{w}_k$: $J(\mathbf{w}_k) - \min_{\boldsymbol{w}} J(\boldsymbol{w})$. Data sets will be constructed explicitly whose resulting objective $J(\mathbf{w})$ will be shown to satisfy Assumption **A1** and attain the lower bound of BMRM. We will focus on the R_{emp} for both the PRBEP loss in (3) and the ROCArea loss in (5).

4. The initial point also matters, as in the best case one can just start from the optimal solution. Thus the quantity of interest is actually $T(\varepsilon; f, A) := \max_{\mathbf{w}_0} \min\{k : f(\mathbf{w}_k) - \min_{\boldsymbol{w}} f(\boldsymbol{w}) \le \varepsilon$, starting point being $\mathbf{w}_0\}$. However, without loss of generality we assume some pre-specified way of initialization.

2.2 Construction Of Lower Bound For PRBEP Problem

Given $\varepsilon > 0$, let $n = \lceil 1/\varepsilon \rceil + 1$ and construct a data set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ as follows. Set $c = \frac{\sqrt{2}}{4}$, and let $\boldsymbol{e}_i$ denote the i-th coordinate vector in $\mathbb{R}^{n+2}$ ($p = n+2$). Let the first example be the only positive example: $y_1 = +1$ and $\boldsymbol{x}_1 = cn\boldsymbol{e}_2$. The rest $n-1$ examples are all negative: $y_i = -1$ and $\boldsymbol{x}_i = -cn\boldsymbol{e}_3 - cn\boldsymbol{e}_{i+2}$ for $i \in [2,n]$. This data set concretizes the R_{emp} in (3) as follows:

$$R_{\text{emp}}(\boldsymbol{w}) = \max_{\boldsymbol{z} \in \{-1,1\}^n} \underbrace{\left[\Delta(\boldsymbol{z}, \boldsymbol{y}) + \frac{1}{n} \langle \boldsymbol{w}, \boldsymbol{x}_1 \rangle (z_1 - 1) + \frac{1}{n} \sum_{i=2}^{n} \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle (z_i + 1) \right]}_{:= \Upsilon(\boldsymbol{z})}.$$

Setting $\lambda = 1$, we obtain the overall objective $J(\boldsymbol{w})$ in (1). Now we have the following lower bound on the rates of convergence of BMRM.

Theorem 4 *Let $\boldsymbol{w}_0 = \boldsymbol{e}_1 + 2c\boldsymbol{e}_2 + 2c\boldsymbol{e}_3$. Suppose running* BMRM *on the above constructed objective $J(\boldsymbol{w})$ produces iterates $\mathbf{w}_1, \ldots, \mathbf{w}_k, \ldots$. Then it takes* BMRM *at least $\left\lfloor \frac{1}{5\varepsilon} \right\rfloor$ steps to find an ε accurate solution. Formally,*

$$\min_{i \in [k]} J(\mathbf{w}_i) - \min_{\boldsymbol{w}} J(\boldsymbol{w}) \geq \frac{1}{4} \left(\frac{1}{k} - \frac{1}{n-1} \right) \text{ for all } k \in [n-1],$$

$$\text{hence} \quad \min_{i \in [k]} J(\mathbf{w}_i) - \min_{\boldsymbol{w}} J(\boldsymbol{w}) > \varepsilon \text{ for all } k < \frac{1}{5\varepsilon}.$$

Proof The crux of our proof is to show

$$\boldsymbol{w}_k = 2c\boldsymbol{e}_2 + 2c\boldsymbol{e}_3 + \frac{2c}{k} \sum_{i=4}^{k+3} \boldsymbol{e}_i, \quad \text{for all } k \in [n-1]. \tag{10}$$

We prove it by induction. Initially at $k = 1$, we note

$$\frac{1}{n} \langle \boldsymbol{w}_0, \boldsymbol{x}_1 \rangle = 2c^2 = \frac{1}{4}, \quad \text{and} \quad \frac{1}{n} \langle \boldsymbol{w}_0, \boldsymbol{x}_i \rangle = -2c^2 = -\frac{1}{4}, \quad \forall\, i \in [2,n]. \tag{11}$$

Since there is only one positive example, $\boldsymbol{z}$ can only have two cases. If $\boldsymbol{z} = \boldsymbol{y}$ (i.e., correct labeling with $b = 0$), then $\Upsilon(\boldsymbol{z}) = \Upsilon(\boldsymbol{y}) = 0$. If $\boldsymbol{z}$ misclassifies the only positive example into negative (i.e., $b = 1$), then $\Delta(\boldsymbol{z}, \boldsymbol{y}) = \frac{b}{n_+} = 1$. Since PRBEP forces $c = b = 1$, there must be one negative example misclassified into positive by $\boldsymbol{z}$. By (11), all $i \in [2,n]$ play the same role in $\Upsilon(\boldsymbol{z})$. Without loss of generality, assume the misclassified negative example is $i = 2$, that is, $\boldsymbol{z} = (-1, 1, -1, -1, \ldots)$. It is easy to check that $\Upsilon(\boldsymbol{z}) = 1 + \frac{2}{n}[-\langle \boldsymbol{w}_0, \boldsymbol{x}_1 \rangle + \langle \boldsymbol{w}_0, \boldsymbol{x}_2 \rangle] = 0 = \Upsilon(\boldsymbol{y})$, which means such $\boldsymbol{z}$ is a maximizer of $\Upsilon(\boldsymbol{z})$. Using this $\boldsymbol{z}$, we can derive a subgradient of $R_{\text{emp}}(\boldsymbol{w}_0)$:

$$\boldsymbol{a}_1 = \frac{2}{n}(-\boldsymbol{x}_1 + \boldsymbol{x}_2) = -2c\boldsymbol{e}_2 - 2c\boldsymbol{e}_3 - 2c\boldsymbol{e}_4 \text{ and } b_1 = 0 - \langle \boldsymbol{a}_1, \boldsymbol{w}_0 \rangle = 4c^2 + 4c^2 = 1,$$

$$\boldsymbol{w}_1 = \operatorname*{argmin}_{\boldsymbol{w}} \left\{ \frac{1}{2} \|\boldsymbol{w}\|^2 + \langle \boldsymbol{a}_1, \boldsymbol{w} \rangle + b_1 \right\} = 2c\boldsymbol{e}_2 + 2c\boldsymbol{e}_3 + 2c\boldsymbol{e}_4.$$

Next assume (10) holds for steps $1, \ldots, k$ ($k \in [n-2]$). Then at step $k+1$ we have

$$\frac{1}{n} \langle \boldsymbol{w}_k, \boldsymbol{x}_1 \rangle = 2c^2 = \frac{1}{4}, \quad \text{and} \quad \frac{1}{n} \langle \boldsymbol{w}_k, \boldsymbol{x}_i \rangle = \begin{cases} -2c^2 - \frac{2c^2}{k} = -\frac{1}{4} - \frac{1}{4k} & \forall\, i \in [2, k+1] \\ -2c^2 = -\frac{1}{4}, & \forall\, i \geq k+2 \end{cases}.$$

Again consider two cases of $\boldsymbol{z}$. If $\boldsymbol{z} = \boldsymbol{y}$ then $\Upsilon(\boldsymbol{z}) = \Upsilon(\boldsymbol{y}) = 0$. If $\boldsymbol{z}$ misclassifies the only positive example into negative, then $\Delta(\boldsymbol{z}, \boldsymbol{y}) = 1$. If $\boldsymbol{z}$ misclassifies any of $i \in [2, k+1]$ into positive, then $\Upsilon(\boldsymbol{z}) = 1 + \frac{2}{n}[-\langle \boldsymbol{w}_k, \boldsymbol{x}_1\rangle + \langle \boldsymbol{w}_k, \boldsymbol{x}_i\rangle] = -\frac{1}{2k} < \Upsilon(\boldsymbol{y})$. So such $\boldsymbol{z}$ cannot be a maximizer of $\Upsilon(\boldsymbol{z})$. But if $\boldsymbol{z}$ misclassifies any of $i \geq k+2$ into positive, then $\Upsilon(\boldsymbol{z}) = 1 + \frac{2}{n}[-\langle \boldsymbol{w}_k, \boldsymbol{x}_1\rangle + \langle \boldsymbol{w}_k, \boldsymbol{x}_i\rangle] = 0 = \Upsilon(\boldsymbol{y})$. So such $\boldsymbol{z}$ is a maximizer. Pick $i = k+2$ and we can derive from this $\boldsymbol{z}$ a subgradient of $R_{\text{emp}}(\boldsymbol{w}_k)$:

$$\boldsymbol{a}_{k+1} = \frac{2}{n}(-\boldsymbol{x}_1 + \boldsymbol{x}_{k+2}) = -2c\boldsymbol{e}_2 - 2c\boldsymbol{e}_3 - 2c\boldsymbol{e}_{k+4}, \quad b_{k+1} = 0 - \langle \boldsymbol{a}_{k+1}, \boldsymbol{w}_k\rangle = 4c^2 + 4c^2 = 1,$$

$$\boldsymbol{w}_{k+1} = \operatorname*{argmin}_{\boldsymbol{w}} J_{k+1}(\boldsymbol{w}) = \operatorname*{argmin}_{\boldsymbol{w}} \left\{ \frac{1}{2}\|\boldsymbol{w}\|^2 + \max_{i\in[k+1]} \{\langle \boldsymbol{a}_i, \boldsymbol{w}\rangle + b_i\} \right\} = 2c\boldsymbol{e}_2 + 2c\boldsymbol{e}_3 + \frac{2c}{k+1}\sum_{i=4}^{k+4} \boldsymbol{e}_i.$$

To verify the last step, note $\langle \boldsymbol{a}_i, \boldsymbol{w}_{k+1}\rangle + b_i = \frac{-1}{2(k+1)}$ for all $i \in [k+1]$, and so $\partial J_{k+1}(\boldsymbol{w}_{k+1}) = \{\boldsymbol{w}_{k+1} + \sum_{i=1}^{k+1} \alpha_i \boldsymbol{a}_i : \boldsymbol{\alpha} \in \Delta_{k+1}\} \ni \boldsymbol{0}$ (just set all $\alpha_i = \frac{1}{k+1}$). So (10) holds for step $k+1$. (End of induction.)

All that remains is to observe $J(\boldsymbol{w}_k) = \frac{1}{4}(2 + \frac{1}{k})$ while $\min_{\boldsymbol{w}} J(\boldsymbol{w}) \leq J(\boldsymbol{w}_{n-1}) = \frac{1}{4}(2 + \frac{1}{n-1})$, from which it follows that $J(\boldsymbol{w}_k) - \min_{\boldsymbol{w}} J(\boldsymbol{w}) \geq \frac{1}{4}(\frac{1}{k} - \frac{1}{n-1})$. ■

Note in the above run of BMRM, all subgradients $\boldsymbol{a}_k$ have norm $\sqrt{\frac{3}{2}}$ which satisfies Assumption 1.

2.3 Construction Of Lower Bound For ROCArea Problem

Next we consider the case of ROCArea. Given $\varepsilon > 0$, define $n = \lfloor 1/\varepsilon \rfloor$ and construct a data set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n+1}$ with $n+1$ examples as follows. Let $\boldsymbol{e}_i$ be the i-th coordinate vector in $\mathbb{R}^{n+2}$ ($p = n+2$). The first example is the only positive example with $y_1 = +1$ and $\boldsymbol{x}_1 = \sqrt{n}\boldsymbol{e}_2$. The rest n examples are all negative, with $y_i = -1$ and $\boldsymbol{x}_i = -n\boldsymbol{e}_{i+1}$ for $i \in [2, n+1]$. Then the corresponding empirical risk for ROCArea loss is

$$R_{\text{emp}}(\boldsymbol{w}) = \frac{1}{n}\sum_{i=2}^{n+1} \max\{0, 1 - \langle \boldsymbol{w}, \boldsymbol{x}_1 - \boldsymbol{x}_i\rangle\} = \frac{1}{n}\sum_{i=2}^{n+1} \max\{0, 1 - \langle \boldsymbol{w}, \underbrace{\sqrt{n}\boldsymbol{e}_2 + n\boldsymbol{e}_{i+1}}_{:=\boldsymbol{u}_i}\rangle\}.$$

By further setting $\lambda = 1$, we obtain the overall objective $J(\boldsymbol{w})$ in (1). It is easy to see that the minimizer of $J(\boldsymbol{w})$ is $\mathbf{w}^* = \frac{1}{2}\left(\frac{1}{\sqrt{n}}\boldsymbol{e}_2 + \frac{1}{n}\sum_{i=3}^{n+2} \boldsymbol{e}_i\right)$ and $J(\mathbf{w}^*) = \frac{1}{4n}$. In fact, simply check that $\langle \mathbf{w}^*, \boldsymbol{u}_i\rangle = 1$ for all i, so $\partial J(\mathbf{w}^*) = \left\{\mathbf{w}^* - \sum_{i=1}^{n} \alpha_i \left(\frac{1}{\sqrt{n}}\boldsymbol{e}_2 + \boldsymbol{e}_{i+2}\right) : \alpha_i \in [0,1]\right\}$, and setting all $\alpha_i = \frac{1}{2n}$ yields the subgradient $\boldsymbol{0}$. Then we have the following lower bound on the rates of convergence for BMRM.

Theorem 5 *Let* $\mathbf{w}_0 = \boldsymbol{e}_1 + \frac{1}{\sqrt{n}}\boldsymbol{e}_2$. *Suppose running* BMRM *on the above constructed objective* $J(\boldsymbol{w})$ *produces iterates* $\mathbf{w}_1, \ldots, \mathbf{w}_k, \ldots$. *Then it takes* BMRM *at least* $\lfloor \frac{2}{3\varepsilon} \rfloor$ *steps to find an* ε *accurate solution. Formally,*

$$\min_{i\in[k]} J(\mathbf{w}_i) - J(\mathbf{w}^*) = \frac{1}{2k} + \frac{1}{4n} \text{ for all } k \in [n], \text{hence } \min_{i\in[k]} J(\mathbf{w}_i) - J(\mathbf{w}^*) > \varepsilon \text{ for all } k < \frac{2}{3\varepsilon}.$$

Proof The crux of the proof is to show that

$$\boldsymbol{w}_k = \frac{1}{\sqrt{n}}\boldsymbol{e}_2 + \frac{1}{k}\sum_{i=3}^{k+2} \boldsymbol{e}_i, \quad \forall\, k \in [n]. \tag{12}$$

We prove (12) by induction. Initially at $k=1$, we note that $\langle \boldsymbol{w}_0, \boldsymbol{u}_i \rangle = 1$ for all $i \in [2, n+1]$. Hence we can derive a subgradient and $\boldsymbol{w}_1$:

$$\boldsymbol{a}_1 = -\frac{1}{n}\boldsymbol{u}_2 = -\frac{1}{\sqrt{n}}\boldsymbol{e}_2 - \boldsymbol{e}_3, \qquad b_1 = -\langle \boldsymbol{a}_1, \boldsymbol{w}_0 \rangle = \frac{1}{n},$$
$$\boldsymbol{w}_1 = \operatorname*{argmin}_{\boldsymbol{w}} \left\{ \frac{1}{2}\|\boldsymbol{w}\|^2 + \langle \boldsymbol{a}_1, \boldsymbol{w} \rangle + b_1 \right\} = -\boldsymbol{a}_1 = \frac{1}{\sqrt{n}}\boldsymbol{e}_2 + \boldsymbol{e}_3.$$

So (12) holds for $k=1$. Now suppose it hold for steps $1, \ldots, k$ ($k \in [n-1]$). Then for step $k+1$ we note

$$\langle \boldsymbol{w}_k, \boldsymbol{u}_i \rangle = \begin{cases} 1 + \frac{n}{k} & \text{if } i \leq k \\ 1 & \text{if } i \geq k+1 \end{cases}.$$

Therefore we can derive a subgradient and compute $\boldsymbol{w}_{k+1}$:

$$\boldsymbol{a}_{k+1} = -\frac{1}{n}\boldsymbol{u}_{k+2} = -\frac{1}{\sqrt{n}}\boldsymbol{e}_2 - \boldsymbol{e}_{k+3}, \qquad b_{k+1} = -\langle \boldsymbol{a}_{k+1}, \boldsymbol{w}_k \rangle = \frac{1}{n},$$
$$\boldsymbol{w}_{k+1} = \operatorname*{argmin}_{\boldsymbol{w}} \underbrace{\left\{ \frac{1}{2}\|\boldsymbol{w}\|^2 + \max_{i \in [k+1]} \{\langle \boldsymbol{a}_i, \boldsymbol{w} \rangle + b_i\} \right\}}_{:= J_{k+1}(\boldsymbol{w})} = -\frac{1}{k+1}\sum_{i=1}^{k+1} \boldsymbol{a}_i = \frac{1}{\sqrt{n}}\boldsymbol{e}_2 + \frac{1}{k+1}\sum_{i=3}^{k+3} \boldsymbol{e}_i.$$

This minimizer can be verified by noting that $\langle \boldsymbol{w}_{k+1}, \boldsymbol{a}_i \rangle + b_i = -\frac{1}{k+1}$ for all $i \in [k+1]$, and so $\partial J_{k+1}(\boldsymbol{w}_{k+1}) = \{\boldsymbol{w}_{k+1} + \sum_{i=1}^{k+1} \alpha_i \boldsymbol{a}_i : \boldsymbol{\alpha} \in \Delta_{k+1}\} \ni \boldsymbol{0}$ (just set all $\alpha_i = \frac{1}{k+1}$). So (12) holds for step $k+1$. (End of induction.)

All that remains is to observe $J(\boldsymbol{w}_k) = \frac{1}{2}(\frac{1}{n} + \frac{1}{k})$. ■

As before, in the above run of BMRM, all subgradients $\boldsymbol{a}_k$ have norm $(1+\frac{1}{n})^{1/2}$ which satisfies Assumption 1.

3. Reformulating The Empirical Risk

In order to bypass the lower bound of CPMs and show that the multivariate score problems can be optimized at faster rates, we propose a smoothing strategy as described in Section 1.1. In this section, we will focus on the design of smooth surrogate functions, with emphasis on the consequent rates of convergence when optimized by Nesterov's accelerated gradient methods (Nesterov, 2005). The computational issues will be tackled in Section 4.

In order to approximate R_{emp} by $g_\mu^\star$ we will start by writing $R_{\text{emp}}(\mathbf{w})$ as $g^*(A^\top \mathbf{w})$ for an appropriate linear transform A and convex function g. Let the domain of g be Q and d be a strongly convex function with modulus 1 defined on Q. Furthermore, assume $\min_{\boldsymbol{\alpha} \in Q} d(\boldsymbol{\alpha}) = 0$ and denote $D = \max_{\boldsymbol{\alpha} \in Q} d(\boldsymbol{\alpha})$. In optimization parlance d is called a *prox-function*. Set

$$g_\mu^\star = (g + \mu d)^\star.$$

Then by the analysis in Appendix A, we can conclude that $g_\mu^\star(A^\top \mathbf{w})$ has a Lipschitz continuous gradient with constant at most $\frac{1}{\mu}\|A\|^2$. In addition,

$$\left| g_\mu^\star(A^\top \mathbf{w}) - R_{\text{emp}}(\mathbf{w}) \right| \leq \mu D. \tag{13}$$

Choosing $\mu = \varepsilon/D$, we can guarantee the approximation is uniformly upper bounded by ε.

There are indeed many different ways of expressing $R_{\text{emp}}(\mathbf{w})$ as $g^\star(A^\top \mathbf{w})$, but the next two sections will demonstrate the advantage of our design.

3.1 Contingency Table Based Loss

Letting $\mathcal{S}^k$ denote the k dimensional probability simplex, we can rewrite (3) as:

$$R_{\text{emp}}(\mathbf{w}) = \max_{\boldsymbol{z}\in\{-1,1\}^n} \left[\Delta(\boldsymbol{z},\boldsymbol{y}) + \frac{1}{n}\sum_{i=1}^n \langle \mathbf{w}, \boldsymbol{x}_i\rangle (z_i - y_i)\right] \tag{14}$$

$$= \max_{\boldsymbol{\alpha}\in\mathcal{S}^{2^n}} \sum_{\boldsymbol{z}\in\{-1,1\}^n} \alpha_{\boldsymbol{z}} \left(\Delta(\boldsymbol{z},\boldsymbol{y}) + \frac{1}{n}\sum_{i=1}^n \langle \mathbf{w}, \boldsymbol{x}_i\rangle (z_i - y_i)\right) \tag{15}$$

$$= \max_{\boldsymbol{\alpha}\in\mathcal{S}^{2^n}} \sum_{\boldsymbol{z}\in\{-1,1\}^n} \alpha_{\boldsymbol{z}}\Delta(\boldsymbol{z},\boldsymbol{y}) - \frac{2}{n}\sum_{i=1}^n y_i \langle \mathbf{w}, \boldsymbol{x}_i\rangle \left(\sum_{\boldsymbol{z}:z_i=-y_i} \alpha_{\boldsymbol{z}}\right).$$

Note from (14) to (15), we defined $0\cdot(-\infty) = 0$ since $\Delta(\boldsymbol{z},\boldsymbol{y})$ can be $-\infty$ as in (4). Introduce $\beta_i = \sum_{\boldsymbol{z}:z_i=-y_i} \alpha_{\boldsymbol{z}}$. Then $\beta_i \in [0,1]$ and one can rewrite the above equation in terms of $\boldsymbol{\beta}$:

$$R_{\text{emp}}(\mathbf{w}) = \max_{\boldsymbol{\beta}\in[0,1]^n} \left\{\frac{-2}{n}\sum_{i=1}^n y_i \langle \mathbf{w}, \boldsymbol{x}_i\rangle \beta_i - g(\boldsymbol{\beta})\right\} \tag{16}$$

$$\text{where}\quad g(\boldsymbol{\beta}) := -\max_{\boldsymbol{\alpha}\in\mathcal{A}(\boldsymbol{\beta})} \sum_{\boldsymbol{z}\in\{-1,1\}^n} \alpha_{\boldsymbol{z}}\Delta(\boldsymbol{z},\boldsymbol{y}). \tag{17}$$

Here $\mathcal{A}(\boldsymbol{\beta})$ is a subset of $\mathcal{S}^{2^n}$ defined via $\mathcal{A}(\boldsymbol{\beta}) = \left\{\boldsymbol{\alpha} \text{ s.t. } \sum_{\boldsymbol{z}:z_i=-y_i} \alpha_{\boldsymbol{z}} = \beta_i \text{ for all } i\right\}$. Indeed, this rewriting only requires that the mapping from $\boldsymbol{\alpha}\in\mathcal{S}^{2^n}$ to $\boldsymbol{\beta}\in Q := [0,1]^n$ is surjective. This is clear because for any $\boldsymbol{\beta}\in[0,1]^n$, a pre-image $\boldsymbol{\alpha}$ can be constructed by setting:

$$\alpha_{\boldsymbol{z}} = \prod_{i=1}^n \gamma_i, \quad \text{where} \quad \gamma_i = \begin{cases} \beta_i & \text{if } z_i = -y_i \\ 1-\beta_i & \text{if } z_i = y_i \end{cases}.$$

Proposition 6 *$g(\boldsymbol{\beta})$ defined by (17) is convex on $\boldsymbol{\beta}\in[0,1]^n$.*

Proof Clearly $\mathcal{A}(\boldsymbol{\beta})$ is closed for any $\boldsymbol{\beta}\in Q$. So without loss of generality assume that for any $\boldsymbol{\beta}$ and $\boldsymbol{\beta}'$ in $[0,1]^n$, the maximum in (17) is attained at $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ respectively. So $\sum_{\boldsymbol{z}:z_i=-y_i} \alpha_{\boldsymbol{z}} = \beta_i$ and $\sum_{\boldsymbol{z}:z_i=-y_i} \alpha'_{\boldsymbol{z}} = \beta'_i$. For any $\rho\in[0,1]$, consider $\boldsymbol{\beta}^\rho := \rho\boldsymbol{\beta} + (1-\rho)\boldsymbol{\beta}'$. Define $\boldsymbol{\alpha}^\rho := \rho\boldsymbol{\alpha} + (1-\rho)\boldsymbol{\alpha}'$. Then clearly $\boldsymbol{\alpha}^\rho$ satisfies

$$\sum_{\boldsymbol{z}:z_i=-y_i} \alpha^\rho_{\boldsymbol{z}} = \rho\beta_i + (1-\rho)\beta'_i = \beta^\rho_i.$$

Therefore $\boldsymbol{\alpha}^\rho$ is admissible in the definition of $g(\boldsymbol{\beta}^\rho)$ (17), and

$$g(\boldsymbol{\beta}^\rho) \le -\sum_{\boldsymbol{z}} \alpha^\rho_{\boldsymbol{z}}\Delta(\boldsymbol{z},\boldsymbol{y}) = -\rho\sum_{\boldsymbol{z}} \alpha_{\boldsymbol{z}}\Delta(\boldsymbol{z},\boldsymbol{y}) - (1-\rho)\sum_{\boldsymbol{z}} \alpha'_{\boldsymbol{z}}\Delta(\boldsymbol{z},\boldsymbol{y})$$
$$= \rho g(\boldsymbol{\beta}) + (1-\rho) g(\boldsymbol{\beta}').$$

which establishes the convexity of g. ■

Using (16) it immediately follows that $R_{\text{emp}}(\mathbf{w}) = g^\star(A^\top\mathbf{w})$ where A is a p-by-n matrix whose i-th column is $\frac{-2}{n} y_i \boldsymbol{x}_i$,.

3.1.1 $\sqrt{D}\,\|A\|$ FOR OUR DESIGN

Let us choose the prox-function $d(\boldsymbol{\beta})$ as $\frac{1}{2}\|\boldsymbol{\beta}\|^2$. Then $D = \max_{\boldsymbol{\beta}\in[0,1]^n} d(\boldsymbol{\beta}) = \frac{n}{2}$. Define $A = \frac{-2}{n}(y_1\boldsymbol{x}_1,\ldots,y_n\boldsymbol{x}_n)$ and M be a n-by-n matrix with $M_{ij} = y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle$. Clearly, $A^\top A = \frac{4}{n^2}M$. Because we assumed that $\|\boldsymbol{x}_i\| \le R$, it follows that $|M_{ij}| \le R^2$. The norm of A can be upper bounded as follows:

$$\|A\|^2 = \lambda_{\max}(A^\top A) = \frac{4}{n^2}\lambda_{\max}(M) \le \frac{4}{n^2}\operatorname{tr}(M) \le \frac{4}{n^2}\cdot nR^2 = \frac{4R^2}{n}. \tag{18}$$

Hence

$$\sqrt{D}\,\|A\| \le \sqrt{\frac{n}{2}}\frac{2R}{\sqrt{n}} = \sqrt{2}R.$$

This constant upper bound is highly desirable because it implies that the rate of convergence in (7) is independent of the size of the data set (n or p). However, the cost per iteration still depends on n and p.

3.1.2 ALTERNATIVES

It is illuminating to see how naive choices for smoothing R_{emp} can lead to excessively large values of $\sqrt{D}\,\|A\|$. For instance, using (14), $R_{\text{emp}}(\mathbf{w})$ can be rewritten as

$$\begin{aligned} R_{\text{emp}}(\mathbf{w}) &= \max_{\boldsymbol{z}\in\{-1,1\}^n} \frac{1}{n}\left[n\Delta(\boldsymbol{z},\boldsymbol{y}) + \sum_{i=1}^n \langle \mathbf{w}, \boldsymbol{x}_i\rangle (z_i - y_i)\right] \\ &= \max_{\boldsymbol{\alpha}\in Q} \sum_{\boldsymbol{z}\in\{-1,1\}^n} \alpha_{\boldsymbol{z}}\left(n\Delta(\boldsymbol{z},\boldsymbol{y}) + \sum_{i=1}^n \langle \mathbf{w}, \boldsymbol{x}_i\rangle (z_i - y_i)\right). \end{aligned}$$

where Q refers to the 2^n dimensional simplex scaled by $\frac{1}{n}$: $Q = \left\{\boldsymbol{\alpha} : \alpha_{\boldsymbol{z}} \in [0, n^{-1}] \text{ and } \sum_{\boldsymbol{z}} \alpha_{\boldsymbol{z}} = \frac{1}{n}\right\}$. Defining A as a p-by-2^n matrix whose $\boldsymbol{z}$-th column is given by $\sum_{i=1}^n \boldsymbol{x}_i(z_i - y_i)$ and writing

$$h(\boldsymbol{\alpha}) = \begin{cases} -n\sum_{\boldsymbol{z}\in\{-1,1\}^n}\Delta(\boldsymbol{z},\boldsymbol{y})\alpha_{\boldsymbol{z}} & \text{if } \boldsymbol{\alpha}\in Q \\ +\infty & \text{otherwise} \end{cases},$$

we can express R_{emp} as

$$R_{\text{emp}}(\mathbf{w}) = \max_{\boldsymbol{\alpha}\in Q}\left[\left\langle A^\top\mathbf{w}, \boldsymbol{\alpha}\right\rangle - h(\boldsymbol{\alpha})\right] = h^\star(A^\top\mathbf{w}). \tag{19}$$

Now let us investigate the value of $\sqrt{D}\,\|A\|$. As Q is a scaled simplex, a natural choice of prox-function is

$$d(\boldsymbol{\alpha}) = \sum_{\boldsymbol{z}} \alpha_{\boldsymbol{z}}\log\alpha_{\boldsymbol{z}} + \frac{1}{n}\log n + \log 2.$$

Clearly, $d(\boldsymbol{\alpha})$ is a strongly convex function in Q with modulus of strong convexity 1 under the L_1 norm (see, e.g., Beck and Teboulle, 2003, Proposition 5.1). In addition, $d(\boldsymbol{\alpha})$ is minimized when $\boldsymbol{\alpha}$ is uniform, that is, $\alpha_{\boldsymbol{z}} = \frac{1}{n2^n}$ for all $\boldsymbol{z}$, and so

$$\min_{\boldsymbol{\alpha}\in Q} d(\boldsymbol{\alpha}) = 2^n\frac{1}{n2^n}\log\frac{1}{n2^n} + \frac{1}{n}\log n + \log 2 = 0.$$

Therefore $d(\boldsymbol{\alpha})$ satisfies all the prerequisites of being a prox-function. Furthermore, $d(\boldsymbol{\alpha})$ is maximized on the corner of the scaled simplex, for example, $\alpha_{\boldsymbol{y}} = \frac{1}{n}$ and $\alpha_{\boldsymbol{z}} = 0$ for all $\boldsymbol{z} \neq \boldsymbol{y}$. So

$$D = \max_{\boldsymbol{\alpha} \in Q} d(\boldsymbol{\alpha}) = \frac{1}{n} \log \frac{1}{n} + \frac{1}{n} \log n + \log 2 = \log 2. \tag{20}$$

Finally, denoting the $\mathbf{z}$-th column of A as $A_{:\boldsymbol{z}}$, we can compute $\|A\|$ as

$$\begin{aligned} \|A\| &= \max_{\mathbf{u}:\|\mathbf{u}\|=1, \mathbf{v}:\sum_{\boldsymbol{z}}|v_{\boldsymbol{z}}|=1} \mathbf{u}^\top A \mathbf{v} = \max_{\mathbf{v}:\sum_{\boldsymbol{z}}|v_{\boldsymbol{z}}|=1} \|A\mathbf{v}\| = \max_{\mathbf{z}} \|A_{:\mathbf{z}}\| \\ &= \max_{\mathbf{z}} \left\| \sum_{i=1}^{n} \boldsymbol{x}_i (z_i - y_i) \right\| \leq 2 \sum_{i=1}^{n} \|\boldsymbol{x}_i\| \leq 2nR. \end{aligned} \tag{21}$$

Note this bound is tight because $\|A\| = 2nR$ can be attained by setting $\boldsymbol{x}_i = R\boldsymbol{e}_1$ for all i, in which case the maximizing $\boldsymbol{z}$ is simply $-\boldsymbol{y}$. Eventually combining (20) and (21), we arrive at

$$\sqrt{D}\,\|A\| \leq 2nR\sqrt{\log 2}.$$

Thus if we re-express R_{emp} by using this alternative form as in (19), $\sqrt{D}\,\|A\|$ will scale as $\Theta(nR)$ which grows linearly with n, the number of training examples.

3.2 ROCArea Loss

We rewrite $R_{\text{emp}}(\mathbf{w})$ from (5) as:

$$\frac{1}{m} \max_{z_{ij} \in \{0,1\}} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} z_{ij} [1 - \mathbf{w}^\top (\mathbf{x}_i - \mathbf{x}_j)] = \frac{1}{m} \max_{\boldsymbol{\beta} \in [0,1]^m} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \beta_{ij} \left[1 - \boldsymbol{w}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right]. \tag{22}$$

The above empirical risk can be identified with $g^\star(A^\top \mathbf{w})$ by setting

$$g(\boldsymbol{\beta}) = \begin{cases} -\frac{1}{m} \sum_{i,j} \beta_{ij} & \text{if } \boldsymbol{\beta} \in [0,1]^m \\ +\infty & \text{elsewhere} \end{cases},$$

and letting A be a p-by-m matrix whose (ij)-th column is $-\frac{1}{m}(\boldsymbol{x}_i - \boldsymbol{x}_j)$. This g is a linear function defined over a convex domain, and hence clearly convex.

3.2.1 $\sqrt{D}\,\|A\|$ FOR OUR DESIGN

Choose prox-function $d(\boldsymbol{\beta}) = \frac{1}{2} \sum_{i,j} \beta_{ij}^2$. By a simple calculation, $D = \max_{\boldsymbol{\beta}} d(\boldsymbol{\beta}) = \frac{m}{2}$. As before, define A to be a matrix such that its (ij)-th column $A_{ij} = -\frac{1}{m}(\boldsymbol{x}_i - \boldsymbol{x}_j)$ and M be a m-by- m matrix with $M_{(ij)(i'j')} = \langle \boldsymbol{x}_i - \boldsymbol{x}_j, \boldsymbol{x}_{i'} - \boldsymbol{x}_{j'} \rangle$. Clearly, $A^\top A = \frac{1}{m^2} M$. Because we assumed that $\|\boldsymbol{x}_i\| \leq R$, it follows that $|M_{ij}| \leq 4R^2$. The norm of A can be upper bounded as follows:

$$\|A\|^2 = \lambda_{\max}(A^\top A) = \frac{1}{m^2} \lambda_{\max}(M) \leq \frac{1}{m^2} \operatorname{tr}(M) \leq \frac{1}{m^2} \cdot 4mR^2 = \frac{4R^2}{m}.$$

Therefore

$$\sqrt{D}\,\|A\| \leq \sqrt{\frac{m}{2}} \cdot \frac{2R}{\sqrt{m}} = \sqrt{2}R. \tag{23}$$

This upper bound is independent of n and guarantees that the rate of convergence is not affected by the size of the data set.

3.2.2 ALTERNATIVES

The advantage of our above design of $g^\star(A^\top \boldsymbol{w})$ can be seen by comparing it with an alternative design. In (22), let us introduce

$$\gamma_i = \sum_{j\in\mathcal{N}} \beta_{ij},\ i \in \mathcal{P}, \qquad \text{and} \qquad \gamma_j = \sum_{i\in\mathcal{P}} \beta_{ij},\ j \in \mathcal{N}.$$

Then clearly $\boldsymbol{\gamma} \in \mathcal{B} := \{\gamma_i \in [0, n_-], \gamma_j \in [0, n_+]\}$. However this map from $\boldsymbol{\beta} \in [0,1]^m$ to $\boldsymbol{\gamma} \in \mathcal{B}$ is obviously *not* surjective, because the image $\boldsymbol{\gamma}$ should at least satisfy $\sum_{i\in\mathcal{P}} \gamma_i = \sum_{j\in\mathcal{N}} \gamma_j$.[5] So let us denote the real range of the mapping as $\mathcal{A}$ and $\mathcal{A} \subseteq \mathcal{B}$. Now rewrite (22) in terms of $\boldsymbol{\gamma}$:

$$\begin{aligned} R_{\text{emp}}(\boldsymbol{w}) &= \frac{1}{m} \max_{\boldsymbol{\gamma}\in\mathcal{A}} \left\{ \sum_{i\in\mathcal{P}} \gamma_i \boldsymbol{w}^\top(-\boldsymbol{x}_i) + \sum_{j\in\mathcal{N}} \gamma_j \boldsymbol{w}^\top \boldsymbol{x}_j + \sum_{i\in\mathcal{P}} \gamma_i + \sum_{j\in\mathcal{N}} \gamma_j \right\} \\ &= g^\star(A^\top \boldsymbol{w}), \quad \text{where} \quad g(\boldsymbol{\gamma}) = \begin{cases} -\frac{1}{m}\sum_{i\in\mathcal{P}} \gamma_i + \frac{1}{m}\sum_{j\in\mathcal{N}} \gamma_j & \text{if } \boldsymbol{\gamma} \in \mathcal{A} \\ +\infty & \text{elsewhere} \end{cases}, \end{aligned}$$

and A is a p-by-n matrix whose i-th column is $-\frac{1}{m}\boldsymbol{x}_i$ ($i \in \mathcal{P}$) and j-th column is $\frac{1}{m}\boldsymbol{x}_j$ ($j \in \mathcal{N}$).

Choose a prox-function $d(\boldsymbol{\gamma}) = \frac{1}{2}\|\boldsymbol{\gamma}\|^2$. Then we can bound $\|A\|$ by $\frac{R}{m}\sqrt{n}$ similar to (18) and bound D by

$$D = \max_{\boldsymbol{\gamma}\in\mathcal{A}} d(\boldsymbol{\gamma}) \le \max_{\boldsymbol{\gamma}\in\mathcal{B}} d(\boldsymbol{\gamma}) = \frac{1}{2}\left(n_+^2 n_- + n_+ n_-^2\right).$$

Note that the $\le$ here is actually an equality because they have a common maximizer: $\gamma_i = n_-$ for all $i \in \mathcal{P}$ and $\gamma_j = n_+$ for all $j \in \mathcal{N}$ (which is mapped from all $\beta_{ij} = 1$). Now we have

$$\sqrt{D}\,\|A\| \le \frac{1}{\sqrt{2}}\sqrt{n_+^2 n_- + n_+ n_-^2}\,\frac{R}{m}\sqrt{n} = \frac{R}{\sqrt{2}}\sqrt{\frac{(n_+ + n_-)^2}{n_+ n_-}}.$$

This bound depends on the ratio between positive and negative examples. If the class is balanced and there exist constants $c_+, c_- \in (0,1)$ such that $n_+ \ge c_+ n$ and $n_- \ge c_- n$, then we easily derive $\sqrt{D}\,\|A\| \le \frac{R}{\sqrt{2c_+c_-}}$ which is a constant. However, if we fix $n_- = 1$ and $n_+ = n-1$, then $\sqrt{D}\,\|A\|$ will scale as $\Theta(\sqrt{n})$. In contrast, the bound in (23) is always a constant.

4. Efficient Evaluation Of The Smoothed Objective And Its Gradient

The last building block required to make our smoothing scheme work is an efficient algorithm to compute the smoothed empirical risk $g_\mu^\star(A^\top \mathbf{w})$ and its gradient. By the chain rule and Corollary X.1.4.4 of Hiriart-Urruty and Lemaréchal (1996), we have

$$\frac{\partial}{\partial \mathbf{w}} g_\mu^\star(A^\top \mathbf{w}) = A\hat{\boldsymbol{\beta}} \quad \text{where} \quad \hat{\boldsymbol{\beta}} = \operatorname*{argmax}_{\boldsymbol{\beta}\in\mathcal{Q}} \left\langle \boldsymbol{\beta}, A^\top \mathbf{w} \right\rangle - g(\boldsymbol{\beta}) - \mu d(\boldsymbol{\beta}). \tag{24}$$

Since g is convex and d is strongly convex, the optimal $\hat{\boldsymbol{\beta}}$ must be unique. Two major difficulties arise in computing the above gradient: the optimal $\hat{\boldsymbol{\beta}}$ can be hard to solve (e.g., in the case of contingency table based loss), and the matrix vector product in (24) can be costly (e.g., $O(n^2 p)$ for ROCArea). Below we show how these difficulties can be overcome to compute the gradient in $O(n \log n)$ time.

5. Note that even adding this equality constraint to the definition of $\mathcal{B}$ will still not make the map surjective.

4.1 Contingency Table Based Loss

Since A is a $p \times n$ dimensional matrix and $\hat{\boldsymbol{\beta}}$ is a n dimensional vector, the matrix vector product in (24) can be computed in $O(np)$ time. Below we focus on solving $\hat{\boldsymbol{\beta}}$ in (24).

To take into account the constraints in the definition of $g(\boldsymbol{\beta})$, we introduce Lagrangian multipliers θ_i and the optimization in (24) becomes

$$\begin{aligned} g_\mu^\star(A^\top \mathbf{w}) = \max_{\boldsymbol{\beta}\in[0,1]^n} \Bigg\{ & \frac{-2}{n}\sum_{i=1}^n y_i \langle \mathbf{w}, \boldsymbol{x}_i\rangle \beta_i - \frac{\mu}{2}\sum_{i=1}^n \beta_i^2 \\ & + \max_{\boldsymbol{\alpha}\in\mathcal{S}^{2^n}} \left[\sum_{\boldsymbol{z}} \alpha_{\boldsymbol{z}} \Delta(\boldsymbol{z},\boldsymbol{y}) + \min_{\boldsymbol{\theta}\in\mathbb{R}^n} \sum_{i=1}^n \theta_i \left(\sum_{\boldsymbol{z}: z_i = -y_i} \alpha_{\boldsymbol{z}} - \beta_i \right) \right] \Bigg\} \end{aligned} \tag{25}$$

$$\begin{aligned} \Leftrightarrow \min_{\boldsymbol{\theta}\in\mathbb{R}^n} \Bigg\{ & \max_{\boldsymbol{\alpha}\in\mathcal{S}^{2^n}} \sum_{\boldsymbol{z}} \alpha_{\boldsymbol{z}} \left[\Delta(\boldsymbol{z},\boldsymbol{y}) + \sum_i \theta_i \delta(z_i = -y_i) \right] \\ & + \max_{\boldsymbol{\beta}\in[0,1]^n} \sum_{i=1}^n \left(\frac{-\mu}{2}\beta_i^2 - \left(\frac{2}{n} y_i \langle \mathbf{w}, \boldsymbol{x}_i\rangle + \theta_i \right) \beta_i \right) \Bigg\} \end{aligned}$$

$$\Leftrightarrow \min_{\boldsymbol{\theta}\in\mathbb{R}^n} \Bigg\{ \max_{\boldsymbol{z}} \underbrace{\left[\Delta(\boldsymbol{z},\boldsymbol{y}) + \sum_i \theta_i \delta(z_i = -y_i) \right]}_{:= q(\boldsymbol{z},\boldsymbol{\theta})} + \sum_{i=1}^n \underbrace{\max_{\beta_i\in[0,1]} \left[\frac{-\mu}{2}\beta_i^2 - \left(\frac{2}{n} y_i \langle \mathbf{w}, \boldsymbol{x}_i\rangle + \theta_i \right) \beta_i \right]}_{:= h_i(\theta_i)} \Bigg\}. \tag{26}$$

The last step is because all β_i are decoupled and can be optimized independently. Let

$$D(\boldsymbol{\theta}) := \max_{\boldsymbol{z}} q(\boldsymbol{z},\boldsymbol{\theta}) + \sum_{i=1}^n h_i(\theta_i),$$

and $\hat{\boldsymbol{\theta}}$ be a minimizer of $D(\boldsymbol{\theta})$: $\hat{\boldsymbol{\theta}} \in \operatorname{argmin}_{\boldsymbol{\theta}} D(\boldsymbol{\theta})$. Given $\hat{\boldsymbol{\theta}}$ and denoting

$$a_i = \frac{-2}{n} y_i \langle \mathbf{w}, \boldsymbol{x}_i \rangle, \tag{27}$$

we can recover the optimal $\beta(\hat{\theta}_i)$ from the definition of $h_i(\hat{\theta}_i)$ as follows:

$$\hat{\beta}_i = \beta_i(\hat{\theta}_i) = \begin{cases} 0 & \text{if } \hat{\theta}_i \geq a_i \\ 1 & \text{if } \hat{\theta}_i \leq a_i - \mu \\ \frac{1}{\mu}(a_i - \hat{\theta}_i) & \text{if } \hat{\theta}_i \in [a_i - \mu, a_i] \end{cases}.$$

So, the main challenge that remains is to compute $\hat{\boldsymbol{\theta}}$. Towards this end, first note that:

$$\nabla_{\theta_i} h_i(\theta_i) = -\beta_i(\theta_i) \quad \text{and} \quad \partial_{\boldsymbol{\theta}} q(\boldsymbol{z},\boldsymbol{\theta}) = \operatorname{co}\left\{ \delta_{\boldsymbol{z}} : \boldsymbol{z} \in \operatorname*{argmax}_{\boldsymbol{z}} q(\boldsymbol{z},\boldsymbol{\theta}) \right\}.$$

Here $\delta_{\boldsymbol{z}} := (\delta(z_1 = -y_1), \ldots, \delta(z_n = -y_n))^\top$ and $\operatorname{co}(\cdot)$ denotes the convex hull of a set. By the first order optimality conditions $\mathbf{0} \in \partial D(\hat{\boldsymbol{\theta}})$, which implies that

$$\hat{\boldsymbol{\beta}} \in \operatorname{co}\left\{ \delta_{\boldsymbol{z}} : \boldsymbol{z} \in \operatorname*{argmax}_{\boldsymbol{z}} q(\boldsymbol{z},\boldsymbol{\theta}) \right\}. \tag{28}$$

We will henceforth restrict our attention to PRBEP as a special case of contingency table based loss Δ. A major contribution of this paper is a characterization of the optimal solution of (26), based on which we propose an $O(n\log n)$ complexity algorithm for finding $\hat{\boldsymbol{\theta}}$ exactly. To state the result, our notation needs to be refined. We will always use i to index positive examples ($i \in \mathcal{P}$) and j to index negative examples ($j \in \mathcal{N}$). Let the positive examples be associated with $\boldsymbol{x}_i^+$, $y_i^+(=1)$, a_i^+, $\hat{\theta}_i^+$, $\hat{\beta}_i^+$, and $h_i^+(\cdot)$, and the negative examples be associated with $\boldsymbol{x}_j^-$, $y_j^-(=-1)$, a_j^-, $\hat{\theta}_j^-$, $\hat{\beta}_j^-$, and $h_j^-(\cdot)$. Given two arbitrary scalars θ^+ and θ^-, let us define $\boldsymbol{\theta} = \boldsymbol{\theta}(\theta^+, \theta^-)$ by setting $\theta_i^+ = \theta^+$ for all $i \in \mathcal{P}$ and $\theta_j^- = \theta^-$ for all $j \in \mathcal{N}$. Finally define $f : \mathbb{R} \mapsto \mathbb{R}$ as follows

$$f(\theta) = \sum_{i\in\mathcal{P}} h_i^+(\theta) + \sum_{j\in\mathcal{N}} h_j^- \left(\frac{-1}{n_+} - \theta\right).$$

Then f is differentiable and convex in θ. Now our result can be stated as

Theorem 7 *The problem* (26) *must have an optimal solution which takes the form* $\boldsymbol{\theta}(\theta^+, \theta^-)$ *with* $\theta^+ + \theta^- = \frac{-1}{n_+}$. *Conversely, suppose* $\theta^+ + \theta^- = \frac{-1}{n_+}$, *then* $\boldsymbol{\theta}(\theta^+, \theta^-)$ *is an optimal solution if, and only if,* $\nabla f(\theta^+) = 0$.

Remark 8 *Although the optimal* $\boldsymbol{\theta}$ *to the problem* (26) *may be not unique, the optimal* $\boldsymbol{\beta}$ *to the problem* (25) *must be unique as we commented below* (24).

4.1.1 PROOF OF THEOREM 7

We start with the first sentence of Theorem 7. Our idea is as follows: given any optimal solution $\hat{\boldsymbol{\theta}}$, we will construct θ^+ and θ^- which satisfy $\theta^+ + \theta^- = \frac{-1}{n_+}$ and $D(\boldsymbol{\theta}(\theta^+, \theta^-)) \le D(\hat{\boldsymbol{\theta}})$.

First $\hat{\boldsymbol{\theta}}$ must be in $\mathbb{R}^n$ because $D(\boldsymbol{\theta})$ tends to ∞ when any component of $\boldsymbol{\theta}$ approaches ∞ or $-\infty$. Without loss of generality, assume $n_+ \le n_-$, $\hat{\theta}_1^+ \ge \hat{\theta}_2^+ \ge \ldots$, and $\hat{\theta}_1^- \ge \hat{\theta}_2^- \ge \ldots$. Define $\hat{\theta}_0^+ = \hat{\theta}_0^- = 0$, and consider the phase transition point

$$k := \max\left\{0 \le i \le n_+ : \hat{\theta}_i^+ + \hat{\theta}_i^- + \frac{1}{n_+} > 0\right\},$$

based on which we have three cases: $k = 0$, $1 \le k \le n_+ - 1$, and $k = n_+$.

Case 1: $1 \le k \le n_+ - 1$. Then there must exist $\theta^+ \in [\hat{\theta}_{k+1}^+, \hat{\theta}_k^+]$ and $\theta^- \in [\hat{\theta}_{k+1}^-, \hat{\theta}_k^-]$ such that $\theta^+ + \theta^- = \frac{-1}{n_+}$. Denote $\boldsymbol{\theta}' := \boldsymbol{\theta}(\theta^+, \theta^-)$ and it suffices to show $D(\boldsymbol{\theta}') \le D(\hat{\boldsymbol{\theta}})$. Since $\theta^+ + \theta^- = \frac{-1}{n_+}$, it is easy to see $\max_{\boldsymbol{z}} q(\boldsymbol{z}, \boldsymbol{\theta}') = 0$, and so

$$D(\boldsymbol{\theta}') = \sum_{i\in\mathcal{P}} h_i^+(\theta^+) + \sum_{j\in\mathcal{N}} h_j^-(\theta^-).$$

To get a lower bound of $D(\hat{\boldsymbol{\theta}})$, consider a specific $\boldsymbol{z}$ where $z_i^+ = -1$ for $1 \le i \le k$, $z_i^+ = 1$ for $i > k$, $z_j^- = 1$ for $1 \le j \le k$, and $z_j^- = -1$ for $j > k$. Then we have

$$D(\hat{\boldsymbol{\theta}}) \ge \frac{k}{n_+} + \sum_{i=1}^{k} \theta_i^+ + \sum_{j=1}^{k} \theta_j^- + \sum_{i\in\mathcal{P}} h_i^+(\theta_i^+) + \sum_{j\in\mathcal{N}} h_j^-(\theta_j^-).$$

Using $\theta^+ + \theta^- = \frac{-1}{n_+}$, we can compare:

$$D(\hat{\boldsymbol{\theta}}) - D(\boldsymbol{\theta}') \geq \sum_{i=1}^{k} \left[h_i^+(\theta_i^+) - h_i^+(\theta^+) + \theta_i^+ - \theta^+\right] + \sum_{j=1}^{k} \left[h_j^-(\theta_j^-) - h_j^-(\theta^-) + \theta_j^- - \theta^-\right] + \sum_{i=k+1}^{n_+} \left[h_i^+(\theta_i^+) - h_i^+(\theta^+)\right] + \sum_{j=k+1}^{n_-} \left[h_j^-(\theta_j^-) - h_j^-(\theta^-)\right]. \quad (29)$$

It is not hard to show all terms in the square bracket are non-negative. For $i \leq k$ and $j \leq k$, by definition of θ^+ and θ^-, we have $\theta_i^+ \geq \theta^+$ and $\theta_j^- \geq \theta^-$. Since the gradient of h_i^+ and h_j^- lies in $[-1,0]$, so by Rolle's mean value theorem we have

$$h_i^+(\theta_i^+) - h_i^+(\theta^+) + \theta_i^+ - \theta^+ \geq (-1)(\theta_i^+ - \theta^+) + \theta_i^+ - \theta^+ = 0,$$
$$h_j^-(\theta_j^-) - h_j^-(\theta^-) + \theta_j^- - \theta^- \geq (-1)(\theta_j^- - \theta^-) + \theta_j^- - \theta^- = 0.$$

For $i \geq k$ and $j \geq k$, by definition we have $\theta_i^+ \leq \theta^+$ and $\theta_j^- \leq \theta^-$, and so $h_i^+(\theta_i^+) \geq h_i^+(\theta^+)$ and $h_j^-(\theta_j^-) \geq h_j^-(\theta^-)$. By (29) we conclude $D(\hat{\boldsymbol{\theta}}) \geq D(\boldsymbol{\theta}')$.

Case 2: $k = n_+$. Then define $\theta^- = \theta_{n_+}^-$ and $\theta^+ = \frac{-1}{n_+} - \theta^-$. Since $k = n_+$ so $\theta^+ < \theta_{n_+}^+$. The rest of the proof for $D(\hat{\boldsymbol{\theta}}) \geq D(\boldsymbol{\theta}(\theta^+, \theta^-))$ is exactly the same as Case 1.

Case 3: $k = 0$. This means $\theta_i^+ + \theta_i^- + \frac{1}{n_+} \leq 0$ for all $1 \leq i \leq n_+$. Then define $\theta^+ = \theta_1^+$ and $\theta^- = \frac{-1}{n_+} - \theta^+$. Clearly, $\theta^- \geq \theta_1^-$. The rest of the proof for $D(\hat{\boldsymbol{\theta}}) \geq D(\boldsymbol{\theta}(\theta^+, \theta^-))$ is exactly the same as Case 1 with k set to 0.

Next we prove the second sentence of Theorem 7. We first show necessity. Denote $\boldsymbol{\theta}' := \boldsymbol{\theta}(\theta^+, \theta^-)$, and $\mathcal{Y} := \operatorname{argmax}_{\boldsymbol{z}} q(\boldsymbol{z}, \boldsymbol{\theta}')$. By the definition of PRBEP, any $\boldsymbol{z} \in \mathcal{Y}$ must satisfy

$$\sum_{i \in \mathcal{P}} \delta(z_i^+ = -1) = \sum_{j \in \mathcal{N}} \delta(z_j^- = 1). \quad (30)$$

Since $\boldsymbol{\theta}'$ is an optimizer of $D(\boldsymbol{\theta})$, so $\mathbf{0} \in \partial D(\boldsymbol{\theta}')$. Hence by (28), there must exist a distribution $\alpha(\boldsymbol{z})$ over $\boldsymbol{z} \in \mathcal{Y}$ such that

$$\nabla h_i^+(\theta^+) = -\sum_{\boldsymbol{z} \in \mathcal{Y}: z_i^+ = -1} \alpha(\boldsymbol{z}),\ \forall i \in \mathcal{P} \quad \text{and} \quad \nabla h_j^-(\theta^-) = -\sum_{\boldsymbol{z} \in \mathcal{Y}: z_j^- = 1} \alpha(\boldsymbol{z}),\ \forall j \in \mathcal{N}.$$

Therefore

$$\sum_{i \in \mathcal{P}} \nabla h_i^+(\theta^+) = -\sum_{i \in \mathcal{P}} \sum_{\boldsymbol{z} \in \mathcal{Y}: z_i^+ = -1} \alpha(\boldsymbol{z}) = -\sum_{\boldsymbol{z} \in \mathcal{Y}} \alpha(\boldsymbol{z}) \sum_{i \in \mathcal{P}} \delta(z_i^+ = -1)$$
$$\stackrel{\text{by (30)}}{=} -\sum_{\boldsymbol{z} \in \mathcal{Y}} \alpha(\boldsymbol{z}) \sum_{j \in \mathcal{N}} \delta(z_j^- = 1) = -\sum_{j \in \mathcal{N}} \sum_{\boldsymbol{z} \in \mathcal{Y}: z_j^- = 1} \alpha(\boldsymbol{z}) = \sum_{j \in \mathcal{N}} \nabla h_j^-(\theta^-).$$

Finally, we prove sufficiency. By the necessity, we know there is an optimal solution $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}(\theta^+, \theta^-)$ which satisfies $\theta^+ + \theta^- = \frac{-1}{n_+}$ and $\nabla f(\theta^+) = 0$. Now suppose there is another pair η^+ and η^- which also satisfy $\eta^+ + \eta^- = \frac{-1}{n_+}$ and $\nabla f(\eta^+) = 0$. Letting $\boldsymbol{\theta}' = \boldsymbol{\theta}(\eta^+, \eta^-)$, it suffices to show that $D(\boldsymbol{\theta}') = D(\hat{\boldsymbol{\theta}})$. Since $\theta^+ + \theta^- = \eta^+ + \eta^- = \frac{-1}{n_+}$, so

$$\max_{\boldsymbol{z}} q(\boldsymbol{z}, \hat{\boldsymbol{\theta}}) = \max_{\boldsymbol{z}} q(\boldsymbol{z}, \boldsymbol{\theta}') = 0. \quad (31)$$

Since f is convex and $\nabla f(\theta^+) = \nabla f(\eta^+) = 0$, so both θ^+ and η^+ minimize $f(\theta)$ and so $f(\theta^+) = f(\eta^+)$. In conjunction with (31), we conclude $D(\boldsymbol{\theta}') = D(\hat{\boldsymbol{\theta}})$. ■

4.1.2 AN $O(n\log n)$ COMPLEXITY ALGORITHM FOR SOLVING (26) UNDER PRBEP

We now design an $O(n\log n)$ complexity algorithm which, given $\{a_i\}$ from (27), finds an optimal solution $\hat{\boldsymbol{\theta}}$ exactly. By Theorem 7, it suffices to find a root of the gradient of f:

$$\nabla f(\theta) = g^+(\theta) - g^-\left(\frac{-1}{n_+} - \theta\right) \quad \text{where} \quad g^+(\theta) := \sum_{i\in\mathcal{P}} \nabla h_i^+(\theta) \ \text{ and } \ g^-(\theta) := \sum_{j\in\mathcal{N}} \nabla h_j^-(\theta).$$

Clearly, g^+ and g^- are piecewise linear. g^+ only has $2n_+$ kink points: $\{a_i^+, a_i^+ - \mu : i \in \mathcal{P}\}$ and we assume they are sorted decreasingly as $\tau_1^+ \geq \ldots \geq \tau_{2n_-}^+$. Similarly, g^- only has $2n_-$ kink points: $\{a_j^-, a_j^- - \mu : j \in \mathcal{N}\}$ and we assume they are sorted decreasingly as $\tau_1^- \geq \ldots \geq \tau_{2n_-}^-$. Further define $\tau_0^+ = \tau_0^- = \infty$ and $\tau_{2n_++1}^+ = \tau_{2n_-+1}^- = -\infty$.

Our main idea is to find a^+, b^+, a^-, b^- which satisfy i) g^+ is linear in $[a^+, b^+]$ and g^- is linear in $[a^-, b^-]$; and ii) there exists $\theta^+ \in [a^+, b^+]$ and $\theta^- \in [a^-, b^-]$ such that $\theta^+ + \theta^- = \frac{-1}{n_+}$ and $g^+(\theta^+) = g^-(\theta^-)$. Once this is done, we can find θ^+ and θ^- by solving a linear system

$$\theta^+ + \theta^- = \frac{-1}{n_+}\,,$$
$$g^+(a^+) + \frac{g^+(a^+) - g^+(b^+)}{a^+ - b^+}(\theta^+ - a^+) = g^-(a^-) + \frac{g^-(a^-) - g^-(b^-)}{a^- - b^-}(\theta^- - a^-). \tag{32}$$

The idea of finding a^+, b^+, a^-, b^- is to detect the sign switch of "some" function. Our framework can be divided into two stages:

- First, we find a sign switching interval $[\tau_i^+, \tau_{i-1}^+]$ such that $\nabla f(\tau_i^+) \leq 0$ and $\nabla f(\tau_{i-1}^+) \geq 0$. Formally, since $f(\tau_{2n_++1}^+) = \lim_{\theta\to-\infty} \nabla f(\theta) = -n_+$ and $f(\tau_0^+) = \lim_{\theta\to\infty} \nabla f(\theta) = n_-$, there must be a cross point $i \in \{1, \ldots, 1+2n_+\}$ such that $\nabla f(\tau_i^+) \leq 0$ and $\nabla f(\tau_{i-1}^+) \geq 0$. We set $a^+ = \tau_i^+$ and $b^+ = \tau_{i-1}^+$.

- Second, within $[a^+, b^+]$ where $c := \frac{-1}{n_+} - \tau_{i-1}^+$ and $d := \frac{-1}{n_+} - \tau_i^+$, we find a sign switching interval $[\tau_j^-, \tau_{j-1}^-]$ such that $\nabla f(\frac{-1}{n_+} - \tau_j^-) \geq 0$ and $\nabla f(\frac{-1}{n_+} - \tau_{j-1}^-) \leq 0$. Formally, suppose $\tau_p^- \geq \ldots \geq \tau_q^-$ are in $[c, d]$. Define a decreasing array by $\chi_0^- = d$, $\chi_1^- = \tau_q^-$, $\ldots$, $\chi_m^- = \tau_r^-$, $\chi_{m+1}^- = c$. Since $\nabla f(\frac{-1}{n_+} - \chi_0^-) = \nabla f(\tau_i^+) \leq 0$ and $\nabla f(\frac{-1}{n_+} - \chi_{m+1}^-) = \nabla f(\tau_{i-1}^+) \geq 0$, there must be a cross point $j \in \{1, \ldots, m+1\}$ such that $\nabla f(\frac{-1}{n_+} - \chi_j^-) \geq 0$ and $\nabla f(\frac{-1}{n_+} - \chi_{j-1}^-) \leq 0$. We set $a^- = \chi_j^-$ and $b^- = \chi_{j-1}^-$.

Then a^+, b^+, a^-, b^- are guaranteed to have the desired property. We formalize the idea in Algorithm 2. Since $\nabla f(\tau_0^+) = n_-$, it simply finds the smallest $i \geq 1$ such that $\nabla f(\tau_i^+) \leq 0$. Similarly, since $\nabla f(\frac{-1}{n_+} - \chi_0^-) \leq 0$, it just finds the smallest $j \geq 1$ such that $\nabla f(\frac{-1}{n_+} - \chi_j^-) \geq 0$.

Algorithm 2 requires an efficient way to evaluate g^+ and g^-. To this end, we design Algorithm 3 for computing g^+, while the case for g^- can be handled in sheer analogy. One key observation made from Algorithm 2 is that the location where g^+ is evaluated grows monotonically in the first and second stage. Therefore, our algorithm only needs to ensure that the *total* cost of evaluating g^+ in both stages is $O(n_+)$.

Our idea to achieve this is to maintain an anchor point $i \in \{1, \ldots, 2n_+\}$, together with the slope s and intercept/bias b of g^+ in the interval $[\tau_{i+1}^+, \tau_i^+]$. Whenever a query point θ is given, we move i

Algorithm 2: Find a^+, b^+, a^-, b^- to facilitate finding θ^+ and θ^- via solving (32).

Input: Arrays $\{\tau_i^+ : 0 \le i \le 2n_+ + 1\}$ and $\{\tau_j^- : 0 \le j \le 2n_- + 1\}$.
Output: a^+, b^+, a^-, b^- which satisfy the desired conditions.

for $i = 1, \ldots, 1 + 2n_+$ **do**
  $g_i^+ = g^+(\tau_i^+)$, $g_i^- = g^-\left(\frac{-1}{n_+} - \tau_i^+\right)$;
  if $g_i^+ - g_i^- \le 0$ **then** break;
Let $c = \frac{-1}{n_+} - \tau_{i-1}^+$, $d = \frac{-1}{n_+} - \tau_i^+$. Suppose $\tau_q^- \ge \ldots \ge \tau_r^-$ are in $[c, d]$;
Define a decreasing array $\chi_0^- = d$, $\chi_1^- = \tau_q^-, \ldots, \chi_m^- = \tau_r^-, \chi_{m+1}^- = c$ ($m = r - q + 1$).
for $j = 1, \ldots, m + 1$ **do**
  $g_j^+ = g^+\left(\frac{-1}{n_+} - \chi_j^-\right)$, $g_j^- = g^-(\chi_j^-)$;
  if $g_j^+ - g_j^- \ge 0$ **then** break;
return $a^+ = \tau_i^+$, $b^+ = \tau_{i-1}^+$, $a^- = \chi_j^-$, $b^- = \chi_{j-1}^-$;

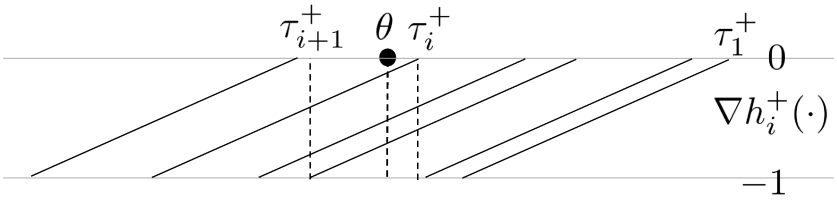


Figure 1: Example of evaluating g^+. At τ_i^+, $s = 3$ and $b = -2$. $g^+(\theta)$ can be computed by $-2 + 3(\theta - \tau_i^+)$. At τ_{i+1}^+, $s = 2$ and $b = -3$.

to the largest possible value such that $\theta \in (\tau_{i+1}^+, \tau_i^+]$ and compute $g^+(\theta)$ by $b + s(\theta - \tau_i^+)$. In other words, when $\theta > \tau_i^+$, we need to decrement i; and when $\theta \le \tau_{i+1}^+$ we increment i. An example is given in Figure 1.

The key operation here is how to update s and b when sliding i. We say i is a head if τ_i^+ corresponds to some a_k^+ (rather than $a_k^+ - \mu$). Otherwise we say i is a tail. We initialize by setting $i = 1$, $s = 1$ and $b = 0$. Then the update rule falls in four cases:

- Increment i and $i' = i + 1$ is a head: update by $s' = s + 1$, and $b' = b + s \cdot \frac{1}{\mu}(\tau_{i'}^+ - \tau_i^+)$.
- Increment i and $i' = i + 1$ is a tail: update by $s' = s - 1$ and $b' = b + s \cdot \frac{1}{\mu}(\tau_{i'}^+ - \tau_i^+)$.
- Decrement i to $i' = i - 1$, and i is a head: update by $s' = s - 1$ and $b' = b - s' \cdot \frac{1}{\mu}(\tau_i^+ - \tau_{i'}^+)$.
- Decrement i to $i' = i - 1$, and i is a tail: update by $s' = s + 1$ and $b' = b - s' \cdot \frac{1}{\mu}(\tau_i^+ - \tau_{i'}^+)$.

Since the query point grows monotonically, the anchor point i only needs to slide monotonically too. Hence the total computational cost in each stage is $O(n_+)$. It is also not hard to see that i can be initialized to $i = 2n_+$, with $s = 0$ and $b = -n_+$. Finally, suppose we store $\{a_i^+, a_i^+ - \mu\}$ by an array A with $A[2k-1] = a_k^+$ and $A[2k] = a_k^+ - \mu$, and the sorting is performed only in terms of the index of A. Then an index i of A is a head (corresponds to some a_k^+) if an only if i is odd.

Algorithm 3: Compute $g^+(\theta)$.

Input: A query point θ.
Output: $g^+(\theta)$.

Initialize: $i = 1$, $s = 1$, $b = 0$. Given $\{a_i\}$, do this only once at the first call of this function. Other initializations are also fine, for example, $i = 2n_+$, with $s = 0$ and $b = -n_+$.
while TRUE do
  if $\tau_i^+ \geq \theta$ **then**
    if $\tau_{i+1}^+ < \theta$ **then return** $b + s \cdot \frac{1}{\mu}(\theta - \tau_i^+)$;
    else
      $b = b + s \cdot \frac{1}{\mu}(\tau_{i+1}^+ - \tau_i^+)$, $i = i + 1$;
      if i is a head **then** $s = s + 1$; **else** $s = s - 1$;
  else
    if $i == 1$ **then return** 0;
    if i is a head **then** $s = s - 1$; **else** $s = s + 1$;
    $i = i - 1$, $b = b - s \cdot \frac{1}{\mu}(\tau_{i+1}^+ - \tau_i^+)$;

4.2 ROCArea Loss

For the ROCArea loss, given the optimal $\hat{\boldsymbol{\beta}}$ in (24) one can compute

$$\frac{\partial}{\partial \mathbf{w}} g_\mu^\star(A^\top \mathbf{w}) = \frac{1}{m} \sum_{i,j} \hat{\beta}_{ij}(\boldsymbol{x}_j - \boldsymbol{x}_i) = -\frac{1}{m}\left[\sum_{i \in \mathcal{P}} \boldsymbol{x}_i \underbrace{\left(\sum_{j \in \mathcal{N}} \hat{\beta}_{ij}\right)}_{:=\gamma_i} - \sum_{j \in \mathcal{N}} \boldsymbol{x}_j \underbrace{\left(\sum_{i \in \mathcal{P}} \hat{\beta}_{ij}\right)}_{:=\gamma_j}\right].$$

If we can efficiently compute all γ_i and γ_j, then the gradient can be computed in $O(np)$ time. Given $\hat{\beta}_{ij}$, a brute-force approach to compute γ_i and γ_j takes $O(m)$ time. We exploit the structure of the problem to reduce this cost to $O(n \log n)$, thus matching the complexity of the separation algorithm in Joachims (2005). Towards this end, we specialize (24) to ROCArea and write

$$g_\mu^\star(A^\top \mathbf{w}) = \max_{\boldsymbol{\beta}} \left(\frac{1}{m} \sum_{i,j} \beta_{ij} \mathbf{w}^\top (\boldsymbol{x}_j - \boldsymbol{x}_i) + \frac{1}{m} \sum_{i,j} \beta_{ij} - \frac{\mu}{2} \sum_{i,j} \beta_{ij}^2\right). \tag{33}$$

Since all β_{ij} are decoupled, their optimal value can be easily found:

$$\begin{aligned} \hat{\beta}_{ij} &= \text{median}\,(1, a_j - a_i, 0) \\ \text{where} \quad a_i &= \frac{1}{\mu m}\left(\mathbf{w}^\top \boldsymbol{x}_i - \frac{1}{2}\right), \text{ and } a_j = \frac{1}{\mu m}\left(\mathbf{w}^\top \boldsymbol{x}_j + \frac{1}{2}\right). \end{aligned} \tag{34}$$

Below we give a high level description of how γ_i for $i \in \mathcal{P}$ can be computed; the scheme for computing γ_j for $j \in \mathcal{N}$ is identical. We omit the details for brevity.

For a given i, suppose we can divide $\mathcal{N}$ into three sets $\mathcal{M}_i^+$, $\mathcal{M}_i$, and $\mathcal{M}_i^-$ such that

- $j \in \mathcal{M}_i^+ \implies 1 < a_j - a_i$, hence $\hat{\beta}_{ij} = 1$.

Algorithm 4: Compute γ_i, s_i, t_i for all $i \in \mathcal{P}$.

Input: Two arrays $\{a_i : i \in \mathcal{P}\}$ and $\{a_j : j \in \mathcal{N}\}$ sorted increasingly: $a_i \leq a_{i+1}$ and $a_j \leq a_{j+1}$.
Output: $\{\gamma_i, s_i, t_i : i \in \mathcal{P}\}$.

```
Initialize: s = t = 0, k = j = 0
for i in 1 to n_+ do
    while j < n_- AND a_{j+1} < a_i do
        j = j + 1; s = s - a_j; t = t - a_j^2
    while k < n_- AND a_{k+1} ≤ a_i + 1 do
        k = k + 1; s = s + a_k; t = t + a_k^2
    s_i = s, t_i = t, γ_i = (n_- - 1 - k) - (k - j)a_i + s
```

- $j \in \mathcal{M}_i \implies a_j - a_i \in [0,1]$, hence $\hat{\beta}_{ij} = a_j - a_i$.
- $j \in \mathcal{M}_i^- \implies a_j - a_i < 0$, hence $\hat{\beta}_{ij} = 0$.

Let $s_i = \sum_{j \in \mathcal{M}_i} a_j$. Then, clearly

$$\gamma_i = \sum_{j \in \mathcal{N}} \hat{\beta}_{ij} = |\mathcal{M}_i^+| + \sum_{j \in \mathcal{M}_i} a_j - |\mathcal{M}_i|\, a_i = |\mathcal{M}_i^+| + s_i - |\mathcal{M}_i|\, a_i.$$

Plugging the optimal $\hat{\beta}_{ij}$ into (33) and using (34), we can compute $g_\mu^\star(A^\top \mathbf{w})$ as

$$g_\mu^\star(A^\top \mathbf{w}) = \mu \sum_{ij} \hat{\beta}_{ij}(a_j - a_i) - \frac{\mu}{2} \sum_{ij} \hat{\beta}_{ij}^2 = \mu \left(\sum_{j \in \mathcal{N}} a_j \gamma_j - \sum_{i \in \mathcal{P}} a_i \gamma_i \right) - \frac{\mu}{2} \sum_{ij} \hat{\beta}_{ij}^2.$$

If we define $t_i := \sum_{j \in \mathcal{M}_i} a_j^2$, then $\sum_{ij} \hat{\beta}_{ij}^2$ can be efficiently computed via

$$\sum_j \hat{\beta}_{ij}^2 = \left|\mathcal{M}_i^+\right| + \sum_{j \in \mathcal{M}_i} (a_j - a_i)^2 = \left|\mathcal{M}_i^+\right| + \left|\mathcal{M}_i\right| a_i^2 - 2 a_i s_i + t_i, \quad \forall\, i \in \mathcal{P}.$$

In order to identify the sets $\mathcal{M}_i^+$, $\mathcal{M}_i$, $\mathcal{M}_i^-$, and compute s_i and t_i, we first sort both $\{a_i : i \in \mathcal{P}\}$ and $\{a_j : j \in \mathcal{N}\}$. We then walk down the sorted lists to identify for each i the first and last indices j such that $a_j - a_i \in [0,1]$. This is very similar to the algorithm used to merge two sorted lists, and takes $O(n_- + n_+) = O(n)$ time and space. The rest of the operations for computing γ_i can be performed in $O(1)$ time with some straightforward book-keeping. The whole algorithm is presented in Algorithm 4 and its overall complexity is dominated by the complexity of sorting the two lists, which is $O(n \log n)$.

5. Empirical Evaluation

We used 24 publicly available data sets and focused our study on two aspects: the reduction in objective value as a function of CPU time, and generalization performance. Since the objective functions we are minimizing are strongly convex, all optimizers will converge the same solution (within numerical precision) and produce the same generalization performance eventually.

Therefore, what we are specifically interested in is the rate at which the objective function and generalization performance decreases. We will refer to our algorithm as SMS, for **S**moothing for **M**ultivariate **S**cores. As our comparator we use BMRM. We downloaded BMRM from `http://users.rsise.anu.edu.au/~chteo/BMRM.html`, and used the default settings in all our experiments.

5.1 Data Sets

Table 1 summarizes the data sets used in our experiments. `adult9`, `astro-ph`, `news20`, `real-sim`, `reuters-c11`, `reuters-ccat` are from the same source as in Hsieh et al. (2008). `aut-avn` is from Andrew McCallum's home page,[6] `covertype` is from the UCI repository (Frank and Asuncion, 2010), `worm` is from Franc and Sonnenburg (2008), `kdd99` is from KDD Cup 1999,[7] while `web8`, `webspam-u`, `webspam-t`,[8] as well as the `kdda` and `kddb`[9] are from the LibSVM binary data collection.[10] The `alpha`, `beta`, `delta`, `dna`, `epsilon`, `fd`, `gamma`, `ocr`, and `zeta` data sets were all obtained from the Pascal Large Scale Learning Workshop website (Sonnenburg et al., 2008). Since the features of `dna` are suboptimal compared with the string kernels used by Sonnenburg and Franc (2010), we downloaded their original DNA sequence (`dna_string`).[11] In particular, we used a weighted degree kernel and two weighted spectrum kernels (one at position 1-59 and one at 62-141, corresponding to the left and right of the splice site respectively), all with degree 8. Following Sonnenburg and Franc (2010), we used the dense explicit feature representations of the kernels, which amounted to 12,670,100 features.

For `dna_string` and the data sets which were also used by Teo et al. (2010) (indicated by an asterisk in Table 1), we used the training test split provided by them. For the remaining data sets we used 80% of the labeled data for training and the remaining 20% for testing. In all cases, we added a constant feature as a bias.

5.2 Optimization Algorithms

Optimizing the smooth objective function $J_\mu(\mathbf{w})$ using the optimization scheme described in Nesterov (2005) requires estimating the Lipschitz constant of the gradient of the $g^\star_\mu(A^\top \mathbf{w})$. Although it can be automatically tuned by, for example, Beck and Teboulle (2009), extra cost is incurred which slows down the optimization empirically. Therefore, we chose to optimize our smooth objective function using L-BFGS, a widely used quasi-Newton solver (Nocedal and Wright, 1999). We implemented our smoothed loss using PETSc[12] and TAO,[13] which allow the *efficient* use of large-scale parallel linear algebra. We used the Limited Memory Variable Metric (`lmvm`) variant of

6. The data set can be found at `http://www.cs.umass.edu/~mccallum/data/sraa.tar.gz`.
7. The data set can be found at `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.
8. `webspam-u` is the webspam-unigram and `webspam-t` is the webspam-trigram data set. Original data set can be found at `http://www.cc.gatech.edu/projects/doi/WebbSpamCorpus.html`.
9. These data sets were derived from KDD CUP 2010. `kdda` is the first problem algebra_2008_2009 and `kddb` is the second problem bridge_to_algebra_2008_2009.
10. The data set can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.
11. The data set can be found at `http://sonnenburgs.de/soeren/projects/coffin/splice_data.tar.xz`.
12. The software can be found at `http://www.mcs.anl.gov/petsc/petsc-2/index.html`. We compiled an optimized version of PETSc (`--with-debugging=0`) and enabled 64-bit index to run for large data sets such as `dna` and `ocr`.
13. TAO can be found at `http://www.mcs.anl.gov/research/projects/tao`.

data set	n	p	$s(\%)$	$n_+ : n_-$	data set	n	p	$s(\%)$	$n_+ : n_-$
adult9*	48,842	123	11.3	0.32	alpha	500,000	500	100	1.00
astro-ph*	94,856	99,757	0.08	0.31	aut-avn*	71,066	20,707	0.25	1.84
beta	500,000	500	100	1.00	covertype*	581,012	6.27 M	22.22	0.57
delta	500,000	500	100	1.00	dna	50.00 M	800	25	3e−3
dna_string	50.00 M	12.7 M	*n.a.*	3e−3	epsilon	500,000	2000	100	1.00
fd	625,880	900	100	0.09	gamma	500,000	500	100	1.00
kdd99*	5.21 M	127	12.86	4.04	kdda	8.92 M	20.22 M	2e−4	5.80
kddb	20.01 M	29.89 M	1e−4	6.18	news20*	19,954	7.26 M	0.033	1.00
ocr	3.50 M	1156	100	0.96	real-sim*	72,201	2.97 M	0.25	0.44
reuters-c11*	804,414	1.76 M	0.16	0.03	reuters-ccat*	804,414	1.76 M	0.16	0.90
web8*	59,245	300	4.24	0.03	webspam-t	350,000	16.61 M	0.022	1.54
webspam-u	350,000	254	33.8	1.54	worm*	1.03 M	804	25	0.06
zeta	500,000	800.4 M	100	1.00					

Table 1: Summary of the data sets used in our experiments. n is the total number of examples, p is the number of features, s is the feature density (% of features that are non-zero), and $n_+ : n_-$ is the ratio of the number of positive vs negative examples. M denotes a million. A data set is marked with an asterisk if it is also used by Teo et al. (2010).

L-BFGS which is implemented in TAO. Open-source code, as well as all the scripts used to run our experiments are available for download from Zhang et al. (2012).

5.3 Implementation And Hardware

Both BMRM and SMS are implemented in C++. Since dna and ocr require more memory than was available on a single machine, we ran both BMRM and SMS with 16 cores spread across 16 machines for these two data sets. As dna_string employs a large number of dense features, it would cost a prohibitively large amount of memory. So following Sonnenburg and Franc (2010), we (repeatedly) computed the explicit features whenever it is multiplied with the weight vector $\boldsymbol{w}$. This entails demanding cost in computation, and therefore we used 32 cores. The other data sets were trained with a single core. All experiments were conducted on the Rossmann computing cluster at Purdue University,[14] where each node has two 2.1 GHz 12-core AMD 6172 processors with 48 GB physical memory per node.

5.4 Experimental Setup

We used $\lambda \in \{10^{-2}, 10^{-4}, 10^{-6}\}$ to test the performance of BMRM and SMS under a reasonably wide range of λ. In line with Chapelle (2007), we observed that solutions with higher accuracy do not improve the generalization performance, and setting $\varepsilon = 0.001$ was often sufficient. Accordingly, we could set $\mu = \varepsilon / D$ as suggested by the uniform deviation bound (13). However, this estimate is very conservative because in (13) we used D as an upper bound on the prox-function, while in practice the quality of the approximation depends on the value of the prox-function *around the optimum*. On the other hand, a larger value of μ proffers more strong convexity in g_μ, which makes

14. The cluster information is at http://www.rcac.purdue.edu/userinfo/resources/rossmann.

data set	PRBEP		ROCArea		data set	PRBEP		ROCArea	
	BMRM	SMS	BMRM	SMS		BMRM	SMS	BMRM	SMS
adult9	43.32	0.40	1.13	0.19	alpha	3595	183.6	802.3	135.4
astro-ph	119.0	48.8	11.8	4.02	aut-avn	2.22	1.77	0.92	0.61
beta	1710	24.4	644.0	46.5	covertype	36.24	7.33	42.73	5.71
delta	*n.a.*	36.03	*n.a.*	44.20	dna	*n.a.*	*n.a.*	12140	561.5
dna_string	*n.a.*	*n.a.*	>30000	27514	epsilon	1716.1	804.5	1182.4	718.4
fd	943.2	154.0	1329.8	327.5	gamma	14612	26.5	8902.0	47.5
kdd99	63.5	152.1	204.9	31.0	kdda	*n.a.*	19204	*n.a.*	6108
kddb	*n.a.*	19363	*n.a.*	10592	news20	6921.2	429.5	1.51	8.8
ocr	893.1	105.9	921.6	98.0	real-sim	1.19	3.52	8.50	2.16
reuters-c11	45.5	46.1	28.7	10.4	reuters-ccat	328.2	162.8	145.0	31.9
web8	4.35	1.26	10.4	0.59	webspam-t	17271	5681.2	5751.9	1805.4
webspam-u	57.8	29.5	58.8	8.47	worm	2058.6	30.4	1901.4	20.2
zeta	679.1	293.4	579.8	386.7					

Table 2: Summary of the wall-clock time taken by BMRM and SMS to find a solution which is within 1% of the minimal regularized risk. All numbers are in seconds, and are obtained from the λ which yields the optimal test performance. There are a few *n.a.* which are explained in Section 5.5.

g^*_μ smoother and hence J_μ easier to optimize. Empirically, we observed that setting $\mu = 10 \cdot \varepsilon / D$ allows SMS to find an ε accurate solution in all cases.

5.5 Results

In Figures 2 to Figure 26 we plot the evolution of the non-smooth objective $J(\mathbf{w})$ (1) as a function of CPU time as well as the generalization performance on the test set for both BMRM and SMS. For a fair comparison the time taken to load the data is excluded for both algorithms. Note that the x-axis on the plots scales logarithmically. The figures also indicate the generalization performance upon convergence, and the best performance among all values of λ is highlighted in boldface.

Across a variety of data sets, for both PRBEP and ROCArea problems, SMS reduces the non-smooth objective $J(\mathbf{w})$ significantly faster than BMRM for all values of λ. For large values of λ (e.g., $\lambda = 10^{-2}$) the difference in performance is not dramatic, but the test performance of both methods is always inferior. This is because most of the data sets we use in our experiments contain a large number of training examples, and hence require very mild regularization. For instance, the optimal generalization performance is obtained by $\lambda = 10^{-6}$ in 40 out of the 48 experiments, by $\lambda = 10^{-4}$ in 14 experiments, and $\lambda = 10^{-2}$ in only 2 experiments (there are 8 ties). For low values of λ we find that BMRM which sports a $O\left(\frac{1}{\lambda\varepsilon}\right)$ rate of convergence slows down significantly. In contrast, SMS is not only able to gracefully handle small values of λ but actually converges significantly faster. To summarize the results, we present in Table 2 the wall-clock time that BMRM and SMS take to minimize the regularized risk to 1% relative accuracy, that is, find $\boldsymbol{w}^*$ such that $J(\boldsymbol{w}^*) \leq 1.01 \cdot \min_{\boldsymbol{w}} J(\boldsymbol{w})$. In many cases, SMS is 10 to 100 times faster than BMRM in achieving the same relative accuracy. Also see the detailed discussion of the results below.

We also studied the evolution of the PRBEP and ROCArea performance on the test data. It is clear that *in general* a lower value in $J(\boldsymbol{w})$ results in better test performance, which confirms the effectiveness of the SVM model. This correlation also translates the faster reduction of the objective value into the faster improvement of the generalization performance. On most data sets, the intermediate models output by SMS achieve comparable (or better) PRBEP and ROCArea scores in time orders of magnitude faster than those generated by BMRM. These results can be explained as follows: Since BMRM is a CPM, only the duality gap is guaranteed to decrease monotonically. This does not translate to reduction in the primal objective value. Consequently, we observe that sometimes the primal objective and generalization performance of BMRM seemingly gets "stuck" for many iterations. In contrast, our smoothing scheme uses L-BFGS which enforces a descent condition, and hence we observe a monotonic decrease in objective value at every iteration.[15]

5.5.1 DETAILED DISCUSSION OF THE RESULTS

On the `adult9` data set (Figure 2), the value of λ that yields the optimal test PRBEP is 10^{-6} (panels (c) and (f)). 100 seconds are needed by BMRM to complete training (i.e., minimize the regularized risk $J(\boldsymbol{w})$ to the desired accuracy and achieve stably optimal test PRBEP), while SMS needs only 0.5 seconds. For ROCArea, the optimal λ is 10^{-4} (panels (h) and (k)), in which case it takes SMS just 0.3 seconds for training, compared with the 2 seconds required by BMRM.

On the `covertype` data set (Figure 7), the optimal λ is 10^{-6} for both PRBEP and ROCArea. In PRBEP-problem (panels (c) and (f)), SMS is able to find the optimal regularized risk in 15 seconds, while BMRM takes about 55 seconds. The optimal test PRBEP is stably achieved by SMS in 10 seconds, while that costs BMRM about 7 times longer (80 seconds). The superiority on ROCArea is even more evident (panels (i) and (l)). SMS takes only 10 seconds to complete training for ROCArea, while BMRM requires 60 seconds. In addition, between 1 second and 10 seconds, BMRM seems to be stuck by a plateau where it is collecting cutting planes to construct a refined model before being able to reduce the regularized risk further.

On the `delta` data set (Figure 8), the optimal λ is again 10^{-6} for both PRBEP and ROCArea. In both cases, BMRM fails to converge to the optimal solution within 9 hours' runtime and is terminated (panels (c), (f), (i), (l), and the "*n.a.*" for `delta` in Table 2). In contrast, SMS optimizes the objective and attains the optimal test PRBEP and ROCArea in only 40 seconds. If we also pay attention to the sub-optimal values of λ, that is, 10^{-2} and 10^{-4}, it is clear that SMS again optimizes the regularized risk and test performance much faster. In fact, the time required by SMS to output the optimal test performance is about only 1% of that of BMRM (panels (d), (e), (j), (k)). Similar phenomena are observed on the `gamma` data set (Figure 13).

`ocr` is another data set where SMS demonstrates a clear edge over BMRM (Figure 18). The optimal test PRBEP and ROCArea are both attained at $\lambda = 10^{-6}$. Training for PRBEP costs BMRM 1000 seconds, while SMS takes only 10% as much time (panels (c) and (f)). As for ROCArea, the training time required by BMRM is 10 times of that of SMS. Similar striking advantage of SMS is also manifested on the `fd` data set (Figure 12). As one can clearly read from panels (c), (f), (i), and (l), SMS completes training for PRBEP in only 150 seconds, 15% of the time needed by BMRM. On ROCArea, training costs SMS 200 seconds, as opposed to 800 seconds for BMRM.

15. L-BFGS enforces monotonic decrease only on the smoothed objective $J_\mu(\boldsymbol{w})$, while what we are plotting is $J(\boldsymbol{w})$. However, empirically we observe that $J(\boldsymbol{w})$ is always monotonically decreasing.

On the two reuters data sets, the advantage of SMS is also manifested. On reuters-c11 (Figure 20), the model achieves about 50% test PRBEP (panel (f)). Since the data set is imbalanced with the ratio of the number of positive versus negative examples being 0.03, this performance is probably not vacuous. SMS and BMRM perform similarly in minimizing the regularized risk (panel (c)), while the test PRBEP fluctuates too much making it hard to tell which method is better (panel (f)). Regarding ROCArea, SMS is clearly faster in reducing the regularized risk (panel (i)), but in terms of test ROCArea it is only slightly faster than BMRM (panel (l)). The advantage of SMS is much clearer on the reuters-ccat data set (Figure 21), whose class ratio is 0.9. With $\lambda = 10^{-6}$ which yields the optimal PRBEP, SMS completes training in about 110 seconds, while BMRM takes 400 seconds (panel (c) and (f)). For ROCArea, the optimal λ is again 10^{-6} and 30 seconds are needed by SMS to complete training, compared with the 100 seconds required by BMRM (panels (i) and (l)).

Let us look at another pair of similar data sets: webspam-u and webspam-t. On webspam-u (Figure 24), SMS demonstrates clear superiority over BMRM. The optimal test PRBEP and ROCArea are both achieved at $\lambda = 10^{-6}$, which outperforms the result of $\lambda = 10^{-4}$ by 0.72% and 0.73% respectively (panels (f) and (l)). As far as ROCArea is concerned, SMS completed training in 9 seconds, while BMRM takes 30 seconds (panels (i) and (l)). In the PRBEP-problem (panels (c) and (f)), training takes 250 seconds for SMS, while the time cost for BMRM is doubled (500 seconds). Very similar results are observed on webspam-t (Figure 23) which differs from webspam-u by using a trigram class of features. On PRBEP, the regularized risk is minimized by SMS in 6,500 seconds (panels (c)), while BMRM requires approximately three times more CPU time (20,000 seconds). The optimal test PRBEP is achieved by SMS in 2,000 seconds (panel (f)), while BMRM takes double as much time (4,000 seconds). Similarly on ROCArea (panels (i) and (l)), SMS is three times more efficient than BMRM, taking 2,000 seconds to minimize the regularized risk (versus 7,000 seconds by BMRM), and 7,000 seconds to optimize the test ROCArea (versus 25,000 seconds by BMRM).

In the PRBEP-problem for kdda and kddb, BMRM is stuck half way because its inner solver for quadratic programming does not converge even after a large number of iterations, and is terminated after running for 9 hours. Hence we mark "*n.a.*" in Table 2. We tried both the QR-LOQO and the Dai_Fletcher solvers provided by BMRM, but they both got stuck. However, SMS had no trouble solving the PRBEP-problem efficiently. Turning our attention to the ROCArea-problems, the optimal λ is 10^{-6} for both kdda and kddb. Here BMRM optimizes the objective very slowly and cannot get close to the optimal solution within 9 hours, hence terminated. SMS again solves the ROCArea-problems efficiently. Note when λ is large (10^{-4} and 10^{-2}), BMRM does manage to find the optimal solution for the ROCArea-problems within 9 hours, but SMS completes training in less than a quarter of the time taken by BMRM.

As is to be expected in such extensive empirical evaluation, there are some anomalies. On the dna data set (Figure 9), the ROCArea is learned very well (panels (j) to (l)), whereas the test PRBEP is pretty poor (panels (d) to (f)). We have enforced the accuracy of regularized risk minimization to very high: 10^{-5}, but in panels (a) to (c), the objective value at convergence is still just slightly below 1, a value which is trivially attained by $\boldsymbol{w} = \mathbf{0}$. Hence we mark "*n.a.*" in Table 2. We tried with smaller λ (e.g., 10^{-8} and 10^{-10}) and got similar test PRBEP. It is noteworthy that dna is the most skewed data set used in our experiment, with 333 times more negatives examples than positive examples. BMRM also struggles on this data set, which leads us to believe that some other learning models may be needed. Similar behavior is also observed on the dna_string data set, which attains

significantly higher test PRBEP and ROCArea than dna does, confirming the superiority of string kernels.

On the beta data set (Figure 6), the optimal PRBEP and ROCArea are around 50% (panels (f) and (l)). Since this data set contains the same number of positive and negative examples, a random labeling of the test data will produce 50% PRBEP and ROCArea in expectation. So for this data set, other modeling methods may be needed as well.

On kdd99 (Figure 14), the objective value is not well correlated with the test performance. For example, although SMS is clearly faster than BMRM in reducing the regularized risk for ROCArea (panels (g) to (i)), the test ROCArea is improved at similar rates by the two algorithms. In fact, the test ROCArea of SMS in panel (l) is even worse. Likewise, SMS and BMRM perform similarly in reducing the regularized risk for PRBEP as shown in panels (a) to (c), but irregular bumps in the test PRBEP are observed on BMRM (panels (d) to (f)). Notice that at convergence, both algorithms still match closely in the test performance.

5.5.2 THE INFLUENCE OF μ

Figures 27 to 29 show the performance of SMS under different values of μ with λ fixed to 10^{-6}. The complete results on all the 24 data sets are available at Zhang et al. (2012).

We tried $\mu = c_\mu \cdot \varepsilon / D$ where $c_\mu \in \{1000, 100, 10\}$. It is clear that $c_\mu = 10$ can always find an ε accurate solution in practice. This guarantee fails to hold when c_μ is too large (e.g., 1000), although in this case SMS sometimes converges faster and may also generalize well on test data. $c_\mu = 100$ often performs very similarly to $c_\mu = 10$.

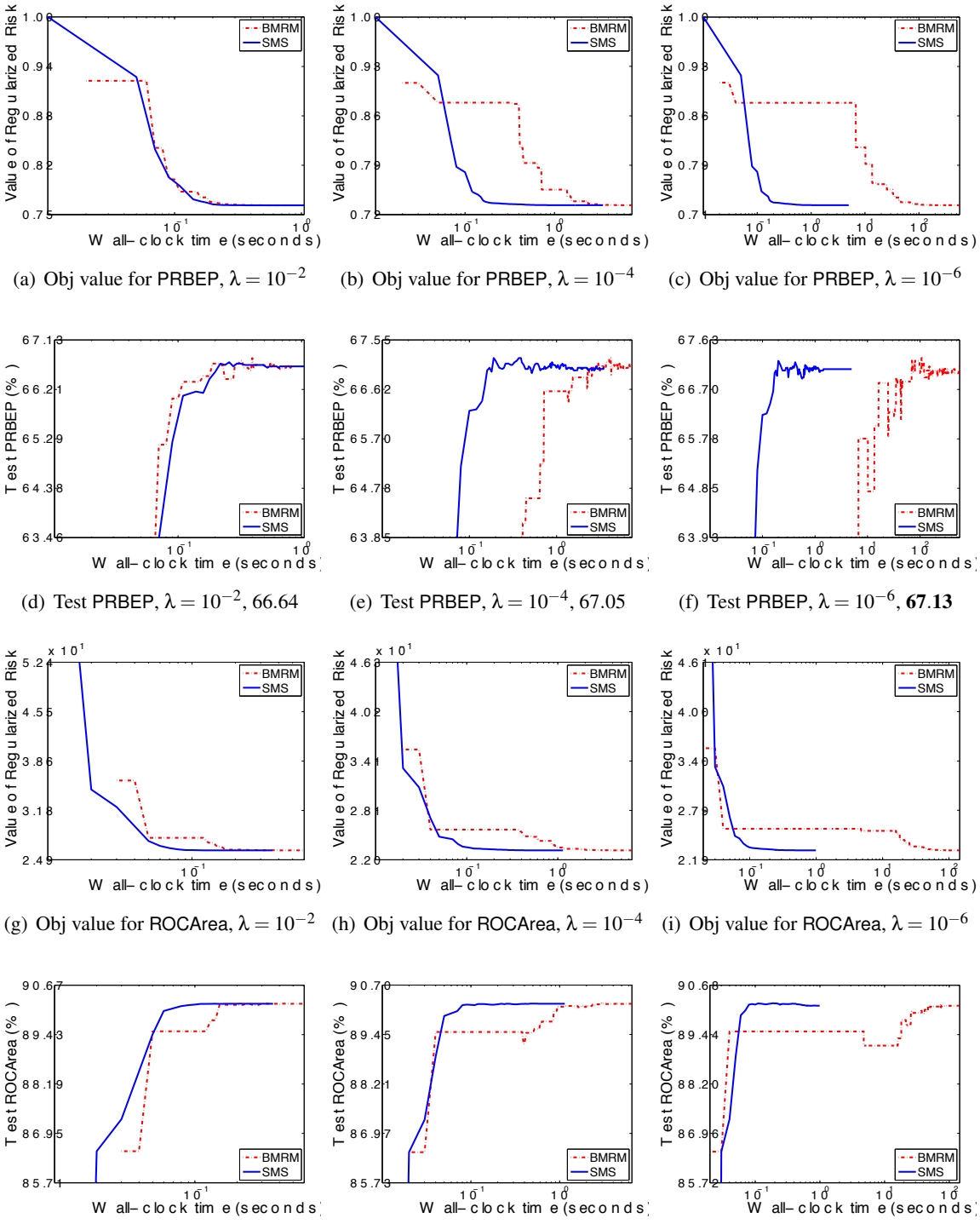
5.5.3 COMPARISON WITH ACCELERATED GRADIENT METHOD

Motivated by the theoretical derivation of the rates of convergence, we tested the performance of Nesterov's accelerated gradient method (AGM) on our smooth objective. In particular, we used the scheme in Nesterov (2007) which adaptively estimates the Lipschitz constant of the gradient. From the results in Figure 30 to 32 where $\lambda = 10^{-6}$ and $c_\mu = 10$, it is clear that AGM converges much slower than L-BFGS. This is not surprising because L-BFGS, which is a second order method, can estimate and use the curvature of the unconstrained and smooth objective. The full set of results on all data sets are available at Zhang et al. (2012).

6. Conclusion And Discussion

The non-smoothness of the loss function is an important consideration for algorithms which employ the kernel trick (Schölkopf and Smola, 2002). This is because such algorithms typically operate in the dual, and the non-smooth losses lead to sparse dual solutions. In many applications such as natural language processing, the kernel trick is not needed because the input data is sufficiently high dimensional. However, now we need to tackle a non-smooth optimization problem in the primal. In this paper we proposed efficient smoothing techniques to approximate the non-smooth function. When combined with a smooth optimization algorithm, our technique outperforms state-of-the-art non-smooth optimization algorithms for multivariate performance scores not only in terms of CPU time but also in terms of generalization performance.

It is also worthwhile emphasizing here that optimization is a means to an end in Machine Learning, and smoothing is not the right approach for every non-smooth problem. For example, although



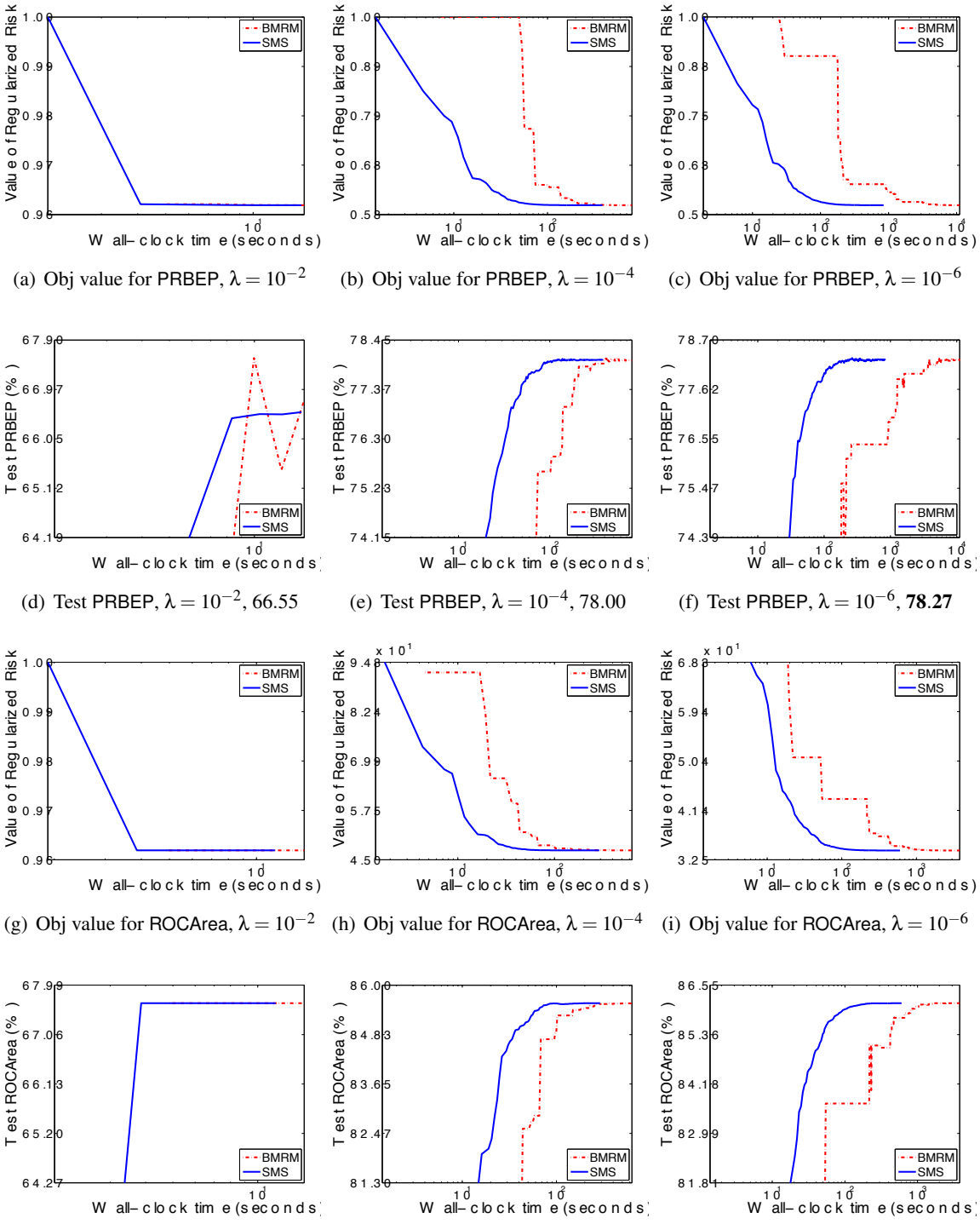
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 66.64 (e) Test PRBEP, $\lambda = 10^{-4}$, 67.05 (f) Test PRBEP, $\lambda = 10^{-6}$, **67.13**

(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 90.21 (k) Test ROCArea, $\lambda = 10^{-4}$, **90.23** (l) Test ROCArea, $\lambda = 10^{-6}$, 90.17

Figure 2: Results for `adult9`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



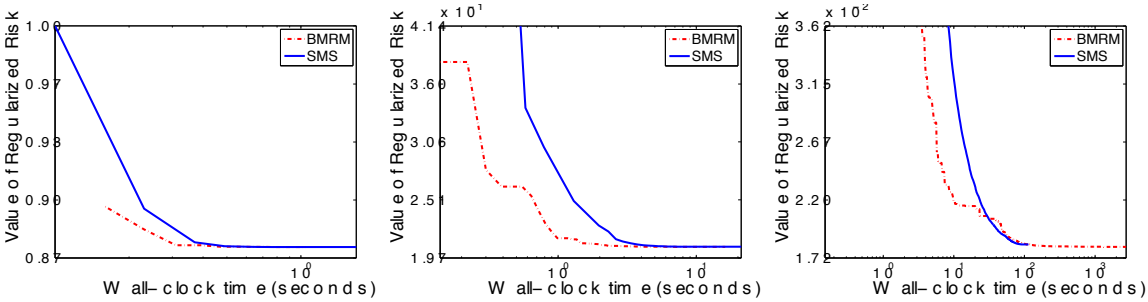
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 66.55 (e) Test PRBEP, $\lambda = 10^{-4}$, 78.00 (f) Test PRBEP, $\lambda = 10^{-6}$, **78.27**

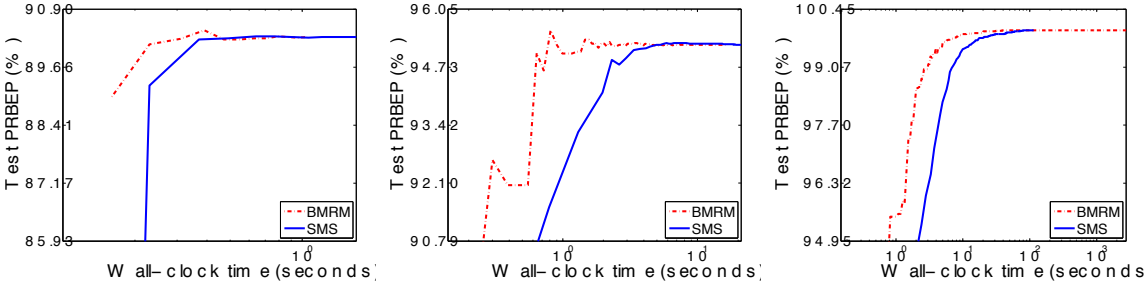
(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 61.84 (k) Test ROCArea, $\lambda = 10^{-4}$, 85.57 (l) Test ROCArea, $\lambda = 10^{-6}$, **86.12**

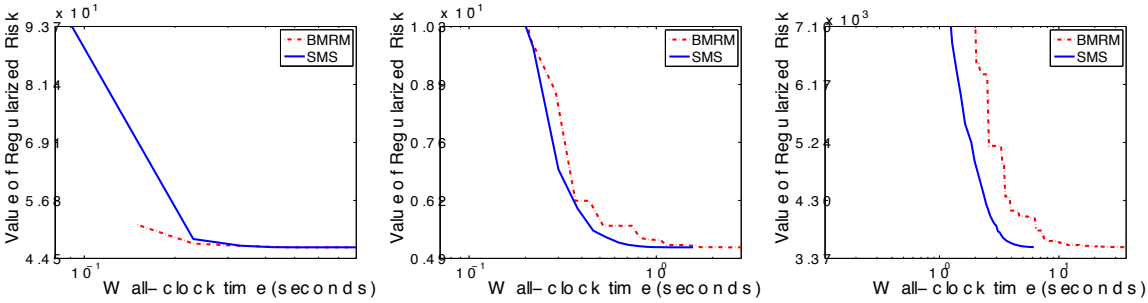
Figure 3: Results for `alpha`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



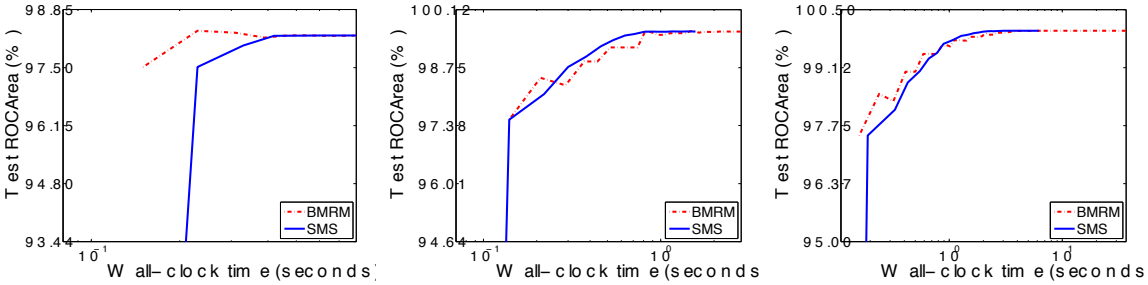
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 90.30 (e) Test PRBEP, $\lambda = 10^{-4}$, 95.24 (f) Test PRBEP, $\lambda = 10^{-6}$, **99.95**

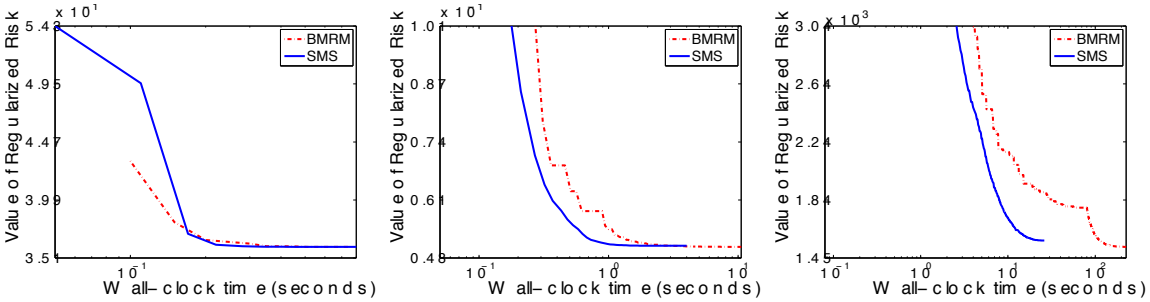


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

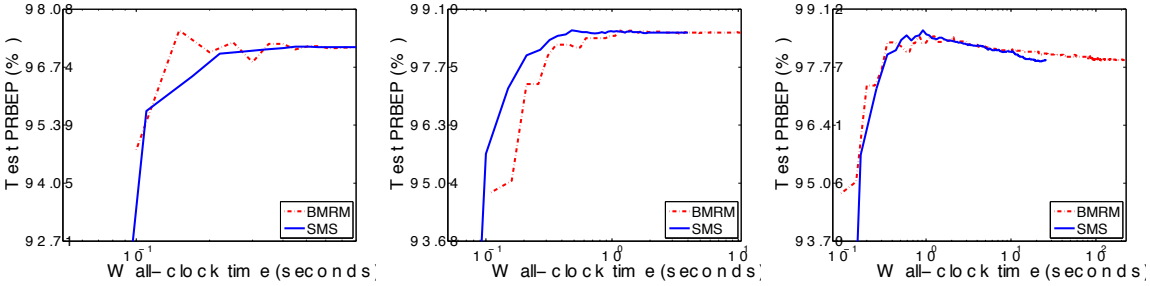


(j) Test ROCArea, $\lambda = 10^{-2}$, 98.25 (k) Test ROCArea, $\lambda = 10^{-4}$, 99.60 (l) Test ROCArea, $\lambda = 10^{-6}$, **100.00**

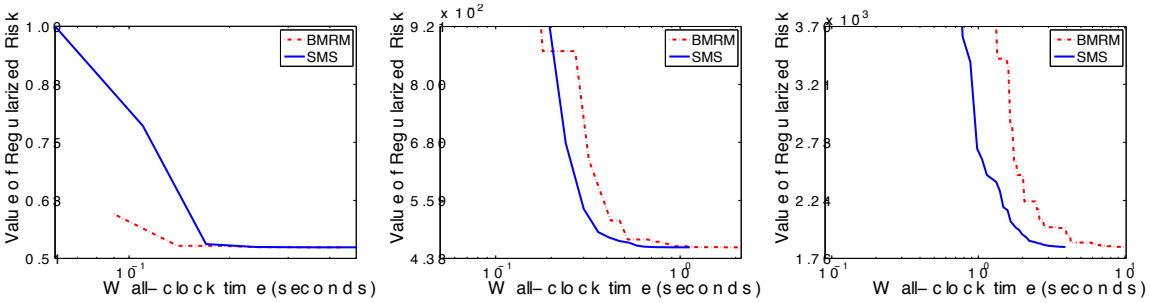
Figure 4: Results for `astro-ph`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



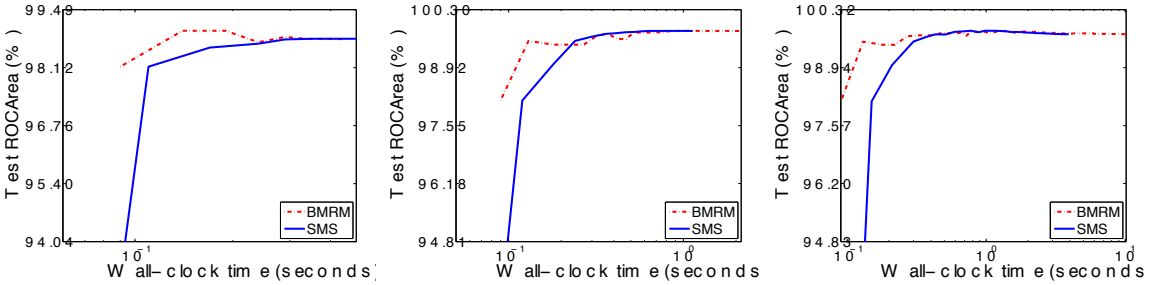
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 97.20 (e) Test PRBEP, $\lambda = 10^{-4}$, **98.56** (f) Test PRBEP, $\lambda = 10^{-6}$, 97.93

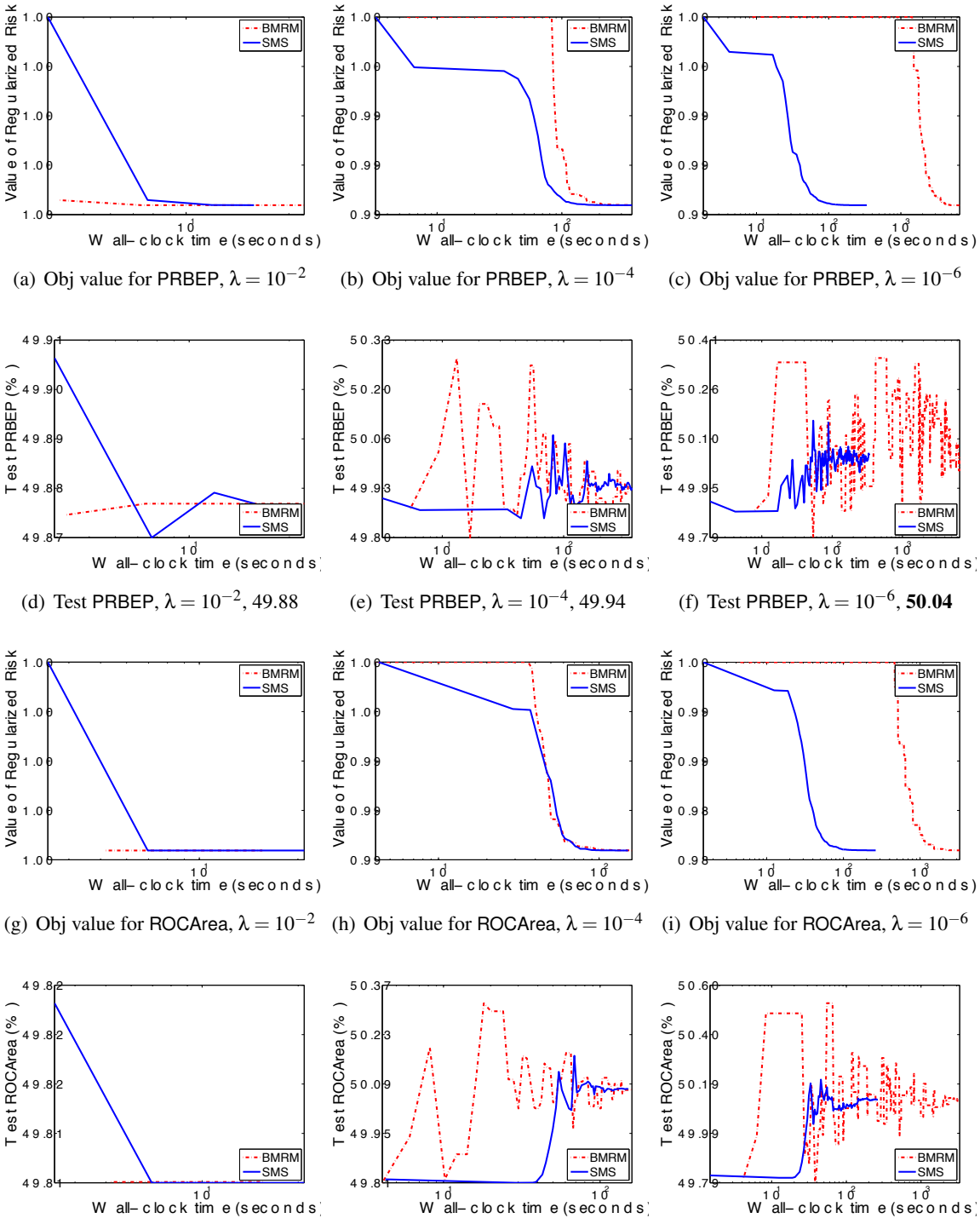


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$



(j) Test ROCArea, $\lambda = 10^{-2}$, 98.80 (k) Test ROCArea, $\lambda = 10^{-4}$, **99.79** (l) Test ROCArea, $\lambda = 10^{-6}$, 99.73

Figure 5: Results for `aut-avn`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



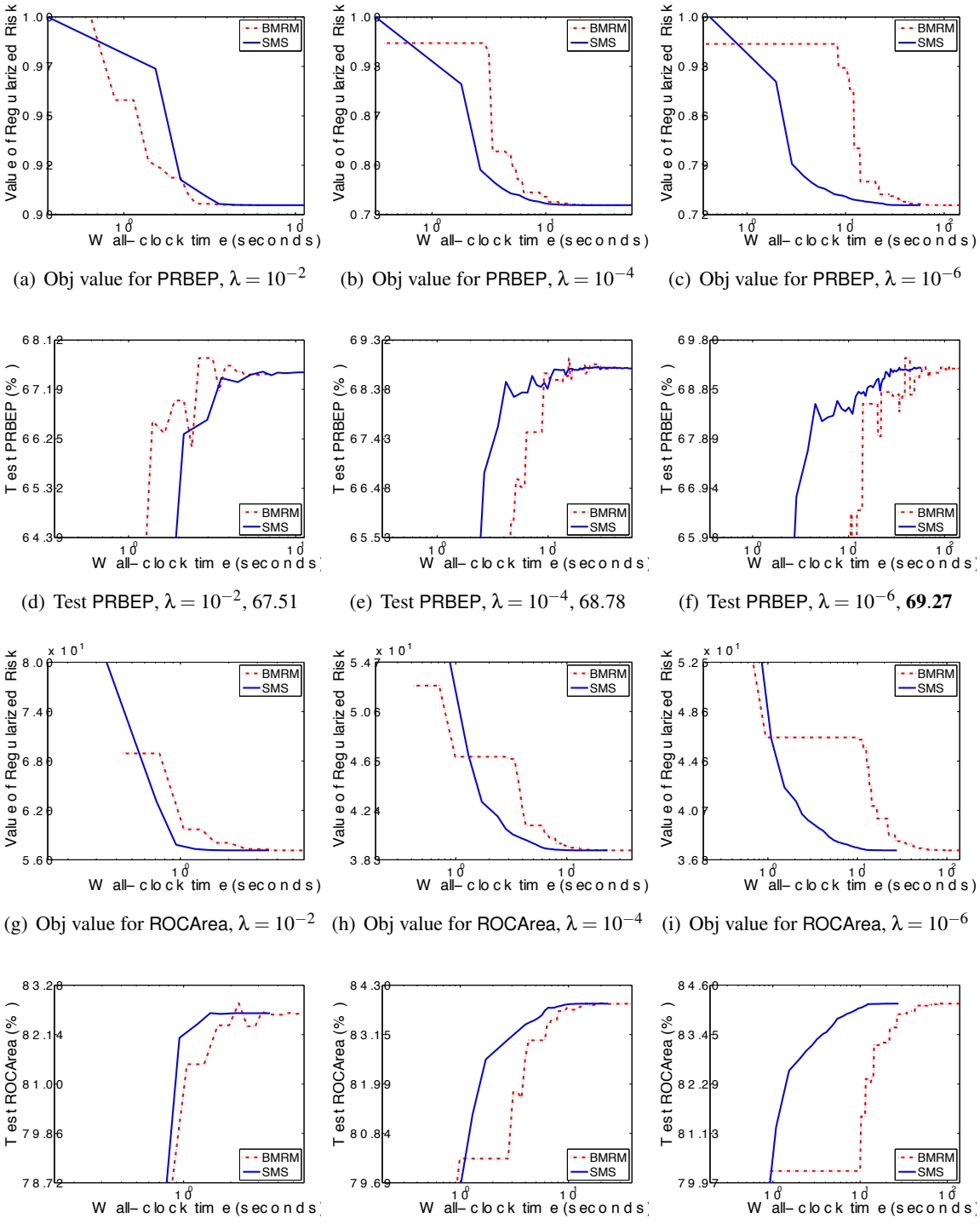
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 49.88 (e) Test PRBEP, $\lambda = 10^{-4}$, 49.94 (f) Test PRBEP, $\lambda = 10^{-6}$, **50.04**

(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 49.81 (k) Test ROCArea, $\lambda = 10^{-4}$, 50.07 (l) Test ROCArea, $\lambda = 10^{-6}$, **50.13**

Figure 6: Results for `beta`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



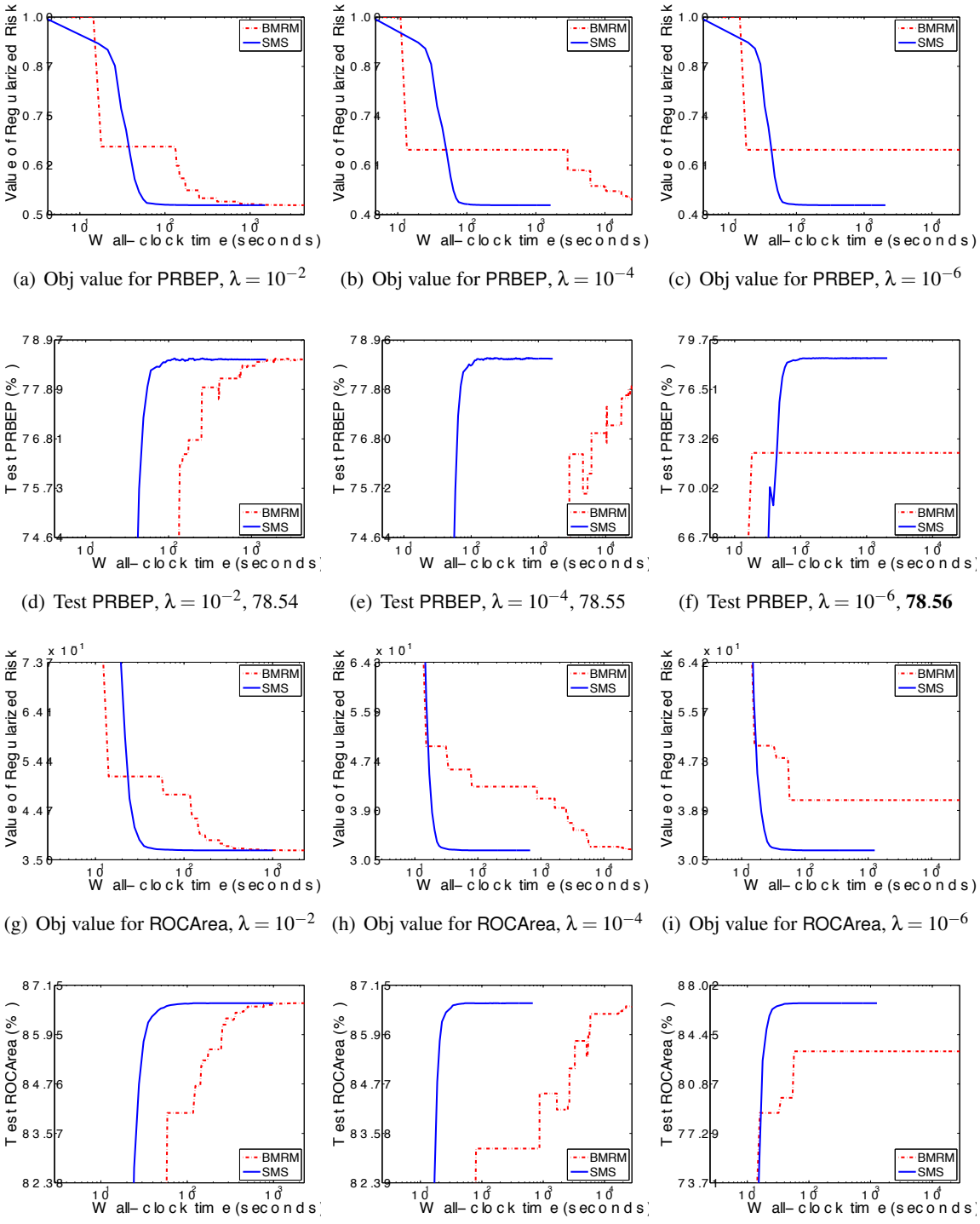
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 67.51 (e) Test PRBEP, $\lambda = 10^{-4}$, 68.78 (f) Test PRBEP, $\lambda = 10^{-6}$, **69.27**

(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 82.64 (k) Test ROCArea, $\lambda = 10^{-4}$, 83.87 (l) Test ROCArea, $\lambda = 10^{-6}$, **84.17**

Figure 7: Results for `covertype`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



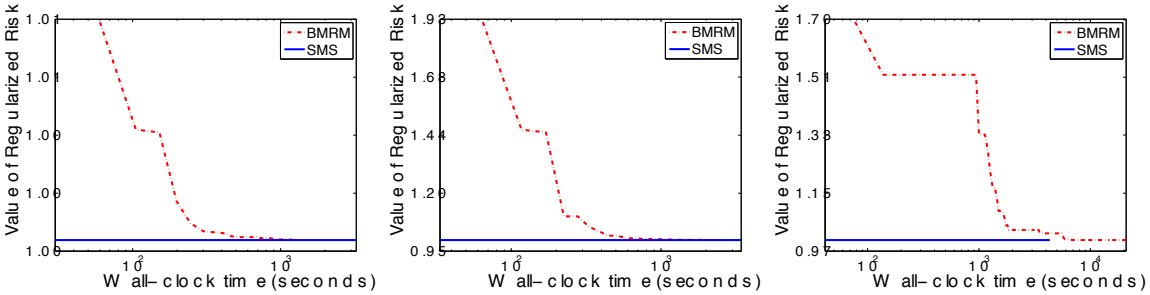
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 78.54 (e) Test PRBEP, $\lambda = 10^{-4}$, 78.55 (f) Test PRBEP, $\lambda = 10^{-6}$, **78.56**

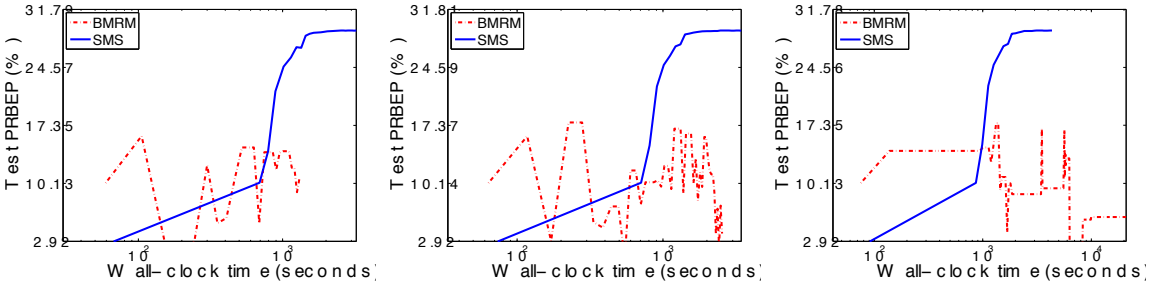
(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 86.71 (k) Test ROCArea, $\lambda = 10^{-4}$, **86.72** (l) Test ROCArea, $\lambda = 10^{-6}$, **86.72**

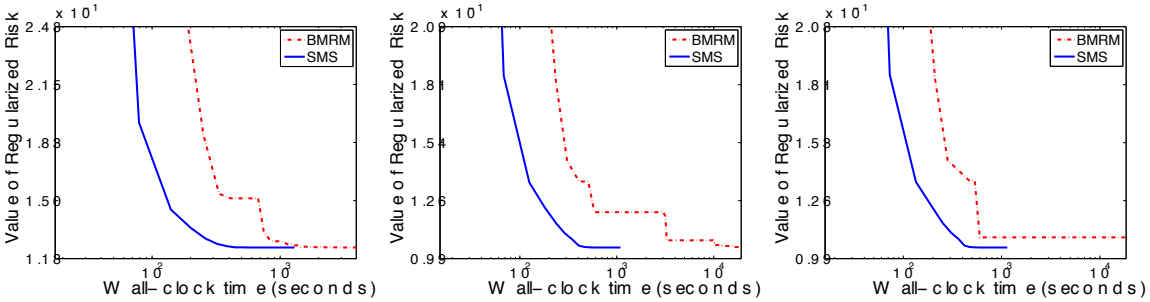
Figure 8: Results for `delta`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



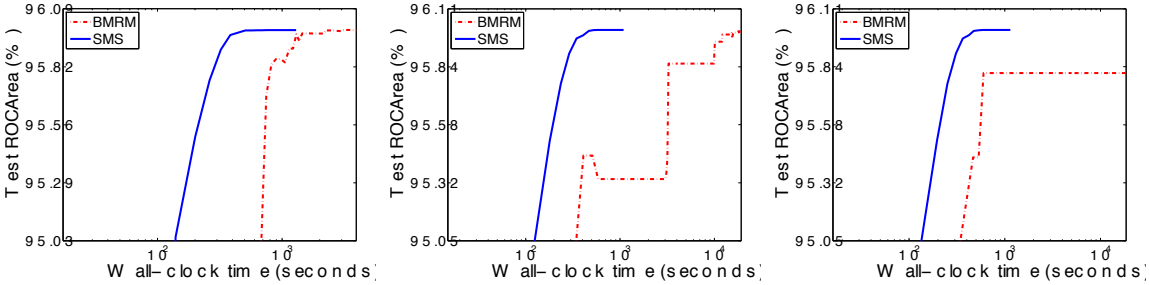
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, **29.14** (e) Test PRBEP, $\lambda = 10^{-4}$, **29.14** (f) Test PRBEP, $\lambda = 10^{-6}$, **29.14**

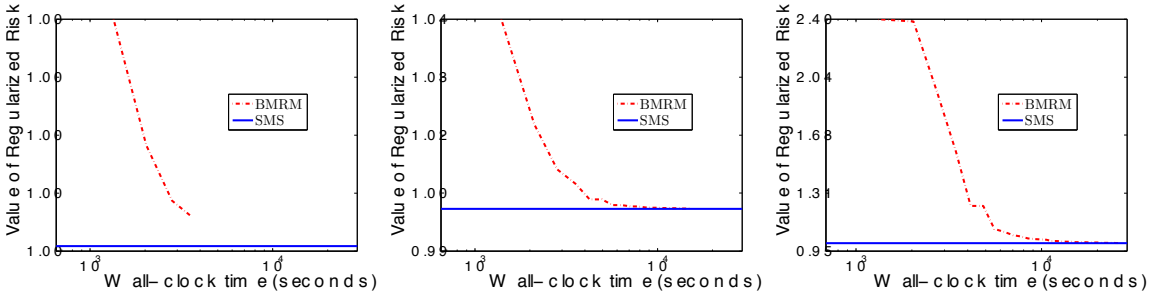


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

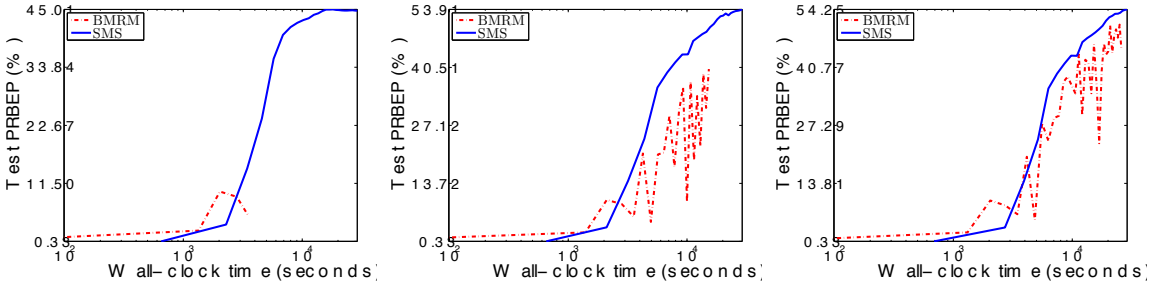


(j) Test ROCArea, $\lambda = 10^{-2}$, 95.99 (k) Test ROCArea, $\lambda = 10^{-4}$, **96.01** (l) Test ROCArea, $\lambda = 10^{-6}$, **96.01**

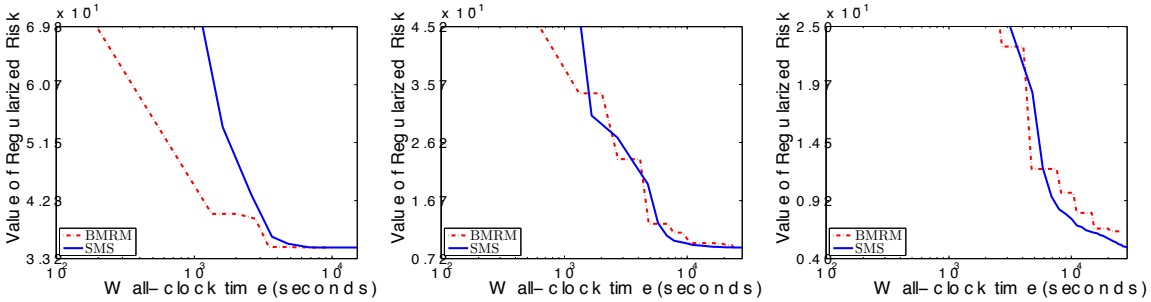
Figure 9: Results for `dna`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions. Both BMRM and our smoothing algorithm are run with 16 processors.



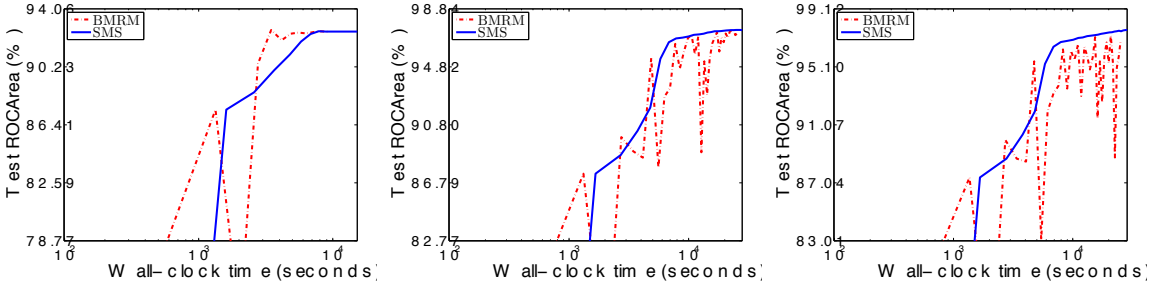
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 45.01 (e) Test PRBEP, $\lambda = 10^{-4}$, 53.91 (f) Test PRBEP, $\lambda = 10^{-6}$, **54.25**

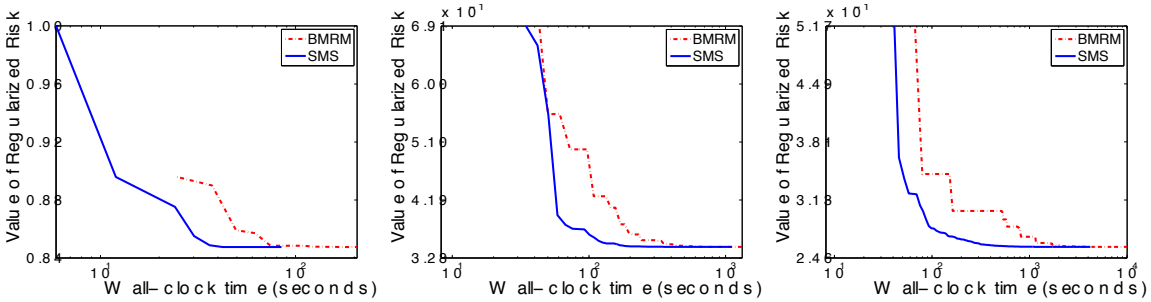


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

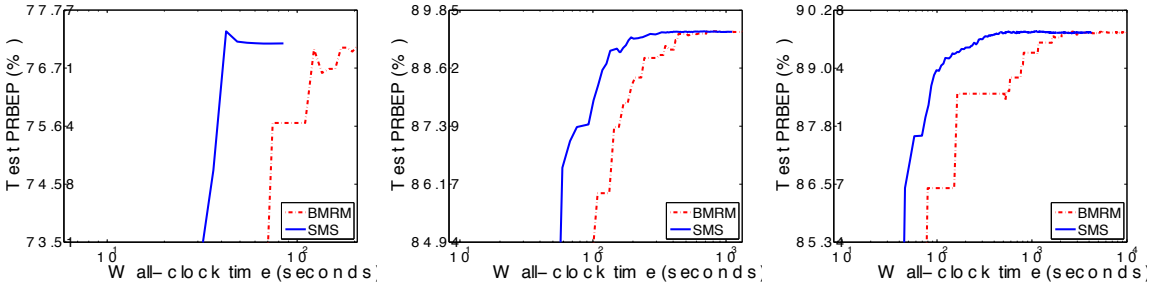


(j) Test ROCArea, $\lambda = 10^{-2}$, 92.57 (k) Test ROCArea, $\lambda = 10^{-4}$, 97.37 (l) Test ROCArea, $\lambda = 10^{-6}$, **97.66**

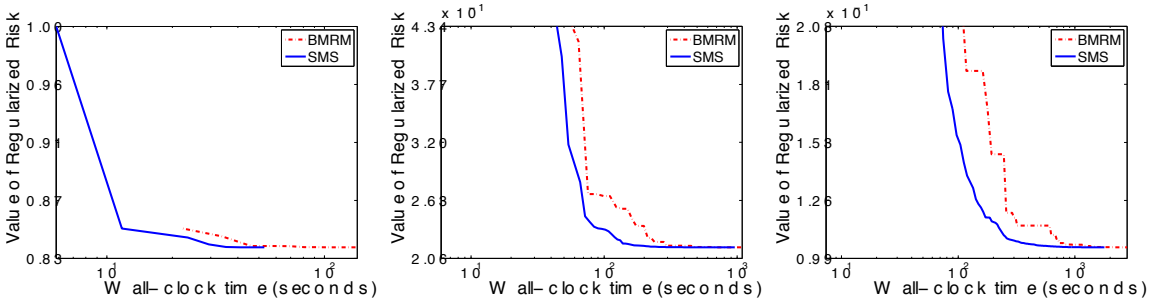
Figure 10: Results for `dna_string`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions. Both BMRM and our smoothing algorithm are run with 32 processors.



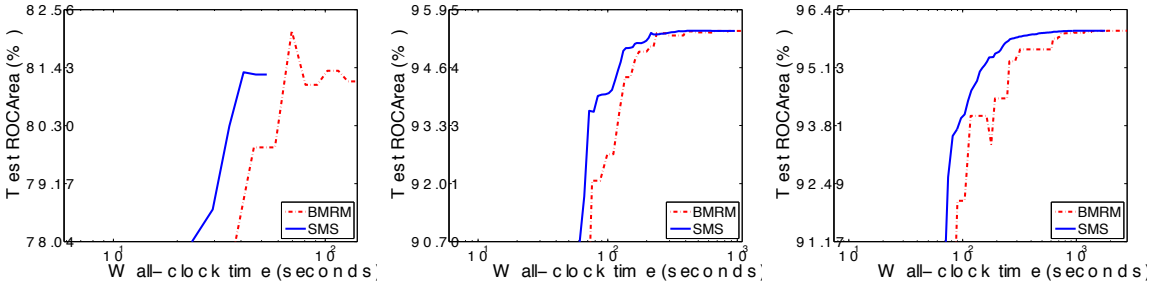
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 77.16 (e) Test PRBEP, $\lambda = 10^{-4}$, 89.39 (f) Test PRBEP, $\lambda = 10^{-6}$, **89.80**

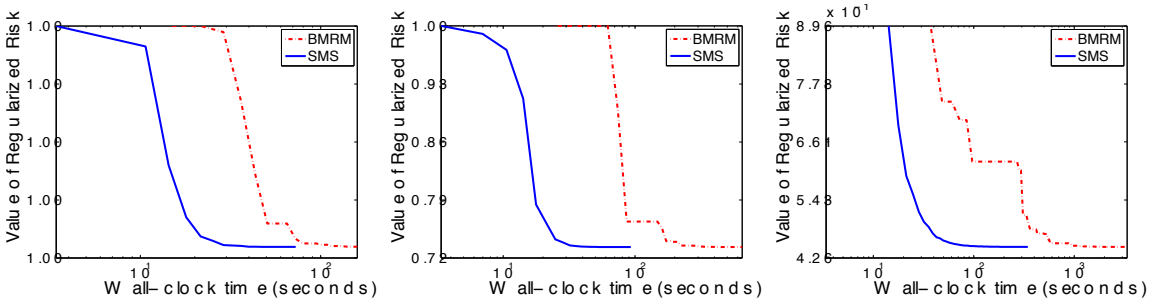


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

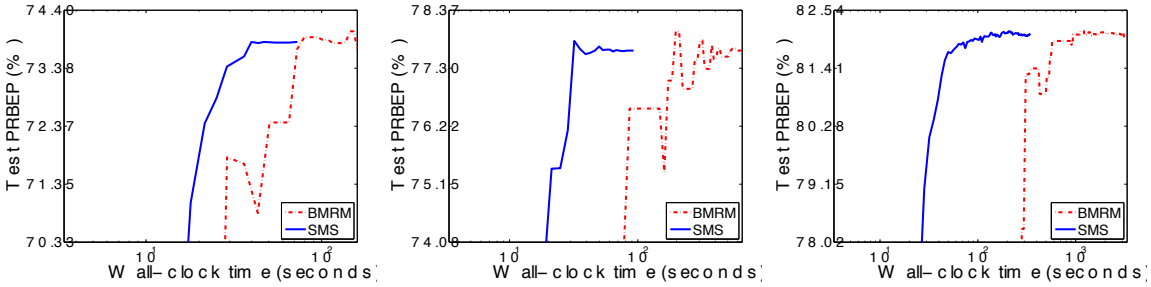


(j) Test ROCArea, $\lambda = 10^{-2}$, 81.30 (k) Test ROCArea, $\lambda = 10^{-4}$, 95.47 (l) Test ROCArea, $\lambda = 10^{-6}$, **95.97**

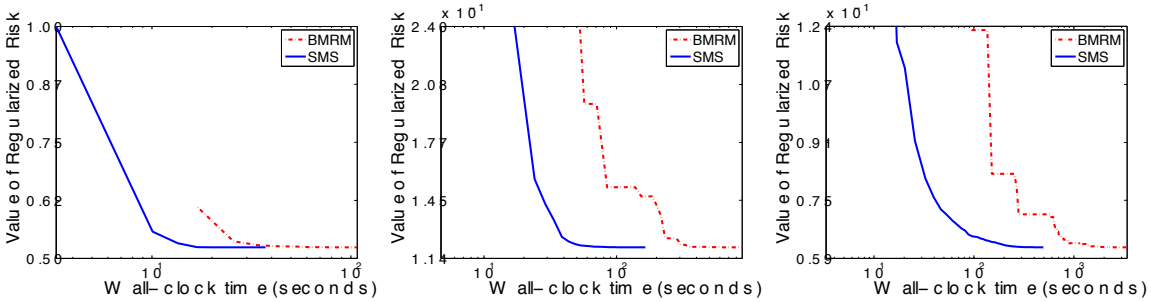
Figure 11: Results for `epsilon`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



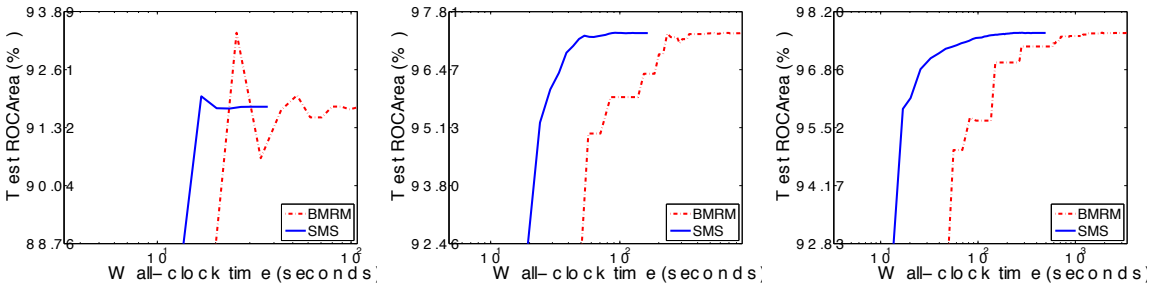
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 73.84 (e) Test PRBEP, $\lambda = 10^{-4}$, 77.62 (f) Test PRBEP, $\lambda = 10^{-6}$, **82.06**

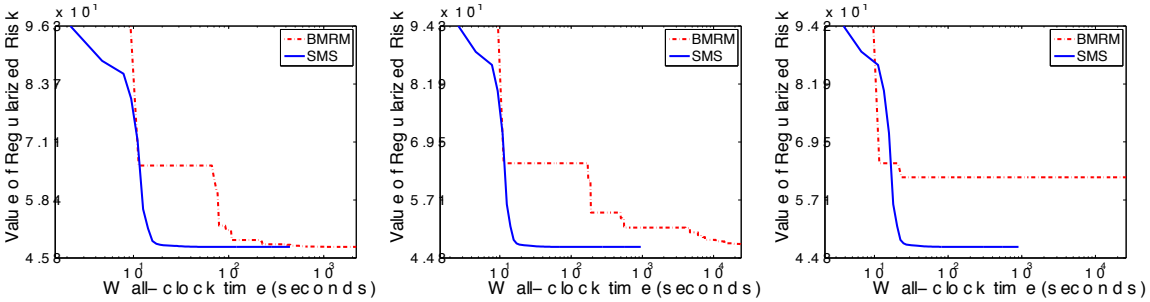


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

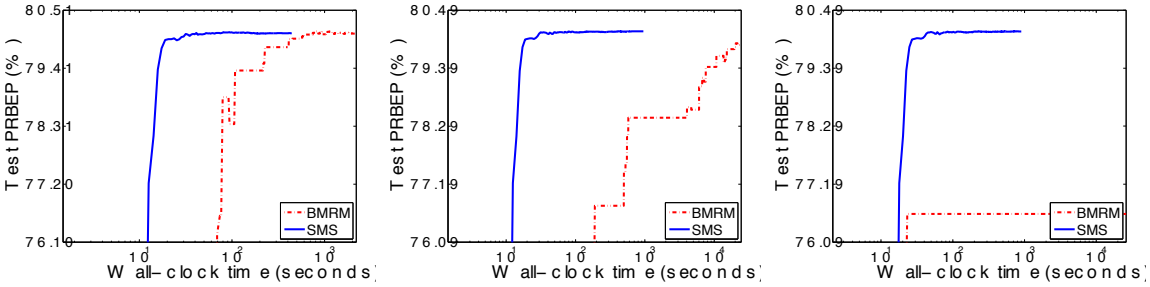


(j) Test ROCArea, $\lambda = 10^{-2}$, 91.79 (k) Test ROCArea, $\lambda = 10^{-4}$, 97.32 (l) Test ROCArea, $\lambda = 10^{-6}$, **97.71**

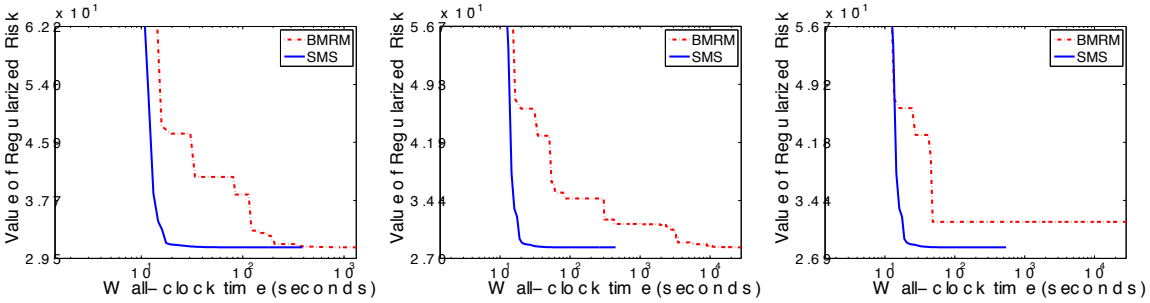
Figure 12: Results for `fd`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



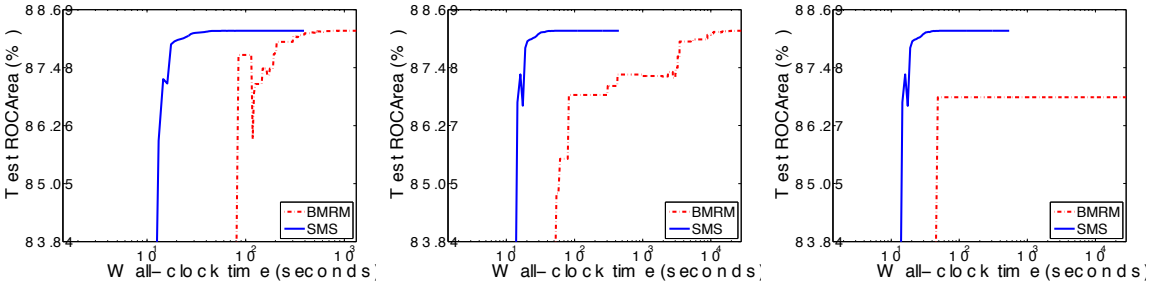
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 80.07 (e) Test PRBEP, $\lambda = 10^{-4}$, **80.09** (f) Test PRBEP, $\lambda = 10^{-6}$, **80.09**

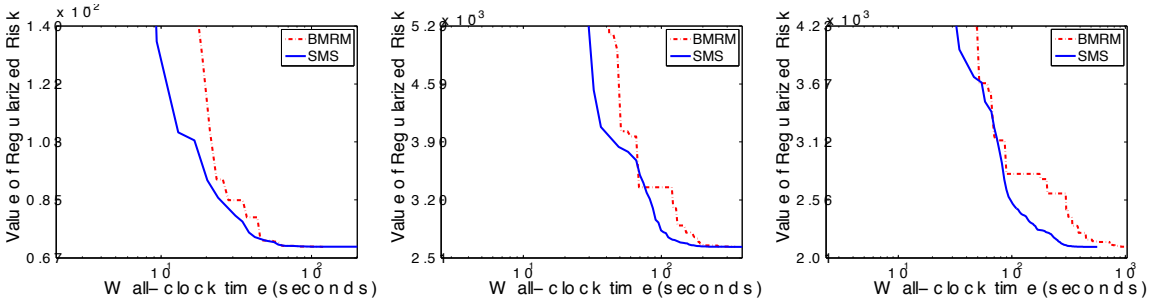


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

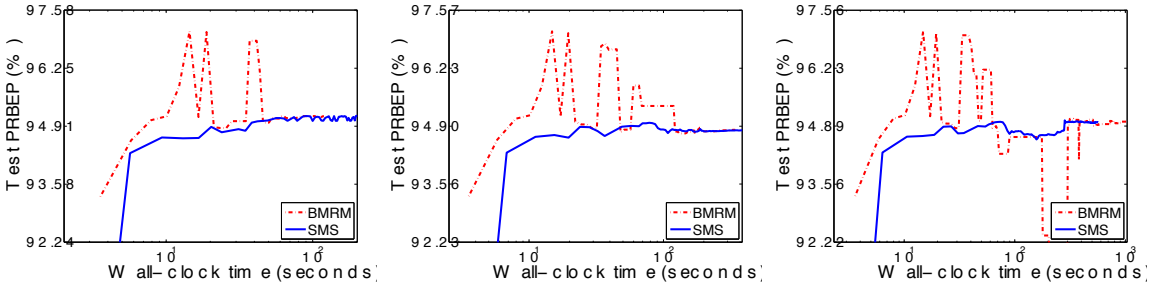


(j) Test ROCArea, $\lambda = 10^{-2}$, **88.25** (k) Test ROCArea, $\lambda = 10^{-4}$, **88.25** (l) Test ROCArea, $\lambda = 10^{-6}$, **88.25**

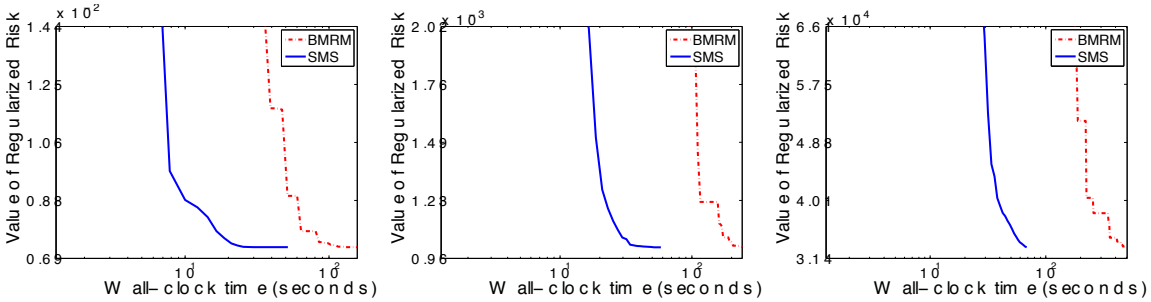
Figure 13: Results for `gamma`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



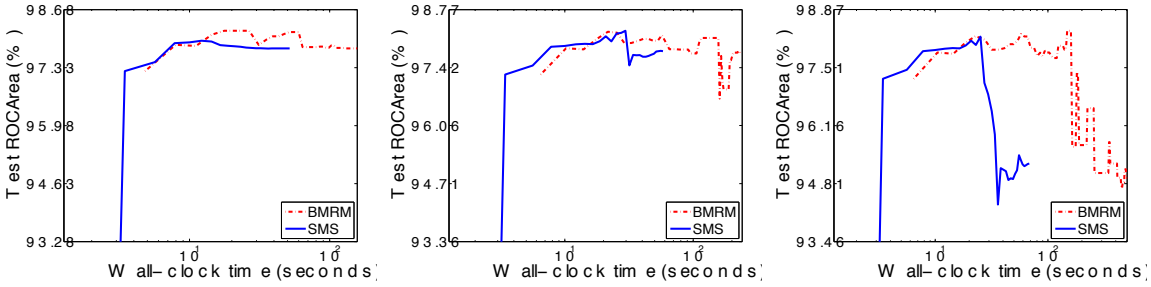
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, **95.07** (e) Test PRBEP, $\lambda = 10^{-4}$, 94.80 (f) Test PRBEP, $\lambda = 10^{-6}$, 94.98

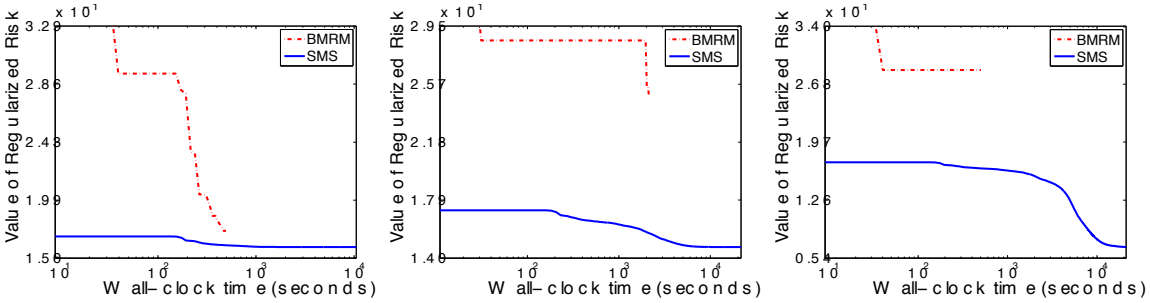


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

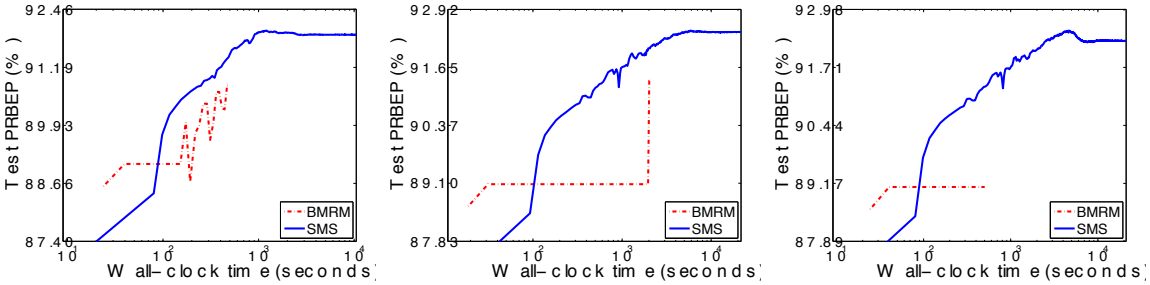


(j) Test ROCArea, $\lambda = 10^{-2}$, 97.78 (k) Test ROCArea, $\lambda = 10^{-4}$, **97.80** (l) Test ROCArea, $\lambda = 10^{-6}$, 95.28

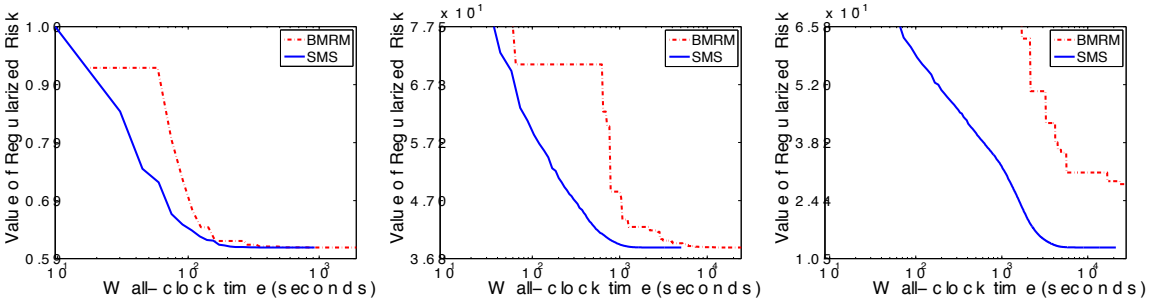
Figure 14: Results for `kdd99`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



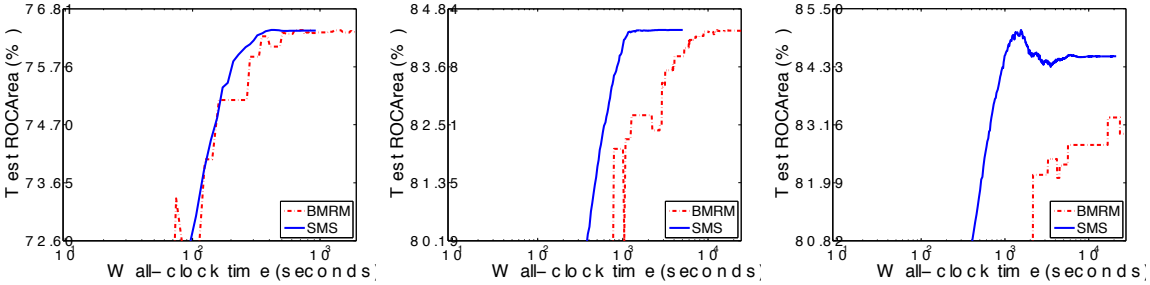
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 91.90 (e) Test PRBEP, $\lambda = 10^{-4}$, 93.02 (f) Test PRBEP, $\lambda = 10^{-6}$, **93.30**

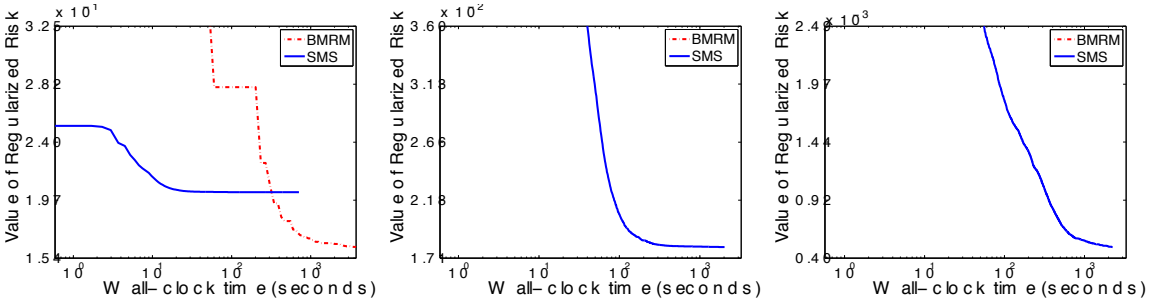


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

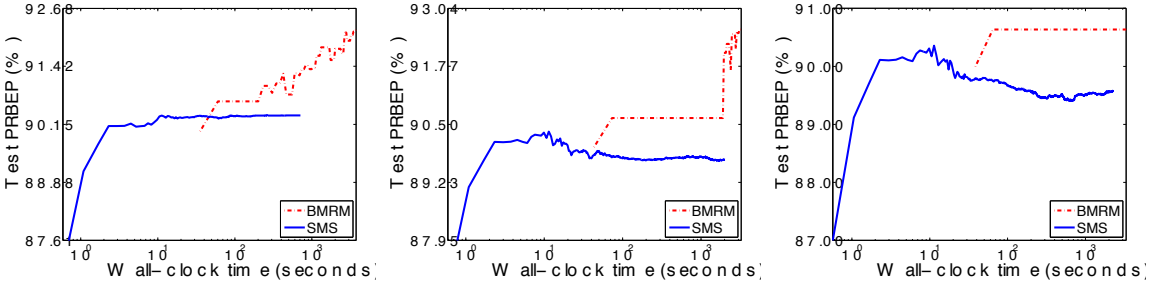


(j) Test ROCArea, $\lambda = 10^{-2}$, 76.41 (k) Test ROCArea, $\lambda = 10^{-4}$, 84.41 (l) Test ROCArea, $\lambda = 10^{-6}$, **84.55**

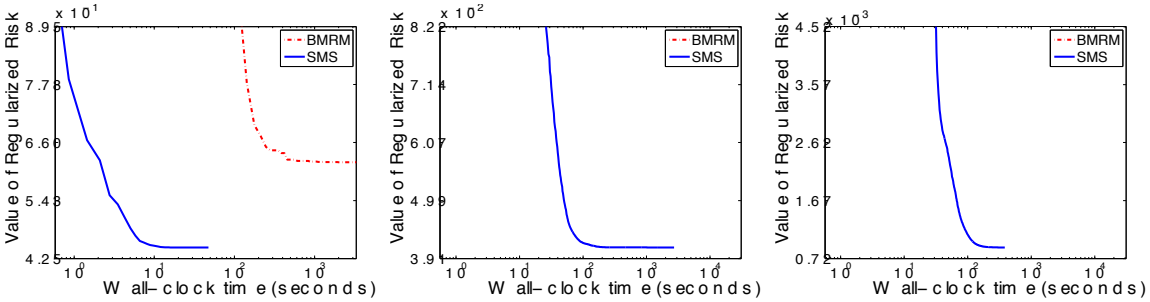
Figure 15: Results for `kdda`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions. In the PRBEP-problem, BMRM fails because the the inner QP solver does not converge even after a large number of iterations.



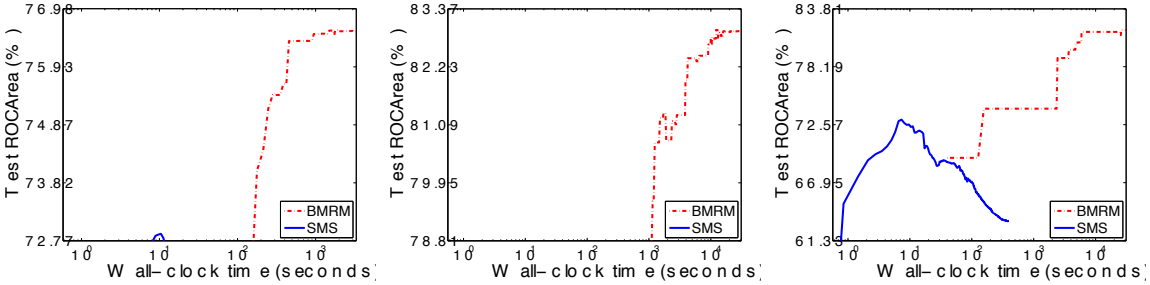
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 92.63 (e) Test PRBEP, $\lambda = 10^{-4}$, 93.02 (f) Test PRBEP, $\lambda = 10^{-6}$, **93.30**

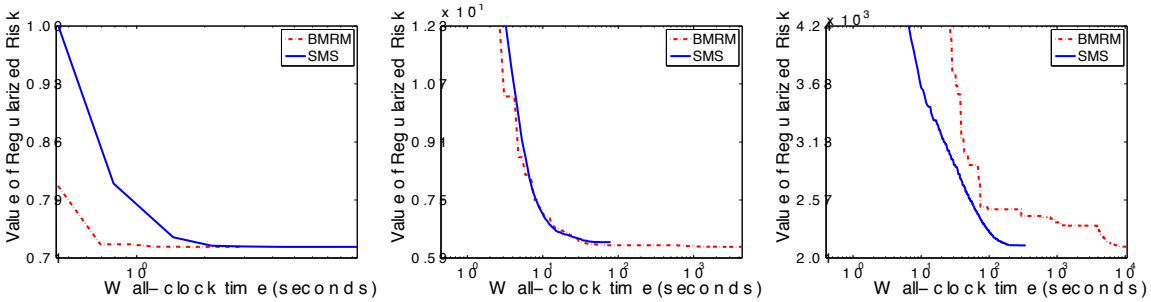


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

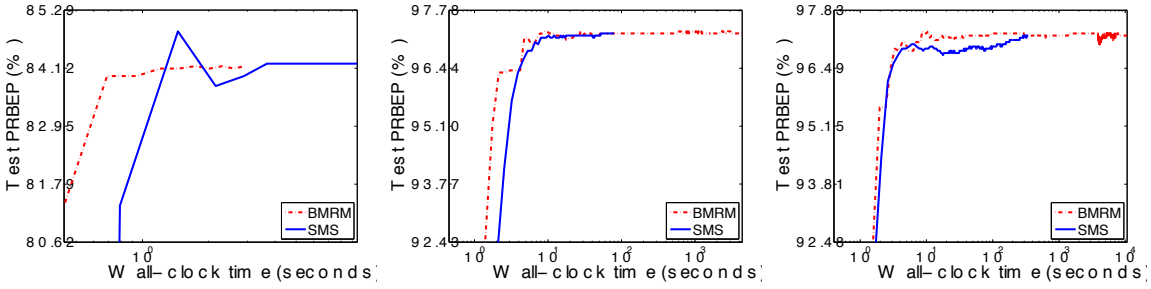


(j) Test ROCArea, $\lambda = 10^{-2}$, 76.58 (k) Test ROCArea, $\lambda = 10^{-4}$, 82.95 (l) Test ROCArea, $\lambda = 10^{-6}$, **85.29**

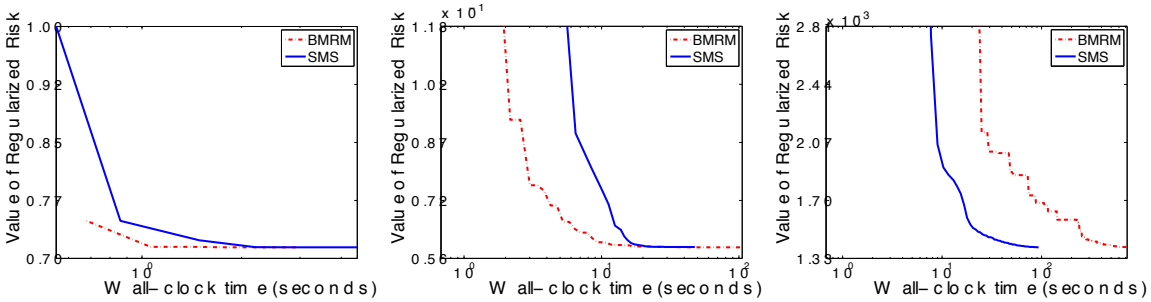
Figure 16: Results for kddb. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions. In the PRBEP-problem, BMRM fails because the inner QP solver does not converge even after a large number of iterations.



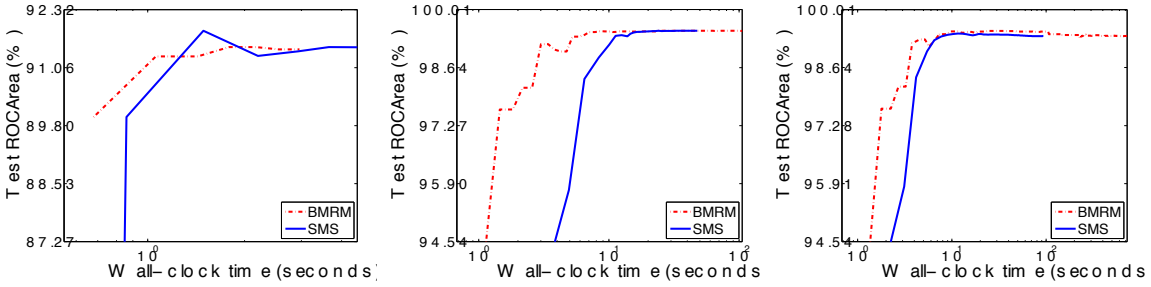
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 84.21 (e) Test PRBEP, $\lambda = 10^{-4}$, **97.24** (f) Test PRBEP, $\lambda = 10^{-6}$, **97.24**

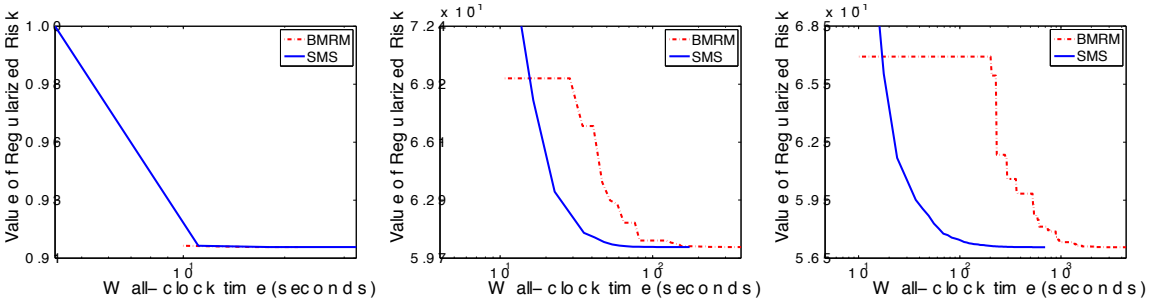


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

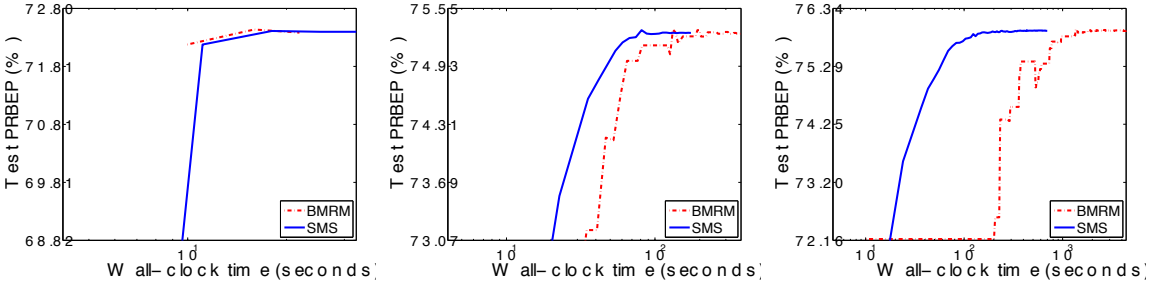


(j) Test ROCArea, $\lambda = 10^{-2}$, 91.51 (k) Test ROCArea, $\lambda = 10^{-4}$, **99.51** (l) Test ROCArea, $\lambda = 10^{-6}$, 99.39

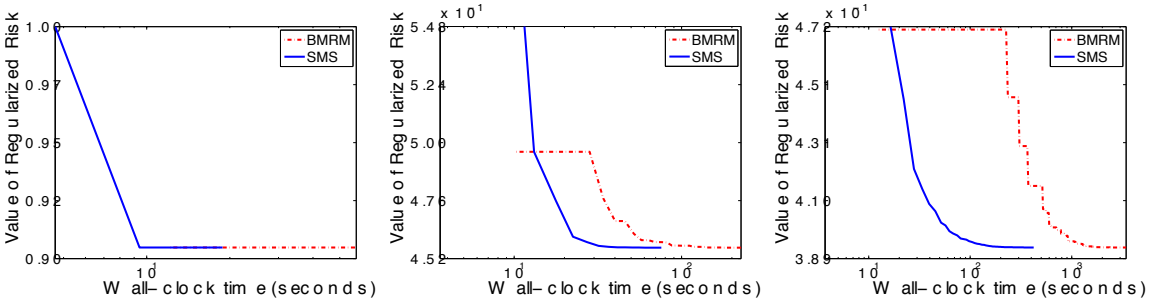
Figure 17: Results for `news20`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



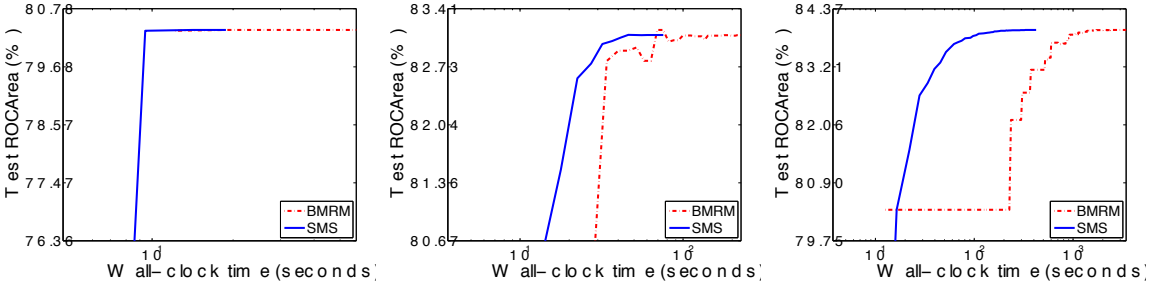
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 72.40 (e) Test PRBEP, $\lambda = 10^{-flff4}$, 75.29 (f) Test PRBEP, $\lambda = 10^{-6}$, **75.93**

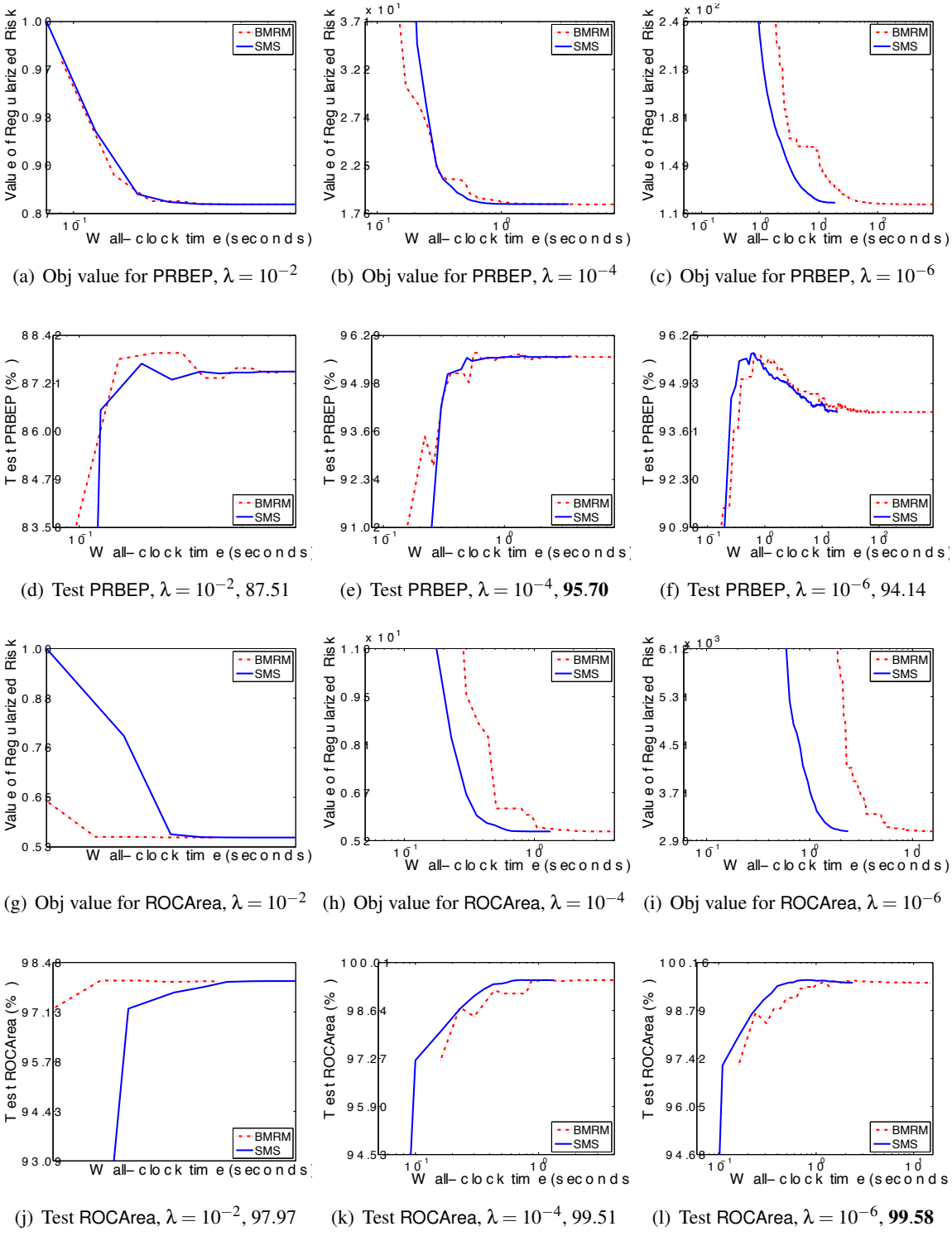


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$



(j) Test ROCArea, $\lambda = 10^{-2}$, 80.38 (k) Test ROCArea, $\lambda = 10^{-4}$, 83.10 (l) Test ROCArea, $\lambda = 10^{-6}$, **83.94**

Figure 18: Results for `ocr`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions. Both BMRM and our smoothing algorithm are run with 16 processors.



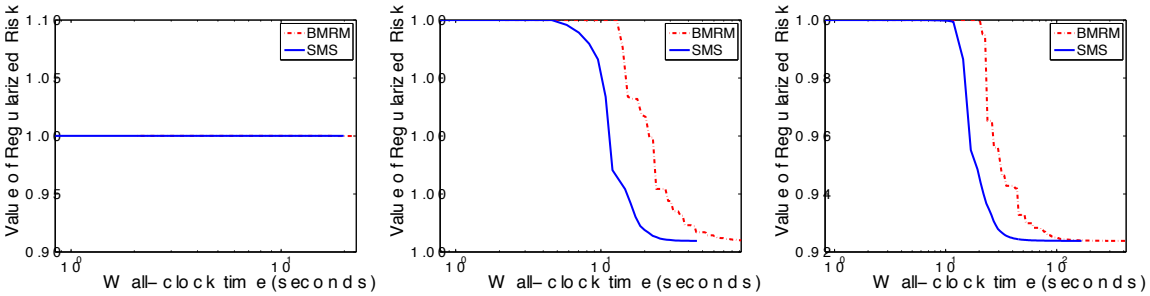
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 87.51 (e) Test PRBEP, $\lambda = 10^{-4}$, **95.70** (f) Test PRBEP, $\lambda = 10^{-6}$, 94.14

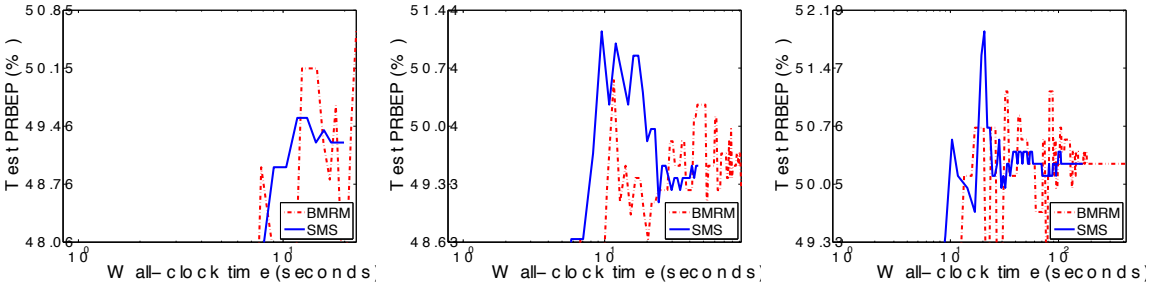
(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 97.97 (k) Test ROCArea, $\lambda = 10^{-4}$, 99.51 (l) Test ROCArea, $\lambda = 10^{-6}$, **99.58**

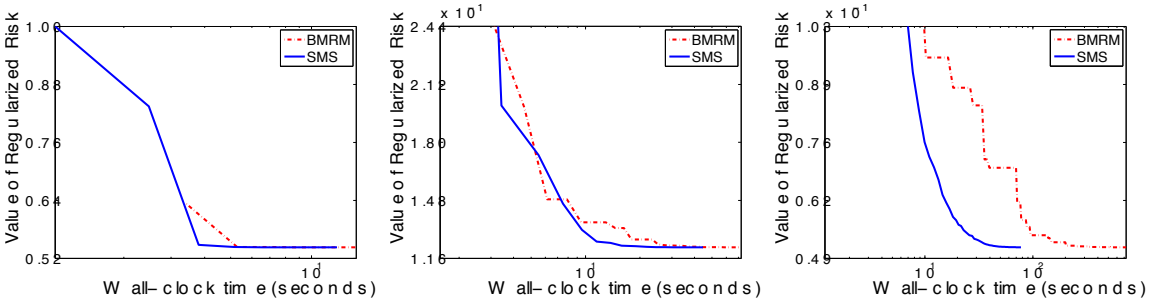
Figure 19: Results for `real-sim`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



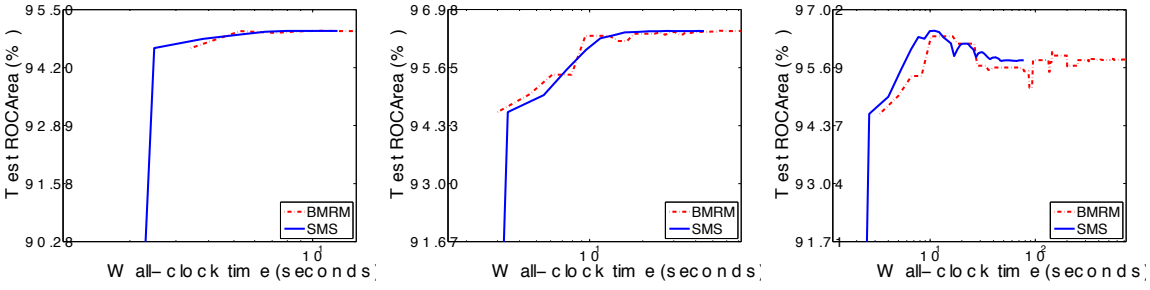
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 49.26 (e) Test PRBEP, $\lambda = 10^{-4}$, 49.55 (f) Test PRBEP, $\lambda = 10^{-6}$, **50.30**

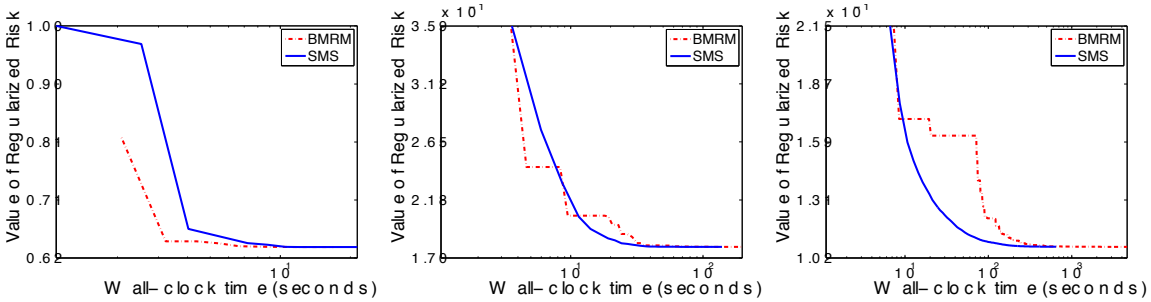


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

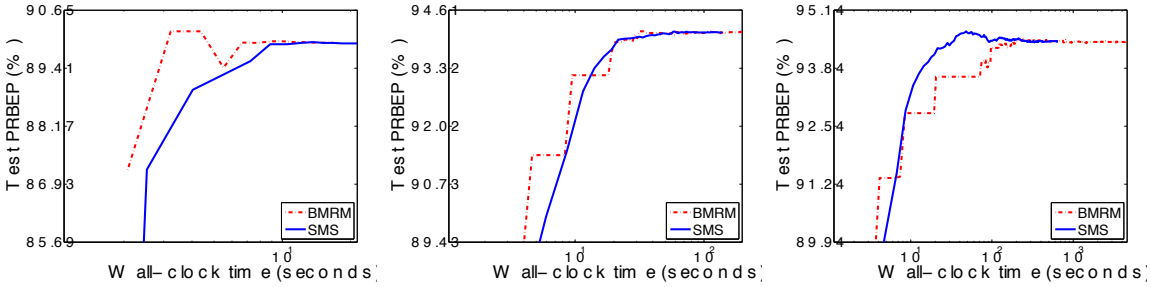


(j) Test ROCArea, $\lambda = 10^{-2}$, 95.02 (k) Test ROCArea, $\lambda = 10^{-4}$, **96.49** (l) Test ROCArea, $\lambda = 10^{-6}$, 95.86

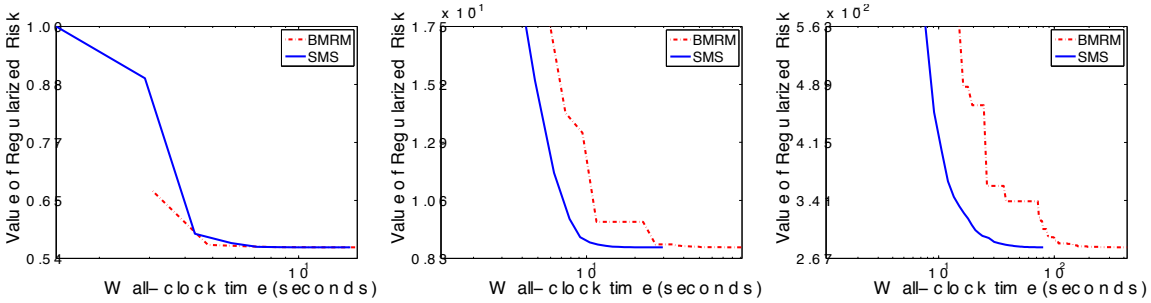
Figure 20: Results for `reuters-c11`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



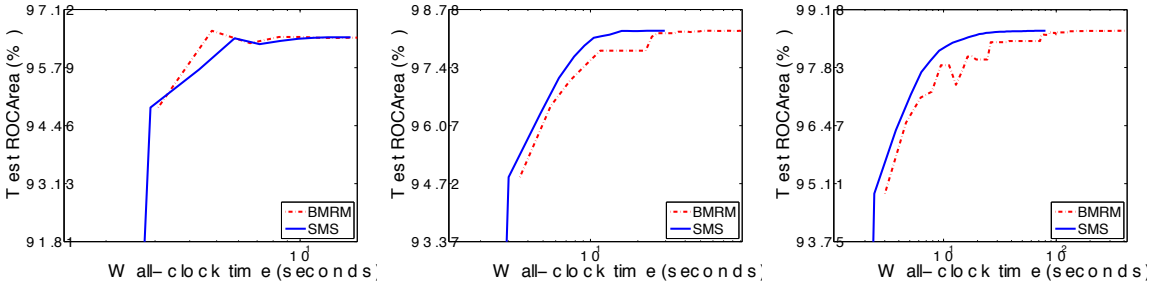
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 89.94 (e) Test PRBEP, $\lambda = 10^{-4}$, 94.11 (f) Test PRBEP, $\lambda = 10^{-6}$, **94.44**

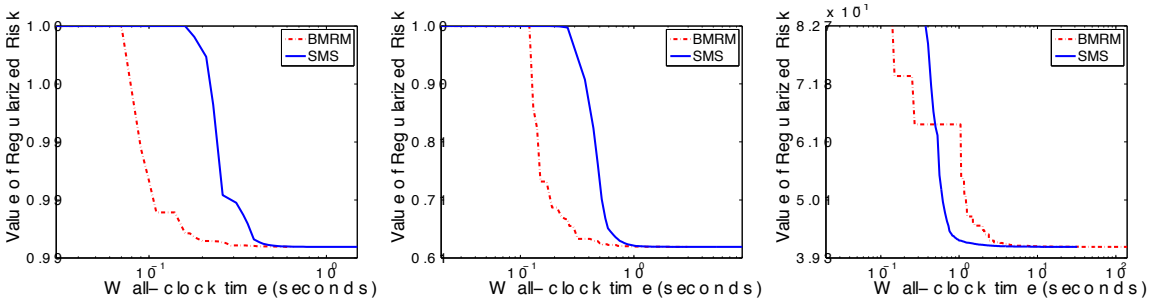


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

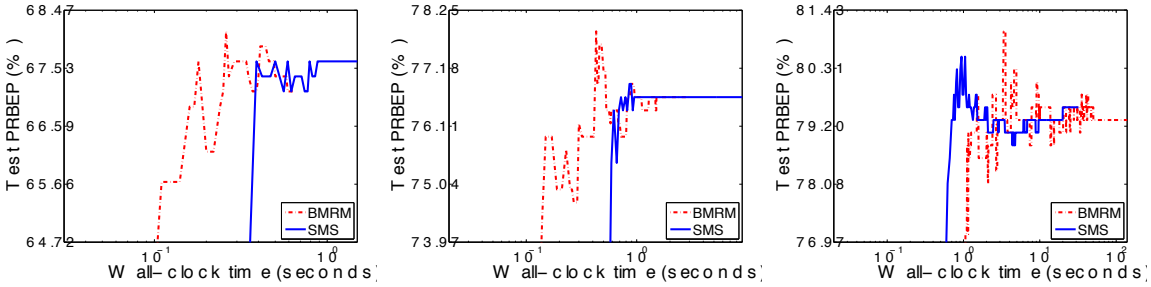


(j) Test ROCArea, $\lambda = 10^{-2}$, 96.49 (k) Test ROCArea, $\lambda = 10^{-4}$, 98.28 (l) Test ROCArea, $\lambda = 10^{-6}$, **98.69**

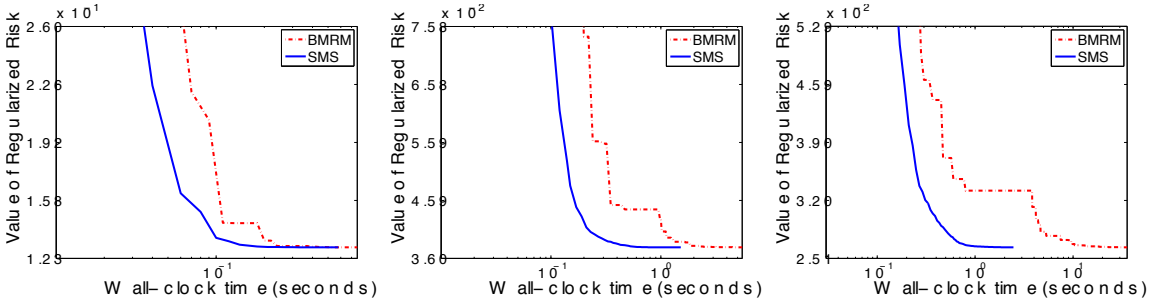
Figure 21: Results for `reuters-ccat`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



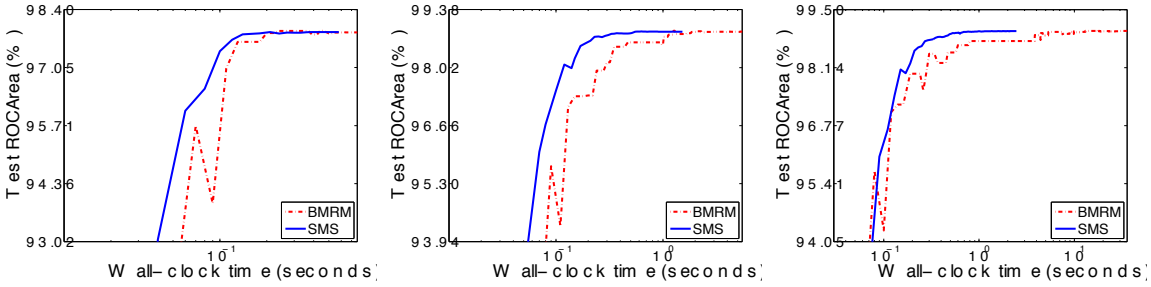
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 67.64 (e) Test PRBEP, $\lambda = 10^{-4}$, 76.64 (f) Test PRBEP, $\lambda = 10^{-6}$, **79.56**

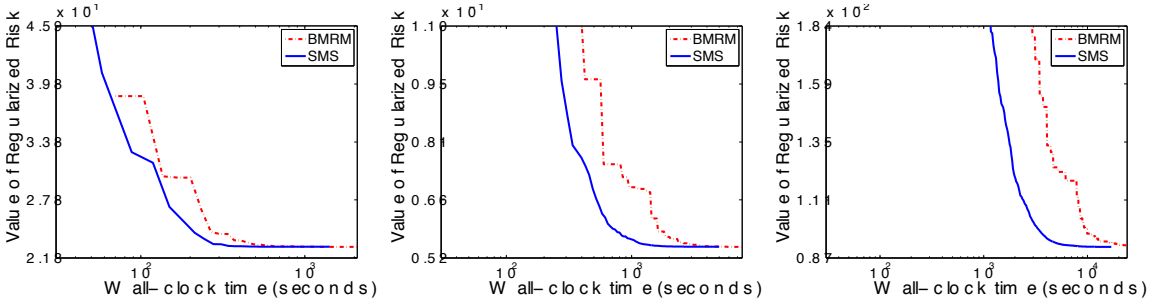


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

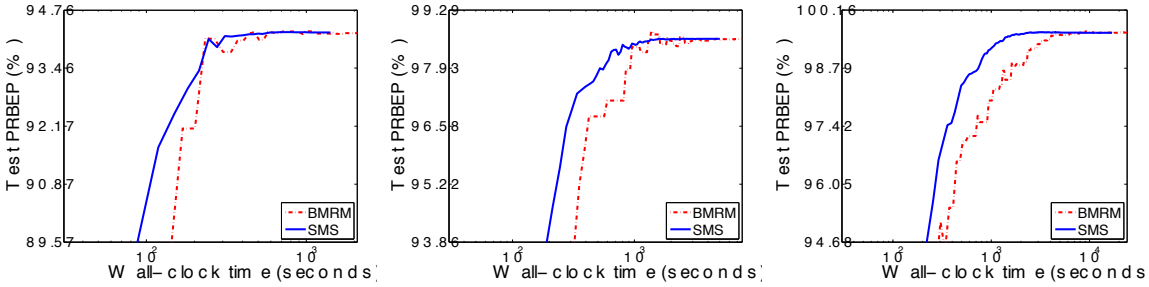


(j) Test ROCArea, $\lambda = 10^{-2}$, 97.88 (k) Test ROCArea, $\lambda = 10^{-4}$, 98.86 (l) Test ROCArea, $\lambda = 10^{-6}$, **98.99**

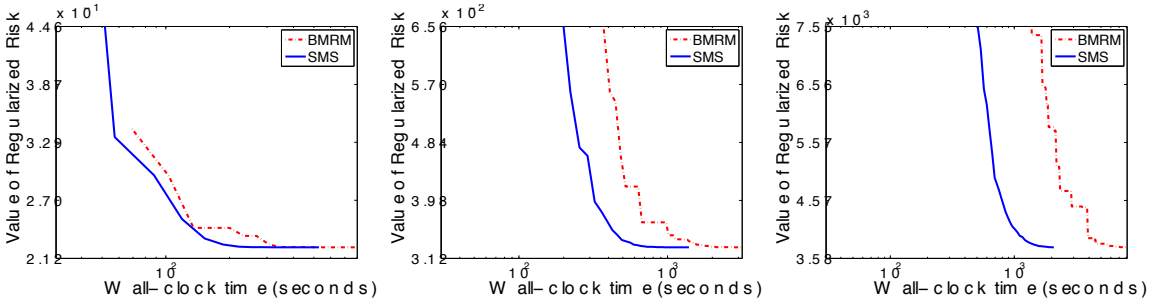
Figure 22: Results for `web8`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



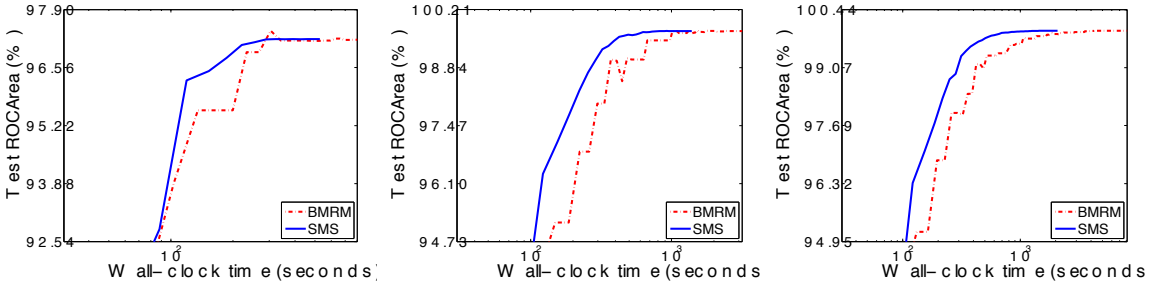
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 94.26 (e) Test PRBEP, $\lambda = 10^{-4}$, 98.62 (f) Test PRBEP, $\lambda = 10^{-6}$, **99.62**

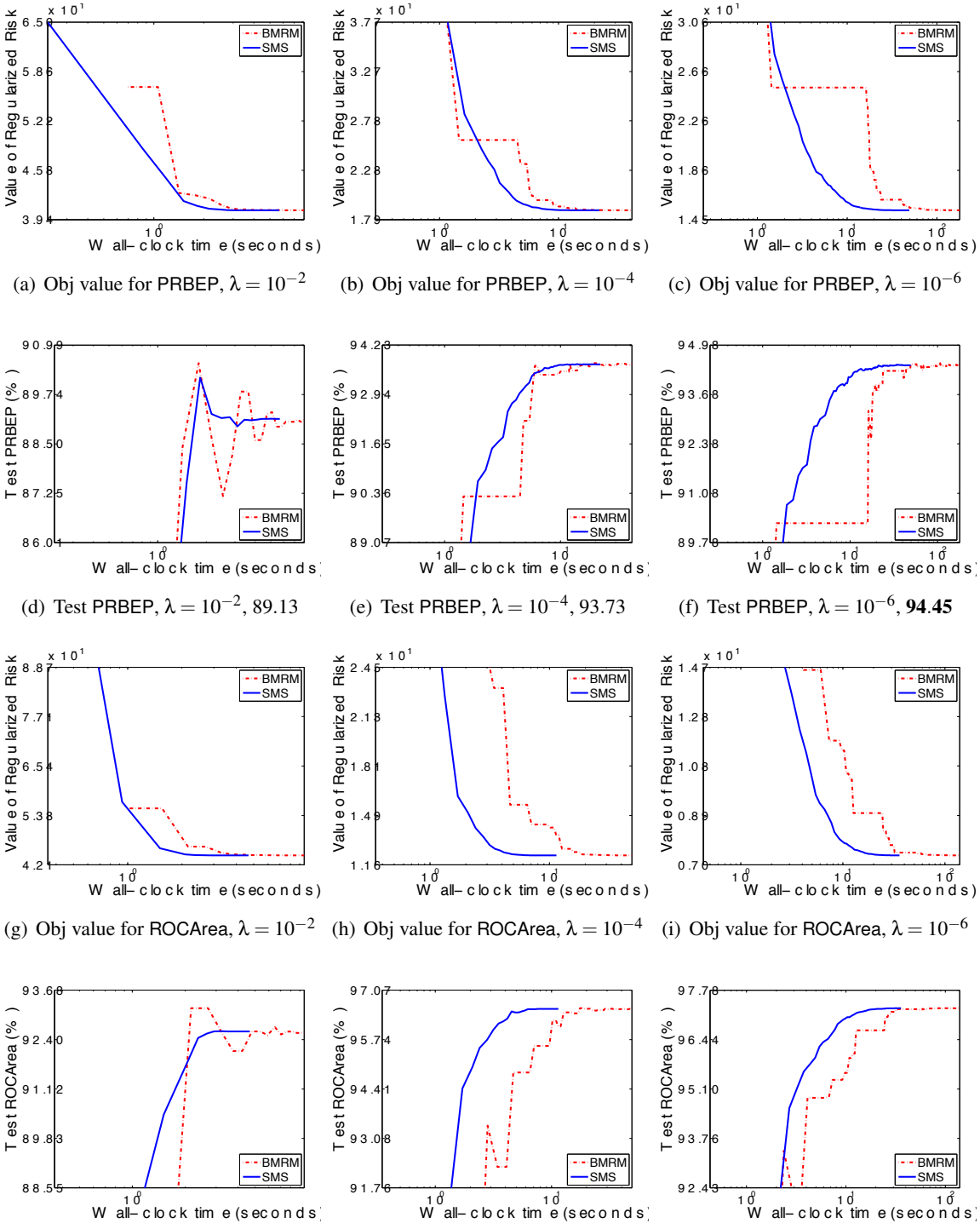


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$



(j) Test ROCArea, $\lambda = 10^{-2}$, 97.21 (k) Test ROCArea, $\lambda = 10^{-4}$, 99.71 (l) Test ROCArea, $\lambda = 10^{-6}$, **99.94**

Figure 23: Results for `webspam-t`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



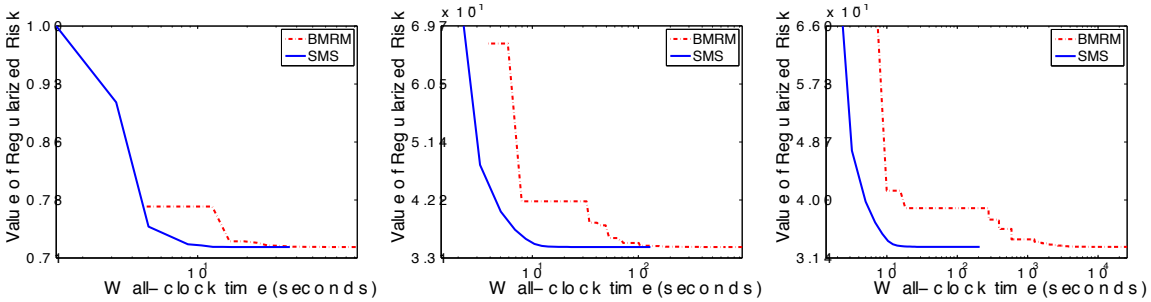
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$

(d) Test PRBEP, $\lambda = 10^{-2}$, 89.13 (e) Test PRBEP, $\lambda = 10^{-4}$, 93.73 (f) Test PRBEP, $\lambda = 10^{-6}$, **94.45**

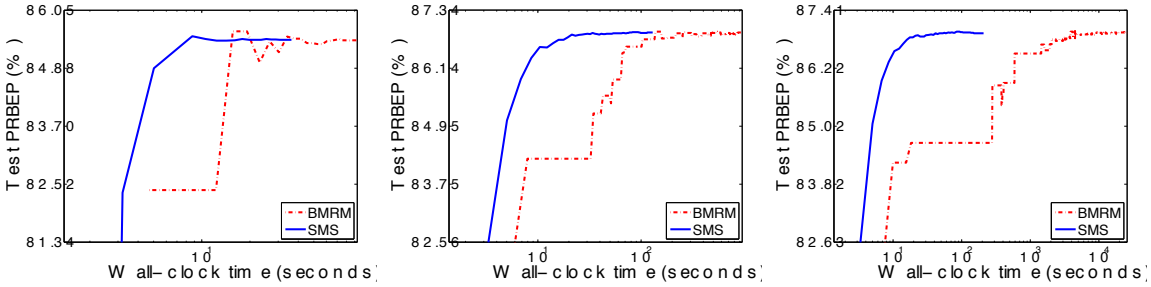
(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

(j) Test ROCArea, $\lambda = 10^{-2}$, 92.61 (k) Test ROCArea, $\lambda = 10^{-4}$, 96.56 (l) Test ROCArea, $\lambda = 10^{-6}$, **97.29**

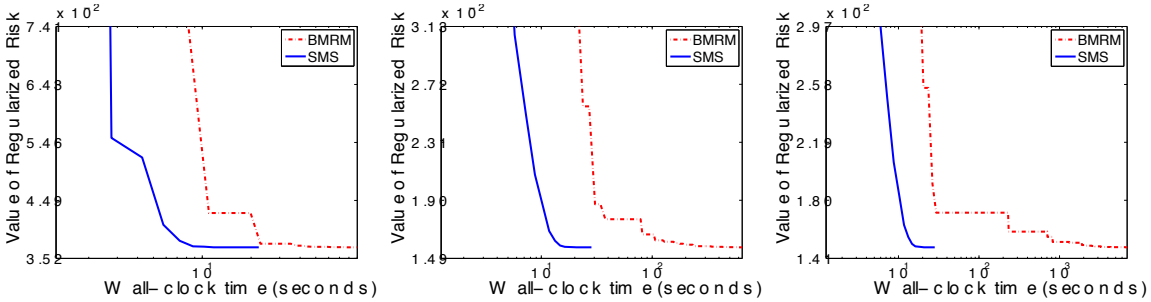
Figure 24: Results for `webspam-u`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



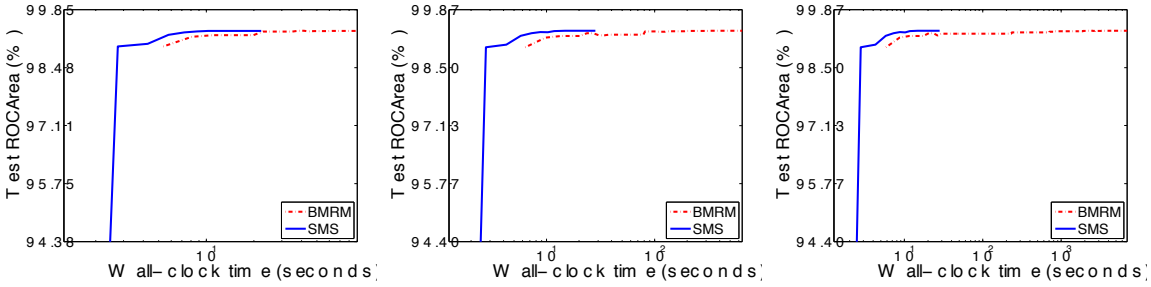
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 85.45 (e) Test PRBEP, $\lambda = 10^{-4}$, 86.88 (f) Test PRBEP, $\lambda = 10^{-6}$, **86.94**

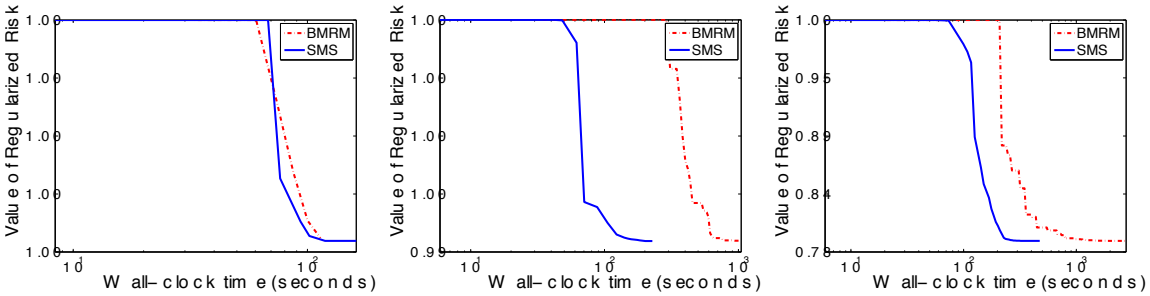


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$

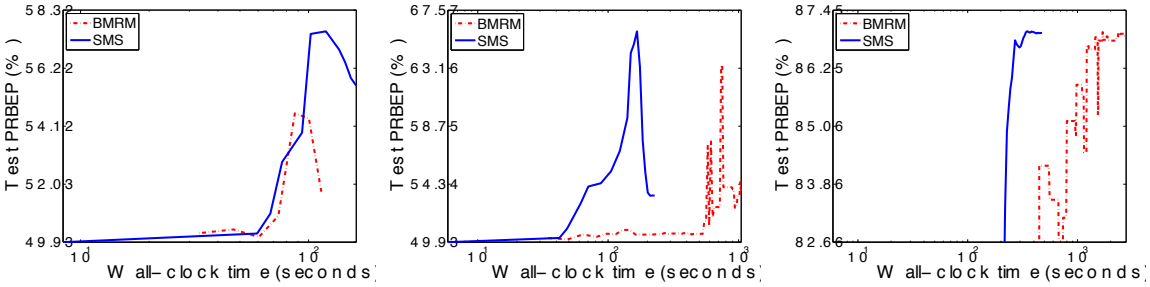


(j) Test ROCArea, $\lambda = 10^{-2}$, 99.34 (k) Test ROCArea, $\lambda = 10^{-4}$, **99.37** (l) Test ROCArea, $\lambda = 10^{-6}$, **99.37**

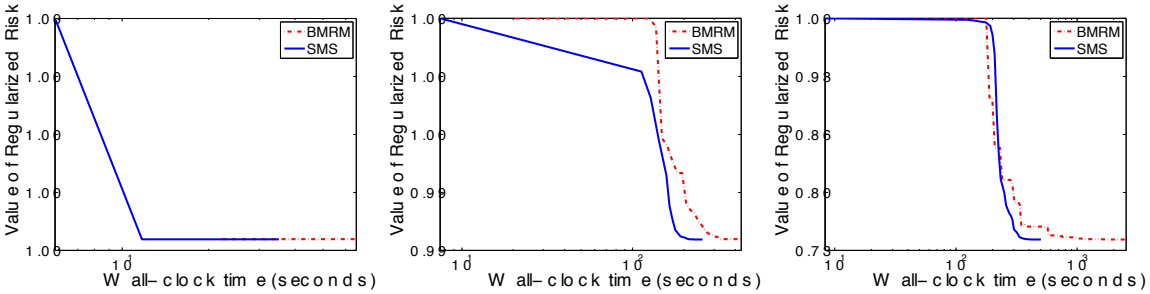
Figure 25: Results for `worm`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



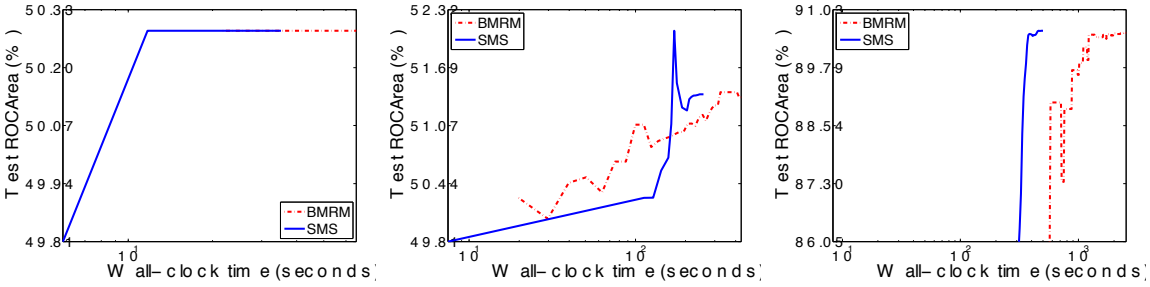
(a) Obj value for PRBEP, $\lambda = 10^{-2}$ (b) Obj value for PRBEP, $\lambda = 10^{-4}$ (c) Obj value for PRBEP, $\lambda = 10^{-6}$



(d) Test PRBEP, $\lambda = 10^{-2}$, 55.59 (e) Test PRBEP, $\lambda = 10^{-4}$, 53.47 (f) Test PRBEP, $\lambda = 10^{-6}$, **86.98**

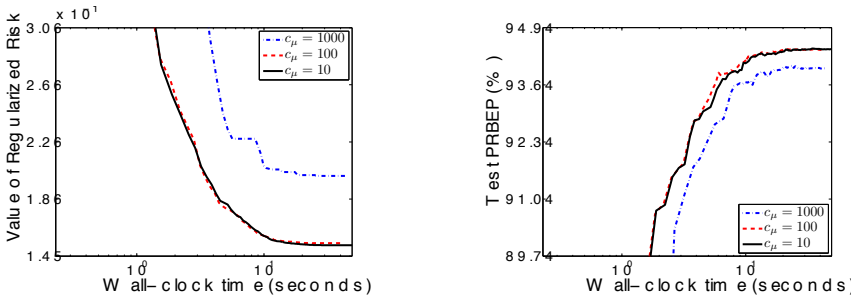


(g) Obj value for ROCArea, $\lambda = 10^{-2}$ (h) Obj value for ROCArea, $\lambda = 10^{-4}$ (i) Obj value for ROCArea, $\lambda = 10^{-6}$



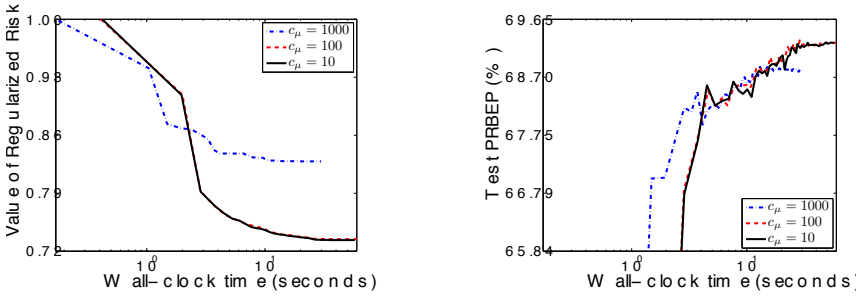
(j) Test ROCArea, $\lambda = 10^{-2}$, 50.28 (k) Test ROCArea, $\lambda = 10^{-4}$, 51.41 (l) Test ROCArea, $\lambda = 10^{-6}$, **90.58**

Figure 26: Results for `zeta`. The optimal test performance among all values of λ is highlighted in boldface in the sub-figure captions.



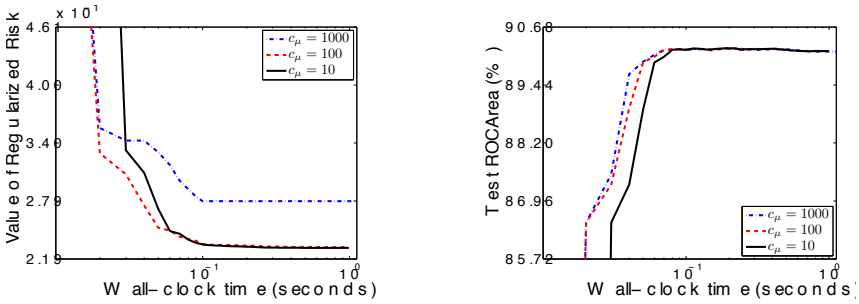
(a) Obj value for PRBEP (b) Test PRBEP

Figure 27: c_μ test: `webspam-u`



(a) Obj value for PRBEP (b) Test PRBEP

Figure 28: c_μ test: `covertype`

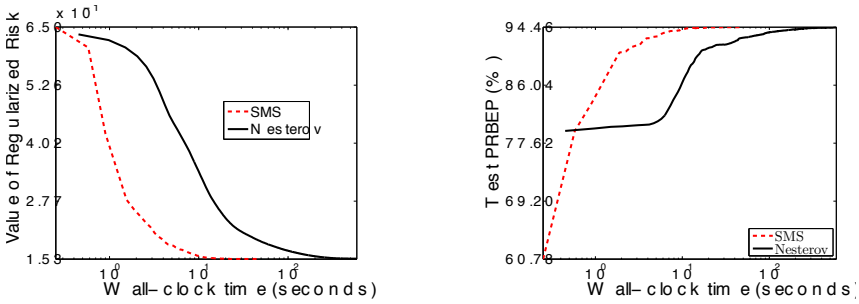


(a) Obj value for ROCArea (b) Test ROCArea

Figure 29: c_μ test: `adult9`

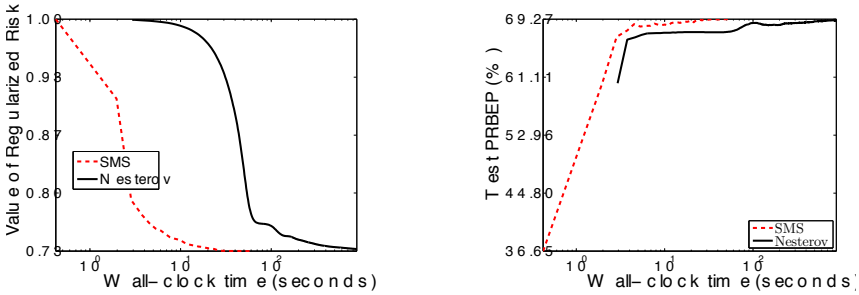
it is easy to smooth the L_1 norm regularizer, it is not recommended; the sparsity of the solution is an important statistical property of these algorithms and smoothing destroys this property.

In future work we would like to extend our techniques to handle more complicated contingency based multivariate performance measures such as the F_1-score. We would also like to extend smoothing to matching loss functions commonly used in ranking, where we believe our techniques will solve a smoothed version of the Hungarian marriage problem.



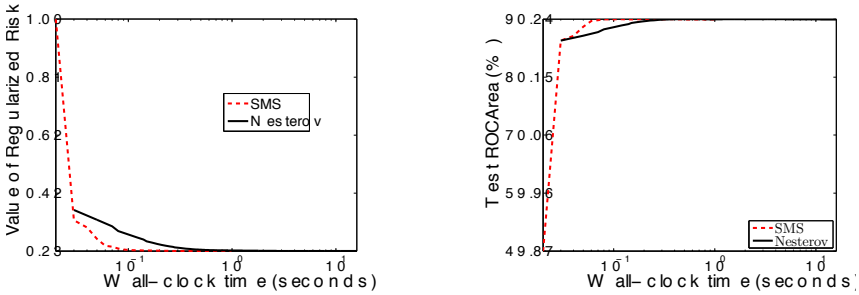
(a) Obj value for PRBEP (b) Test PRBEP

Figure 30: SMS vs AGM: `webspam-u`



(a) Obj value for PRBEP (b) Test PRBEP

Figure 31: SMS vs AGM: `covertype`



(a) Obj value for ROCArea (b) Test ROCArea

Figure 32: SMS vs AGM: `adult9`

Acknowledgments

We thank the PETSc and TAO developers for making available their code, updating their software based on our feature requests, and patiently answering all our questions. We thank the anonymous reviewers of the earlier versions of this paper for their helpful comments and suggestions. Xinhua Zhang would like to acknowledge support from the Alberta Innovates Centre for Machine Learning. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. Ankan Saha would like to acknowledge the fellowship support from the department of Computer Science at the University of Chicago. The work of S.V.N. Vishwanathan is partially supported by a grant from Google and NSF grant IIS-1117705.

Appendix A. The Smoothing Procedure

The idea of the smoothing technique in Nesterov (2005) can be motivated by using the Theorem 4.2.1 and 4.2.2 in Hiriart-Urruty and Lemaréchal (1996).

Lemma 9 *If $f:\mathbb{R}^n\to\mathbb{R}$ is convex and differentiable, and ∇f is Lipschitz continuous with constant L (called L-l.c.g), then $f^\star$ is strongly convex with modulus $\frac{1}{L}$ (called $\frac{1}{L}$-sc). Conversely, if $f:\mathbb{R}^n\to\mathbb{R}\cup\{\infty\}$ is σ-sc, then $f^\star$ is finite on $\mathbb{R}^n$ and is $\frac{1}{\sigma}$-l.c.g.*

Since $g+\mu d$ is μ-sc, Lemma 9 implies $g_\mu^\star$ is $\frac{1}{\mu}$-*l.c.g*. By chain rule, one can show that $g_\mu^\star(A^\top\mathbf{w})$ is L_μ-*l.c.g* where $L_\mu\le\frac{1}{\mu}\|A\|^2$. Further, the definition of Fenchel dual implies the following uniform deviation bound:

$$g_\mu^\star(\boldsymbol{u})=\max_{\boldsymbol{\alpha}\in Q}\{\langle\boldsymbol{\alpha},\boldsymbol{u}\rangle-g(\boldsymbol{\alpha})-\mu d(\boldsymbol{\alpha})\}\begin{cases}\le\max\limits_{\boldsymbol{\alpha}\in Q}\{\langle\boldsymbol{\alpha},\boldsymbol{u}\rangle-g(\boldsymbol{\alpha})\}=g^\star(\boldsymbol{u})\\ \ge\max\limits_{\boldsymbol{\alpha}\in Q}\{\langle\boldsymbol{\alpha},\boldsymbol{u}\rangle-g(\boldsymbol{\alpha})-\mu D\}=g^\star(\boldsymbol{u})-\mu D\end{cases}$$

$$\implies\qquad g^\star(\mathbf{u})-\mu D\le g_\mu^\star(\mathbf{u})\le g^\star(\mathbf{u}),\quad\forall\,\boldsymbol{u}\in\mathbb{R}^n. \tag{35}$$

Note that the derivation of (35) does not require g be convex, but the strong convexity of $g+\mu d$ (required by Lemma 9) relies on the convexity of g. By (35), to find an ε accurate solution to $J(\mathbf{w})$, it suffices to set the maximum deviation $\mu D<\frac{\varepsilon}{2}$ (i.e., $\mu<\frac{\varepsilon}{2D}$), and then find a $\frac{\varepsilon}{2}$ accurate solution to J_μ in (6). Initialize $\mathbf{w}$ to $\mathbf{0}$ and apply Nesterov's accelerated gradient method in Nesterov (2007) to J_μ, this takes at most

$$k=\min\left\{\sqrt{\frac{4L_\mu\Delta_0}{\varepsilon}},\log\frac{L_\mu\Delta_0}{\varepsilon}\Big/\log\left(1-\sqrt{\lambda/L_\mu}\right)\right\}$$

number of steps where $\Delta_0=\frac{1}{2}\|\mathbf{w}^*\|^2$ and $\mathbf{w}^*$ is the minimizer of $J(\mathbf{w})$. Each step involves one gradient query of $g_\mu^\star(A^\top\mathbf{w})$ and some cheap updates. Plugging in $L_\mu\le\frac{2D}{\varepsilon}\|A\|^2$ and using $\log(1+\delta)\approx\delta$ when $\delta\approx0$, we get the iteration bound in (7).

References

Alekh Agarwal, Peter Bartlett, Pradeep Ravikumar, and Martin Wainwright. Information-theoretic lower bounds on the oracle complexity of convex optimization. In *Neural Information Processing Systems*, 2009.

Satish Balay, Jed Brown, Kris Buschelman, William Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2011. http://www.mcs.anl.gov/petsc.

Peter Bartlett, Michael Jordan, and Jon McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Steve Benson, Lois Curfman McInnes, Jorge Moré, Todd Munson, and Jason Sarich. TAO user manual (revision 1.10.1). Technical Report ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory, 2010. http://www.mcs.anl.gov/tao.

Léon Bottou. Stochastic gradient SVMs. `http://leon.bottou.org/projects/sgd`, 2008.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.

Olivier Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5): 1155–1178, 2007.

Chuong Do, Quoc Le, and Chuan-Sheng Foo. Proximal regularization for online and batch learning. In *International Conference on Machine Learning ICML*, 2009.

Michael Ferris and Todd Munson. Interior-point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783–804, 2002.

Vojtěch Franc and Sören Sonnenburg. Optimized cutting plane algorithm for support vector machines. In Andrew McCallum and Sam Roweis, editors, *ICML*, pages 320–327. Omnipress, 2008.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms, I and II*, volume 305 and 306. Springer-Verlag, 1996.

Cho Jui Hsieh, Kai Wei Chang, Chih Jen Lin, Sathiya Keerthi, and Sundararajan Sellamanickam. A dual coordinate descent method for large-scale linear SVM. In William Cohen, Andrew McCallum, and Sam Roweis, editors, *ICML*, pages 408–415. ACM, 2008.

Thorsten Joachims. A support vector method for multivariate performance measures. In *Proc. Intl. Conf. Machine Learning*, pages 377–384, San Francisco, California, 2005. Morgan Kaufmann Publishers.

Thorsten Joachims. Training linear SVMs in linear time. In *Proc. ACM Conf. Knowledge Discovery and Data Mining (KDD)*. ACM, 2006.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

Nikolas List and Hans Ulrich Simon. Svm-optimization and steepest-descent line search. In Sanjoy Dasgupta and Adam Klivans, editors, *Proc. Annual Conf. Computational Learning Theory*, LNCS. Springer, 2009.

Arkadi Nemirovski and David Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley and Sons, 1983.

Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. *Soviet Math. Docl.*, 269:543–547, 1983.

Yurii Nesterov. *Introductory Lectures On Convex Optimization: A Basic Course*. Springer, 2003.

Yurii Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.

Yurii Nesterov. Gradient methods for minimizing composite objective function. Technical Report 76, CORE Discussion Paper, UCL, 2007.

Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 2nd edition, 2006.

John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, 1998.

Bernhard Schölkopf and Alexander Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *Proc. Intl. Conf. Machine Learning*, 2007.

Soeren Sonnenburg, Vojtech Franc, Elad Yom-Tov, and Michele Sebag. Pascal large scale learning challenge. 2008. URL `http://largescale.ml.tu-berlin.de/workshop/`.

Sören Sonnenburg and Vojtěch Franc. COFFIN: A computational framework for linear SVMs. In *Proceedings of the International Conference on Machine Learning*, Haifa, 2010.

Choon Hui Teo, S. V. N. Vishwanthan, Alexander Smola, and Quoc Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11:311–365, January 2010.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Ann. Statist.*, 32(1):56–85, 2004.

Xinhua Zhang, Ankan Saha, and S. V. N. Vishwanathan. Regularized risk minimization by Nesterov's accelerated gradient methods: Algorithmic extensions and empirical studies. Technical report arXiv:1011.0472, 2010. URL `http://arxiv.org/abs/1011.0472`.

Xinhua Zhang, Ankan Saha, and S. V. N. Vishwanathan. Lower bounds on rate of convergence of cutting plane methods. In *Advances in Neural Information Processing Systems 23*, 2011a.

Xinhua Zhang, Ankan Saha, and S. V. N. Vishwanathan. Smoothing multivariate performance measures. In *Proceedings of UAI*, 2011b.

Xinhua Zhang, Ankan Saha, and S. V. N. Vishwanathan. Smoothopt, 2012. URL `http://webdocs.cs.ualberta.ca/~xinhua2/SmoothOPT`.

Tianyi Zhou, Dacheng Tao, and Xindong Wu. NESVM: a fast gradient method for support vector machines. In *Proc. Intl. Conf. Data Mining*, 2010.

Journal of Machine Learning Research 13 (2012) 3681-3724 Submitted 2/10; Revised 2/11; Published 11/12

Security Analysis of Online Centroid Anomaly Detection

Marius Kloft[*] KLOFT@TU-BERLIN.DE
Machine Learing Group
Technische Universität Berlin
Franklinstr. 28/29
10587 Berlin, Germany

Pavel Laskov PAVEL.LASKOV@UNI-TUEBINGEN.DE
Wilhelm-Schickard Institute for Computer Science
University of Tübingen
Sand 1
72076 Tübingen, Germany

Editor: Ingo Steinwart

Abstract

Security issues are crucial in a number of machine learning applications, especially in scenarios dealing with human activity rather than natural phenomena (e.g., information ranking, spam detection, malware detection, etc.). In such cases, learning algorithms may have to cope with manipulated data aimed at hampering decision making. Although some previous work addressed the issue of handling malicious data in the context of supervised learning, very little is known about the behavior of anomaly detection methods in such scenarios. In this contribution,[1] we analyze the performance of a particular method—online centroid anomaly detection—in the presence of adversarial noise. Our analysis addresses the following security-related issues: formalization of learning and attack processes, derivation of an optimal attack, and analysis of attack efficiency and limitations. We derive bounds on the effectiveness of a poisoning attack against centroid anomaly detection under different conditions: attacker's full or limited control over the traffic and bounded false positive rate. Our bounds show that whereas a poisoning attack can be effectively staged in the unconstrained case, it can be made arbitrarily difficult (a strict upper bound on the attacker's gain) if external constraints are properly used. Our experimental evaluation, carried out on real traces of HTTP and exploit traffic, confirms the tightness of our theoretical bounds and the practicality of our protection mechanisms.

Keywords: anomaly detection, adversarial, security analysis, support vector data description, computer security, network intrusion detection

1. Introduction

Machine learning methods have been instrumental in enabling novel data analysis applications. Numerous currently indispensable technologies—object recognition, user preference analysis, spam filtering, to name only a few—rely on accurate analysis of massive amounts of data. Unfortunately, the increasing *use* of machine learning methods gives rise to a threat of their *abuse*. A convinc-

∗. MK is also with Department of Brain and Cognitive Engineering, Korea University, Anam-dong, Seongbuk-gu, Seoul 136-713, Korea. Parts of the work were done while MK was with Computer Science Division and Department of Statistics, University of California, Berkeley, CA 94720-1758, USA.

1. A short version of this paper appeared previously in AISTATS 2010.

ing example of this phenomenon are emails that bypass spam protection tools. Abuse of machine learning can take on various forms. A malicious party may affect the training data, for example, when it is gathered from the real operation of a system and cannot be manually verified. Another possibility is to manipulate objects observed by a deployed learning system (test data) so as to bias its decisions in favor of an attacker. Yet another way to defeat a learning system is to send a large amount of nonsense data in order to produce an unacceptable number of false alarms and hence force the system's operator to turn it off. Manipulation of a learning system may thus range from simple cheating to complete disruption of its operation.

A potential insecurity of machine learning methods stems from the fact that they are usually not designed with adversarial input in mind. Starting from the mainstream computational learning theory (Vapnik, 1998; Schölkopf and Smola, 2002), a prevalent assumption is that training and test data are generated from the same fixed, but unknown, probability distribution. This assumption obviously does not hold for adversarial scenarios. Furthermore, even the recent work on learning with non-i.i.d. data (Steinwart et al., 2009; Mohri and Rostamizadeh, 2010) or differing training and test distributions (Sugiyama et al., 2007) is not necessarily appropriate for adversarial input, because in the latter case one must account for a specific worst-case difference while all the aforementioned papers assume that the data is generated stochastically.

Computer security is the most important application field in which robustness of learning algorithms against adversarial input is crucial. Modern security infrastructures are facing an increasing professionalization of attacks motivated by monetary profit. A widespread deployment of evasion techniques, such as encryption, obfuscation and polymorphism, is manifested in a rapidly increasing diversity of malicious software observed by security experts. Machine learning methods offer a powerful tool to counter a rapid evolution of security threats. For example, anomaly detection can identify unusual events that potentially contain novel, previously unseen exploits (Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007). Another typical application of learning methods is automatic signature generation which drastically reduces the time needed for development and deployment of attack signatures (Newsome et al., 2006; Li et al., 2006). Machine learning methods can also help researchers better understand the design of malicious software by using classification or clustering techniques together with special malware acquisition and monitoring tools (Bailey et al., 2007; Rieck et al., 2008).

In order for machine learning methods to be successful in security applications—and in general in any application where adversarial input may be encountered—they should be equipped with countermeasures against potential attacks. The current understanding of security properties of learning algorithms is rather incomplete. Earlier work in the PAC-framework addressed some scenarios in which training data is deliberately corrupted (Angluin and Laird, 1988; Littlestone, 1988; Kearns and Li, 1993; Auer, 1997; Bschouty et al., 1999). These results, however, are not connected to modern learning algorithms used in classification, regression and anomaly detection problems. Several examples of effective attacks were demonstrated in the context of specific security and spam detection applications (Lowd and Meek, 2005a; Fogla et al., 2006; Fogla and Lee, 2006; Perdisci et al., 2006; Newsome et al., 2006; Nelson et al., 2008), which motivated a recent work on taxonomization of such attacks (Barreno et al., 2006, 2008, 2010). However, it remains largely unclear whether machine learning methods can be protected against adversarial impact.

We believe that an unequivocal answer to the problem of the "security of machine learning" does not exist. Security guarantees cannot be established experimentally, because the notion of security addresses events that do not just happen on average but rather only potentially may happen. Hence,

a theoretical analysis of machine learning algorithms for adversarial scenarios is indispensable. It is hard to imagine, however, that such an analysis can offer meaningful results for any attack in every circumstance. Hence, to be a useful guide for practical applications of machine learning in adversarial environments, such an analysis must address *specific attacks against specific learning algorithms*. This is precisely the approach followed in this contribution.

The main focus of our work is the security analysis of online centroid anomaly detection against the so-called "poisoning" attacks. Centroid anomaly detection is a very simple method which has been widely used in computer security applications (e.g., Forrest et al., 1996; Warrender et al., 1999; Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007). In the learning phase, centroid anomaly detection computes the mean of all training data points:

$$\mathbf{c} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{x}_j \,.$$

Detection is carried out by computing the distance of a new example $\mathbf{x}$ from the centroid $\mathbf{c}$ and comparing it with an appropriate threshold:

$$f(\mathbf{x}) = \begin{cases} 1, & \text{if } ||\mathbf{x} - \mathbf{c}|| > \theta \\ 0, & \text{otherwise} \,. \end{cases}$$

Notice that all operations can be carried out using kernel functions—a standard trick known since the development of support vector machines and kernel PCA (Boser et al., 1992; Schölkopf et al., 1998; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004), which substantially increases the discriminative power of this method.

More often than not, anomaly detection algorithms are deployed in non-stationary environments and need to be regularly re-trained. Since the data is fed into the learning phase without any verification, an adversary has an opportunity to force a learning algorithm to learn a representation suitable for him. One particular kind of attack is the so-called "poisoning" in which specially crafted data points are injected to cause the decision function to misclassify a given malicious point as benign. This attack makes sense when an attacker does not have "write" permission to the training data, hence cannot manipulate it directly. Therefore, his goal is to trick an algorithm by merely using an "append" permission, by sending new data that looks innocuous to the learning algorithm but changes the algorithm's state in a way that favors the attacker; for example, forcing the algorithm to accept a specific attack point later during the testing stage.

The poisoning attack against online centroid anomaly detection has been considered by Nelson and Joseph (2006) for the case of an infinite window, that is, when a learning algorithm memorizes all data seen so far. Their main result was surprisingly optimistic: it was shown that the number of attack data points that must be injected grows exponentially as a function of the impact over a learned hypothesis. However, the assumption of an infinite window also hinders the ability of a learning algorithm to adjust to legitimate changes in the data distribution.

1.1 Contributions of This Work

As a main contribution, we present the security analysis of online centroid anomaly detection for the *finite window* case; that is, when only a fixed number of data points can be used at any time to form a hypothesis. We show that, in this case, an attacker can easily compromise a learning algorithm by

using only a linear amount of injected data unless additional constraints are imposed. As a further contribution, we analyze the algorithm under two additional constraints: (a) the fraction of the traffic controlled by an attacker is bounded by ν, and (b) the false positive rate induced by an attack is bounded by α. Both constraints can be motivated by an operational practice of anomaly detection systems. Overall, we significantly extend the analysis of Nelson and Joseph (2006) by considering a more realistic learning scenario, explicitly treating potential constraints on the attacker's part and providing tighter bounds.

Our analysis methodology follows the following framework, which can be used for any kind of *quantitative security analysis* of learning algorithms (Laskov and Kloft, 2009):

1. *Axiomatic formalization of the learning and the attack processes*. Such a formalization includes definitions of data sources and objective (risk) functions used by each party, as well as the attacker's goal. It specifies the knowledge available to an attacker, that is, whether he knows an algorithm, its parameters and internal state, and which data he can potentially manipulate.

2. *Specification of an attacker's constraints*. Potential constraints on the attacker's part may include: percentage of traffic under his control, amount of additional data that must be injected, an upper bound on the norm of a manipulated part, a maximal allowable false-positive rate (in case an attack must be stealthy), etc. Such constraints must be incorporated into the axiomatic formalization.

3. *Investigation of the optimal attack policy*. Such a policy may be long-term; that is, over multiple attack iterations, as well as short-term, for a single iteration. The investigation can be carried out either as a formal proof or numerically, by casting the search for an attack policy as an optimization problem.

4. *Bounding the attacker's gain under the optimal policy*. The ultimate goal of our analysis is to quantify the attacker's gain or effort under his optimal policy. Such an analysis may take different forms, for example calculation of the probability for an attack to succeed, estimation of the required number of attack iterations, calculation of the geometric impact of an attack (a shift towards an insecure state), etc.

Organization of this paper reflects the main steps of the proposed methodology. In the preliminary Section 2, the models of the learning and attack processes are introduced. The analytical part is arranged in two sections. In Sections 3 and 4, we address the steps (1), (3) and (4) under the assumption that the attacker has full control of the network traffic, first assuming that a learning algorithm can memorize all previously seen examples, followed by the finite memory case. Section 5 introduces the assumption that the attacker's control is limited to a fixed fraction of network traffic, as required in step (2). Another constraint (bounded false positive rate) is considered in Section 6. This section also removes the somewhat unrealistic assumption of Section 5 that all innocuous points are accepted by the algorithm. Analytic results are experimentally verified in Section 8 on real HTTP data and attacks used in intrusion detection systems. Some proofs and auxiliary technical material are presented in the Appendix. The notation used in the paper is summarized in Table 1.

r	centroid's radius
i	attack iteration index, $i \in \mathbb{N}_0$
$\mathbf{c}_i$	center of centroid in i-th attack iteration
$\mathbf{A}$	attack point
$\mathbf{a}$	attack direction vector
D_i	i-th relative displacement of a centroid in radii into direction of $\mathbf{a}$
n	number of patterns used for initial training of the centroid
f	attack strategy function
ν	fraction of adversarial training points
B_i	Bernoulli random variable
$\mathbf{x}_i$	training data
α	false alarm rate
I_S	indicator function of a set S

Table 1: Notation summary.

1.2 Poisoning and Related Attacks Against Learning Algorithms

For two-class learning problems, attacks against learning algorithms can be generally classified according to the following two criteria (the terminology in the taxonomy of Barreno et al. (2006) is given in brackets):

- whether an attack is staged during the training (causative) or the deployment of an algorithm (exploratory), or
- whether an attack attempts to increase the false negative or the false positive rate at the deployment stage (integrity/availability).

The poisoning attack addressed in our work can be classified as a causative integrity attack. This scenario is quite natural, for example, in web application scenarios in which the data on a server can be assumed to be secure but the subsequent injection of adversarial data cannot be easily prevented. Other common attack types are the mimicry attack—alteration of malicious data to resemble innocuous data (an exploratory integrity attack) and the "red herring" attack—sending junk data that causes false alarms (an exploratory availability attack). Attacks of the latter two kinds are beyond the scope of our investigation.

As a final remark, we must consider the extent to which the attacker is familiar with the learning algorithm and trained model. One of the key principles of computer security, known as *Kerckhoff's principle*, is that the robustness of any security instrument must not depend on keeping its operational functionality secret. Similar to modern cryptographic methods, we must assume that the attacker knows which machine learning algorithm is deployed and how it operates (he can even use machine learning to reverse engineer deployed classifiers, as shown by Lowd and Meek, 2005b). However, it may be more difficult for an attacker to obtain the training data or the particular learned model. In the case of anomaly detection, it is relatively easy for an attacker to retrieve the learned model: it suffices to sniff on the same application that is protected by an algorithm to get approxi-

mately the same innocuous data the algorithm is trained on. Hence, we will assume that the attacker has precise knowledge of the trained model throughout the attack.

2. Learning and Attack Models

Before proceeding with the analysis, we first present the precise models of the learning and the attack processes.

2.1 Centroid Anomaly Detection

Given a data set $X_0 = \{\mathbf{x}_0^{(1)}, \ldots, \mathbf{x}_0^{(n)}\} \subset \mathbb{R}^d$, the goal of anomaly detection (also often referred to as "novelty detection") is to determine whether an example $\mathbf{x}$ is unlikely to have been generated by the same distribution as the set X_0. A natural way to perform anomaly detection is to estimate the probability density function of the distribution from which the set X_0 was drawn and mark $\mathbf{x}$ as anomalous if it comes from a region with low density. In general, however, density estimation is a difficult problem, especially in high dimensions. A large amount of data is usually needed to reliably estimate the density in all regions of the space. For anomaly detection, knowing the density in the entire space is superfluous, as we are only interested in deciding whether a specific point lies within a "sparsely populated" area. Hence several direct methods have been proposed for anomaly detection, for example, one-class SVM (Schölkopf et al., 2001), support vector data description (SVDD) (Tax and Duin, 1999a,b), and density level set estimation (Polonik, 1995; Tsybakov, 1997; Steinwart et al., 2005). A comprehensive survey of anomaly detection techniques can be found in Markou and Singh (2003a,b).

In the centroid anomaly detection, an Euclidean distance from the empirical mean of the data is used as a measure of abnormality:

$$f(\mathbf{x}) = \left|\left|\mathbf{x} - \frac{1}{n}\sum_{j=1}^{n} \mathbf{x}_0^{(j)}\right|\right|.$$

If a hard decision is desired instead of a soft abnormality score, the data point is considered anomalous if its score exceeds a fixed threshold r.

Despite its straightforwardness, a centroid model can represent complex density level sets using a kernel mapping (Müller et al., 2001; Schölkopf and Smola, 2002) (see Figure 1). Centroid anomaly detection can be seen as a special case of the SVDD with outlier fraction $\eta = 1$ and of the Parzen window density estimator (Parzen, 1962) with Gaussian kernel function $k(\mathbf{x},\mathbf{y}) = \frac{1}{(\sqrt{2\pi}\sigma)^d}\exp(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|^2)$.

Centroid anomaly detection has been successfully used in a variety of anomaly detection applications such as intrusion detection (Hofmeyr et al., 1998; Yeung and Chow, 2002; Laskov et al., 2004a; Wang and Stolfo, 2004; Rieck and Laskov, 2006; Wang et al., 2006; Rieck and Laskov, 2007), wireless sensor networks (Rajasegarar et al., 2007) and jet engine vibration data analysis (Nairac et al., 1999). It has been shown (see, for example, Section 4.1 in Shawe-Taylor and Cristianini, 2004; Vert and Vert, 2006) that even in high-dimensional spaces induced by nonlinear feature maps, the empirical estimator of the center of mass of the data is stable and the radius of the sphere anchored at the center of mass is related to the level set of the corresponding probability density.

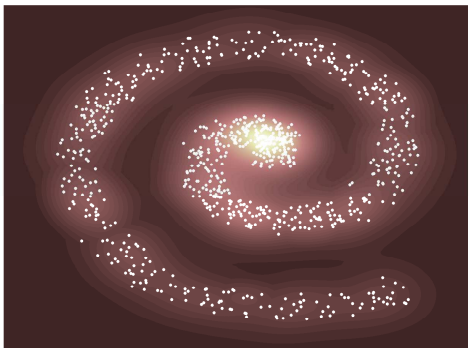


Figure 1: Illustration of the density level estimation using a centroid model with a non-linear kernel.

2.2 Online Anomaly Detection

The majority of anomaly detection applications have to deal with non-stationary data. This is especially typical for computer security, as the processes being monitored usually change over time. For example, network traffic characteristics are strongly influenced by the time of the day; system call sequences depend on the applications running on a computer. The model of normality constructed by anomaly detection algorithms hence needs to be regularly updated, in the extreme case—after the arrival of each data point. Obviously, retraining the model from scratch every time is computationally inefficient; however, the incorporation of new data points and the removal of irrelevant ones can be done with acceptable effort (e.g., Laskov et al., 2006).

For centroid anomaly detection, we assume that the initial state of the learner comprises a center $\mathbf{c}_0$ and a given radius r. We further assume that this state has been obtained from purely innocuous data. Whenever a new data point $\mathbf{x}_i$ arrives at iteration $i \in \mathbb{N}$, the learner's center of mass $\mathbf{c}_i$ is updated if and only if the new data point is considered non-anomalous; otherwise, it is rejected and not used for re-training. The radius r stays fixed over time.[2] Recalculation of the center of mass is straightforward and requires $O(1)$ work. If all examples are "memorized", that is, $X_i = X_{i-1} \cup \{x_i\}$,[3] the update is computed as

$$\mathbf{c}_{i+1} = \left(1 - \frac{1}{n+i}\right)\mathbf{c}_i + \frac{1}{n+i}\mathbf{x}_i\,. \tag{1}$$

For a finite horizon, that is, $\forall i : |X_i| = n$, in each iteration i, some previous example $\mathbf{x}_{\text{old}} \in X_i$ is replaced by the newly arriving $\mathbf{x}_i$, and the update is thus performed as

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}(\mathbf{x}_i - \mathbf{x}_{\text{old}})\,. \tag{2}$$

The update formula can be generalized to $\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{\kappa}{i}(\mathbf{x}_i - \mathbf{x}_{\text{old}})$, with fixed $\kappa \geq 0$. This changes the bounds in the upcoming analysis only by a constant factor in this case, which is negligible.

2. From the operational standpoint, this assumption is reasonable as the radius serves as a threshold and is usually set and manually tuned after inspection of (false) alarms. Treating the radius as a model parameter would necessitate complex rules for its update and would open the system to poisoning attacks against the radius.
3. Note that the data need not be physically stored, as the contributions from individual data points are accumulated in a current location of the center.

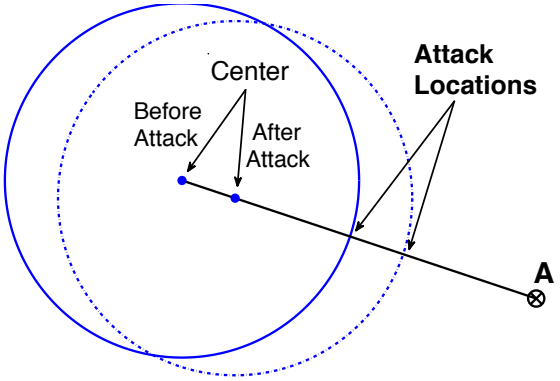


Figure 2: Illustration of a poisoning attack. By iteratively inserting malicious training points an attacker can gradually "drag" the centroid into a direction of an attack. The radius is assumed to stay fixed over time. Figure taken from Nelson and Joseph (2006).

Various strategies can be used to determine the "least relevant" point $\mathbf{x}_{\text{old}}$ to be removed from the working set:

(a) oldest-out: the point with the oldest timestamp is removed.

(b) random-out: a randomly chosen point is removed.

(c) nearest-out: the nearest-neighbor of the new point $\mathbf{x}$ is removed.

(d) average-out: the center of mass is removed. The new center of mass is recalculated as $\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}(\mathbf{x}_i - \mathbf{c}_i)$, which is equivalent to Equation (1) with constant n.

The strategies (a)–(c) require the storage of all points in the working set, whereas the strategy (d) can be implemented by holding only the center of mass in memory.

2.3 Poisoning Attack

The goal of a poisoning attack is to force an algorithm, at some learning iteration i, to accept the attack point $\mathbf{A}$ that lies outside of the normality ball, that is, $||\mathbf{A} - \mathbf{c}_i|| > r$. It is assumed that the attacker knows the anomaly detection algorithm and the training data. Furthermore, the attacker cannot modify any existing data except for adding new points. Although the attacker can send any data point, it obviously only makes sense for him to send points lying within the current sphere as they are otherwise discarded by the learning algorithm. As illustrated in Figure 2, the poisoning attack amounts to injecting specially crafted points that are accepted as innocuous but shift the center of mass in the direction of the attack point until the latter appears innocuous as well.

What points should be used by an attacker in order to subvert online anomaly detection? Intuitively, one can expect that the optimal one-step displacement of the center of mass is achieved by placing attack point $\mathbf{x}_i$ along the line connecting $\mathbf{c}$ and $\mathbf{A}$ such that $||\mathbf{x}_i - \mathbf{c}|| = r$. A formal proof of the optimality of such a strategy and the estimation of its efficiency constitutes the first step of security analysis of online anomaly detection and is provided in the following sections for various scenarios. In our analysis, we will use the following measure of attack effectiveness.

Definition 1 (Relative displacement) *Let* $\mathbf{A}$ *be an attack point and* $\mathbf{a} = \frac{\mathbf{A}-\mathbf{c}_0}{||\mathbf{A}-\mathbf{c}_0||}$ *be the attack direction unit vector. The* i-th relative displacement D_i *of an online centroid learner is defined as*

$$D_i = \frac{(\mathbf{c}_i - \mathbf{c}_0)\cdot\mathbf{a}}{r}\,.$$

The relative displacement measures the length of the projection of accrued change to $\mathbf{c}_i$ onto the attack direction $\mathbf{a}$ in terms of the radius of the normality ball. Note that the displacement is a relative quantity, that is, we may without loss of generality translate the coordinate system so that the center of mass lies at the origin (i.e., $\mathbf{c}_0 = \mathbf{0}$) and subsequently isotropically normalize the space so that the centroid has unit radius $r = 1$. After this transformation, the formula for the displacement can be simplified to

$$D_i = \mathbf{c}_i \cdot \mathbf{a}\,.$$

Definition 2 *An attack strategy that maximizes the displacement* D_i *in each iteration* i *is referred to as* greedy-optimal.

3. Attack Effectiveness for Infinite Horizon Centroid Learner

The effectiveness of a poisoning attack for the infinite horizon learner has been analyzed in Nelson and Joseph (2006). We provide an alternative proof that follows the main steps of the framework proposed in Section 1.1.

Theorem 3 *The* i*-th relative displacement* D_i *of the online centroid learner with an infinite horizon under a poisoning attack is bounded by*

$$D_i \leq \ln\left(1 + \frac{i}{n}\right), \tag{3}$$

where i *is the number of attack points and* n *is the number of initial training points.*

Proof We first determine the greedy-optimal attack strategy and then bound the attack progress.

(a) Let $\mathbf{A}$ be an attack point and denote by $\mathbf{a}$ the corresponding attack direction vector. Let $\{\mathbf{x}_i | i \in \mathbb{N}\}$ be adversarial training points. The center of mass at the ith iteration is given in the following recursion:

$$\mathbf{c}_{i+1} = \left(1 - \frac{1}{n+i}\right)\mathbf{c}_i + \frac{1}{n+i}\mathbf{x}_{i+1}\,, \tag{4}$$

with initial value $\mathbf{c}_0 = 0$. By the construction of the poisoning attack, $||\mathbf{x}_i - \mathbf{c}_i|| \leq r$, which is equivalent to $\mathbf{x}_i = \mathbf{c}_i + \mathbf{b}_i$ with $||\mathbf{b}_i|| \leq r$. Equation (4) can thus be transformed into

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n+i}\mathbf{b}_i\,.$$

Taking the scalar product with $\mathbf{a}$ and using the definition of a relative displacement, we obtain:

$$D_{i+1} = D_i + \frac{1}{n+i}\cdot\frac{\mathbf{b}_i\cdot\mathbf{a}}{r}\,, \tag{5}$$

with $D_0 = 0$. The right-hand side of the Equation (5) is clearly maximized under $||\mathbf{b}_i|| \leq 1$ by setting $\mathbf{b}_i = r\mathbf{a}$. Thus the greedy-optimal attack is given by

$$\mathbf{x}_i = \mathbf{c}_i + r\mathbf{a}\,. \tag{6}$$

(b) Plugging the optimal strategy $\mathbf{b}_i = r\mathbf{a}$ into Equation (5), we obtain:

$$D_{i+1} = D_i + \frac{1}{n+i}\,.$$

This recursion can be explicitly solved, taking into account that $D_0 = 0$, resulting in:

$$D_i = \sum_{k=1}^{i} \frac{1}{n+k} = \sum_{k=1}^{n+i} \frac{1}{k} - \sum_{k=1}^{n} \frac{1}{k}\,.$$

Inserting the upper bound on the harmonic series, $\sum_{k=1}^{m} \frac{1}{k} = \ln(m) + \varepsilon_m$, with $\varepsilon_m \geq 0$, into the above formula and noting that ε_m is monotonically decreasing, we obtain:

$$D_i \leq \ln(n+i) - \ln(n) = \ln\left(\frac{n+i}{n}\right) = \ln\left(1 + \frac{i}{n}\right)\,,$$

which completes the proof. ■

Since the bound in Equation (3) is monotonically increasing, we can invert it to obtain a bound on the effort needed by an attacker to achieve his goal:

$$i \geq n \cdot (\exp(D_i) - 1)\,.$$

It can be seen that the effort needed to poison an online centroid learner is *exponential* in terms of the relative displacement of the center of mass.[4] In other words, an attacker's effort grows prohibitively fast with respect to the separability of the attack from innocuous data. For a kernelized centroid learner, the greedy-optimal attack may not be valid, as there may not exist a point in the input space corresponding to the optimal attack image in the feature space. However, an attacker can construct points in the input space that are close enough to the greedy-optimal point for the attack to succeed, with a moderate constant cost factor; cf., Section 8.5.

4. Poisoning Attack against Finite Horizon Centroid Learner

The optimistic result presented in Section 3 is unfortunately not quite useful. In practice, memorization of all training points essentially defeats the main purpose of an online algorithm, that is, its ability to adjust to non-stationarity. Hence it is important to understand how the removal of data points from the working set affects the security of online anomaly detection. To this end, the specific removal strategies presented in Section 2.2 must be considered. The analysis can be carried out theoretically for the average-out and random-out update rules; for the nearest-out rule, an optimal attack can be stated as an optimization problem and the attack effectiveness can be analyzed empirically.

4. Even constraining a maximum number of online update steps does not remove this bound's exponential growth (Nelson and Joseph, 2006).

4.1 Poisoning Attack for Average-out, Random-out and Oldest-out Rules

We begin our analysis with the average-out learner which follows exactly the same update rule as the infinite-horizon online centroid learner with the exception that the window size n remains fixed instead of growing indefinitely (cf. Section 2.2). Despite the similarity to the infinite-horizon case, the result presented in the following theorem is surprisingly pessimistic.

Theorem 4 *The i-th relative displacement D_i of the online centroid learner with the average-out update rule under a worst-case optimal poisoning attack is*

$$D_i = \frac{i}{n},$$

where i is the number of attack points and n is the training window size.

Proof The proof is similar to the proof of Theorem 3. By explicitly writing out the recurrence between subsequent displacements, we conclude that the greedy-optimal attack is also attained by placing an attack point along the line connecting $\mathbf{c}_i$ and $\mathbf{A}$ at the edge of the sphere (cf. Equation (6)):

$$\mathbf{x}_i = \mathbf{c}_i + r\mathbf{a}\,.$$

It follows that the relative displacement under the greedy-optimal attack is

$$D_{i+1} = D_i + \frac{1}{n}\,.$$

Since this recurrence is independent of the running index i, the displacement is simply accumulated over each iteration, which yields the bound of the theorem. ■

One can see that, unlike the logarithmic bound in Theorem 3, the average-out learner is characterized by a linear bound on the displacement. As a result, an attacker only needs a linear number of injected points—instead of an exponential one—in order to subvert an average-out learner. This cannot be considered secure.

A similar result, in terms of the *expectation* of the relative displacement, can be obtained for the random-out removal strategy. The proof is based on the observation that in expectation, the average-out rule is equivalent to the random-out rule. The oldest-out rule can also be handled similarly to the average-out rule by observing that in both cases some fixed point known in advance is removed from a working set, which allows an attacker to easily find an optimal attack point.

4.2 Poisoning Attack for Nearest-out Rule

One might expect that the nearest-out strategy poses a stronger challenge to an attacker as it tries to retain working set diversity by eliminating the most similar data to the new point. However, even this strategy can be broken with a feasible amount of work if the attacker follows a greedy-optimal strategy. The latter is the subject of our investigation in this section.

4.2.1 GREEDY-OPTIMAL ATTACK

The *greedy-optimal* attack should provide a maximal gain for an attacker in a single iteration. For the infinite-horizon learner (and hence also for the average-out learner, as it uses the same recurrence in a proof), it is possible to show that the greedy-optimal attack yields the maximum gain for the entire sequence of attack iterations; that is, it is (globally) optimal. For the nearest-out learner, it is hard to analyze the full sequence of attack iterations, hence we limit our analysis to a single-iteration gain. Empirically, even a greedy-optimal attack turns out to be effective.

To construct a greedy-optimal attack, we partition the sphere spanned by the centroid into Voronoi cells V_j centered at the training data points $\mathbf{x}_j$, $j = 1,\dots,n$. Each Voronoi cell comprises points for which $\mathbf{x}_j$ is the nearest neighbor. Whenever a new training point "falls into" the sphere, the center of the corresponding Voronoi cell is removed according to the nearest-out rule.

The optimal attack strategy is now straightforward. First, we determine the optimal attack location within each cell. This can be done by solving the following optimization problem for a fixed $\mathbf{x}_j$:

Optimization Problem 5 (greedy-optimal attack)

$$\mathbf{x}_j^* = \operatorname*{argmax}_{\mathbf{x}} \; y_j(\mathbf{x}) := (\mathbf{x}-\mathbf{x}_j)\cdot\mathbf{a} \tag{7}$$

$$s.t. \;\; \|\mathbf{x}-\mathbf{x}_j\| \le \|\mathbf{x}-\mathbf{x}_k\|, \quad \forall k = 1,\dots,n \tag{8}$$

$$\|\mathbf{x}-\tfrac{1}{n}\textstyle\sum_{k=1}^n \mathbf{x}_k\| \le r\,. \tag{9}$$

The objective of the optimization problem (7) reflects the goal of maximizing the projection of $\mathbf{x}-\mathbf{x}_j$ onto the attack direction $\mathbf{a}$. The constraint (8) stipulates that the point $\mathbf{x}_j$ is the nearest neighbor of $\mathbf{x}$. The constraint (9), when active, enforces that no solution lies outside of the sphere.

The geometry of the greedy-optimal attack is illustrated in Figure 3. An optimal attack point is placed at the "corner" of a Voronoi cell (including possibly a round boundary of the centroid) to cause the largest displacement of the centroid along the attack direction.

Once the candidate attack locations are found for each of the n Voronoi cells, the one that has the highest value of the objective function $y_j(\mathbf{x}_j^*)$ is injected and the respective center $\mathbf{x}_{j^*}$ of the Voronoi cell is expunged from the training set:

$$j^* = \operatorname{argmax}_{j\in 1,\dots,n} y_j(\mathbf{x}_j^*)\,. \tag{10}$$

The optimization problem (7) can be simplified by plugging in the definition of the Euclidean norm. In the resulting optimization problem, all but one of the norm constraints are reduced to simpler linear constraints:

$$\begin{aligned} \mathbf{x}_j^* = \operatorname*{argmax}_{\mathbf{x}} \; & (\mathbf{x}-\mathbf{x}_j)\cdot\mathbf{a} \\ \text{s.t. } & 2(\mathbf{x}_k-\mathbf{x}_j)\cdot\mathbf{x} \le \mathbf{x}_k\cdot\mathbf{x}_k - \mathbf{x}_j\cdot\mathbf{x}_j, \quad \forall k = 1,\dots,n \\ & \mathbf{x}\cdot\mathbf{x} - \tfrac{2}{n}\textstyle\sum_{k=1}^n \mathbf{x}\cdot\mathbf{x}_k \le r^2 - \frac{1}{n^2}\sum_{k,l=1}^n \mathbf{x}_k\cdot\mathbf{x}_l\,. \end{aligned} \tag{11}$$

Due to the quadratic constraint, the inner optimization task is not as simple as a linear or a quadratic program. However, several standard optimization packages; for example, CPLEX or MOSEK,

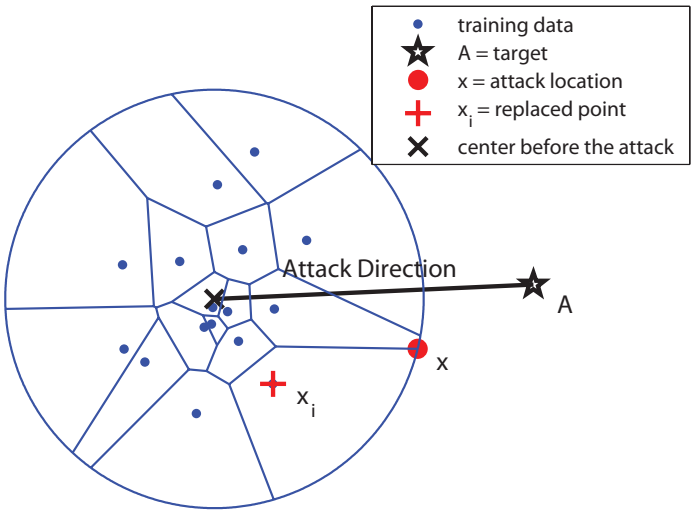


Figure 3: The geometry of a poisoning attack for the nearest-out rule. A greedy-optimal attack is injected at the boundary of the respective Voronoi cell.

can optimize such quadratically constrained linear programs (QCLP) with high efficiency, especially when there is only a single quadratic constraint. Alternatively, one can use specialized algorithms for linear programming with a single quadratic constraint (van de Panne, 1966; Martein and Schaible, 2005) or convert the quadratic constraint to a second-order cone (SOC) constraint and use general-purpose conic optimization methods.

4.2.2 IMPLEMENTATION OF A GREEDY-OPTIMAL ATTACK

For the practical implementation of the attack specified by problem (11), some additional processing steps must be carried out.

A point can become "immune" to a poisoning attack, if starting from some iteration i' its Voronoi cell does not overlap with the hypersphere of radius r centered at $\mathbf{c}_{i'}$. The quadratic constraint (9) is never satisfied in this case, and the inner optimization problem (7) becomes infeasible. These immune points remain in the working set forever and slow down the attack's progress. To avoid this situation, an attacker must keep track of all optimal solutions $\mathbf{x}_j^*$ of the inner optimization problems. If an online update would cause some Voronoi cell V_j to completely slip out of the hypersphere an attacker should ignore the outer loop decision (10) and expunge $\mathbf{x}_j$ instead of $\mathbf{x}_{j^*}$.

A significant speedup can be attained by avoiding the solution of unnecessary QCLP problems. Let $S = \{1,\ldots,j-1\}$ and α_S be the current best solution of the outer loop problem (10) over the set S. Let y_{α_S} be the corresponding objective value of an inner optimization problem (11). Consider the following auxiliary quadratic program (QP):

$$\max_{\mathbf{x}} \ \|\mathbf{x} - \tfrac{1}{n}\textstyle\sum_{k=1}^{n} \mathbf{x}_k\| \tag{12}$$

$$\text{s.t. } 2(\mathbf{x}_k - \mathbf{x}_j)\cdot\mathbf{x} \le \mathbf{x}_k\cdot\mathbf{x}_k - \mathbf{x}_j\cdot\mathbf{x}_j, \quad \forall k = 1,\ldots,n \tag{13}$$

$$(\mathbf{x} - \mathbf{x}_j)\cdot\mathbf{a} \ge y_{\alpha_S}. \tag{14}$$

Its feasible set comprises the Voronoi cell of $\mathbf{x}_j$, defined by constraint (13), further reduced by constraint (14) to the points that improve the current value y_{α_s} of the global objective function. If the objective function value provided by the solution of the auxiliary QP (12) exceeds r then the solution of the local QCLP (11) does not improve the global objective function y_{α_s}. Hence the expensive QCLP optimization can be skipped.

4.2.3 ATTACK EFFECTIVENESS

To evaluate the effectiveness of the greedy-optimal attack, we perform a simulation on artificial geometric data. The goal of this simulation is to investigate the behavior of the relative displacement D_i during the progress of the greedy-optimal attack.

An initial working set of size $n = 100$ is sampled from a d-dimensional Gaussian distribution with unit covariance (experiments are repeated for various values of $d \in \{2, ..., 100\}$). The radius r of the online centroid learner is chosen such that the expected false positive rate is bounded by $\alpha = 0.001$. An attack direction $\mathbf{a}$, $\|\mathbf{a}\| = 1$, is chosen randomly, and 500 attack iterations ($5 * n$) are generated using the procedure presented in Sections 4.2.1–4.2.2. The relative displacement of the center in the direction of the attack is measured at each iteration. For statistical significance, the results are averaged over 10 runs.

Figure 4(a) shows the observed progress of the greedy-optimal attack against the nearest-out learner and compares it to the behavior of the theoretical bounds for the infinite-horizon learner (the bound of Nelson and Joseph, 2006) and the average-out learner. The attack effectiveness is measured for all three cases by the relative displacement as a function of the number of iterations. Plots for the nearest-out learner are presented for various dimensions d of the artificial problems tested in simulations. The following observations can be made from the plot provided in Figure 4(a).

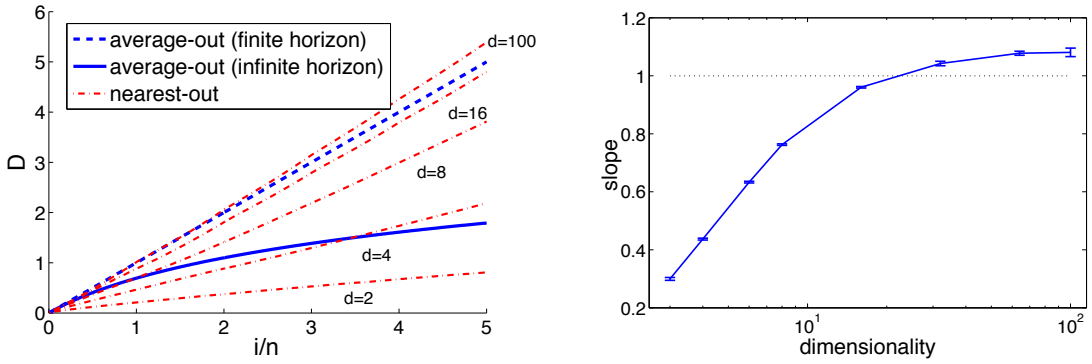
First, the attack progress, that is, the functional dependence of the relative displacement of the greedy-optimal attack against the nearest-out learner with respect to the number of iterations, is *linear*. Hence, contrary to our initial intuition, the removal of nearest neighbors of incoming points does not lead to better security against poisoning attacks.

Second, the slope of the linear attack progress *increases with the dimensionality of the problem*. For low dimensionality, the relative displacement of the nearest-out learner is comparable, in absolute terms, with that of the infinite-horizon learner. For high dimensionality, the nearest-out learner becomes even less secure than the simple average-out learner. By increasing the dimensionality beyond $d > n$ the attack effectiveness cannot be increased. Mathematical reasons for this behavior are investigated in Section A.1.

A further illustration of the behavior of the greedy-optimal attack is given in Figure 4(b), showing the dependence of the average attack slope on the dimensionality. One can see that the attack slope increases logarithmically with the dimensionality and wanes out to a constant factor after the dimensionality exceeds the number of training data points. A theoretical explanation of the observed experimental results is given in Appendix A.1.2.

4.3 Concluding Remarks

To summarize our analysis for the case of the attacker's full control over the training data, we conclude that an optimal poisoning attack successfully subverts a finite-horizon online centroid learner for all outgoing point selection rules. This conclusion contrasts with the analysis of the infinite-horizon learner carried out in Barreno et al. (2006) that yields a logarithmic attack progress.



(a) (b)

Figure 4: Effectiveness of the poisoning attack for the nearest-out rule as a function of input space dimensionality. (a) The displacement of the centroid along the attack direction grows linearly with the number of injected points. Upper bounds on the displacement of the average-out rule are plotted for comparison. (b) The slope of the linear growth increases with the input space dimensionality.

As a compromise, one can in practice choose a large working set size n, which reduces the slope of a linear attack progress.

Among the different outgoing point selection rules, the nearest-out rule presents the most challenges for the implementation of an optimal attack. Some approximations make such an attack feasible while still maintaining a reasonable progress rate. The key factor for the success of a poisoning attack in the nearest-out case lies in high dimensionality of the feature space. The progress of an optimal poisoning attack depends on the size of the Voronoi cells induced by the training data points. The size of the Voronoi cells is related linearly to the volume of the sphere corresponding to the attack's feasible region (see Appendix A.1.2 for a theoretical discussion of this effect). With the increasing dimensionality of the feature space, the volume of the sphere increases exponentially, which leads to a higher attack progress rate.

In the following sections, we analyze two additional factors that affect the progress of a poisoning attack. First, we consider the case when the attacker controls only a fixed fraction ν of the training data. Subsequently we analyze a scenario in which an attacker is not allowed to exceed a certain false positive rate α; for example, by stopping online learning when a high false positive rate is observed. It will be shown that both of these possible constraints significantly reduce the effectiveness of poisoning attacks.

5. Poisoning Attack with Limited Bandwidth Constraint

Until now we assumed that an attacker has unlimited control over the training data; that is, in the worst case, all the data seen by the learner may be adversarial. This assumption is too pessimistic, as typically a deployed learning algorithm would also receive normal data during its operation. We would now analyze centroid anomaly detection under the assumption that only a fraction ν of the training data is adversarial. Our goal is to analyze the impact of this fraction of control on the

difficulty of an attack. The choice of realistic values of ν is application-specific. For simplicity, we restrict ourselves to the average-out learner as we have seen that it only differs by a constant from the nearest-out one and is equivalent in expectation to the random-out one.

5.1 Learning and Attack Models

As discussed in Section 2, we assume that the initial online centroid learner is centered at the position $\mathbf{c}_0$ and has a fixed radius r (without loss of generality $\mathbf{c}_0 = \mathbf{0}$ and $r = 1$; cf. discussion after Definition 1 in Section 2). At each iteration a new training point arrives—which is either inserted by an adversary or drawn independently from the distribution of innocuous points—and a new center of mass $\mathbf{c}_i$ is calculated. The mixing of innocuous and attack points is modeled by a Bernoulli random variable with the parameter ν which denotes the probability that an adversarial point is presented to the learner. Adversarial points $\mathbf{A}_i$ are chosen according to the attack function f depending on the actual state of the learner $\mathbf{c}_i$. The innocuous pool is modeled by a probability distribution from which the innocuous points $\mathbf{x}_i$ are independently drawn. We assume that the expectation of innocuous points $\mathbf{x}_i$ coincides with the initial center of mass: $E(\mathbf{x}_i) = \mathbf{c}_0$.

For simplicity, we make one additional assumption in this chapter: *all innocuous points are accepted by the learner at any time of the attack independent of their actual distance to the center of mass*. In the absence of this assumption, we would need special treatment for the case that a truly innocuous point is disregarded by the learner as the center of mass is getting displaced by the attack. In the next section we drop this assumption so that the learner only accepts points which fall within the actual radius.

The described probabilistic model is formalized by the following axiom.

Axiom 6 *$\{B_i | i \in \mathbb{N}\}$ are independent Bernoulli random variables with parameter $\nu > 0$. $\mathbf{x}_i$ are i.i.d. random variables in a Euclidean space $\mathbb{R}^d$, drawn from a fixed but unknown distribution $P_{\mathbf{x}}$, satisfying $E(\mathbf{x}) = \mathbf{0}$ and $\|\mathbf{x}\| \leq r \overset{w.l.o.g.}{=} 1$. B_i and $\mathbf{x}_j$ are mutually independent for each i, j. $f : \mathbb{R}^d \to \mathbb{R}^d$ is a function $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$ that we call* attack strategy. *$\{\mathbf{c}_i | i \in \mathbb{N}\}$ is a collection of random vectors such that w.l.o.g. $\mathbf{c}_0 = 0$ and*

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n} \left(B_i f(\mathbf{c}_i) + (1 - B_i)\mathbf{x}_i - \mathbf{c}_i \right) . \tag{15}$$

Moreover, we denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

For simplicity of notation, in this section we refer to a collection of random vectors $\{\mathbf{c}_i | i \in \mathbb{N}\}$ satisfying Axiom 6 as an *online centroid learner*. Any function f satisfying Axiom 6 is called an *attack strategy*. The attack strategy is a function that maps a vector (the center) to an attack location.

According to the above axiom, the adversary's attack strategy is formalized by an *arbitrary* function f. This raises the question of which attack strategies are optimal in the sense that an attacker reaches his goal of concealing a predefined attack direction vector in a minimal number of iterations. As in the previous sections, an attack's progress is measured by projecting the current center of mass onto the attack direction vector:

$$D_i = \mathbf{c}_i \cdot \mathbf{a} .$$

Attack strategies maximizing the displacement D_i in each iteration i are referred to as *greedy-optimal attack strategies*.

5.2 Greedy-Optimal Attack

The following result characterizes the greedy-optimal attack strategy for the model specified in Axiom 6.

Proposition 7 *Let* $\mathbf{a}$ *be an attack direction vector. Then the* greedy-optimal attack strategy f *against the online centroid learner is given by*

$$f(\mathbf{c}_i) := \mathbf{c}_i + \mathbf{a}\,. \tag{16}$$

Proof Since by Axiom 6 we have $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$, any valid attack strategy can be written as $f(\mathbf{x}) = \mathbf{x} + g(\mathbf{x})$, such that $\|g\| \leq r = 1$. It follows that

$$\begin{aligned} D_{i+1} &= \mathbf{c}_{i+1} \cdot \mathbf{a} \\ &= \left(\mathbf{c}_i + \frac{1}{n}\left(B_i f(\mathbf{c}_i) + (1-B_i)\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a} \\ &= D_i + \frac{1}{n}(B_i D_i + B_i g(\mathbf{c}_i)\cdot\mathbf{a} + (1-B_i)\mathbf{x}\cdot\mathbf{a} - D_i)\,. \end{aligned}$$

Since $B_i \geq 0$, the greedy-optimal attack strategy should maximize $g(\mathbf{c}_i)\cdot\mathbf{a}$ subject to $||g(\mathbf{c}_i)|| \leq 1$. The maximum is clearly attained by setting $g(\mathbf{c}_i) = \mathbf{a}$. ■

Note that the displacement measures the projection of the change of the centroid onto the attack direction vector. Hence it is not surprising that the optimal attack strategy is independent of the actual position of the learner.

5.3 Attack Effectiveness

The effectiveness of the greedy-optimal attack in the limited control case is characterized in the following theorem.

Theorem 8 *For the displacement* D_i *of the centroid learner under an optimal poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) \;&&=\; (1-a_i)\frac{\nu}{1-\nu} \\ &\text{(b)} \qquad \mathrm{Var}(D_i) \;&&\leq\; \gamma_i\left(\frac{\nu}{1-\nu}\right)^2 + \delta_n\,, \end{aligned}$$

where $a_i := \left(1 - \frac{1-\nu}{n}\right)^i$, $b_i = \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right)^i$, $\gamma_i = a_i - b_i$, *and* $\delta_n := \frac{\nu^2 + (1-b_i)}{(2n-1)(1-\nu)^2}$.

Proof (a) Inserting the greedy-optimal attack strategy of Equation (16) into Equation (15) of Axiom 6, we have:

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}\left(B_i\left(\mathbf{c}_i + \mathbf{a}\right) + (1-B_i)\mathbf{x}_i - \mathbf{c}_i\right)\,,$$

which can be rewritten as:

$$\mathbf{c}_{i+1} = \left(1 - \frac{1-B_i}{n}\right)\mathbf{c}_i + \frac{B_i}{n}\mathbf{a} + \frac{(1-B_i)}{n}\mathbf{x}_i\,. \tag{17}$$

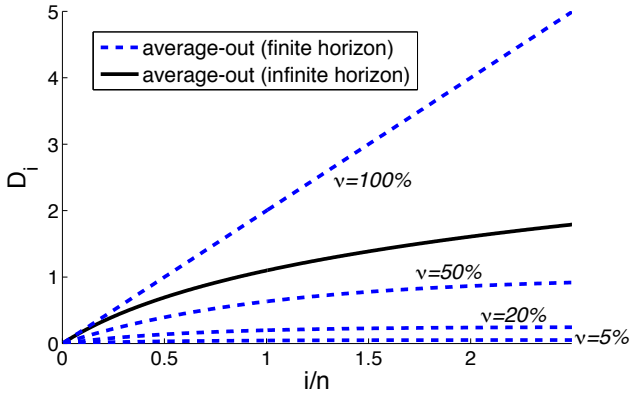


Figure 5: Theoretical behavior of the displacement of a centroid under a poisoning attack for a bounded fraction of traffic under attacker's control. The infinite horizon bound of Nelson et al. is shown for comparison (solid line).

Taking the expectation on the latter equation and noting that by Axiom 6, $E(\mathbf{x}_i) = 0$ and $E(B_i) = \nu$, we have

$$E(\mathbf{c}_{i+1}) = \left(1 - \frac{1-\nu}{n}\right) E(\mathbf{c}_i) + \frac{\nu}{n}\mathbf{a},$$

which by the definition of the displacement translates to

$$E(D_{i+1}) = \left(1 - \frac{1-\nu}{n}\right) E(D_i) + \frac{\nu}{n}.$$

The statement (a) follows from the latter recursive equation by Proposition 17 (formula of the geometric series). For the more demanding proof of (b), see Appendix A.2. ■

The following corollary shows the asymptotic behavior of the above theorem.

Corollary 9 *For the displacement D_i of the centroid learner under an optimal poisoning attack,*

$$\begin{array}{lrcll} \text{(a)} & E(D_i) & \leq & \dfrac{\nu}{1-\nu} & \textit{for all } i \\ \text{(b)} & \mathrm{Var}(D_i) & \to & 0 & \textit{for } i, n \to \infty. \end{array}$$

Proof The corollary follows from the fact that $\gamma_i, \delta_n \to 0$ for $i, n \to \infty$. ■

The behavior of the above bounds as a function of the number of attack iterations is illustrated in Figure 5. One can see that the attack's progress depends on the fraction of the training data controlled by the attacker. For any $\nu < 1$, the attack progress is *bounded by a constant*. Hence the attack's success critically depends on the value of this constant: if $||\mathbf{A} - \mathbf{c}_0|| < \nu/(1-\nu)$, the attack fails *even with infinite effort*. This result provides a much stronger security guarantee than the exponential bound for the infinite horizon case.

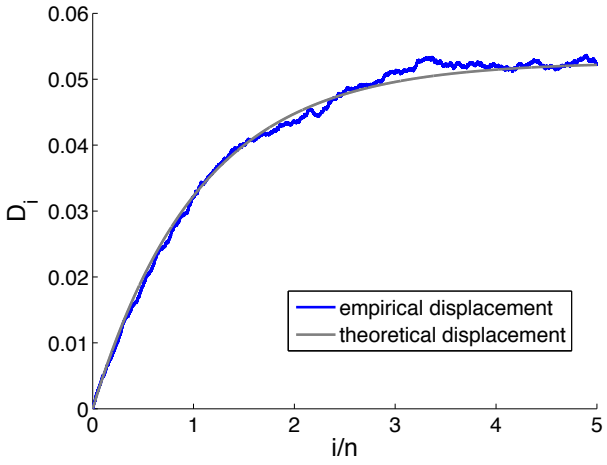


Figure 6: Comparison of the empirical displacement of the centroid under a poisoning attack with attacker's limited control ($\nu = 0.05$) with the theoretical bound for the same setup. The empirical results are averaged over 10 runs.

To empirically investigate the tightness of the derived bound we compute a Monte Carlo simulation of the scenario defined in Axiom 6 with the parameters $\nu = 0.05$, $n = 100000$, $\mathcal{H} = \mathbb{R}^2$, and $P_{\mathbf{x}}$ being a uniform distribution over the unit circle. Figure 6 shows a typical displacement curve over the first $500,000$ attack iterations. One can clearly see that the theoretical bound is closely followed by the empirical simulation.

6. Poisoning Attack under False Positive Constraints

In the last section we have assumed, that innocuous training points $\mathbf{x}_i$ are always accepted by the online centroid learner. It may, however, happen that some innocuous points fall outside of the hypersphere boundary while an attacker displaces the hypersphere. We have seen that the attacker's impact highly depends on the fraction of points he controls. If an attacker succeeds in pushing the hypersphere far enough for innocuous points to start dropping out, the speed of the hypersphere displacement increases. Hence additional protection mechanisms are needed to prevent the success of an attack.

6.1 Learning and Attack Models

Motivated by the above considerations we modify the probabilistic model of the last section as follows. Again we consider the online centroid learner initially anchored at a position $\mathbf{c}_0$ having a radius r. For the sake of simplicity and without loss of generality we can again assume $\mathbf{c}_0 = 0$ and $r = 1$. Innocuous and adversarial points are mixed into the training data according to a fixed fraction controlled by a binary random variable B_i. In contrast to Section 5, innocuous points $\mathbf{x}_i$ are accepted if and only if they fall within the radius r from the hypersphere's center $\mathbf{c}_i$. In addition, to avoid the learner being quickly displaced, we require that the false alarm rate, that is, the number of

innocuous points rejected by the learner, is bounded by α. If the latter is exceeded, we assume the adversary's attack to have failed and a safe state of the learner to be loaded.

We formalize this probabilistic model as follows:

Axiom 10 *$\{B_i | i \in \mathbb{N}\}$ are independent Bernoulli random variables with parameter $\nu > 0$. $\mathbf{x}_i$ are i.i.d. random variables in a reproducing kernel Hilbert space $\mathcal{H}$ drawn from a fixed but unknown distribution $P_{\mathbf{x}}$, satisfying $E(\mathbf{x}) = \mathbf{0}$, $\|\mathbf{x}\| \leq r = 1$, and $P_{\mathbf{x}\cdot\mathbf{a}} = P_{-\mathbf{x}\cdot\mathbf{a}}$ (symmetry w.r.t. the attack direction). B_i and $\mathbf{x}_j$ are mutually independent for each i, j. $f : \mathbb{R}^d \to \mathbb{R}^d$ is an attack strategy satisfying $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$. $\{\mathbf{c}_i | i \in \mathbb{N}\}$ is a collection of random vectors such that $\mathbf{c}_0 = \mathbf{0}$ and*

$$\mathbf{c}_{i+1} = \mathbf{c}_i + \frac{1}{n}\left(B_i\left(f(\mathbf{c}_i) - \mathbf{c}_i\right) + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(\mathbf{x}_i - \mathbf{c}_i\right)\right) ,$$

if $E_{\mathbf{x}}\left(I_{\{\|\mathbf{x}-\mathbf{c}_i\| \leq r\}}\right) \leq 1 - \alpha$ and by $\mathbf{c}_{i+1} = \mathbf{0}$ otherwise. Moreover, we denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

For simplicity of notation, we refer to a collection of random vectors $\{\mathbf{c}_i | i \in \mathbb{N}\}$ satisfying Axiom 10 as an *online centroid learner with maximal false positive rate* α in this section. Any function f satisfying Axiom 10 is called an *attack strategy*. Optimal attack strategies are characterized in terms of the displacement as in the previous sections. Note that $E_{\mathbf{x}}(\cdot)$ denotes the conditional expectation given all remaining random quantities except for $\mathbf{x}$.

The intuition behind the symmetry assumption in Axiom 10 is that it ensures that resetting the centroid's center to zero (initiated by the false positive protection) does not lead to a positive shift of the centroid toward the attack direction.

6.2 Greedy-Optimal Attack and Attack Effectiveness

The following result characterizes the greedy-optimal attack strategy for the model specified in Axiom 10. We restrict our analysis to greedy-optimal strategies, that is, the ones that maximize the displacement in each successive iteration.

Proposition 11 *Let $\mathbf{a}$ be an attack direction vector and consider the centroid learner with maximal false positive rate α as defined in Axiom 10. Then the* greedy-optimal *attack strategy f is given by*

$$f(\mathbf{c}_i) := \mathbf{c}_i + \mathbf{a} .$$

Proof Since by Axiom 10 we have $\|f(\mathbf{x}) - \mathbf{x}\| \leq r$, any valid attack strategy can be written as $f(\mathbf{x}) = \mathbf{x} + g(\mathbf{x})$, such that $\|g\| \leq r = 1$. It follows that either $D_{i+1} = 0$, in which case the optimal f is arbitrary, or we have

$$\begin{aligned} D_{i+1} &= \mathbf{c}_{i+1} \cdot \mathbf{a} \\ &= \left(\mathbf{c}_i + \frac{1}{n}\left(B_i f(\mathbf{c}_i) + (1 - B_i)\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a} \\ &= D_i + \frac{1}{n}\left(B_i\left(D_i + g(\mathbf{c}_i)\right) + (1 - B_i)\mathbf{x}_i - D_i\right) . \end{aligned}$$

Since $B_i \geq 0$, the greedy-optimal attack strategy should maximize $g(\mathbf{c}_i) \cdot \mathbf{a}$ subject to $||g(\mathbf{c}_i)|| \leq 1$. The maximum is clearly attained by setting $g(\mathbf{c}_i) = \mathbf{a}$. ■

The estimate of effectiveness of the greedy-optimal attack in the limited control case is given in the following theorem.

Theorem 12 *For the displacement D_i of a centroid learner with maximal false positive rate α under a poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) &&\leq (1-a_i)\frac{\nu+\alpha(1-\nu)}{(1-\nu)(1-\alpha)} \\ &\text{(b)} \qquad \mathrm{Var}(D_i) &&\leq \gamma_i \frac{\nu^2}{(1-\alpha)^2(1-\nu)^2} + \rho(\alpha) + \delta_n\,, \end{aligned}$$

where $a_i := \left(1-\frac{(1-\nu)(1-\alpha)}{n}\right)^i$, $b_i = \left(1-\frac{1-\nu}{n}(2-\frac{1}{n})(1-\alpha)\right)^i$, $\gamma_i = (a_i - b_i)$, $\rho(\alpha) = \alpha\frac{(1-a_i)(1-b_i)(2\nu(1-\alpha)+\alpha)}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2}$, *and* $\delta_n = \frac{(1-b_i)(\nu+(1-\nu)E(x_i^2))}{(2n-1)(1-\nu)(1-\alpha)}$, $x_i := \mathbf{x}_i \cdot \mathbf{a}$.

The proof is technically demanding and is given in Appendix A.3. Despite the more general proof reasoning, we recover the tightness of the bounds of the previous section for the special case of $\alpha = 0$, as shown by the following corollary.

Corollary 13 *Suppose a maximal false positive rate of $\alpha = 0$. Then, the bounds on the expected displacement D_i, as given by Theorem 8 and Theorem 12, coincide. Furthermore, the variance bound of Theorem 12 upper bounds the one of Theorem 8.*

Proof We start by setting $\alpha = 0$ in Theorem 12(a). Clearly the latter bound coincides with its counterpart in Theorem 8. For the proof of the second part of the corollary, we observe that $\rho(0) = 0$ and that the quantities a_i, b_i, and γ_i coincide with their counterparts in Theorem 8. Moreover, removing the distribution dependence by upper bounding $E(x_i) \leq 1$ reveals that δ_i is upper bounded by its counterpart of Theorem 8. Hence, the whole expression on the right hand side of Theorem 12(b) is upper bounded by its counterpart in Theorem 8(b). ■

The following corollary shows the asymptotic behavior of the above theorem. It follows from $\gamma_i, \delta_n, \rho(\alpha) \to 0$ for $i, n \to \infty$, and $\alpha \to 0$, respectively.

Corollary 14 *For the displacement D_i of the centroid learner with maximal false positive rate α under an optimal poisoning attack,*

$$\begin{aligned} &\text{(a)} \qquad E(D_i) &&\leq \frac{\nu+\alpha(1-\nu)}{(1-\nu)(1-\alpha)} && \text{for all } i \\ &\text{(b)} \qquad \mathrm{Var}(D_i) &&\to 0 && \text{for } i, n \to \infty, \alpha \to 0\,. \end{aligned}$$

From the previous theorem, we can see that for small false positive rates $\alpha \approx 0$, which are common in many applications, for example, intrusion detection (see Section 8 for an extensive empirical analysis), the bound approximately equals the one of the previous section; that is, we have $E(D_i) \leq \frac{\nu}{1-\nu} + \delta$ where $\delta > 0$ is a small constant with $\delta \to 0$. Inverting the bound we obtain the useful formula

$$\nu \geq \frac{E(D_i)}{1+E(D_i)}\,, \tag{18}$$

which gives a lower bound on the minimal ν an adversary has to employ for an attack to succeed.

The illustration of the bound in Theorem 12 is given in Figure 7 for different levels of the false positive protection $\alpha \in [0, 0.025]$. We are especially interested in low false positive rates. One can see that tightness of the bounds of the previous section is almost entirely preserved. In the extreme case $\alpha = 0$, the bounds coincide, as it was shown in Corollary 13.

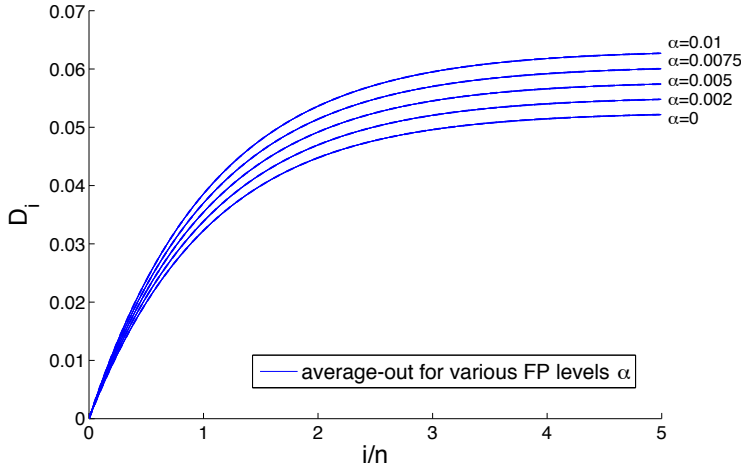


Figure 7: Theoretical behavior of the displacement of a centroid under a poisoning attack for different levels of false positive protection α. The predicted displacement curve for $\alpha = 0$ coincides with the one shown in Figure 6.

7. Generalization to Kernels

For simplicity, we have assumed in the previous sections that the data $\mathbf{x}_i \in \mathbb{R}^d$ lies in an Euclidean space. This assumption does not add any limitations, as all our results can be generalized to the so-called *kernel functions* (Schölkopf and Smola, 2002). This is remarkable, as nonlinear kernels allow one to obtain complex decision functions from the simple centroid model (cf. Figure 1).

Definition 15 (Def. 2.8 in Shawe-Taylor and Cristianini, 2004) *A function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is called a* kernel *if and only if there exists a Hilbert space $(\mathcal{H}, \langle\cdot\rangle)$ and a map $\phi : \mathbb{R}^d \to \mathcal{H}$ such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ it holds $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. Given a sample $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the matrix K with ij-th entry $k(\mathbf{x}_i, \mathbf{x}_j)$ is called* kernel matrix.

Examination of the theoretical results and proofs of the proceeding sections reveals that they, at no point, require special properties of Euclidean spaces. Instead, all calculations can be carried out in terms of arbitrary kernels. To this end, we only need to substitute all occurrences of $\mathbf{x} \in \mathbb{R}^d$ in the proofs by feature vectors $\phi(\mathbf{x}) \in \mathcal{H}$. Likewise, all occurrences of inner products $\langle \mathbf{x}, \mathbf{y} \rangle$ generalize to scalar products $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$ in the kernel Hilbert space. The scalar product also induces a norm, defined by $\|\mathbf{w}\| := \sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$ for all $\mathbf{w} \in \mathcal{H}$. The expectation operator $E : \mathcal{H} \mapsto \mathcal{H}$ is well-defined as long as $E\|\mathbf{x}\|$ is finite, which will always be assumed (as a matter of fact, we assume that $\|\mathbf{x}\|$ is bounded almost surely; cf. Axiom 6 and Axiom 10).

It is interesting to discuss whether the assumptions on the distribution generating the innocuous data as imposed by Axiom 6 and Axiom 10 can be fulfilled in Hilbert spaces. For Axiom 6, there is no restriction at all, but in Axiom 10 we use the assertion $P_{(\mathbf{x}-\mathbf{c}_0)\cdot\mathbf{a}} = P_{-(\mathbf{x}-\mathbf{c}_0)\cdot\mathbf{a}}$, which means that the distribution of the data projection onto the attack direction is required to be point symmetric w.r.t. $\mathbf{c}_0 \cdot \mathbf{a}$, where $\mathbf{c}_0$ is the center of the hypersphere before the attack takes place. For example, this

is fulfilled by distributions that are symmetric with respect to $\mathbf{c}_0$ in feature space, and thus naturally fulfilled by Gaussian distributions with mean $\mathbf{c}_0$ (using $\phi = \mathrm{id}$) or truncated Gaussians. When using kernels the symmetry assumption can be invalid, for example, for RBF kernels. However, our empirical analysis (see next section) shows that our bounds are, nevertheless, sharp in practice.

It is also worth noting that the use of kernels can impose geometric constraints on the optimal attack. Note that, in practice, the attacker can only construct attack points in the input space and not directly in the feature space. The attack is then embedded into the feature space. Thus, strictly speaking, we would need to restrict the search space to feature vectors that have a valid pre-image when using kernels. However, this can be a hard problem to solve in general (Fogla and Lee, 2006). In Proposition 11, we do not take this additional complication into account. Therefore, we *over*estimate the attacker. This is admissible for security analysis; it is the *under*estimation of the attack capability that would have been problematic.

8. Case Study: Application to Intrusion Detection

In this section we present an experimental evaluation of the developed analytical instruments in the context of a particular computer security application: intrusion detection. After a short presentation of the application, data collection, preprocessing and model selection, we report on experiments aimed at the verification of the theoretically obtained growth rates for the attack progress as well as the computation of constant factors for specific real-life exploits.[5]

8.1 Anomaly-Based Intrusion Detection

Computer systems linked to the Internet are exposed to a plethora of network attacks and malicious code. Numerous threats, which range from simple "drive-by-downloads" of malicious code to sophisticated self-proliferating worms, target network hosts every day; networked systems are generally at risk to be remotely compromised and abused for illegal purposes. Sometimes it suffices for malware to send a single HTTP-request to a vulnerable webserver to infect a vast majority of computers within minutes (e.g., the Nimda worm). While early attacks were developed rather for fun than for profit, proliferation of current network attacks is now driven by a criminal underground economy. Compromised systems are often abused for monetary gains including the distribution of spam messages and theft of confidential data. The success of these illegal businesses poses a severe threat to the security of network infrastructures. Alarming reports on an expanding dissemination of advanced attacks render sophisticated security systems indispensable (e.g., Microsoft, 2008; Symantec, 2008).

Conventional defenses against such malicious software rest on abuse detection; that is, identifying attacks using known patterns of abuse, so-called attack *signatures*. While abuse detection effectively protects from known threats, it increasingly fails to be able to cope with the amount and diversity of attacks. The time span required for crafting a signature from a newly discovered attack is insufficient for protecting against rapidly propagating malicious code (e.g., Moore et al., 2002; Shannon and Moore, 2004). Moreover, recent attacks frequently use polymorphic modifications, which strongly impedes the creation of accurate signatures (Song et al., 2007). Consequently, there

5. The term *exploit* denotes a sequence of bytes which, given as an input to a vulnerable program, causes execution of arbitrary, potentially harmful code.

is currently a strong demand for alternative techniques for detection of attacks during the course of their propagation.

Anomaly detection methods provide a means for identifying unknown and novel attacks in network traffic and thereby complement regular security defenses. The centroid anomaly detection is especially appealing because of its low computational complexity.[6] It has been successfully used in several well-known intrusion detection systems (e.g., Hofmeyr et al., 1998; Lazarevic et al., 2003; Wang and Stolfo, 2004; Laskov et al., 2004b; Wang et al., 2005, 2006; Rieck and Laskov, 2007).

8.2 Data Corpus and Preprocessing

The data to be used in our case study was collected by recording real HTTP traffic for 10 days at Fraunhofer Institute FIRST. We consider the data at the level of HTTP requests which are the basic syntactic elements of the HTTP protocol. To transform the raw data into HTTP requests, we remove packet headers from the Ethernet, IP and TCP layers, and merge requests spread across multiple packets. After this point, we consider only request bodies (viewed as a byte string) to be our data point. The resulting data set comprises 145,069 requests of the average length of 489 bytes, from which we randomly drew a representative subset of 2950 data points. This data is referred to as the normal data pool.

The malicious data pool is obtained by a similar procedure applied to the network traffic generated by examples of real attacks used for penetration testing.[7] It contains 69 attack instances from 20 real exploits obtained from the Metasploit penetration testing framework.[8] Attacks were launched in a virtual network and normalized to match the characteristics of the innocuous HTTP requests (e.g., URLs were changed to that of a real web server).

As byte sequences are not directly suitable for the application of machine learning algorithms, we deploy a k-gram spectrum kernel (Leslie et al., 2002; Shawe-Taylor and Cristianini, 2004) for the computation of inner products. This kernel represents a linear product in a feature space in which dimensions correspond to subsequences of length k contained in input sequences. To enable fast comparison of large byte sequences (a typical sequence length is 500–1000 bytes), efficient algorithms using sorted arrays (Rieck and Laskov, 2008) were implemented. Furthermore, kernel values are normalized according to

$$k(\mathbf{x},\bar{\mathbf{x}}) \longmapsto \frac{k(\mathbf{x},\bar{\mathbf{x}})}{\sqrt{k(\mathbf{x},\mathbf{x})k(\bar{\mathbf{x}},\bar{\mathbf{x}})}} \tag{19}$$

to avoid a dependence on the length of a request payload. The resulting inner products were subsequently used by an RBF kernel. Notice that if k is a kernel (in our case it is; see Leslie et al., 2002), then the kernel normalized by Equation (19) is a kernel, too. For example, this can be seen by noting that, for any kernel matrix, this normalization preserves its positive definiteness as it is a columnwise operation and thus can only change the principal minors by a constant factor.

8.3 Learning Model

The feature space selected for our experiments depends on two parameters: the k-gram length and the RBF kernel width σ. Prior to the main experiments aimed at the validation of the proposed

6. With suitable parallelization for multicore architectures, processing speeds of over 3 Gbps can be attained.
7. *Penetration testing* refers to launching real exploits against computer systems to identify potential vulnerabilities.
8. Metasploit can be found at `http://www.metasploit.com/`.

security analysis techniques, we investigate optimal model parameters in our feature space. The parameter range considered is $k = 1,2,3$ and $\sigma = 2^{-5}, 2^{-4.5}, ..., 2^{5}$. Each of the 69 attack instances is represented by a feature vector. We refer to these embedded attacks as the *attack points*; that is, the points in the feature space that the adversary would like to have declared as non-anomalous.

To carry out model selection, we randomly partitioned the innocuous corpus into disjoint training, validation and test sets (of sizes 1000, 500 and 500). The training set is used for computing the centroid, the validation set is used for model selection, and the test set is used to evaluate the detection performance of the centroid. The training set is comprised of the innocuous data only, as the online centroid learner assumes clean training data. The validation and test sets are mixed with 10 randomly chosen attack instances. We thereby ensure that none of the attack instances mixed into the validation set has a class label that also occurs in the test data set. When sampling, we realize this requirement by simply skipping instances that would violate this condition.[9] For each partition, an online centroid learner model is trained on a training set and evaluated on a validation and a test set, using the normalized $\text{AUC}_{[0,0.01]}$ (area under the ROC-curve for false positive rates less that 0.01) as a performance measure.[10] For statistical significance, model selection is repeated 1000 times with different randomly drawn partitions. The average values of the normalized $\text{AUC}_{[0,0.01]}$ for the different k values on test partitions are given in Table 2.

It can be seen that the 3-gram model consistently shows better AUC values for both the linear and the best RBF kernels. We have chosen the linear kernel for the remaining experiments since it allows one to carry out computations directly in the input space with only a marginal penalty in detection accuracy.

	linear	best RBF kernel	optimal σ
1-grams	0.913 ± 0.051	0.985 ± 0.021	$2^{-2.5}$
2-grams	0.979 ± 0.026	0.985 ± 0.025	$2^{-1.5}$
3-grams	0.987 ± 0.018	0.989 ± 0.017	$2^{-0.5}$

Table 2: AUC for the linear kernel, the best RBF kernel and the optimal bandwidth σ.

8.4 Intrinsic HTTP Data Dimensionality

As it was shown in Sec. 4.2.3, the dimensionality of the training data plays a crucial role in the (in)security of the online centroid learner when using the nearest-out update rule. In contrast, the displacement under the average-out rule is independent of the input dimensionality; we therefore focus on the nearest-out rule in this section. For the intrusion detection application at hand, the dimensionality of the chosen feature space (k-grams with $k = 3$) is 256^3, that is, it is rather high. One would thus expect a dramatic impact of the dimensionality on the displacement (an thus the insecurity) of the learner. However, the real progress rate depends on the *intrinsic* dimensionality of the data. When the latter is smaller than the size of the training data, an attacker can compute a PCA of the data matrix (Schölkopf et al., 1998) and project the original data into the subspace spanned by a smaller number of informative components. The following theorem shows that the dimensionality of the relevant subspace in which attack takes place is bounded by the size of the

9. The latter requirement reflects the goal of anomaly detection to recognize *previously unknown* attacks.
10. The normalization is such that AUC = 1 holds for a perfect detector.

training data n, which can be much smaller than the input dimensionality, typically in the range of 100–1000 for realistic applications.

Theorem 16 *There exists an optimal solution of problem (11) satisfying*

$$\mathbf{x}_i^* \in \text{span}(\mathbf{a}, \mathbf{x}_1, ..., \mathbf{x}_n) .$$

The above theorem, which also can be used as a representer theorem for "kernelization" of the optimal greedy attack, shows that the attack's efficiency cannot be increased beyond dimensions with $d \geq n+1$. The proof is given in Appendix A.1.

To determine the intrinsic dimensionality of possible training sets drawn from HTTP traffic, we randomly draw 1000 elements from the innocuous pool, calculate a linear kernel matrix in the space of 3-grams and compute its eigenvalue decomposition. We then determine the number of leading eigen-components as a function of the percentage of variance preserved. The results averaged over 100 repetitions are shown in Figure 8.

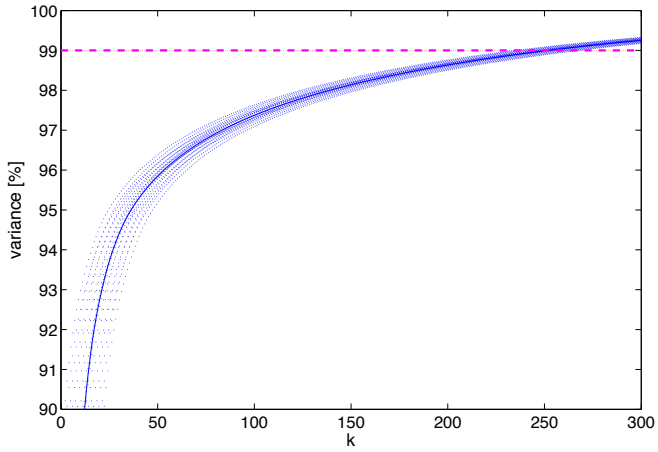


Figure 8: Intrinsic dimensionality of the embedded HTTP data. The preserved variance is plotted as a function of the number of eigencomponents, k, employed for calculation of variance (solid blue line). The tube indicates standard deviations.

It can be seen that 250 kernel PCA components are needed to preserve 99% of the variance. This implies that, although the effective dimensionality of the HTTP traffic is significantly smaller than the number of training data points, it still remains sufficiently high so that the attack progress rate approaches 1, which is similar to the simple average-out learner.

8.5 Geometrical Constraints of HTTP Data

Several technical difficulties arising from data geometry have to be overcome in launching a poisoning attack in practice. However, consideration of the training data geometry provides an attacker with efficient tools for finding reasonable approximations for the above mentioned tasks.

(1) Even for the linear kernel, it is hard to craft a poisoning point in the 3-gram input space due to the high dimensionality of the 3-gram space. An approximately equivalent explicit feature space can be constructed by applying kernel PCA to the kernel matrix K. By pruning the eigenvalues "responsible" for dimensions with low variance one can reduce the size of the feature space to the implicit dimensionality of the problem if the kernel matches the data (Braun et al., 2008). In all subsequent experiments we used $d = 256$ as suggested by the experiments in Section 8.4.

(2) The normalization condition (19) requires that a solution lies on a unit sphere.[11] Unfortunately, this renders the calculation of the greedy-optimal attack point non-convex. Therefore, we pursue the following heuristic procedure to enforce normalization: we explicitly project local solutions (for each Voronoi cell) to a unit sphere, verify their feasibility (the radius and the cell constraints) and remove infeasible points from the outer loop (10).

(3) In general one cannot expect each feature space vector to correspond to a valid byte sequence since not all combinations of k-grams can be "glued" to a valid byte sequence. In fact, finding a sequence with the best approximation to a given k-gram feature vector is NP-hard (Fogla and Lee, 2006). Since the optimal attack lies in the span of the training data (cf. Theorem 16) we can construct an attack byte sequence by concatenating original training sequences with rational coefficients that approximately match the coefficients of a linear combination. A potential disadvantage of this method is the increase of sequence lengths. Large requests are conspicuous and may consume significant resources on the attacker's part.

(4) An attack byte sequence must be embedded in a valid HTML protocol frame so that a request does not cause an error on a server. An HTTP request consists of fixed format headers and a variable format body. A straightforward way to stealthily introduce arbitrary content is to provide a body in a request whose method (e.g., GET) does not require one (it is ignored by the server). Alternatively one can introduce custom headers that are not expected by the server and will be ignored as well.

8.6 Poisoning Attack for Finite Horizon Centroid Learner

The analysis carried out in Section 4 shows that an online centroid learner, in general, does not provide sufficient security if an attacker fully controls the data. Practical efficiency of a poisoning attack, however, depends on the dimensionality and geometry of the training data analyzed in the previous section. Theoretical results were validated in simulations on artificial data presented in Section 4.2.3. The experiments in this section are intended to verify the results presented in Section 4.2.3 in the context of real attacks against HTTP applications. Our experiments focus on the nearest-out learner as other update rules can be easily attacked with trivial methods.

The experimental protocol is as follows. We randomly draw $n = 250$ training points from the innocuous corpus, calculate the center of mass and fix the radius such that the false positive rate on the training data is $\alpha = 0.001$. Then we draw a random instance from each of the 20 attack classes and for each of these 20 attack instances generate a poisoning attack as described in Section 8.5. An attack succeeds when the attack point is accepted as innocuous by the learning algorithm.

For each attack instance, the number of iterations needed for an attack to succeed and the respective displacement of the center of mass is recorded. Figure 9 shows, for each attack instance, the behavior of the relative displacement at the point of success as a function of the number of iterations. We interpolate a "displacement curve" from these pointwise values by linear least-squares

11. In the absence of normalization, the high variability of the byte sequence lengths leads to poor accuracy of centroid anomaly detection.

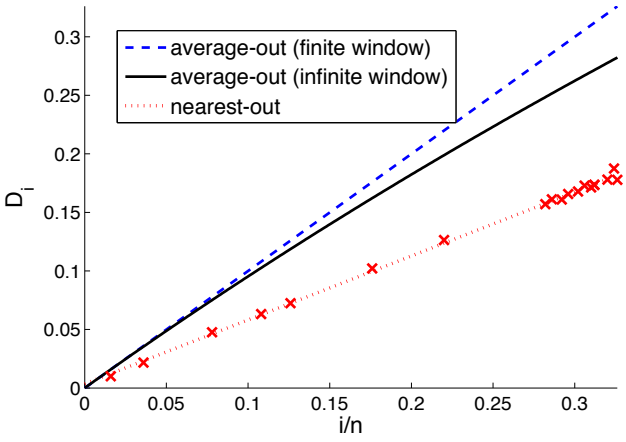


Figure 9: Empirical displacement of the nearest-out centroid for 20 different exploits (crosses, linear fit shown by a red dotted line). Displacement values are shown at the point of success for each attack. Theoretical bounds are shown for comparison (blue and black lines).

regression. For comparison, the theoretical upper bounds for the average-out and all-in cases are shown ("all-in" hereby refers to the update strategy given by Equation (1), where points are added to the training set without removing old ones). Notice that the bound for the all-in strategy is also almost linear for the small i/n ratios observed in this experiment.

The observed results *confirm that the linear progress rate in the full control scenario can be attained for real data*. Compared to the simulations of Section 8.5, the progress rate of an attack is approximately half the one for the average-out case. This can be attributed to multiple approximations performed in the attack implementation for real byte sequences. For example, we use a k-gram spectrum kernel, so each poisoning attack point is restricted to have unit norm in feature space. The practicality of the poisoning attack is further emphasized by the small number of iterations needed for an attack to succeed: it suffices to overwrite between 2 and 35 percent of the initial number of points in the training data to subvert the nearest-out learner.

8.7 Critical Traffic Ratios of HTTP Attacks

For the case of the attacker's limited control over the data, the success of a poisoning attack largely depends on attacker's constraints, as shown in the analysis in Sections 5 and 6. The main goal of the experiments in this section is therefore to investigate the impact of potential constraints in practice. In particular, we are interested in the impact of the traffic ratio ν and the false positive rate α.

The analysis in Section 5 (cf. Theorem 8 and Figure 5) shows that the displacement of a poisoning attack is bounded from above by a constant depending on the traffic ratio ν controlled by an attacker. Hence the susceptibility of the learner to a particular attack depends on the *value of this constant*. If an attacker does not control a sufficiently large traffic portion and the potential displacement is bounded by a constant smaller than the distance from the initial center of mass to

the attack point, then the attack fails. To illustrate this observation, we compute critical traffic rates needed for the success of attacks from each of the 20 attack classes in our malicious pool.

We randomly draw a 1000-element training set from the innocuous pool and calculate its center of mass (in the space of 3-grams). The radius is fixed such that the false positive rate $\alpha = 0.001$ on innocuous data is attained. For each of the 20 attack classes we compute the class-wise median distance to the centroid's boundary. Using these distance values we calculate the "critical value" ν_{crit} by solving Theorem 8(c) for ν (cf. Equation (18)). The experiments were repeated 10 times, with results shown in Table 3.

Attacks	**Rel. dist.**	ν_{crit}
ALT-N WebAdmin Overflow	0.058 ± 0.002	0.055 ± 0.002
ApacheChunkedEncoding	0.176 ± 0.002	0.150 ± 0.001
AWStats ConfigDir Execution	0.067 ± 0.002	0.063 ± 0.002
Badblue Ext Overflow	0.168 ± 0.002	0.144 ± 0.001
Barracuda Image Execution	0.073 ± 0.002	0.068 ± 0.002
Edirectory Host	0.153 ± 0.002	0.132 ± 0.001
IAWebmail	0.178 ± 0.002	0.151 ± 0.001
IIS 5.0 IDQ exploit	0.162 ± 0.002	0.140 ± 0.001
Pajax Execute	0.107 ± 0.002	0.097 ± 0.002
PEERCAST URL	0.163 ± 0.002	0.140 ± 0.001
PHP Include	0.097 ± 0.002	0.088 ± 0.002
PHP vBulletin	0.176 ± 0.002	0.150 ± 0.001
PHP XML RPC	0.172 ± 0.002	0.147 ± 0.001
HTTP tunnel	0.160 ± 0.002	0.138 ± 0.001
IIS 4.0 HTR exploit	0.176 ± 0.002	0.149 ± 0.002
IIS 5.0 printer exploit	0.161 ± 0.002	0.138 ± 0.001
IIS unicode attack	0.153 ± 0.002	0.133 ± 0.001
IIS w3who exploit	0.168 ± 0.002	0.144 ± 0.001
IIS 5.0 WebDAV exploit	0.179 ± 0.002	0.152 ± 0.001
rproxy exploit	0.155 ± 0.002	0.134 ± 0.001

Table 3: Relative distances (in radii) of exploits to the boundary of a centroid enclosing all training points and critical values of parameter ν.

The results indicate that in order to subvert an online centroid learner an attacker needs to control on average from 5 to 20 percent of traffic (with small variance). This could be a significant limitation on highly visible sites. Generating sufficiently high bandwidths in this case is likely to make the attacker's cost exorbitantly high.

On the other hand, one can see that the traffic rate limiting alone cannot be seen as a sufficient protection instrument due to its passive nature. In the following section we investigate a different protection scheme using both the traffic ratio and the false positive rate control.

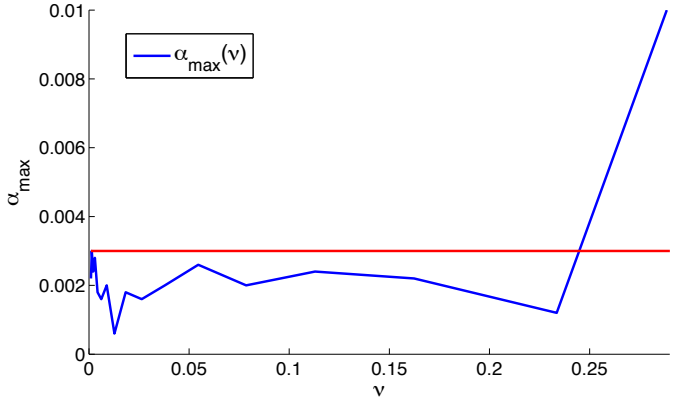


Figure 10: Maximal false positive rate within 10000 attack iterations as a function of ν (maximum taken over 10 runs).

8.8 Poisoning Attack Against Learner with False Positive Protection

The analysis in Section 5 (cf. Theorem 8 and Figure 5) shows that the displacement of a poisoning attack is bounded from above by a number, depending on the traffic ratio ν and the maximal false positive rate α. The dependence of this number on α can be used for *constructive* protection of a learner against a poisoning attack. The idea is to use the observed false positive rate as a measure of the attack's progress, and to turn online updates off if some critical false alarm rate is attained.

8.8.1 EXPERIMENT 1: PRACTICABILITY OF FALSE POSITIVE PROTECTION

The first issue to be decided is what values of α are reasonable from the operational perspective. A false positive threshold that is too low would lead to a quasi-permanent shutdown of online updates; a high tolerance to false positives would allow an attack to slip in unnoticed. Hence, we need to investigate the dependence of the observed false positive rate on the traffic ratio ν under the attacker's control.

To this end, we randomly draw 100 samples from the innocuous pool (each sample containing 1000 points), and compute the mean and the radius that encompasses 99.9% of the points. We may hence view the average radius r (averaged over the 100 samples) as the empirical estimate of the $\alpha = 0.001$-quantile of the innocuous pool. Then, we randomly draw another 1000-element training set from the innocuous pool and use it to calculate the center $\mathbf{c}$. We thus may expect the resulting initial centroid $(\mathbf{c}, r)$ to have a false positive rate of $\alpha = 0.001$ on the innocuous pool. Next, the learner is switched online. We randomly draw a 500-element online training set and a 500-element *hold-out set* from the innocuous pool. The hold-out set is used for the estimation of the false positive rate for each location of the center. We then proceed by presenting normal data (drawn with replacement from the online training set) mixed with poisoning attack points (using the IIS 5.0 WebDAV exploit as target) and measuring the false positive rate for each attack iteration. Notice that normal data points may also get rejected if they do not fall within the radius r from the (poisoned) center.

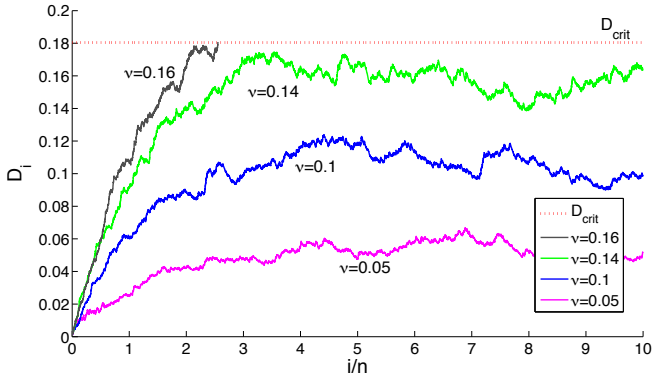


Figure 11: Simulation of a poisoning attack against *IIS 5.0 WebDAV exploit* under limited control.

In Figure 10 the maximal observed false positive rate is shown for various values of ν, where the maximum is taken over all attack iterations and 10 runs. One can see from the plot that $\alpha = 0.005$ is a reasonable threshold in our setting to ensure that the system's normal operation is not disrupted by the false positive protection.

8.8.2 EXPERIMENT 2: ATTACK SIMULATION FOR FALSE POSITIVE PROTECTION

In the previous experiment we have seen that $\alpha = 0.005$ is a reasonable threshold for the false positive protection. In this section we illustrate that the critical values from Section 8.7 computed on the basis of Theorem 8 for the maximal false positive rate of $\alpha = 0.005$ give a good approximation of the true impact of a poisoning attack.

To this end, we fix a particular attack in our malicious corpus (IIS WebDAV 5.0 exploit) and run a poisoning attack against the average-out centroid learner for various values of $\nu \in [0.05, 0.10, 0.14, 0.16]$ recording the actual displacement curves. One can see from Figure 11 that the attack succeeds for $\nu = 0.16$ but fails to reach the required relative displacement of $D_{crit} = 0.18$ for $\nu = 0.14$. The theoretical critical traffic ratio for this attack (with false positive rate bounded by $\alpha \leq 0.005$) according to Table 3 is $\nu_{\text{crit}} = 0.152$. The experiment shows that the derived bounds are surprisingly tight in practice.

The IIS WebDAV 5.0 exploit is, in a sense, an extreme case as it constitutes the most pessimistic scenario for an attacker (farthest away from the normality centroid). In another experiment, we therefore consider the ALT-N WebAdmin Overflow; that is, the most *optimistic* scenario for the attacker and the closest attack to the centroid. The result is shown in Figure 12. We observe from the figure that the experiment again supports our theory: the predicted critical fraction is $\nu = 0.055$ and indeed this quickly leads to a successful attack. For a slightly smaller $\nu = 0.45$, the attack fails and the IDS can be considered safe.

8.8.3 IMPLEMENTATION OF A POISONING PROTECTION IN PRACTICE

In Section 5, we have seen that an attacker's impact on corrupting the training data crucially depends on the fraction of adversarial points in the training data stream. This implies that a large amount of innocuous training points must be processed in order for the system to be secure. In Section 6, we have seen that we can secure the learner by setting a threshold on the false positive rate α. When a false positive rate exceeds the threshold, further countermeasures such as disabling the online

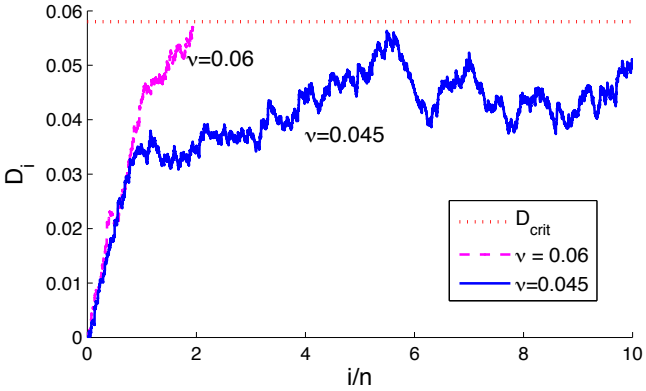


Figure 12: Simulation of a poisoning attack against *ALT-N WebAdmin Overflow* under limited control.

training process can be triggered. This raises an issue of a reliable estimation of the allowable false positive rate.

In practice, this can be done; for example, by caching the training data. When the cache exceeds a certain value at which we have a confident estimation of α (e.g., after 24 hours), the cached training data can be applied to the learner. Since in applications including intrusion detection we usually deal with a very large amount of training data, a confident estimation is already possible after a short time period.

9. Discussion and Conclusions

Understanding of security properties of learning algorithms is essential for their protection against abuse. In order to prove the immunity of a learning algorithm against manipulated data attacks, certain security properties must be *proved*. To this end, we have developed a methodology for security analysis of learning algorithms and applied it for a specific scenario of online centroid anomaly detection. Our analysis highlights conditions under which an attacker's effort to subvert this algorithm is prohibitively high. We further propose some constructive countermeasures for protecting online anomaly detection against poisoning attacks.

Our work is related to other research in machine learning for adversarial environments that has gained significant attention in recent years. A comprehensive survey of a large body of work in this field can be found in Barreno et al. (2010). The majority of related work targets classification problems (e.g., Kearns and Li, 1993; Dalvi et al., 2004; Globerson and Roweis, 2006; Dekel and Shamir, 2008; Dekel et al., 2009). In contrast, our work is focused on anomaly detection methods that have received much less previous attention. Compared to the closest related work of Nelson and Joseph (2006), our methods address more realistic attack and learning scenarios. In particular, we dispense with the assumption of an infinite amount of training data accumulated in the course of training and analyze more advanced scenarios in which the attacker's impact is limited by certain constraints.

Other related work for anomaly detection methods is concerned with the PAYL (Wang and Stolfo, 2004) and Anagram (Wang et al., 2006) algorithms. Both can be seen as a special case of centroid anomaly detection with appropriate distance functions. The blending attack considered by

Fogla et al. (2006); Fogla and Lee (2006) is aimed at evading a trained model at the detection stage by modifying malicious data to resemble the normal one. It differs from the scenario considered in our work in that the attack takes place *after* the model has been trained, whereas the poisoning attack affects the model in the course of training. Although the exact blending was shown to be NP-hard for k-gram feature spaces, approximations have been proposed that are good enough in practice, at least for low values of k. Another related technique is sanitization proposed by Cretu et al. (2008). It deals with the general problem of cleaning the training data from potential attacks. It was shown that small amounts of unintentional attacks can be filtered out by building micro-models based on parts of the training data and rejecting data deemed as anomalous by some micro-models. This work differs from ours in that it considers offline learning; that is, the sanitization takes place *before* before the training data is presented to the learning algorithm. An extension of sanitization to online learning scenarios has been proposed in Cretu-Ciocarlie et al. (2009). The extended method uses previously constructed micro-models to sanitize data from which the most recent model is learned. However, once the attacker has knowledge about individual micro-models, a poisoning attack similar the one considered in our work can be constructed. If suffices for an attacker to craft points that are accepted by all micro-models and inject them into the training data to poison a new micro-model.

Some of the previously developed methods can also be seen to contain parts of the general analysis methodology proposed in Section 1.1. Dalvi et al. (2004) analyzed the robustness of Bayesian classification against adversarial impact. The choice of their classifier is motivated by widespread application of the naive Bayes classification in the domain of spam detection where real examples of adversarial impact have long been observed. The adversarial classification is considered as a game between an attacker and a learner. Due to the complexity of analysis, only one move by each party can be analyzed. Similar to our approach, Dalvi et al. (2004) formalize the problem by defining cost functions of an attacker and a learner (Step 1) and determine an optimal adversarial strategy (Step 3). Although the attacker's constraints are not explicitly treated theoretically, several scenarios using specific constraints have been tested experimentally. No analysis of the attacker's gain is carried out; instead, the learner's direct response to adversarial impact is considered.

A somewhat related approach has been developed for handling worst-case random noise; for example, random feature deletion (Globerson and Roweis, 2006; Dekel and Shamir, 2008). Similar to Dalvi et al. (2004), both of these methods construct a classifier that automatically reacts to the worst-case noise or, equivalently, the optimal adversarial strategy. In both methods, the learning problem is formulated as large-margin classification using a specially constructed risk function. An important role in this approach is played by the consideration of constraints (Step 2); for example, in the form of the maximal number of corruptible features. Although these approaches do not quantitatively analyze the attacker's gain, Dekel and Shamir (2008) contains an interesting learning-theoretic argument that relates classification accuracy and sparseness with the robustness against adversarial noise.

To summarize, we believe that despite recent evidence of possible attacks against machine learning and the currently lacking theoretical foundations for learning under adversarial impact, machine learning algorithms *can* be protected against such an impact. The key to such a protection lies in quantitative analysis of the security of machine learning. We have shown that such an analysis can be rigorously carried out for specific algorithms and attacks. Further work should extend this analysis to more complex learning algorithms and a wider attack spectrum.

Acknowledgments

The authors greatly thank Peter Bartlett, Ulf Brefeld, Vojtech Franc, and Klaus-Robert Müller, Konrad Rieck for fruitful discussions and helpful comments. Furthermore we thank Konrad Rieck for providing the network traffic and an anonymous reviewer for many helpful comments. Most of the work was done when MK and PL were supported by the German Bundesministerium für Bildung und Forschung (BMBF) under the project REMIND (FKZ 01-IS07007A), and by the FP7-ICT Programme of the European Community, under the PASCAL2 Network of Excellence, ICT-216886. MK achknowledges a postdoctorial fellowship and PL a Heisenberg fellowship, both by the National Research Foundation of Germany (DFG). MK is also supported by the German Science Foundation under DFG MU 987/6-1, RA 1894/1-1, and by the World Class University Program through the National Research Foundation of Korea funded by the Korean Ministry of Education, Science, and Technology, under Grant R31-10008.

Appendix A. Auxiliary Material and Proofs

In this appendix we present some helpful Lemmas as well as detailed proofs of the theorems.

A.1 Auxiliary Material for Section 4

This appendix start with a characterization of the greedy-optimal optimization problem in terms of dual variables.

A.1.1 Proof of Theorem 16

Proof The Lagrangian of optimization problem (11) is given by:

$$\begin{aligned} L(\mathbf{x},\alpha,\beta) &= -(\mathbf{x}-\mathbf{x}_i)\cdot\mathbf{a}+\sum_{j=1}^{n}\alpha_j\left(2(\mathbf{x}_j-\mathbf{x}_i)\cdot\mathbf{x}-\mathbf{x}_j\cdot\mathbf{x}_j+\mathbf{x}_i\cdot\mathbf{x}_i\right) \\ &\quad +\beta\left(\mathbf{x}\cdot\mathbf{x}-\frac{2}{n}\sum_{j=1}^{n}\mathbf{x}\cdot\mathbf{x}_j+\frac{1}{n^2}\sum_{j,k=1}^{n}\mathbf{x}_j\cdot\mathbf{x}_k-r^2\right). \end{aligned}$$

Since the feasible set of problem (11) is bounded by the spherical constraint and is not empty ($\mathbf{x}_i$ trivially is contained in the feasible set), there exists at least one optimal solution $\mathbf{x}_i^*$ to the primal. For optimal $\mathbf{x}_i^*$, α^* and β^*, we have the following first order optimality conditions

$$\frac{\delta L}{\delta \mathbf{x}}=0: \quad -\mathbf{a}-\frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_j+2\sum_{j=1}^{n}\alpha_j^*(\mathbf{x}_j-\mathbf{x}_i)+\beta^*\left(2\mathbf{x}_i^*-\frac{2}{n}\sum_{j=1}^{n}\mathbf{x}_j\right)=0. \tag{20}$$

If $\beta^*\neq 0$, the latter equation can be resolved for $\mathbf{x}_i^*$ leading to:

$$\mathbf{x}_i^*=\frac{1}{2\beta^*}\mathbf{a}+\sum_{j=1}^{n}\left(\frac{1}{2\beta^* n}-\frac{\alpha_j^*}{\beta^*}+\frac{1}{n}\right)\mathbf{x}_j+\frac{1}{\beta^*}\sum_{j=1}^{n}\alpha_j^*\mathbf{x}_i.$$

From the latter equation, we see that $\mathbf{x}$ is contained in $S:=\operatorname{span}(\mathbf{x}_1,...,\mathbf{x}_n \text{ and } \mathbf{a})$.

Now assume $\beta^*=0$ and $\mathbf{x}_i^*\notin S$. At first, since $\beta^*=0$, we see from Equation (20) that $\mathbf{a}$ is contained in the subspace $S:=\operatorname{span}(\mathbf{x}_1,...,\mathbf{x}_n)$. Hence the objective, $(\mathbf{x}-\mathbf{x}_i)\cdot\mathbf{a}$, only depends on the

optimal $\mathbf{x}$ via inner products with the data $\mathbf{x}_i$. The same naturally holds for the constraints. Hence, both the objective value and the constraints are invariant under the projection of $\mathbf{x}_i^*$ onto S, denoted by P. Hence $P(x_i^*)$ also is an optimal point. Moreover, by construction, $P(\mathbf{x}_i^*) \in S = \text{span}(\mathbf{x}_1^*, ..., \mathbf{x}_n^*)$. ■

A.1.2 THEORETICAL ANALYSIS FOR THE OPTIMAL GREEDY ATTACK

The dependence of an attack's effectiveness on data dimensionality results from the geometry of Voronoi cells. Intuitively, the displacement at a single iteration depends on the size of the largest Voronoi cell in a current working set. Although it is hard to derive a precise estimate on the latter, the following "average-case" argument sheds some light on the attack's behavior, especially since it is the average-case geometry of the working set that determines the overall attack progress.

Consider a simplified case where each of the Voronoi cells C_j constitutes a ball of radius r centered at a data point $\mathbf{x}_j$, $j = 1, \ldots, n$. Clearly, the greedy attack will result in a progress of r/n (we will move one of the points by r but the center's displacement will be discounted by $1/n$). We will now use the relationship between the volumes of balls in $\mathbb{R}^d$ to relate r, R and d.

The volume of each Voronoi cell C_j is given by

$$\text{Vol}(C_j) = \frac{\pi^{\frac{d}{2}} r^d}{\Gamma\left(\frac{d}{2}+1\right)}.$$

Likewise, the volume of the hypersphere S of radius R is

$$\text{Vol}(S) = \frac{\pi^{\frac{d}{2}} R^d}{\Gamma\left(\frac{d}{2}+1\right)}.$$

Assuming that the Voronoi cells are "tightly packed" in S, we obtain

$$\text{Vol}(S) \approx n\,\text{Vol}(C_j).$$

Hence we conclude that

$$r \approx \sqrt[d]{\frac{1}{n}}\, R.$$

One can see that the attacker's gain approximately represented by the cell radius r is a constant fraction of the threshold R, which explains the linear progress of the poisoning attack. The slope of this linear dependence is controlled by two opposing factors: the size of the training data decreases the attack speed whereas the intrinsic dimensionality of the feature space increases it. Both factors depend on fixed parameters of the learning problem and cannot be controlled by the algorithm. In the limit, when d approaches n (the effective dimension is limited by the training data set according to Theorem 16) the attack progress rate is approximately described by the function $\sqrt[n]{\frac{1}{n}}$ which approaches 1 with increasing n.

A.2 Proofs of Section 5

This following proposition is a well-known fact from algebra.

Proposition 17 (Geometric series) *Let $(s)_{i\in\mathbb{N}_0}$ be a sequence of real numbers satisfying $s_0 = 0$ and $s_{i+1} = qs_i + p$ (or $s_{i+1} \leq qs_i + p$ or $s_{i+1} \geq qs_i + p$) for some $p, q > 0$. Then*

$$s_i = p\frac{1-q^i}{1-q}\,, \quad \left(\text{and } s_i \leq p\frac{1-q^i}{1-q} \text{ or } s_i \geq p\frac{1-q^i}{1-q}\right), \tag{21}$$

respectively.

Proof We prove part (a) of the theorem by induction over $i \in \mathbb{N}_0$, the case of $i = 0$ being obvious.

In the inductive step we show that if Equation (21) holds for an arbitrary fixed i it also holds for $i+1$:

$$\begin{aligned} s_{i+1} &= qs_i + p = q\left(p\frac{1-q^i}{1-q}\right) + p = p\left(q\frac{1-q^i}{1-q} + 1\right) \\ &= p\left(\frac{q - q^{i+1} + 1 - q}{1-q}\right) = p\left(\frac{1-q^{i+1}}{1-q}\right). \end{aligned}$$

The proof of part (b) is analogous. ■

Proof of Theorem 8(b) Multiplying both sides of Equation (17) with $\mathbf{a}$ and substituting $D_i = \mathbf{c}_i \cdot \mathbf{a}$ results in

$$D_{i+1} = \left(1 - \frac{1-B_i}{n}\right) D_i + \frac{B_i}{n} + \frac{(1-B_i)}{n}\mathbf{x}_i \cdot \mathbf{a}\,.$$

Inserting $B_i^2 = B_i$ and $B_i(1-B_i) = 0$, which holds because B_i is Bernoulli, into the latter equation, we have:

$$D_{i+1}^2 = \left(1 - 2\frac{1-B_i}{n} + \frac{1-B_i}{n^2}\right) D_i^2 + \frac{B_i}{n^2} + \frac{(1-B_i)}{n^2}\|x_i\|^2 + 2\frac{B_i}{n}D_i + 2\left(1 - \frac{1}{n}\right)\frac{(1-B_i)}{n}D_i x_i\,,$$

where $x_i := \mathbf{x}_i \cdot \mathbf{a}$. Taking the expectation on the latter equation, and noting that, by Axiom 6, $\mathbf{x}_i$ and $\mathbf{D}_i$ are independent, we have:

$$\begin{aligned} E\left(D_{i+1}^2\right) &= \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2\frac{\nu}{n}E(D_i) + \frac{\nu}{n^2} + \frac{1-\nu}{n^2}E(\|\mathbf{x}_i \cdot \mathbf{a}\|^2) \\ &\overset{(1)}{\leq} \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2\frac{\nu}{n}E(D_i) + \frac{1}{n^2}\,, \end{aligned} \tag{22}$$

where (1) holds because, by Axiom 6, we have $\|\mathbf{x}_i\|^2 \leq r$ and moreover $\|\mathbf{a}\| = r$, $r = 1$. Inserting the result of Theorem 8(a) in the latter equation results in the following recursive formula:

$$E\left(D_{i+1}^2\right) \leq \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right) E\left(D_i^2\right) + 2(1-a_i)\frac{\nu}{n}\frac{\nu}{1-\nu} + \frac{1}{n^2}\,.$$

By the formula of the geometric series, that is, by Proposition 17, we have:

$$E\left(D_i^2\right) \leq \left(2(1-a_i)\frac{\nu}{n}\frac{\nu}{1-\nu} + \frac{1}{n^2}\right)\frac{1-b_i}{\frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)}\,,$$

where $b_i := \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)\right)^i$. Furthermore, by some algebra

$$E\left(D_i^2\right) \leq \frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} \frac{\nu^2}{(1-\nu)^2} + \frac{1-b_i}{(2n-1)(1-\nu)}\,. \tag{23}$$

We will need the auxiliary formula

$$\frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} - (1-a_i)^2 \leq \frac{1}{2n-1} + a_i - b_i\,, \tag{24}$$

which can be verified by some more algebra and employing $b_i < a_i$ and that a_i ranges in the real interval $[0,1]$. We finally conclude

$$\begin{aligned}
\mathrm{Var}(D_i) &= E(D_i^2) - (E(D_i))^2 \\
&\overset{\text{Th.12}(a);\ \text{Equation(23)}}{\leq} \left(\frac{(1-a_i)(1-b_i)}{1-\frac{1}{2n}} - (1-a_i)^2\right)\left(\frac{\nu}{1-\nu}\right)^2 + \frac{1-b_i}{(2n-1)(1-\nu)^2} \\
&\overset{\text{Equation(24)}}{\leq} \gamma_i \left(\frac{\nu}{1-\nu}\right)^2 + \delta_n
\end{aligned}$$

where we denote $\gamma_i := a_i - b_i$ and $\delta_n := \frac{\nu^2 + (1-b_i)}{(2n-1)(1-\nu)^2}$, and use $(1-\nu)^2 \leq 1-\nu \leq 1$ in the inequalities. This completes the proof.

■

A.3 Proofs of Section 6

In this appendix we prove one of our main results.

Lemma 18 *Let $\mathcal{C}$ be a online centroid learner with maximal false positive rate α satisfying the optimal attack strategy. Denote $x_i := \mathbf{x}_i \cdot \mathbf{a}$. Then we have:*

$$\begin{aligned}
&\text{(a)} \quad 0 \leq E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) \leq \alpha E(D_i^q)\,, \quad q = 1,2 \\
&\text{(b)} \quad 0 \leq E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i\right) \leq \alpha \\
&\text{(c)} \quad E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i D_i\right) \leq \alpha E(D_i)\,.
\end{aligned}$$

Proof

(a) Let $q = 1$ or $q = 2$. Since D_i is independent of $\mathbf{x}_i$ (hence constant under the operator $E_{\mathbf{x}_i}$), we have

$$E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) = (D_i)^q E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right)\,.$$

Hence by Axiom 10

$$E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i^q\right) = 0\,,$$

if $E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right) > \alpha$ and $0 \leq E_{\mathbf{x}_i}\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}\right) \leq \alpha$ otherwise. Moreover, $E(D_i) \geq 0$ by the symmetry assumption in Axiom 10. Taking the full expectation $E = E_{\mathbf{c}_i} E_{\mathbf{x}_i}$ yields the assertion (a).

(b) We denote $I_{\leq} := I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}$ and $I_{>} := I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}$. Since

$$E(I_{\leq} x_i) + E(I_{>} x_i) = E\left((I_{\leq} + I_{>}) x_i\right) = E(x_i) = 0\,,$$

we conclude

$$E(I_{\leq} x_i) = -E(I_{>} x_i) = E(I_{>}(-x_i)) \overset{(1)}{\leq} \alpha\,,$$

where (1) holds because $||x_i|| \leq 1$ and by Axiom 10 we have $E(I_{>}) \leq \alpha$.

Furthermore $E(I_{\leq} x_i) \geq 0$ is clear.

(c) The proof of (c) is analogous to that of (a) and (b). ■

Proof of Theorem 12

(a) By Axiom 10, we have

$$D_{i+1} \leq \max\left(0, D_i + \frac{1}{n}\left(B_i\left(f(\mathbf{c}_i) - \mathbf{c}_i\right) + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(\mathbf{x}_i - \mathbf{c}_i\right)\right) \cdot \mathbf{a}\right). \tag{25}$$

By Proposition 11 an greedy-optimal attack strategy can be defined by

$$f(\mathbf{x}) = \mathbf{x} + \mathbf{a}\,.$$

Inserting the latter into Equation (25), using $D_i \overset{\text{Def.}}{=} \mathbf{c}_i \cdot \mathbf{a}$, and taking the expectation, we have

$$E(D_{i+1}) \leq E\left[I_{\left\{E_{\mathbf{x}}\left(I_{\{\|\mathbf{x} - \mathbf{c}_i\| \leq r\}}\right) \leq 1 - \alpha\right\}}\left(D_i + \frac{1}{n}\left(B_i + (1 - B_i) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}\left(x_i - D_i\right)\right)\right)\right], \tag{26}$$

where $x_i = \mathbf{x}_i \cdot \mathbf{a}$. By the symmetry assumption in Axiom 10, the first term can be omitted, hence the above equation can be rewritten as

$$\begin{aligned} E(D_{i+1}) \quad &\leq \quad \left(1 - \frac{1 - \nu}{n}\right) E(D_i) + \frac{\nu}{n} \\ &\quad + \frac{1 - \nu}{n}\left(E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i\right) + E\left(I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i\right)\right). \end{aligned}$$

Inserting the inequalities (a) and (b) of Lemma 18 into the above equation results in:

$$\begin{aligned} E(D_{i+1}) \quad &\leq \quad \left(1 - \frac{1 - \nu}{n}\right) E(D_i) + \frac{\nu}{n} + \frac{1 - \nu}{n}\left(\alpha E(D_i) + \alpha\right) \\ &= \quad \left(1 - \frac{(1 - \nu)(1 - \alpha)}{n}\right) E(D_i) + \frac{\nu + \alpha(1 - \nu)}{n}\,. \end{aligned}$$

By the formula of the geometric series, that is, Proposition 17, we have

$$E(D_{i+1}) \leq (1 - a_i) \frac{\nu + \alpha(1 - \nu)}{(1 - \nu)(1 - \alpha)}\,,$$

where $a_i = \left(1 - \frac{(1-\nu)(1-\alpha)}{n}\right)^i$. Moreover we have

$$E(D_{i+1}) \geq (1 - b_i) \frac{\nu}{1 - \nu}\,, \tag{27}$$

where $b_i = \left(1 - \frac{1-\nu}{n}\right)^i$, by analogous reasoning. We schematically show this by starting from Equation (26) and subsequently applying Jensen's inequality and using the lower bounds of Lemma 18 and the formula for the geometric series. Since $b_i \leq a_i$, we conclude that

$$E(D_{i+1}) \geq (1-a_i)\frac{\nu}{1-\nu}\,.$$

(b) Rearranging the terms in Equation (25), we have

$$\begin{aligned} D_{i+1} \quad \leq \quad & \max\left(0, \left(1 - \frac{1-B_i}{n}\right) D_i + \frac{B_i}{n} + \frac{1-B_i}{n} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i \right. \\ & \left. + \frac{1-B_i}{n} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i \right). \end{aligned}$$

Squaring the latter equation on both sides and using the fact that B_i, $I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}}$ and $I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}}$ are binary-valued yields

$$\begin{aligned} D_{i+1}^2 \leq & \left(1 - \frac{1-B_i}{n}\left(2 - \frac{1}{n}\right)\right) D_i^2 + 2\frac{B_i}{n} D_i + \left(\frac{1-B_i}{n}\left(2 - \frac{1}{n}\right)\right) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| > r\}} D_i \\ & + 2\frac{1-B_i}{n}\left(1 - \frac{1}{n}\right) I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} x_i D_i + \frac{1-B_i}{n^2} I_{\{\|\mathbf{x}_i - \mathbf{c}_i\| \leq r\}} {x_i}^2 + \frac{B_i}{n^2}\,. \end{aligned}$$

Taking expectation of the above equation, by Lemma 18, we have

$$\begin{aligned} E(D_{i+1}^2) \quad \leq \quad & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) \\ & + 2\left(\frac{\nu}{n} + \alpha\frac{1-\nu}{n}\left(1 - \frac{1}{n}\right)\right) E(D_i) + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\,. \end{aligned}$$

We are now in an equivalent situation as in the proof of Theorem 7, right after Equation (22). Similary, we insert the result of (a) into the above equation, obtaining

$$\begin{aligned} E(D_{i+1}^2) \leq & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) \\ & + 2\left(\frac{\nu}{n} + \alpha\frac{1-\nu}{n}\left(1 - \frac{1}{n}\right)\right)(1-a_i)\frac{\nu + \alpha(1-\nu)}{(1-\nu)(1-\alpha)} + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2} \\ \leq & \left(1 - \frac{1-\nu}{n}\left(2 - \frac{1}{n}\right)(1-\alpha)\right) E(D_i^2) + 2(1-a_i)\frac{(\nu + \alpha(1-\nu))^2}{n(1-\nu)(1-\alpha)} \\ & + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\,. \end{aligned}$$

By the formula of the geometric series we obtain

$$\begin{aligned} E(D_i^2) \quad \leq \quad & \left(2(1-a_i)\frac{(\nu + \alpha(1-\nu))^2}{n(1-\nu)(1-\alpha)} + \frac{\nu + (1-\nu)E({x_i}^2)}{n^2}\right) \frac{1-b_i}{\frac{1-\nu}{n}(2 - \frac{1}{n})(1-\alpha)} \\ \leq \quad & \frac{(1-a_i)(1-b_i)(\nu + \alpha(1-\nu))^2}{(1 - \frac{1}{2n})(1-\nu)^2(1-\alpha)^2} + \frac{(1-b_i)(\nu + (1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)}\,, \end{aligned} \tag{28}$$

where $b_i = \left(1 - \frac{1-\nu}{n}(2-\frac{1}{n})(1-\alpha)\right)^i$. We finally conclude

$$\begin{aligned}
\mathrm{Var}(D_i) &= E(D_i^2) - (E(D_i))^2 \\
&\overset{(27),(28)}{\leq} \frac{(1-a_i)(1-b_i)(\nu+\alpha(1-\nu))^2}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2} + \frac{(1-b_i)(\nu+(1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)} - (1-a_i)^2\frac{\nu^2}{(1-\nu)^2} \\
&\overset{(1)}{\leq} \gamma_i \frac{\nu^2}{(1-\alpha)^2(1-\nu)^2} + \rho(\alpha) + \delta_n\,,
\end{aligned}$$

defining $\gamma_i = a_i - b_i$, $\rho(\alpha) = \alpha\frac{(1-a_i)(1-b_i)(2\nu(1-\alpha)+\alpha)}{(1-\frac{1}{2n})(1-\nu)^2(1-\alpha)^2}$, and $\delta_n = \frac{(1-b_i)(\nu+(1-\nu)E({x_i}^2))}{(2n-1)(1-\nu)(1-\alpha)}$, where (1) can be verified employing some algebra and using the auxiliary formula Equation (24), which holds for all $0 < b_i < a_i < 1$. This completes the proof of (b).

Statements (c) and (d) are easily derived from (a) and (b) by noting hat $0 \leq a_i < 1$, $a_i \to 1$ for $i \to \infty$ and $\delta(n) \to 0$ for $n \to \infty$. This completes the proof of the theorem. ■

References

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):434–470, 1988.

P. Auer. Learning nested differences in the presence of malicious noise. *Theoretical Computer Science*, 185(1):159–175, 1997.

M. Bailey, J. Oberheide, J. Andersen, Z. M. Mao, F. Jahanian, and J. Nazario. Automated classification and analysis of internet malware. In *Recent Adances in Intrusion Detection (RAID)*, pages 178–197, 2007.

M. Barreno, B. Nelson, R. Sears, A. Joseph, and J. Tygar. Can machine learning be secure? In *ACM Symposium on Information, Computer and Communication Security*, pages 16–25, 2006.

M. Barreno, P. L. Bartlett, F. J. Chi, A. D. Joseph, B. Nelson, B. I. Rubinstein, U. Saini, and J. D. Tygar. Open problems in the security of learning. In *AISec '08: Proceedings of the 1st ACM workshop on Workshop on AISec*, pages 19–26, New York, NY, USA, 2008. ACM.

M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

M. L. Braun, J. Buhmann, and K.-R. Müller. On relevant dimensions in kernel feature spaces. *Journal of Machine Learning Research*, 9:1875–1908, Aug 2008.

N. H. Bschouty, N. Eiron, and E. Kushilevitz. PAC learning with nasty noise. In *Algorithmic Learning Theory (ALT 1999)*, pages 206–218, 1999.

G. Cretu, A. Stavrou, M. Locasto, S. Stolfo, and A. Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. In *Proc. of IEEE Symposium on Security and Privacy*, pages 81–95, 2008.

G. F. Cretu-Ciocarlie, A. Stavrou, M. E. Locasto, and S. J. Stolfo. Adaptive anomaly detection via self-calibration and dynamic updating. In *Recent Adances in Intrusion Detection (RAID)*, pages 41–60, 2009.

N. Dalvi, P. Domingos, M. Sumit, and S. D. Verma. Adversarial classification. In *In KDD*, pages 99–108. ACM Press, 2004.

O. Dekel and O. Shamir. Learning to classify with missing and corrupted features. In *International Conference on Machine Learning (ICML)*, pages 216–223, 2008.

O. Dekel, O. Shamir, and L. Xiao. Learning to classify with missing and corrupted features. *Machine Learning*, 2009.

P. Fogla and W. Lee. Evading network anomaly detection systems: formal reasoning and practical techniques. In *ACM Conference on Computer and Communications Security*, pages 59–68, 2006.

P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee. Polymorphic blending attacks. In *Proc. of USENIX Security Symposium*, pages 241–256, 2006.

S. Forrest, S. Hofmeyr, A. Somayaji, and T. Longstaff. A sense of self for unix processes. In *Proc. of IEEE Symposium on Security and Privacy*, pages 120–128, Oakland, CA, USA, 1996.

A. Globerson and S. Roweis. Nightmare at test time: Robust learning by feature deletion. In *International Conference on Machine Learning (ICML)*, pages 353–360, 2006.

S. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6(3):151–180, 1998.

M. Kearns and M. Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, 1993.

P. Laskov and M. Kloft. A framework for quantitative security analysis of machine learning. In D. Balfanz and J. Staddon, editors, *AISec*, pages 1–4. ACM, 2009. ISBN 978-1-60558-781-3.

P. Laskov, C. Schäfer, and I. Kotenko. Intrusion detection in unlabeled data with quarter-sphere support vector machines. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of DIMVA Conference*, pages 71–82, 2004a.

P. Laskov, C. Schäfer, I. Kotenko, and K.-R. Müller. Intrusion detection in unlabeled data with quarter-sphere support vector machines (extended version). *Praxis der Informationsverarbeitung und Kommunikation*, 27:228–236, 2004b.

P. Laskov, C. Gehl, S. Krüger, and K. R. Müller. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, 7:1909–1936, Sept. 2006.

A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proc. of SIAM International Conference on Data Mining (SDM)*, 2003.

C. Leslie, E. Eskin, and W. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proc. Pacific Symp. Biocomputing*, pages 564–575, 2002.

Z. Li, M. Sandhi, Y. Chen, M.-Y. Kao, and B. Chavez. Hamsa: fast signature generation for zero-day polymorphic worms with provable attack resilience. In *IEEE Security and Privacy*, pages 32–47, 2006.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear threshold algorithm. *Machine Learning*, 2:285–318, 1988.

D. Lowd and C. Meek. Good word attacks on statistical spam filters. In *Proceedings of the* 11^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 641–647, 2005a.

D. Lowd and C. Meek. Adversarial learning. In *Conference on Email and Anti-Spam*, 2005b.

M. Markou and S. Singh. Novelty detection: a review – part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 2003a.

M. Markou and S. Singh. Novelty detection: a review – part 2: neural network based approaches. *Signal Processing*, 83:2499–2521, 2003b.

L. Martein and S. Schaible. On solving a linear program with one quadratic constraint. *Decisions in Economics and Finance*, 10:75–90, 2005.

Microsoft. Microsoft security intelligence report: January to June 2008. Microsoft Corporation, 2008.

M. Mohri and A. Rostamizadeh. Stability bounds for stationary φ-mixing and β-mixing processes. *J. Mach. Learn. Res.*, 11:789–814, March 2010. ISSN 1532-4435. URL `http://portal.acm.org/citation.cfm?id=1756006.1756032`.

D. Moore, C. Shannon, and J. Brown. Code-Red: a case study on the spread and victims of an internet worm. In *Proc. of Internet Measurement Workshop (IMW)*, pages 273–284, 2002.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Neural Networks*, 12(2):181–201, May 2001.

A. Nairac, T. N., R. Carr, S. King, P. Cowley, and L. Tarassenko. A system for the analysis fo jet vibration data. *Integrated Computer-Aided Engineering*, 1999.

B. Nelson and A. D. Joseph. Bounding an attack's complexity for a simple learning model. In *Proc. of the First Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML)*, Saint-Malo, France, 2006.

B. Nelson, M. Barreno, F. Chi, A. Joseph, B. Rubinstein, U. Saini, C. Sutton, J. Tygar, and K. Xia. Exploiting machine learning to subvert your spam filter. In *Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, 2008.

J. Newsome, B. Karp, and D. Song. Paragraph: Thwarting signature learning by training maliciously. In *Recent Adances in Intrusion Detection (RAID)*, pages 81–105, 2006.

E. Parzen. On estimation of probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif. Misleading worm signature generators using deliberate noise injection. In *Proc. of IEEE Symposium on Security and Privacy*, pages 17–31, 2006.

W. Polonik. Measuring mass concentration and estimating density contour clusters – an excess mass approach. *Annals of Statistics*, 23:855–881, 1995.

S. Rajasegarar, C. Leckie, M. Palaniswami, and J. Bezdek. Quarter sphere based distributed anomaly detection in wireless sensor networks. In *IEEE International Conference on Communications (ICC)*, pages 3864–3869, 2007.

K. Rieck and P. Laskov. Detecting unknown network attacks using language models. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of 3rd DIMVA Conference*, LNCS, pages 74–90, July 2006.

K. Rieck and P. Laskov. Language models for detection of unknown attacks in network traffic. *Journal in Computer Virology*, 2(4):243–256, 2007.

K. Rieck and P. Laskov. Linear-time computation of similarity measures for sequential data. *Journal of Machine Learning Research*, 9(Jan):23–48, 2008.

K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and classification of malware behavior. In *Detection of Intrusions and Malware, and Vulnerability Assessment, Proc. of 5th DIMVA Conference*, LNCS, pages 108–125, 2008.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

C. Shannon and D. Moore. The spread of the Witty worm. *IEEE Security and Privacy*, 2(4):46–50, 2004.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Y. Song, M. Locasto, A. Stavrou, A. Keromytis, and S. Stolfo. On the infeasibility of modeling polymorphic shellcode. In *Conference on Computer and Communications Security (CCS)*, pages 541–551, 2007.

I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.

I. Steinwart, D. Hush, and C. Scovel. Learning from dependent observations. *J. Multivar. Anal.*, 100:175–194, January 2009. ISSN 0047-259X.

M. Sugiyama, M. Krauledat, and K.-R. Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:1027–1061, 2007.

Symantec. Symantex report on the underground economy: July 07 to June 08. Symantec Corporation, 2008.

D. Tax and R. Duin. Data domain description by support vectors. In M. Verleysen, editor, *Proc. ESANN*, pages 251–256, Brussels, 1999a. D. Facto Press.

D. Tax and R. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11–13): 1191–1199, 1999b.

A. Tsybakov. On nonparametric estimation of density level sets. *Annals of Statistics*, 25:948–969, 1997.

C. van de Panne. Programming with a quadratic constraint. *Management Science*, 12:798–815, 1966.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

R. Vert and J.-P. Vert. Consistency and convergence rates of one-class svms and related algorithms. *J. Mach. Learn. Res.*, 7:789–814, December 2006. ISSN 1532-4435.

K. Wang and S. Stolfo. Anomalous payload-based network intrusion detection. In *Recent Adances in Intrusion Detection (RAID)*, pages 203–222, 2004.

K. Wang, G. Cretu, and S. Stolfo. Anomalous payload-based worm detection and signature generation. In *Recent Adances in Intrusion Detection (RAID)*, 2005.

K. Wang, J. Parekh, and S. Stolfo. Anagram: A content anomaly detector resistant to mimicry attack. In *Recent Adances in Intrusion Detection (RAID)*, pages 226–248, 2006.

C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: alternative data methods. In *Proc. of IEEE Symposium on Security and Privacy*, pages 133–145, 1999.

D.-Y. Yeung and C. Chow. Parzen-window network intrusion detectors. In *Sixteenth International Conference on Pattern Recognition (ICPR)*, pages 385–388, 2002.

Journal of Machine Learning Research 13 (2012) 3725-3768 Submitted 12/10; Revised 12/11; Published 12/12

Exploration in Relational Domains for Model-based Reinforcement Learning

Tobias Lang TOBIAS.LANG@FU-BERLIN.DE
Marc Toussaint MARC.TOUSSAINT@FU-BERLIN.DE
Freie Universität Berlin
Machine Learning and Robotics Group
Arnimallee 7, 14195 Berlin, Germany

Kristian Kersting KRISTIAN.KERSTING@IAIS.FRAUNHOFER.DE
Fraunhofer Institute for Intelligent Analysis and Information Systems
Knowledge Discovery Department
Schloss Birlinghoven, 53754 Sankt Augustin, Germany

Editor: Satinder Baveja

Abstract

A fundamental problem in reinforcement learning is balancing exploration and exploitation. We address this problem in the context of model-based reinforcement learning in large stochastic relational domains by developing relational extensions of the concepts of the E^3 and R-MAX algorithms. Efficient exploration in exponentially large state spaces needs to exploit the generalization of the learned model: what in a propositional setting would be considered a novel situation and worth exploration may in the relational setting be a well-known context in which exploitation is promising. To address this we introduce relational count functions which generalize the classical notion of state and action visitation counts. We provide guarantees on the exploration efficiency of our framework using count functions under the assumption that we had a relational KWIK learner and a near-optimal planner. We propose a concrete exploration algorithm which integrates a practically efficient probabilistic rule learner and a relational planner (for which there are no guarantees, however) and employs the contexts of learned relational rules as features to model the novelty of states and actions. Our results in noisy 3D simulated robot manipulation problems and in domains of the international planning competition demonstrate that our approach is more effective than existing propositional and factored exploration techniques.

Keywords: reinforcement learning, statistical relational learning, exploration, relational transition models, robotics

1. Introduction

Acting optimally under uncertainty is a central problem of artificial intelligence. In reinforcement learning (RL), an agent's learning task is to find a policy for action selection that maximizes its reward over the long run. Model-based approaches learn models of the underlying transition process, usually formalized as Markov decision processes, from the agent's interactions with the environment. These models are then analyzed to compute optimal plans.

Generally, an agent has limited data from its interaction with an environment and its model only approximates the true dynamics therein. One of the key challenges in reinforcement learning is thus the exploration-exploitation tradeoff, which strives to balance two competing types of behavior of an

autonomous agent in an unknown environment: the agent can either make use of its current model of the environment to maximize its cumulative reward (that is, to exploit), or sacrifice short-term rewards to gather information about the environment (that is, to explore) in the hope of increasing future long-term return by improving its model. This exploration-exploitation tradeoff has received considerable attention in unstructured, non-relational domains. There exist algorithms which define unique optimal solutions (such as Bayesian reinforcement learning, Poupart et al., 2006) or provably polynomial time solutions (E^3, Kearns and Singh, 2002, and R-MAX, Brafman and Tennenholtz, 2002; Strehl et al., 2009) to the exploration-exploitation trade-off problem. However, while they give a clear idea of how in principle exploration and exploitation can be organized, the basic algorithms in their original formulation only work on discrete enumerated state spaces. Therefore, we believe that the core scientific problem is not to find new exploration-exploitation theories, but how these principles can be realized on non-trivial representations (representations other than enumerated state spaces), and how the generalization and abstraction implicit in non-trivial representations interferes with these exploration-exploitation principles.

The environment of the agent typically contains varying numbers of objects with relations among them. Learning and acting in such large relational domains is a second key challenge in reinforcement learning. Relational approaches (Getoor and Taskar, 2007) can generalize information about one object to reach conclusions about other objects and thereby exploit the relational structure of natural environments. Such domains are hard—or even impossible—to represent meaningfully using an enumerated or a propositional state space. As an example, consider a hypothetical household robot which, after taken out of the shipping box and turned on, explores autonomously the environment in order to learn how to perform its cleaning chores. Without a compact knowledge representation that supports abstraction and generalization of previous experiences to the current state and potential future states, it seems to be hopeless for such a "robot-out-of-the-box" to explore one's home in reasonable time. For instance, after having opened one or two water-taps in bathrooms, the priority for exploring further water-taps in bathrooms, and also in other rooms such as the kitchen, should be reduced. Generalization over object types is crucial for any autonomous agent in realistic environments, but cannot be expressed in a propositional setting where every new object implies a new and therefore non-modeled situation.

The problem of exploration in stochastic relational worlds has so far received little attention. State-of-the-art relational reinforcement learning approaches (Džeroski et al., 2001; Driessens et al., 2006) are mostly model-free and use ε-greedy exploration which does not make use of relational knowledge. Exploiting the relational knowledge for exploration is the problem we address in the current paper. Applying existing, propositional exploration techniques is likely to fail: what in a propositional setting would be considered a novel situation and worth exploration may in the relational setting be an instance of a well-known abstract context in which exploitation is promising. In other terms, the key idea underlying our approach is: *The inherent generalization of learned knowledge in the relational representation has profound implications also on the exploration strategy.*

1.1 Our Approach

We first outline our approach so that we can better discuss related work afterwards. We present a general framework for model-based RL in relational domains. As is typical in model-based RL approaches like E^3 and R-MAX, our system will be composed of a *model learner*, a *planner* and a relational *exploration-exploitation strategy* that integrates both. We introduce a concrete instan-

tiation in this setting where the learner and planner components are based on previous work: The learner is a relational learning algorithm of noisy indeterministic deictic (NID) rules (Pasula et al., 2007) which extracts a compact stochastic relational rule-based model from experience. As a planner we employ PRADA (Lang and Toussaint, 2010) which translates the learned relational model to a grounded dynamic Bayesian network (DBN) and uses approximate inference (a factored frontier) to estimate the expected return of sampled action sequences. Given a learner and a planner, the relational exploration-exploitation strategy needs to realize an estimation of novelty in the relational setting. The classical way to estimate state novelty is based on state (and action) visitation counts which, in an enumerated representation, directly reflect model certainty. We generalize the notion of state counts to relational count functions such that visitation of a single state increases this measure of knownness also for "related" states. What is considered related depends on the choice of features used to model these count functions: similar to density estimation with mixtures we assume count functions to be a mixture of basic (relational) features. The inherent generalization in these count functions thus depends on the choice of features. We propose several possible choices of such features in a relational setting, including one that exploits the specific relational context features that are implicitly learned by the relational rule learner.

Ideally, we would like the learner to fulfill guarantees in the KWIK (*knows what it knows*) framework (Li et al., 2011) and the planner to guarantee near-optimality (in our case, exact inference in the corresponding DBN). Clearly, our specific choices for the learner and planner do *not* fulfill these guarantees but target at being efficient in challenging applications as we demonstrate in the experimental section. Nevertheless, we will establish theoretical guarantees of our relational exploration-exploitation strategy under the assumption that we had a KWIK learner and near-optimal planning. This will allow us to draw clear connections *(i)* to the basic R-MAX and E^3 framework for exploration in reinforcement learning and *(ii)* to the pioneering work of Walsh (2010) on KWIK learning in a relational reinforcement learning setting.

Walsh's work proved the existence of a KWIK learning algorithm in a relational RL setting by nesting several KWIK algorithms for learning different parts of relational transition models. As Walsh points out himself, however, such an integrated KWIK learner, allowing provably efficient exploration, has never been realized or tested and would be "clearly approaching the edge of tractability" (Walsh, 2010). Further, his conceptual algorithm makes limiting assumptions on the model representation which are violated by the more general relational rule framework of Pasula et al. (2007). This is the reason why we choose a practically efficient but heuristic learner which has not been proven to be a KWIK learner. Similarly, it is clear that the computational complexity of *optimal* planning or exact inference in our corresponding DBN is exponential in the number of objects. Therefore, we choose a practically efficient approximate inference technique for planning in relational domains, as given by PRADA.

1.2 Related Work

The first studies on effective exploration in multi-state control problems developed a number of concepts for describing explorative behavior, including curiosity (Schmidhuber, 1991), seeking to minimize the variance of action value estimates (Kaelbling et al., 1996) and counters on the occurrences of states and actions (Thrun, 1992). Thereafter, efficient exploration solutions have been developed for propositional and continuous domains where the environment is represented as an enumerated or vector space. Bayesian reinforcement learning (Poupart et al., 2006) provides an

optimal solution in a Bayesian framework by taking all potential models weighted by their posteriors into account at once. This solution is intractable in all but small problem settings, although there have been advances recently such as the near-Bayesian approach by Kolter and Ng (2009). An alternative approach to optimal exploration are algorithms studied in the probabilistically approximately correct (PAC) framework applied to Markov decision processes (MDPs) (so called PAC-MDP approaches). The seminal algorithms E^3 (Kearns and Singh, 2002) and R-MAX (Brafman and Tennenholtz, 2002; Strehl et al., 2009) execute near-optimal actions in all but a polynomial number of steps (or, in alternative problem formulations, only require a polynomial number of steps before they return a near-optimal policy for the current state). Despite the theoretical guarantees of these algorithms, in practice the required exploration steps are often unrealistically large. More importantly, these approaches are designed for enumerated representations of finite domains and thus difficult to apply in domains of everyday life involving many objects.

E^3 has been extended to parameter learning in factored propositional MDPs with a known structure (Kearns and Koller, 1999) and to Metric E^3 (Kakade et al., 2003) for state spaces where a metric allows to construct accurate local models. Both approaches assume the existence of efficient near-optimal planning algorithms. However, general algorithms with the required guarantees are not known. Supposedly, this is among the reasons why both approaches have not been empirically demonstrated: "it is thus unlikely that the Factored E^3 [for parameter learning] can ever be feasibly implemented" (Guestrin et al., 2002); similar statements can be made for Metric E^3. Therefore, Guestrin et al. (2002) propose an exploration strategy for factored propositional MDPs which is tailored towards a specific planning algorithm based on linear programming. The idea of characterizing relevant subspaces for exploration has been pursued in continuous domains using dimensionality reduction methods (Nouri and Littman, 2010). R-MAX has been extended to continuous MDPs with linearly parameterized dynamics (Strehl and Littman, 2007). All approaches discussed so far as well as other function approximation methods are propositional, that is, they do not generalize over object types. This can only be achieved by incorporating relational features (which subsume attributes of individual objects), which has not been pursued in the mentioned methods.

In recent years, there has been a growing interest in using expressive representations such as relational languages for reinforcement learning (RL) (Džeroski et al., 2001). While typical traditional RL approaches require explicit state and action enumeration, symbolic approaches seek to avoid explicit state and action enumeration through a symbolic representation of states and actions. Most work in this context has focused on model-free approaches (estimating a value function) and has not developed relational exploration strategies. Essentially, a number of relational regression algorithms have been developed for use in these relational RL systems such as relational regression trees (Džeroski et al., 2001) and Gaussian processes with graph kernels (Driessens et al., 2006). Kersting and Driessens (2008) introduce relational policy gradients. All of these approaches use some form of ε-greedy strategy to handle exploration; no special attention has been paid to the exploration-exploitation problem as done in the current paper. Driessens and Džeroski (2004) propose the use of "reasonable policies" in model-free relational RL to provide guidance, that is, to increase the chance to discover sparse rewards in large relational state spaces, also known as reward shaping. Sanner (2005, 2006) combines feature discovery with model-free relational reinforcement learning but does not discuss count function estimation for known states in the exploration-exploitation problem in general terms. Ramon et al. (2007) present an incremental relational regression tree algorithm that is capable of dealing with concept drift and showed that it enables a relational Q-learner to transfer knowledge from one task to another. They do not learn a model of the domain and again,

relational exploration strategies were not developed. Generally, model-free RL approaches are not suited to realize the type of planned exploration as exemplified in R-MAX, E^3 or Bayesian RL. Croonenborghs et al. (2007) learn a relational world model online and additionally use lookahead trees to give the agent more informed Q-values by looking some steps into the future when selecting an action. Exploration is based on sampling random actions instead of informed exploration. Diuk (Diuk, 2010; Diuk et al., 2008) presents an algorithm for efficient exploration under certain assumptions in an alternative object-oriented representation of MDPs focused on object attributes. This representation does not account for noisy dynamics in realistic domains where actions may have a large number of low-probability effects. Efficient planning algorithms for this representation still need to be developed.

The pioneering work of Walsh (Walsh, 2010; Walsh et al., 2009; Walsh and Littman, 2008) provides the first principled investigation into the exploration-exploitation tradeoff in relational domains. His work lifts ideas from R-MAX and efficient RL algorithms for feature selection (such as the algorithms by Diuk et al., 2009) to relational domains. Walsh establishes sample complexity bounds for specific relational MDP learning problems which scale polynomially in the relational action operator descriptions and the number of objects—in contrast to the original R-MAX and E^3 which scale exponentially in the number of objects in relational domains due to the corresponding exponential state and action spaces. Walsh provides an evaluation for some of his algorithms in settings with 2-3 objects; however, his approaches for learning the effects of actions and, more importantly, for learning full action operators have not been demonstrated in practice. Despite their theoretical significance, it is uncertain whether an implementation of these algorithms is feasible; this might show an "inherent limitation of the KWIK-learning paradigm [Walsh's learning framework, discussed below] and likely of online learning itself" (Walsh, 2010). Furthermore, to derive theoretical guarantees Walsh assumes a limited language to represent the learned model. Our approach will use the more expressive language of relational NID rules (Pasula et al., 2007), which are necessary to capture the dynamics in realistic domains, for which Walsh's algorithms would not be applicable. We will detail the differences in representations in Section 3.4.

There is also an increasing number of (approximate) dynamic programming approaches for solving relational MDPs, see for example Boutilier et al. (2001), Kersting et al. (2004), Hölldobler et al. (2006), Wang et al. (2008) and Sanner and Boutilier (2009). In contrast to the current paper, however, their work assumes a given model of the world. Recently, Lang and Toussaint (2009) and Joshi et al. (2010) have shown that successful planning typically involves only a small subset of relevant objects or states. This can speed up symbolic dynamic programming significantly. A principled approach to exploration, however, has not been developed. Guestrin et al. (2003) calculate approximate value functions for relational MDPs from sampled (grounded) environments and provide guarantees for accurate planning in terms of the number of samples; they do not consider an agent which explores its environment step by step to learn a transition model.

The question of optimal exploration in model-based RL where we learn a transition model has similarities to the problem of active learning (Cohn et al., 1996). Epshteyn et al. (2008) investigate active RL in enumerated domains to focus the exploration on the regions of the state space to which the optimal policy is most sensitive. In statistical relational learning (Getoor and Taskar, 2007; de Raedt et al., 2008), which combines expressive knowledge representation formalisms with statistical approaches to perform probabilistic inference and learning in relational domains, active learning has only recently started to attract attention (Bilgic et al., 2010; Xu et al., 2010).

1.3 Contributions

The previous section outlined previous work on (mostly model-free) relational RL, which neglected explicit exploration, and the fundamental work by Walsh et al., which proved existence of KWIK learning in a model-based relational RL setting but falls short of practical applicability. The goal of this work is to propose a practically feasible online relational RL system that integrates efficient exploration in relational domains with fully unknown transition dynamics and learning complete action operators (including contexts, effects, and effect distributions).

Our approach extends Kearns and Singh's theoretically justified exploration technique E^3 and its successor R-MAX and outperforms existing non-relational techniques in a large number of relevant and challenging problems. More precisely, our contributions are the following:

- We introduce the problem of learning relational count functions which generalize the classical notion of state (action) visitation counts.

- We develop a general relational model-based reinforcement learning framework called REX (short for relational explorer), which lifts the concepts of E^3 and R-MAX to relational representations and uses learned relational count functions to estimate empirical model confidence in the relational RL setting.

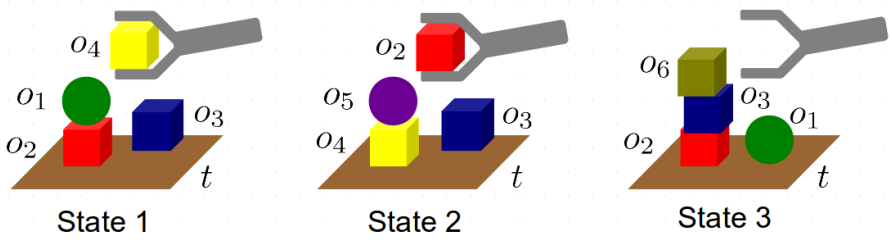
- We provide guarantees on the exploration efficiency of the general REX framework *under the assumption that we had* a relational KWIK learner and *were* capable of near-optimal planning in our domain.

- As a concrete instance of our REX framework, we integrate the state-of-the-art relational planner PRADA (Lang and Toussaint, 2010) and a learner for probabilistic relational rules (Pasula et al., 2007) into our framework. The resulting system is the first practically feasible efficient solution for relational domains with fully unknown transition dynamics.

Our extensive experimental evaluation in a 3D simulated complex robot manipulation environment with an articulated manipulator and realistic physics and in domains of the international planning competition (IPPC) shows that our methods can solve tasks in complex worlds where existing propositional methods fail. With these evaluations we also show that relational representations are a promising technique to formalize the idea of curriculum learning (Bengio et al., 2009). Our work has interesting parallels in cognitive science: Windridge and Kittler (2010) employ ideas of relational exploration for cognitive bootstrapping, that is, to progressively learn more abstract representations of an agent's environment on the basis of its action capabilities.

1.4 Outline

In the following section, we introduce previous work and background on which our methods build. In Section 3, we develop our general REX framework, including our model of relational count functions, and derive theoretical guarantees of this framework under the assumption that we had an ideal KWIK learner and near-exact inference. We then assume relational rule learning and PRADA as concrete learner and planner components for REX. In Section 4, we present the experimental evaluation of this overall system. Finally, we conclude in Section 5.



(a)

State	Enumerated	Factored	Relational
1	s_1	$on\_o_1\_o_2$, $on\_o_2\_t$, $on\_o_3\_t$, $inhand\_o_4$	$on(o_1,o_2)$, $on(o_2,t)$, $on(o_3,t)$, $inhand(o_4)$, $ball(o_1)$, $cube(o_2)$, $cube(o_3)$, $cube(o_4)$, $table(t)$
2	s_2	$on\_o_3\_t$, $on\_o_4\_t$, $on\_o_5\_o_4$, $inhand\_o_2$	$on(o_3,t)$, $on(o_4,t)$, $on(o_5,o_4)$, $inhand(o_2)$, $cube(o_2)$, $cube(o_3)$, $cube(o_4)$, $ball(o_5)$, $table(t)$
3	s_3	$on\_o_1\_t$, $on\_o_2\_t$, $on\_o_3\_o_2$, $on\_o_6\_o_3$	$on(o_1,t)$, $on(o_2,t)$, $on(o_3,o_2)$, $on(o_6,o_3)$, $cube(o_2)$, $cube(o_3)$, $cube(o_6)$, $ball(o_1)$, $table(t)$

(b)

Table 1: Illustration of three world representation types in a robot manipulation domain

2. Background on MDPs, Representations, Exploration and Transition Models

In this section, we set up the theoretical background for the relational exploration framework and algorithms presented later. First, we review briefly Markov decision processes (MDPs). Then, we describe different methods to represent states and actions in MDPs. Thereafter, we discuss exploration in MDPs including the algorithms E^3 and R-MAX. Finally, we discuss in detail compact relational transition models.

2.1 Markov Decision Processes

A Markov decision process (**MDP**) is a discrete-time stochastic control process used to model the interaction of an agent with its environment. At each time-step, the process is in one of a fixed set of discrete states S and the agent can choose an action from a set A. The transition model T specifies the conditional transition distribution $P(s' \,|\, s,a)$ over successor states s' when executing an action a in a given state s. The agent receives rewards in states according to a function $R : S \to \mathbb{R}_{\geq 0}$ (we assume non-negative rewards in this paper without loss of generality). The goal of planning in an MDP is to find a policy $\pi : S \to A$, specifying for each state the action to take, which maximizes the expected future rewards. For a discount factor $0 < \gamma < 1$, the value of a policy π for a state s is defined as the expected sum of discounted rewards $V^\pi(s) = E[\sum_t \gamma^t R(s_t) \,|\, s_0 = s, \pi]$. In our context, we face the problem of reinforcement learning (RL): we do not know the transition model T. Without loss of generality, we assume in this paper that the reward function R is given. We pursue a model-based approach and estimate T from our experiences. Based on our estimate $\hat{T}$ we compute (approximately) optimal policies.

2.2 State and Action Representations

The choice of representation for states S and actions A has important consequences for reinforcement learning techniques on the conceptual and the algorithmic level. Three different representation

types dominate AI research on discrete representations: *(i)* unstructured enumerated representations, *(ii)* factored propositional representations, and *(iii)* relational representations. Table 1 presents three states in a robot manipulation domain together with their translations to the respective representations.

The simplest representation of states and actions is the **enumerated** (or flat) representation. States and actions are represented by single distinct symbols. Hence, the state and action spaces S and A are simply enumerated lists. In Table 1, the three states are represented by s_1, s_2 and s_3. This representation cannot capture the structure of states and does not provide a concept of objects. Therefore, it is impossible to express commonalities among states and actions. In the example, all three states appear equally different in this representation.

A **factored** propositional representation represents a state as a list of attributes. These attributes capture the state structure and hence commonalities among states. MDPs based on factored representations, called factored MDPs, have been investigated extensively in RL and planning research. The disadvantage of factored representations is their lack of a notion of objects. This makes it impossible to express commonalities among attributes. For instance, in Table 1 the attributes $on\_o_1\_o_2$ and $on\_o_5\_o_4$ are treated as completely different and therefore State 1 is perceived as equally different from State 2 as from State 3. Similar arguments hold for actions which are also represented by individual symbols.

Relational representations account for state structure and objects explicitly. The state space S is described by means of a relational vocabulary consisting of predicates $\mathcal{P}$ and functions $\mathcal{F}$, which yield the set of ground atoms with arguments taken from the set of domain objects O. A state is defined by a list of true ground literals. The action space A is defined by atoms $\mathcal{A}$ with arguments from O. In MDPs based on relational representations, called relational MDPs, the commonalities of state structures, actions and objects can be expressed. They enable compact representations since atoms containing logical variables allow for abstraction from concrete objects and situations. We will speak of grounding an abstract formula ψ if we apply a substitution σ that maps all of the variables appearing in ψ to objects in O. In Table 1, abstract atoms capture the greater similarity of State 1 and State 2 in contrast to the one of State 1 and State 3: we can generalize State 1 and State 2 to the (semi-) abstract state $on(A,B), on(B,table), on(o_3,table), inhand(C)$, which is impossible for State 3.

The choice of representation determines the expressivity of models and functions in reinforcement learning. In particular, it influences the compactness and generalization of models and the efficiency of learning and exploration as we discuss next.

2.3 Exploration

A central challenge in reinforcement learning is the exploration-exploitation tradeoff. We need to ensure that we learn enough about the environment to accurately understand the domain and to be able to plan for high-value states (explore). At the same time, we have to ensure not to spend too much time in low-value parts of the state space (exploit). The discount factor γ of the MDP influences this tradeoff: if states are too far from the agent, the agent does not need to explore them as their potential rewards are strongly discounted; large values for γ necessitate more exploration.

The exploration efficiency of a RL algorithm can be measured in terms of its sample complexity: this is the number of time-steps it acts non-optimally, that is, without achieving near-optimal rewards. More formally, let $R_{max} > 0$ denote the maximal reward and m the number of

unknown parameters of the MDP. m captures the complexity of the learning problem and corresponds to the number of parameters of the transition model T in our context. Let $V_t(s_t) = E[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,|\, s_0, a_0, r_0 \ldots, s_t]$ be the value function of the algorithm's policy (which is non-stationary and depends on its history), and V^* of the optimal policy. We define the sample complexity along the lines of Kakade (2003):

Definition 1 *Let $\varepsilon > 0$ be a prescribed accuracy and $\delta > 0$ be an allowed probability of failure. The expression $\eta(\varepsilon, \delta, m, \gamma, R_{max})$ is a **sample complexity** bound for the algorithm if independently of the choice of s_0, with probability at least $1-\delta$, the number of timesteps such that $V_t(s_t) < V^*(s_t) - \varepsilon$ is at most $\eta(\varepsilon, \delta, m, \gamma, R_{max})$.*

An algorithm with a sample complexity polynomial in $1/\varepsilon$, $\log(1/\delta)$, m, $1/(1-\gamma)$ and R_{max} is called probably approximately correct in MDPs, **PAC-MDP** (Strehl et al., 2009).

The seminal approach R-MAX (Brafman and Tennenholtz, 2002) provides a PAC-MDP solution to the exploration-exploitation problem in unstructured enumerated state spaces: its sample complexity is polynomial in the number of states and actions (Strehl et al., 2009). R-MAX generalizes the fundamental approach E^3 (Explicit Explore or Exploit) (Kearns and Singh, 2002) for which a similar result has been established in a slightly different formulation: E^3 finds a near-optimal policy after a number of steps which is polynomial in the number of states and actions. Both E^3 and R-MAX focus on the concept of *known states* where all actions have been observed sufficiently often, defined in terms of a threshold ζ. For this purpose, they maintain state-action counts $\kappa(s,a)$ for all state-action pairs. E^3 (Algorithm 1) distinguishes explicitly between exploitation and exploration phases. If E^3 enters an *unknown* state, it takes the action it has tried the fewest times there (**direct exploration**). If it enters a *known* state, it tries to calculate a high-value policy within a model M_{exploit} including all known states with their sufficiently accurate model estimates and a special self-looping state $\tilde{s}$ with zero reward which absorbs unknown state-action pairs (assuming non-negative rewards in the MDP). If it finds a near-optimal policy in M_{exploit} this policy is executed (**exploitation**). Otherwise, E^3 plans in a different model M_{explore} which is equal to M_{exploit} except that all known states achieve zero reward and $\tilde{s}$ achieves maximal reward. This "optimism in the face of uncertainty" ensures that the agent explores unknown states efficiently (**planned exploration**). The value of the known-state threshold ζ depends on several factors: the discount factor γ, the maximum reward R_{max} and the complexity m of the MDP defined by the number of states and actions as well as the desired accuracy ε and confidence δ for the RL algorithm. The original formulation of E^3 assumes knowledge of the optimal value function V^* to decide for exploitation. Kearns and Singh discuss, however, how this decision can be made without that knowledge. In contrast to E^3, R-MAX decides implicitly for exploration or exploitation and maintains only one model $M_{\text{R-MAX}}$: it uses its model estimates for the known states, while unknown state-action transitions lead to the absorbing state $\tilde{s}$ with maximum reward R_{max}.

The theoretical guarantees of E^3 and R-MAX are strong. In practice, however, the number of exploratory actions becomes huge so that in case of the large state spaces of relational worlds, it is unrealistic to meet the theoretical thresholds of state visits. To address this drawback, variants of E^3 for factored but propositional MDP representations have been explored (Kearns and Koller, 1999; Guestrin et al., 2002). Our evaluations will include variants of factored exploration strategies where the factorization is based on grounded relational formulas. However, factored MDPs still do not enable generalization over objects. In this paper, we are investigating exploration strategies for relational representations and lift E^3 and R-MAX to relational domains. This may strongly improve

Algorithm 1 Sketch of E^3

Input: State s
Output: Action a

if $\forall a : \kappa(s,a) \geq \zeta$ **then** ▷ State is known
  Plan in M_{exploit} with zero-reward for $\tilde{s}$
  if resulting plan has value above some threshold **then**
    return first action of plan ▷ Exploitation
  else
    Plan in M_{explore} with maximum reward for $\tilde{s}$ and zero-reward for known states
    return first action of plan ▷ Planned exploration
  end if
else ▷ State is unknown
  return action $a = \text{argmin}_a \kappa(s,a)$ ▷ Direct exploration
end if

the exploration efficiency and the performance of a reinforcement learning agent. The key idea is to generalize the state-action counts $\kappa(s,a)$ of the original E^3 algorithm over states, actions and objects.

Sample complexity guarantees for more general MDPs can be developed within the **KWIK** (*knows what it knows*) framework (Li et al., 2011). In a model-based RL context, a KWIK learning algorithm can be used to estimate the unknown parts of the MDP. This learner accounts for its uncertainty explicitly: instead of being forced to make a prediction (for instance, of the probability of a successor state for a given state-action pair), it can instead signal its uncertainty about the prediction and return a unique symbol $\perp$. In this case, the (potentially noisy) outcome is provided to the learner which it can use for further learning to increase its certainty. For required model accuracy ε and confidence δ, a model class is called **KWIK-learnable** if there is a learner satisfying the following conditions: (1) if the learner does not predict $\perp$, its prediction is ε-accurate, and (2) the number of $\perp$-predictions is bounded by a polynomial function of the problem description (here, this is m in addition to ε and δ). The KWIK-R-MAX algorithm (Li, 2009) uses a KWIK learner $\mathcal{L}$ to learn the transition model T. The predictions of $\mathcal{L}$ define the known states in the sense of E^3 and R-MAX: a state is known if for all actions $\mathcal{L}$ makes ε-accurate predictions (with some failure probability) and unknown otherwise (where $\mathcal{L}$ predicts $\perp$). Where the learner is uncertain and predicts $\perp$, KWIK-R-MAX assumes a transition to $\tilde{s}$ with reward R_{max}. Li (2009) shows that if T can be efficiently KWIK-learned by $\mathcal{L}$, then KWIK-R-MAX using $\mathcal{L}$ is PAC-MDP. The overall accuracy ε for the sample complexity determines the required individual accuracies ε_T for the KWIK model learner and ε_P for the planner. Hence, one can derive an efficient RL exploration algorithm by developing an efficient KWIK learner $\mathcal{L}$ for the associated model learning problem.

2.4 Learning Generalizing Transition Models

The central learning task in model-based reinforcement learning is to estimate a transition model $\hat{T}$ from a set of experiences $\mathcal{E} = \{(s_t, a_t, s_{t+1})\}_{t=0}^{T-1}$ which can be used for decision-making and planning. (We assume the reward function is provided to the agent.) An example of $\mathcal{E}$ is given in Table 2. The learned model $\hat{T}$ defines a conditional distribution $P(s' \mid s,a)$. Generally, our view is that $\hat{T}$ does not have to be a precise map of the truth—the point of a model is to abstract and partition the space in such a way that this model is a good basis for accurate and efficient decision making.

$\mathcal{E} = \{$			
	$grab(d)$:		$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), on(c,a), \underline{on(d,b)} \ldots$
		$\rightarrow$	$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), on(c,a), \underline{inhand(d)} \ldots$
	$puton(t)$:		$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), on(c,a), \underline{inhand(d)} \ldots$
		$\rightarrow$	$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), on(c,a), \underline{on(d,t)} \ldots$
	$grab(c)$:		$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), \underline{on(c,a)}, on(d,t) \ldots$
		$\rightarrow$	$cube(a), cube(b), ball(c), ball(d), table(t), on(a,t), on(b,t), on(d,t), \underline{inhand(c)} \ldots$
	$\vdots$		$\vdots$
$\}$			

Table 2: The reinforcement learning agent collects a series $\mathcal{E}$ of relational state transitions consisting of an action (on the left), a predecessor state (first line) and a successor state (second line after the arrow). The changing state features are underlined. The agent uses such experiences to learn a transition model resulting in a compression of the state transitions.

This can be achieved by compressing the experiences $\mathcal{E}$ in a compact model $\hat{T}$. Before describing a specific learning algorithm, we discuss conceptual points of learning generalizing models from experience. Compression of the experiences can exploit three opportunities:

- The *frame assumption* states that all state features which are not explicitly changed by an action persist over time. This simplifies the learning problem by deliberately ignoring large parts of the world.
- *Abstraction* allows to exploit the set of experiences efficiently by means of generalization. It can be achieved with relational representations.
- Assuming *uncertainty* in the observations is essential to fit a generalizing and regularized function. It allows to tradeoff the exact modeling of every observation (model likelihood) with generalization capabilities and to find low-complexity explanations of our experience: singleton events can be "explained away" as noise. In our point of view, the assumption of uncertainty is crucial to get relational representations working; it "unleashes" the model and opens the door for simplification, abstraction and compactification. Note that even in deterministic domains, it may be advantageous to learn a probabilistic model because it can be more abstract, more compact and neglect irrelevant details. In this sense, modeling uncertainty can also be understood as regularization.

The generalization capability of $\hat{T}$ and in turn the efficiency to learn it depend on the chosen representation. In an enumerated representation, we need to collect experiences for each relevant state-action pair (s,a) separately. In a factored propositional representation, one can—to some degree—generalize over states and actions by means of the structure imposed by the state attributes, but not over objects. For instance in Table 2, in a factored representation we need to learn the effects for $grab(d)$ and $grab(c)$ independently. In contrast, a *relational* representation enables compact transition models $P(s' \mid s,a)$ by using abstract formulas to generalize from concrete situations and object identities. From a statistical learning point of view, the purpose of such relational logic

$$grab(X): \quad on(X,Y),\ ball(X),\ cube(Y),\ table(Z) \quad \rightarrow \quad \begin{cases} 0.7 & : & inhand(X),\ \neg on(X,Y) \\ 0.2 & : & on(X,Z),\ \neg on(X,Y) \\ 0.1 & : & \text{noise} \end{cases}$$

Table 3: Example NID rule for a robot manipulation scenario, which models to try to grab a ball X. The cube Y is implicitly defined as the one below X (deictic referencing). X ends up in the robot's hand with high probability, but might also fall on the table. With a small probability something unpredictable happens. Refer to Figure 1 for an example application.

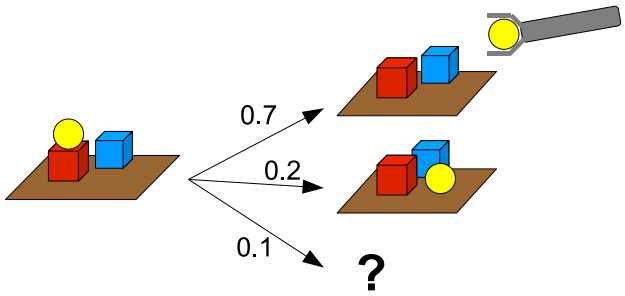


Figure 1: The NID rule defined in Table 3 can be used to predict the effects of action $grab(specific\_ball)$ in the situation on the left side. The right side shows the possible successor states as predicted by the rule. The noise outcome is indicated by a question mark and does not define a unique successor state.

descriptions is simply to provide complex feature descriptors. For instance in Table 2, both $grab(d)$ and $grab(c)$ are used to learn one general model for the abstract action $grab(X)$. In turn, the learned model also models situations with previously unseen objects (which is impossible in enumerated and factored propositional representations). In this sense, we view a relational transition model $\hat{T}$ as a (noisy) compressor of the experiences $\mathcal{E}$ whose compactness enables generalization.

Transition models which generalize over objects and states play a crucial role in our relational exploration algorithms. While our ideas work with any type of relational model that can be learned from experience, for illustration and to empirically evaluate our ideas we employ noisy indeterministic deictic (NID) rules which we present next.

2.4.1 NOISY INDETERMINISTIC DEICTIC RULES

An example of noisy indeterministic deictic (**NID**) rules (Pasula et al., 2007) is shown in Table 3 for our robot manipulation domain. Figure 1 depicts a situation where this rule can be used for

prediction. Formally, a NID rule r is given as

$$a_r(\mathcal{X}):\ \phi_r(\mathcal{X}) \quad \rightarrow \quad \left\{ \begin{array}{lcl} p_{r,1} & : & \Omega_{r,1}(\mathcal{X}) \\ & \vdots & \\ p_{r,m_r} & : & \Omega_{r,m_r}(\mathcal{X}) \\ p_{r,0} & : & \Omega_{r,0} \end{array} \right. ,$$

where $\mathcal{X}$ is a set of logic variables in the rule (which represents a (sub-)set of abstract objects). The rule r consists of preconditions, namely that action a_r is applied on $\mathcal{X}$ and that the abstract state context ϕ_r is fulfilled, and m_r+1 different abstract outcomes with associated probabilities $p_{r,i} > 0$, $\sum_{i=0} p_{r,i} = 1$. Each outcome $\Omega_{r,i}(\mathcal{X})$ describes which literals are predicted to change when the rule is applied. The context $\phi_r(\mathcal{X})$ and outcomes $\Omega_{r,i}(\mathcal{X})$ are conjunctions of literals constructed from the predicates in $\mathcal{P}$ as well as equality statements comparing functions from $\mathcal{F}$ to constant values. The so-called *noise outcome* $\Omega_{r,0}$ subsumes all possible action outcomes which are not explicitly specified by one of the other $\Omega_{r,i}$. This includes rare and overly complex outcomes which are typical for noisy domains and which we do not want to cover explicitly for compactness and generalization reasons. For instance, in the context of the rule depicted in Figure 1 a potential, but highly improbable outcome is to grab the blue cube while pushing all other objects off the table: the noise outcome accounts for this without the burden of explicitly stating it. The arguments of the action $a(\mathcal{X}_a)$ may be a proper subset $\mathcal{X}_a \subset \mathcal{X}$ of the variables $\mathcal{X}$ of the rule. The remaining variables are called deictic references $DR = \mathcal{X} \setminus \mathcal{X}_a$ and denote objects relative to the agent or action being performed.

So, how do we apply NID rules? Let σ denote a substitution that maps variables to constant objects, $\sigma : \mathcal{X} \rightarrow O$. Applying σ to an abstract rule $r(\mathcal{X})$ yields a *grounded rule* $r(\sigma(\mathcal{X}))$. We say a grounded rule r *covers* a state s and a ground action a if $s \models \phi_r$ and $a = a_r$. Let Γ be our set of ground rules and $\Gamma(s,a) \subset \Gamma$ the set of rules covering (s,a). If there is a unique covering rule $r_{s,a} \in \Gamma(s,a)$ with $|\Gamma(s,a)| = 1$, we use it to model the effects of action a in state s. We calculate $P(s' \mid s,a)$ by taking all outcomes of $r_{s,a}$ (omitting subscripts in the following) into account weighted by their respective probabilities,

$$P(s' \mid s,a) \ = \ P(s' \mid s,r) \ = \ \sum_{i=1}^{m_r} p_{r,i} P(s' | \Omega_{r,i}, s) + p_{r,0} P(s' \mid \Omega_{r,0}, s),$$

where, for $i > 0$, $P(s' \mid \Omega_{r,i}, s) = I(s \wedge \Omega_{r,i} \models s')$ is a deterministic distribution that is one for the unique state constructed from s taking the changes of $\Omega_{r,i}$ into account. (The function $I(\cdot)$ maps logical truth values to 0 or 1.) The distribution given the noise outcome, $P(s' \mid \Omega_{r,0}, s)$, is unknown and needs to be estimated. Pasula et al. use a worst-case constant bound $p_{min} \leq P(s' | \Omega_{r,0}, s)$ to lower bound $P(s' | s,a)$. If a state-action pair (s,a) does *not* have a unique covering rule r (including the case that more than one rule covers the state-action pair, resulting in potentially conflicting predictions), we use a noisy default rule r_ν which predicts all effects as noise: $P(s' | s, r_\nu) = P(s' \mid \Omega_{r_\nu,0}, s)$.

The ability to learn models of the environment from experience is a crucial requirement for autonomous agents. The problem of learning rule-sets is in general NP-hard, but with suitable assumptions and restrictions efficiency guarantees on the sample complexity can be given for many learning subtasks (Walsh, 2010). Pasula et al. (2007) have proposed a supervised batch learning algorithm for *complete* NID rules based on ideas developed in inductive logic programming (Nienhuys-Cheng and de Wolf, 1997). This algorithm learns the structure of rules (contexts and outcomes) as well as

their parameters from experience triples $\mathcal{E} = \{(s_t, a_t, s_{t+1})_{t=0}^{T-1}\}$, relying on the frame assumption. Efficient exploration strategies to collect useful data $\mathcal{E}$ for learning, however, were not investigated by Pasula et al.—this will be achieved with our proposed methods. The learning algorithm performs a greedy search through the space of rule-sets, maintaining the thus far best performing rule-set. It optimizes the tradeoff between maximizing the likelihood of the experience triples and minimizing the complexity of the current hypothesis rule-set Γ by optimizing the scoring metric (Pasula et al., 2007)

$$S(\Gamma) = \sum_{(s,a,s')} \log P(s' \,|\, s, r_{s,a}) - \alpha \sum_{r \in \Gamma} \mathrm{PEN}(r) \,, \tag{1}$$

where $r_{s,a}$ is either the unique covering rule for (s,a) or the noisy default rule r_ν. α is a scaling parameter that controls the influence of regularization. $\mathrm{PEN}(r)$ penalizes the complexity of a rule and is defined as the total number of literals in r. The larger α is set, the more compact are the learned rule-sets—and thus, the more general, but potentially also more uncertain and inaccurate. For instance, if we set $\alpha = \infty$, the resulting model will consist of only the default rule, explaining all state transitions as noise. In contrast, if we set $\alpha = 0$, the resulting model explains each experience as accurately as possible, potentially overfitting and not generalizing the data. The learning algorithm is initialized with a rule-set comprising only the noisy default rule r_ν and then iteratively adds new rules or modifies existing ones using a set of search operators. More precisely, these search operators take the current rule-set and the set of experiences as input, repeatedly select individual rules, modify them with respect to the experiences, and thereby produce new rule-sets. If the best new rule-set scores better than the current rule-set according to Equation (1), it becomes the new rule-set maintained by the algorithm. For instance, the most complex search operator, called *ExplainExamples*, creates new rules from experiences which are modeled by the default rule thus far. First, it builds a complex specific rule for an experience; then, it tries to trim this rule to generalize it to other experiences. Other search operators work directly on existing rules: they add new literals to rule contexts or delete them from contexts, add new deictic references to rules in form of literal sets or delete them, delete complete rules from rule-sets, or generalize comparison literals involving function values in rule contexts.

The noise outcome of NID rules is crucial for learning: it permits the iterative formulation of the learning algorithm as the corresponding noisy default rule covers all experiences without unique covering rule so far; and more importantly, it avoids overfitting by refusing to model rare and overly complex experiences. This advantage has been demonstrated empirically by Pasula et al. (2007). Its drawback is that the successor state distribution $P(s' \,|\, \Omega_{r,0}, s)$ is unknown. To deal with this problem, the learning algorithm uses a lower bound p_{min} to approximate this distribution, as described above. In our experience, while the resulting rule-sets are robust to minor changes in the two parameters p_{min} and α, the concrete choice for their values is important for learning expressive rule-sets, and their mutual influence needs to be taken into account. These parameters, however, cannot be optimized by the rule-learning algorithm itself: the required degree of rule compactness is not only determined by the complexity of the modeled domain (in terms of the vocabulary size, stochasticity and context sensitivity of actions), but also by the purpose the rules have to serve (for example, the required accuracy for planning). Nonetheless, in our experiments no significant efforts were necessary to find appropriate values for p_{min} and α: preliminary testings of a small number of typical value-combinations were sufficient to set the parameters effectively for each domain. All

other choices in the learning algorithm (in particular, the choice of search operators and their order) were the same across all reported experiments.

The rule learning algorithm of Pasula et al. uses greedy heuristics in its attempt to learn complete rules. Hence, one cannot give guarantees on its efficiency, correctness or convergence. It has been shown empirically, however, that learned NID rules provide accurate transition models in noisy robot manipulation domains (Pasula et al., 2007; Lang and Toussaint, 2010). In this paper, we show further that the transition dynamics of many domains of the international planning competition can be learned reliably with NID rules. In all our investigated domains, independent learning runs converged to the same or very similar rule-sets; in particular, the learned rule contexts are usually the same.

2.5 Planning with Probabilistic Relational Rules

Model-based RL requires efficient planning for exploitation and planned (directed) exploration. The semantics of NID rules allow one to find a "satisficing" action sequence in relational domains that will lead with high probability to states with large rewards. In this paper, we use the **PRADA** algorithm (Lang and Toussaint, 2010) for planning in grounded relational domains. Empirical results have shown that PRADA finds effective and reliable plans in difficult scenarios. PRADA grounds a given set of abstract NID rules with respect to the objects in the domain and converts the grounded rules to factored dynamic Bayesian networks (DBNs). The random variables (nodes) of the DBNs represent the state literals, actions, rules, rule contexts and outcomes and rewards at different time-steps; factors on these variables define the stochastic transition dynamics according to the NID rules. For planning, PRADA samples sequences of actions in an informed way taking into account the effects of previous actions. To evaluate an action sequence, it uses fast approximate inference to calculate posterior beliefs over states and rewards. If exact instead of approximate inference is used, PRADA is guaranteed to find the optimal plan with high probability given a sufficient number of samples (Lang, 2011). PRADA can plan for different types of rewards, including conjunctions of abstract and grounded literals. Thus, it can be used in model-based RL for both exploiting the learned model to plan for high-reward states as well as for exploring unknown states and actions using the R_{max} reward. The number of sampled action sequences trades off computation time and plan quality: the more samples are taken, the higher are the chances to find a good plan. In our experiments, we set this number sufficiently high to ensure that good plans are found with high probability. If we know the maximum reward, we can determine PRADA's planning horizon for a desired level of accuracy from the discount factor γ of the MDP: we can calculate after which time-step the remaining maximally possible rewards can be ignored.

3. Exploration in Relational Domains

Relational representations enable generalization of experiences over states, actions and objects. Our contribution in this paper are strategies to exploit this for exploration in model-based reinforcement learning. First, we discuss the implications of a relational knowledge representation for exploration on a conceptual level (Section 3.1). We show how to quantify the knowledge of states and actions by means of a generalized, relational notion of state-action counts, namely a *relational count function*. This opens the door to a large variety of possible exploration strategies. Thereafter, we propose a relational model-based reinforcement learning framework lifting the ideas of the algorithms E^3 and R-MAX to symbolic representations (Section 3.2). Then, we discuss theoretical guarantees

concerning the exploration efficiency in our framework (Section 3.3). Finally, we present a concrete algorithm in this framework which uses some of the previously introduced exploration strategies (Section 3.4), including an illustrative example (Section 3.5).

3.1 Relational Count Functions for Known States and Actions

The theoretical derivations of the efficient non-relational exploration algorithms E^3 and R-MAX show that the concept of known states is crucial. On the one hand, the confidence in estimates in known states drives exploitation. On the other hand, exploration is guided by seeking for novel (yet unknown) states and actions. For instance, the direct exploration phase in E^3 chooses novel actions, which have been tried the fewest; the planned exploration phase seeks to visit novel states, which are labeled as yet unknown.

In the original E^3 and R-MAX algorithms operating in an enumerated state space, states and actions are considered known based directly on their **counts**: the number of times they have been visited. In relational domains, we should go beyond simply counting state-action visits to estimate the novelty of states and actions:

- The size of the state space is exponential in the number of objects. If we base our notion of known states directly on visitation counts, then the overwhelming majority of all states will be labeled yet-unknown and the exploration time required to meet the criteria for known states of E^3 and R-MAX even for a small relevant fraction of the state space becomes exponential in the number of objects.

- The key benefit of relational learning is the ability to generalize over yet unobserved instances of the world based on relational abstractions. This implies a fundamentally different perspective on what is novel and what is known and permits qualitatively different exploration strategies compared to the propositional view.

We propose to generalize the notion of counts to a *relational count function* that quantifies the degree to which states and actions are known. Similar to using mixtures (e.g., of Gaussians) for density modeling, we model a count function over the state space as a linear superposition of features f_k ("mixture components") such that visitation of a single state generalizes to "neighboring" states—where "neighboring" is defined by the structure of the features f_k. The only technical difference between count functions and mixture density models is that our count functions are not normalized.

Theoretical guarantees on the convergence and accuracy for this model are discussed in Section 3.3. Given sets of features f_k and mixture weights w_k, a count function over states s can be written as

$$\kappa(s) = \sum_k w_k f_k(s) \ .$$

The state features f_k can be arbitrary. An example for a relational feature are binary tests that have value 1 if some relational query is true for s; otherwise they have value 0. Estimating such mixture models is a type of unsupervised learning or clustering and involves two problems: finding the features f_k themselves (structure learning) and estimating the feature weights w_k (parameter learning).

While the use of relational features offers a great compactness and generalization it complicates the feature selection (structure learning) problem: there are no prior restrictions on the length and

complexity of features and hence, we have essentially infinitely many features to choose from. The longer a relational feature is (e.g., a long conjunction of abstract literals) and the more variables it contains, the larger is the number of possible ways to bind the variables and the larger is the set of refined features that we potentially have to consider when evaluating a feature. In addition, the space of the instances s is very large in our case. Recall that even very small relational models can have hundreds of ground atoms and it would be impossible to represent all possible states. For instance, for just a single binary relation and 10 objects there would be 100 ground atoms and hence 2^{100} distinct states, which is clearly intractable. Thus, we need to focus on a compact, structured representation of the count functions. Furthermore, we only have a finite set of positive (that is, experienced) states from which we have to generalize to other states, while taking care not to overgeneralize wrongly. Note that in contrast the original E^3 and R-MAX algorithms for unstructured domains do not face this problem as they do not generalize experiences to other states. Thus, we essentially face the problem of structure learning from positive examples only (Muggleton, 1997). This is more difficult than the traditional setting considered in relational learning that additionally assumes that negative (impossible) examples are given.

If the features are given, however, the estimation of the count function becomes simpler: only the weights need to be learned. For instance, in 1-class SVMs for density estimation assumptions about the feature structure are embedded in the kernel function; in the mixture of Gaussians model, the functional form of Gaussians is given a-priori and provides the structure. We propose a "patch up" approach in this paper. We examine different choices of features in the relational setting whose weights can be estimated based on empirical counts. While they are only approximations, we then show that we can "patch up" and improve some of these approximations, namely the context-based features (see below) through learning NID rules. In our relational model-based RL algorithms as well as in our evaluation, we focus on context-based features. In other words, our methods implicitly solve both structure and parameter learning from positive examples only.

Let us now introduce different choices of features for the estimation of relational count functions. These imply different approaches to quantify known states and actions in a relational RL setting. We focus on a specific type of features, namely queries $q \in Q$. Queries are simply relational formulas, for instance conjunctions of ground or abstract literals, which evaluate to 0 or 1 for a given state s. We discuss different choices of Q in detail below including examples. Our count functions use queries q in combination with the set of experiences $\mathcal{E}$ of the agent and the estimated transition model $\hat{T}$. Given a set Q of queries we model the state count function as

$$\kappa_Q(s) \;=\; \textstyle\sum_{q\in Q} c_{\mathcal{E}}(q)\, I(\exists\sigma : s \models \sigma(q)) \tag{2}$$

$$\text{with} \quad c_{\mathcal{E}}(q) \;=\; \textstyle\sum_{(s_e,a_e,s'_e)\in\mathcal{E}} I(\exists\sigma : s_e \models \sigma(q)) \;. \tag{3}$$

This function combines the confidences of all queries $q \in Q$ which are fulfilled in state s. The second term in Equation (2) examines whether query q is fulfilled in s: the substitution σ is used to ground potentially abstract queries; the function $I(\cdot)$ maps logical statements to 1 if they are satisfied and to 0 otherwise. $c_{\mathcal{E}}(q)$, the first term in Equation (2), is an experience-count of query q: it quantifies the number of times query q held in previously experienced predecessor states in the agent's set of observed state transitions $\mathcal{E} = \{(s_t, a_t, s_{t+1})\}_{t=1}^{T-1}$. Overall, a state s has a high count $\kappa_Q(s)$ if it satisfies queries $q \in Q$ with large experience-counts $c_{\mathcal{E}}(q)$. This implies that all states with low $\kappa_Q(s)$ are considered novel and should be explored, as in E^3 and R-MAX. The model for state-action count functions is analogous. In the following, we discuss different choices for queries q and the accompanying count estimates which emphasize different aspects of relational data. We use the

three states shown in Table 1 as our running example: we assume that our experiences consist of exactly State 1, that is $\mathcal{E} = \{(s_1, a_1, s_1')\}$ (the action a_1 and the successor state s_1' are ignored by $\kappa_Q(s)$), while State 2 and State 3 have not been experienced.

Enumerated: Let us first consider briefly the propositional enumerated setting. We have a finite enumerated state space S and action space A. The set of queries,

$$Q_{enum} = \{s \,|\, \exists (s_e, a_e, s_e') \in \mathcal{E} : s_e = s\},$$

corresponds to predecessor states $s \in S$ which have been visited in $\mathcal{E}$. Thus, queries are conjunctions of positive and negated ground atoms which fully describe a state. This translates directly to the count function

$$\kappa_{enum}(s) = c_{\mathcal{E}}(s)\,, \quad \text{with } c_{\mathcal{E}}(s) = \sum\nolimits_{(s_e, a_e, s_e') \in \mathcal{E}} I(s_e = s)\,.$$

The experience-count $c_{\mathcal{E}}(q)$ in the previous equation counts the number of occasions state s has been visited in $\mathcal{E}$ (in the spirit of Thrun, 1992). There is no generalization in this notion of known states. Similar arguments can be applied on the level of state-action counts $\kappa(s,a)$. As an illustration, in our running example of Table 1 both State 2 and State 3 are equally unknown and novel: both are not the experienced State 1.

Literal-based: Given a relational structure with the set of logical predicates $\mathcal{P}$, an alternative approach to describe what are known states is based on counting how often a (ground) literal (a potentially negated atom) has been observed true or false in the experiences $\mathcal{E}$ (all statements equally apply to functions $\mathcal{F}$, but we neglect this case here). A literal l for a predicate $P \in \mathcal{P}$ containing variables is abstract in that it represents the set of all corresponding ground, that is variable-free, literals for P. Ground literals then play the role of the traditional factors used in mixture models. First, we consider ground literals $l \in \mathcal{L}^G$ with arguments taken from the domain objects O. This leads to the set of queries $Q_{lit} = \mathcal{L}^G$ and in turn to the count function

$$\begin{aligned} \kappa_{lit}(s) \;&=\; \textstyle\sum_{l \in \mathcal{L}^G} c_{\mathcal{E}}(l)\, I(s \models l) \\ \text{with} \quad c_{\mathcal{E}}(l) &:= \textstyle\sum_{(s_e, a_e, s_e') \in \mathcal{E}} I(s_e \models l). \end{aligned}$$

Each query $l \in \mathcal{L}^G$ examines whether l has the same truth values in s as in experienced states. This implies that a state is considered familiar (with $\kappa_{lit}(s) > 0$) if a ground literal that is true (false) in this state has been observed true (false) before. Thus abstraction over states can be achieved by means of ground literals. We can follow the same approach for abstract literals $\mathcal{L}^A$ and set $Q_{lit} = \mathcal{L}^A$. For $l \in \mathcal{L}^A$ and a state s, we examine whether there are groundings of the logical variables in l such that s covers l. More formally, we replace $s \models l$ by $\exists \sigma : s \models \sigma(l)$. For instance, we may count how often a specific blue ball was on top of *some* other object. If this was rarely the case this implies a notion of novelty which guides exploration. For example, in Table 1 State 1 and State 3 share the ground literal $on(o_2, t)$ while this literal does not hold in State 2. Thus, if $Q_{lit} = \{on(o_2, t)\}$ and State 1 is the sole experienced state, then State 3 is perceived as better known than State 2. In contrast, if we use the abstract query $inhand(X)$ (expressing there was *something* held inhand) and set $Q_{lit} = \{inhand(X)\}$, then State 3 is perceived as more novel, since in both State 1 and State 2 some object was held inhand. Note that this second query abstracts from the identities of the inhand held objects.

Context-based: Assume that we are given a finite set Φ of contexts, which are queries consisting of formulas over predicates and functions. While many relational knowledge representations

have some notion of context or rule precondition, in our running example of NID rules these may correspond to the set of NID rule contexts $\{\phi_r\}$. These are learned from the experiences $\mathcal{E}$ and have specifically been optimized to be a compact context representation (cf. Section 2). Given a set Φ of such queries, setting $Q_\Phi = \Phi$ results in the count function

$$\kappa_\Phi(s) \;=\; \textstyle\sum_{\phi\in\Phi} c_\mathcal{E}(\phi)\, I(\exists\sigma : s \models \sigma(\phi))$$
$$\text{with} \quad c_\mathcal{E}(\phi) \;=\; \textstyle\sum_{(s_e,a_e,s'_e)\in\mathcal{E}} I(\exists\sigma : s_e \models \sigma(\phi)).$$

$c_\mathcal{E}(\phi)$ counts in how many experiences $\mathcal{E}$ the context respectively query ϕ was covered with arbitrary groundings. Intuitively, contexts may be understood as describing situation classes based on whether the same abstract prediction models can be applied. Taking this approach, states are considered novel if they are not covered by any existing context ($\kappa_\Phi(s) = 0$) or covered by a context that has rarely occurred in $\mathcal{E}$ ($\kappa_\Phi(s)$ is low). That is, the description of novelty which drives exploration is lifted to the level of abstraction of these relational contexts. Similarly, we estimate a count function for known state-action pairs based on state-action contexts. For instance, in the case of a set Γ of NID rules, each rule defines a state-action context, resulting in the count

$$\kappa_\Phi(s,a) \;=\; \textstyle\sum_{r\in\Gamma} c_\mathcal{E}(r)\, I(r = r_{s,a}), \quad \text{with} \quad c_\mathcal{E}(r) \;:=\; |\mathcal{E}(r)|,$$

which is based on counting how many experiences are covered by the unique covering rule $r_{s,a}$ for a in s. $\mathcal{E}(r)$ are the experiences in $\mathcal{E}$ covered by r, $\mathcal{E}(r) = \{(s,a,s') \in \mathcal{E} \,|\, r = r_{s,a}\}$. Thus, the number of experiences covered by the rule $r_{s,a}$ modeling (s,a) can be understood as a measure of confidence in $r_{s,a}$ and determines $\kappa_\Phi(s,a)$. We will use $\kappa_\Phi(s,a)$ in our proposed algorithm below. In the example of Table 1, assume in State 1 we perform $puton(o_3)$ successfully, that is, we put the inhand cube o_4 on o_3. From this experience, we learn a rule for the action $puton(X)$ with the context $\phi = clear(X) \wedge inhand(Y)$. Thereafter, State 3 is perceived as more novel than State 2: the effects of $puton(o_3)$ can be predicted in State 2, but not in State 3.

Similarity-based: Different methods to estimate the similarity of relational states exist. For instance, Driessens et al. (2006) and Halbritter and Geibel (2007) present relational reinforcement learning approaches which use relational graph kernels to estimate the similarity of relational states. These can be used to estimate relational counts which, when applied in our context, would readily lead to alternative notions of novelty and thereby exploration strategies. This count estimation technique bears similarities to Metric E^3 (Kakade et al., 2003). Applying such a method to model $\kappa(s)$ from $\mathcal{E}$ implies that states are considered novel (with low $\kappa(s)$) if they have a low kernel value (high "distance") to previous explored states. Let $k(\cdot,\cdot) \in [0,1]$ denote an appropriate kernel for the queries $q \in Q$, for instance based on relational graph kernels. We replace the hard indicator function $I(\exists\sigma : s \models \sigma(q))$ in Equation (2) by the kernel function, resulting in the more general kernel-based count function

$$\kappa_{k,Q}(s) \;=\; \textstyle\sum_{q\in Q} c_\mathcal{E}(q)\, k(s,q)\ .$$

If one sets $Q = \{s \,|\, s \in S\}$ to the set of all states, the previous count function measures the distance to all observed predecessor states multiplied by experience-counts. In the example of Table 1 with $\mathcal{E} = \{(s_1,a_1,s'_1)\}$, if we use relational graph kernels, the isomorphism of the graph representations of State 1 and State 2 leads to a large kernel estimate $k(s_1,s_2)$, while the different graph structure of State 3 causes $k(s_1,s_3)$ to be small, therefore $\kappa_{k,Q}(s_2) > \kappa_{k,Q}(s_3)$.

The above approaches are based on counts over the set of experiences $\mathcal{E}$. As a simple extension, we propose using the **variability** within $\mathcal{E}$. Consider two different series of experiences $\mathcal{E}_1$ and $\mathcal{E}_2$ both of size n. Imagine that $\mathcal{E}_1$ consists of n times the same experience, while the experiences in $\mathcal{E}_2$ differ. For instance, $\mathcal{E}_1$ might list repeatedly grabbing the same object in the same context, while $\mathcal{E}_2$ might list grabbing different objects in varying conditions. $\mathcal{E}_2$ has a higher variability. Although the experience-counts $c_{\mathcal{E}_1}(q)$ and $c_{\mathcal{E}_2}(q)$ as defined in Equation (3) are the same, one might be tempted to say that $\mathcal{E}_2$ confirms q better as the query q succeeded in more heterogeneous situations, supporting the claim of generalization. We formalize this by redefining the experience-counts in Equation (3) using a distance estimate $d(s,s') \in [0,1]$ for two states s and s' as

$$\begin{aligned} c^V_{\mathcal{E}}(q) \;=\; & \sum_{(s_t,a_t,s'_t)\in\mathcal{E}} I(\exists\sigma : s_t \models \sigma(q)) \cdot \delta(s_t, \mathcal{E}^{(q,t)}) \;, \\ & \text{with} \quad \mathcal{E}^{(q,t)} = \{(s_e,a_e,s'_e) \in \mathcal{E} \,|\, \exists\sigma : s_e \models \sigma(q) \wedge e < t\} \\ & \text{and} \quad \delta(s,\mathcal{E}) = \min_{(s_e,a_e,s'_e)\in\mathcal{E}} d(s,s_e) \;. \end{aligned}$$

The experience-count $c^V_{\mathcal{E}}(q)$ weights each experience based on its distance $\delta(s_t, \mathcal{E}^{(q,t)})$ to prior experiences (while the original $c_{\mathcal{E}}(q)$ assigns all experiences the same weight irrespective of other experiences). Here, the measure $d(s,s')$ computes the distance between two ground states, but calculations on partial or (partially) lifted states are likewise conceivable. To compute $d(s,s')$, a kernel $k(s,s') \in [0,1]$ as above might be employed, $d(s,s') \propto 1-k(s,s')$, for instance using a simple distance estimate based on least general unifiers (Ramon, 2002). For illustration, consider again the states in Table 1. Assume two series of experiences $\mathcal{E}_1 = \{(s_1,a_1,s'_1),(s_2,a_2,s'_2)\}$ and $\mathcal{E}_2 = \{(s_1,a_1,s'_1),(s_3,a_3,s'_3)\}$ and the query $q = on(X,Y) \wedge ball(X)$ (that there is a ball on-top of some other object). All State 1, State 2 and State 3 cover this query, therefore the standard counts for $\mathcal{E}_1$ and $\mathcal{E}_2$ according to Equation (3) are the same, $c_{\mathcal{E}_1}(q) = c_{\mathcal{E}_2}(q)$. As State 1 and State 2 share the same structure, however, the variability within $\mathcal{E}_1$ is smaller than within $\mathcal{E}_2$. Thus, $\mathcal{E}_2$ provides more heterogeneous evidence for q and therefore $c^V_{\mathcal{E}_2}(q) > c^V_{\mathcal{E}_1}(q)$.

3.2 Relational Exploration Framework

The approaches to estimate relational state and action counts which have been discussed above open a large variety of possibilities for concrete exploration strategies. In the following, we derive a relational model-based reinforcement learning framework we call REX (short for relational explorer) in which these strategies can be applied. This framework is presented in Algorithm 2. REX lifts E^3 and R-MAX to relational exploration.

At each time-step, the relational explorer performs the following general steps:

1. It adapts its estimated relational transition model $\hat{T}$ with the set of experiences $\mathcal{E}$.

2. Based on $\mathcal{E}$ and $\hat{T}$, it estimates the count function for known states and actions, $\kappa(s)$ and $\kappa(s,a)$. For instance, $\hat{T}$ might be used to provide formulas and contexts to estimate a specific relational count function.

3. The estimated count function is used to decide for the next action a_t based on the strategy of either E^3 or R-MAX.

   In the case of relational E^3, the phase-ordering is exactly the same as in the original E^3: if the current state s_t is known, exploitation is tried and if exploitation fails (the planner does not find

Algorithm 2 Relational Exploration (REX)

Input: Start state s_0, reward function R, confidence threshold ζ
Set of experiences $\mathcal{E} = \emptyset$
for t=0,1,2 ... **do**
  Update transition model $\hat{T}$ according to $\mathcal{E}$
  Estimate $\kappa(s)$ and $\kappa(s,a)$ from $\mathcal{E}$ and $\hat{T}$ ▷ Relational representation enables generalization
  if E^3-exploration **then** ▷ E^3-exploration
    if $\forall a \in \mathcal{A} : \kappa(s_t, a) \geq \zeta$ **then** ▷ State is known → **uses relational generalization**
      Plan in M_{exploit} with zero reward for $\tilde{s}$ ▷ **uses relational generalization**
      if resulting plan has value above some threshold **then**
        a_t = first action of plan ▷ Exploitation
      else
        Plan in M_{explore} with maximum reward for $\tilde{s}$ and zero-reward for known states
          ▷ **uses relational generalization**
        a_t = first action of plan ▷ Planned exploration
      end if
    else ▷ State is unknown
      $a_t = \text{argmin}_a \kappa(s_t, a)$ ▷ Direct exploration → **uses relational generalization**
    end if
  else ▷ R-MAX-exploration
    Plan in $M_{\text{R-MAX}}$ with maximum reward for $\tilde{s}$ and given reward R for all known states
      ▷ **uses relational generalization**
    a_t = first action of plan
  end if
  Execute a_t
  Observe new state s_{t+1}
  Update set of experiences $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s_t, a_t, s_{t+1})\}$
end for

a policy with a sufficiently high value), planned exploration to unknown states is undertaken; otherwise (if the current state s_t is not known), direct exploration is performed. Like in the original non-relational E^3, the decision for exploitation in relational E^3 assumes knowledge of the optimal values. To remove this assumption, the relational explorer can instead attempt planned exploration first along the lines described by Kearns and Singh for the original E^3.

In the case of relational R-MAX, a single model is built in which all unknown states according to the estimated counts lead to the absorbing special state $\tilde{s}$ with reward R_{max}.

4. Finally, the action a_t is executed, the resulting state s_{t+1} observed and added to the experiences $\mathcal{E}$, and the process repeated.

The estimation of relational count functions for known states and actions, $\kappa(s)$ and $\kappa(s,a)$, and the resulting generalization over states, actions and objects play a crucial role at several places in the algorithm; for instance, in the case of relational E^3,

- to decide whether the current state is considered known or novel;
- to determine the set of known states where to try to exploit;
- to determine the set of both novel and known states in planned exploration to decide for target states and plan-through states;

- and to determine which action to execute in direct exploration where the least known action with the lowest $\kappa(s,a)$ is chosen; this combines the original E^3 (choosing the action with the fewest "visits") with relational generalization (defining "visits" by means of state abstraction).

The parameter ζ in our relational exploration framework REX defines the threshold to decide whether states and actions are known or unknown. For the original E^3 and R-MAX algorithms for enumerated state and action spaces, a value for ζ can be derived such that efficient exploration in polynomial time can be guaranteed. This derivation is closely tied to the enumerated representation of state and action spaces, not straightforward to apply in practice and will lead to overly large thresholds (see Section 2.3). In the next subsection, we discuss how to develop theoretical guarantees on the sample complexity for relational domains within our REX framework.

3.3 Theoretical Guarantees on the Sample Complexity

The original R-MAX algorithm operating in enumerated state and action spaces is PAC-MDP: its sample complexity is polynomial in the number of states and actions (Strehl et al., 2009). A similar result for E^3 has been established in a slightly different formulation: E^3 finds a near-optimal policy after a number of steps which is polynomial in the number of states and actions. In the following, we briefly establish conditions for which similar guarantees can be made in our relational exploration framework REX using count functions: we state simple conditions for a KWIK learner to "match" with the used count function such that REX using this learner is PAC-MDP. This clarifies also a relation to Walsh's assumptions about mutually exclusive contexts in his KWIK learning setup.

Let $\mathcal{E}_{\kappa(s,a)} \subseteq \mathcal{E}$ be the subset of experiences such that any experience $(s_t, a_t, s_{t+1}) \in \mathcal{E}_{\kappa(s,a)}$ would lead to $\kappa(s,a) > 0$ given $\kappa(s,a) = 0$ before. Recall that m is the number of unknown parameters of the (relational) transition model T. As described in Section 2.3, we distinguish two different accuracy levels ε_P and ε_T which both influence the overall accuracy ε in the sample complexity of a PAC-MDP reinforcement learning algorithm (Definition 1). An ε_P -accurate planner finds plans a whose value differs by at most ε_P from the value of the optimal plan a^*: $Q(s,\mathrm{a}) > Q(s,\mathrm{a}^*) - \varepsilon_P$. An ε_T-accurate model learner $\mathcal{L}$ makes predictions $\hat{y} = \mathcal{L}(x)$ for input x which differ by at most ε_T from the true value y: $|\hat{y} - y| < \varepsilon_T$. We get the following lemma on the exploration efficiency of REX:

Lemma 2 *Assume* REX *uses an ε_P-accurate planner and a KWIK learner $\mathcal{L}$ to learn the transition model $\hat{T}$. If for a given probability of failure $\delta \in (0,1)$* REX *employs a count function $\kappa(s,a)$ such that it holds with confidence at least $1-\delta$ that*

$$\forall s,a: \quad \kappa(s,a) \geq \zeta \;\Rightarrow\; \mathcal{L}(s,a) \text{ is } \varepsilon_T\text{-accurate and } \mathcal{L}(s,a) \neq \perp \quad \text{and} \tag{4}$$

$$|\mathcal{E}_{\kappa(s,a)}| \geq \text{polynomial}(m) \;\Rightarrow\; \kappa(s,a) \geq \zeta\,, \tag{5}$$

then REX *in the* R-MAX *variant is PAC-MDP.*

Proof First assume we had equivalence $\Longleftrightarrow$ in Equation (4). Then, for the R-MAX option in Algorithm 2, this corollary is a direct implication of the original results for KWIK-R-MAX (Li, 2009): in line 19 of Algorithm 2 the condition $\kappa(s,a) \geq \zeta$, using the count function to decide knownness of states and actions, is identical to using the KWIK learner condition $\mathcal{L}(s,a) \neq \perp$. To show the lemma with one-sided implication in Equation (4), we define a second learner $\mathcal{L}'$ with $\kappa(s,a) < \zeta \Rightarrow \mathcal{L}'(s,a) = \perp$ and $\mathcal{L}'(s,a) \neq \perp \Rightarrow \mathcal{L}'(s,a) = \mathcal{L}(s,a)$ for all s,a. $\mathcal{L}'$ is a KWIK learner

due to the condition in Equation (5) and for $\mathcal{L}'$ equivalence in Equation (4) holds. ■

The condition $\kappa(s,a) \geq \zeta \;\Rightarrow\; L(s,a) \neq \bot$ describes a "matching" between the KWIK learner and the definition of the count function. Motivated by Walsh's concrete relational KWIK learner and our concrete REX instance described below, we mention another, more special case condition for a KWIK learner to match with a count function.

Corollary 3 *Assume* REX *uses an ε_P-accurate planner and a KWIK learner $\mathcal{L}$ to learn the transition model $\hat{T}$. If it employs a count function $\kappa_Q(s,a)$ defined as in Equation (2) indirectly via a set $Q = \{q_1,..,q_n\}$ of queries where the queries are mutually exclusive ($q_i(s,a) = true \rightarrow \forall k \neq i : q_k(s,a) = false$) and with confidence at least $1-\delta$*

$$\forall q \in Q: \quad c_{\mathcal{E}}(q) \geq \zeta \;\Rightarrow\; \forall s,a \text{ with } q(s,a) = true : \\ \mathcal{L}(s,a) \text{ is } \varepsilon_T\text{-accurate and } \mathcal{L}(s,a) \neq \bot \\ \text{and } \; c_{\mathcal{E}}(q) \geq polynomial(m) \;\Rightarrow\; c_{\mathcal{E}}(q) \geq \zeta\,,$$

then REX *in the* R-MAX *variant is PAC-MDP.*

The corollary follows directly from the previous lemma since if $\kappa_Q(s,a) \geq \zeta$, then there is a $q \in Q$ with $\exists \sigma : s \wedge a \models \sigma(q)$ and $c_{\mathcal{E}}(q) \geq \zeta$.

This special case is interesting since Walsh assumes such mutually exclusive contexts in his learning algorithms. Similarly, the learning algorithm of Pasula et al. tries to ensure that experienced states map to unique contexts (thus, heuristic disjoint queries). Therefore, in those special cases of mutually exclusive queries, a KWIK learner that matches a count function has to ensure to respond only a polynomial number of times with $\bot$ for each abstract context.

The above results are limited to REX in the R-MAX variant. Analogous to KWIK-R-MAX, one may conceive KWIK-E^3 which uses a KWIK learner $\mathcal{L}$ to estimate a transition model $\hat{T}$, relating to REX in the E^3 variant. This generalizes the original E^3 by replacing the counts on states and actions with the certainty estimate of $\mathcal{L}$: if $\mathcal{L}(s,a) = \bot$, then this state-action pair is considered unknown with a transition to the special state $\tilde{s}$ (which has reward 0 in M_{exploit} and reward R_{max} in M_{explore}). While in our experiments we investigate both REX in the R-MAX and E^3 variant, a proof that KWIK-E^3 is PAC-MDP is beyond the scope of this paper. To our knowledge, plain E^3 has not been shown to be strictly PAC-MDP in the sense of Definition 1, either. It is informative to see why we cannot make a proof analogous to the one of Li (2009) for KWIK-R-MAX. Li's proof is based on a theorem of Strehl et al. (2009) tailored to R-MAX-style exploration. For a given exploration algorithm, this theorem defines sufficient conditions on the value function of the algorithm's policy which ensure that the algorithm is PAC-MDP: the "optimism" condition states that this value function is lower bounded by the optimal value function in the true MDP—hence, the algorithm never misses the opportunity to harvest large rewards; the "accuracy" condition states that the value function is upper bounded by the value in a model with the true transitions for the known states—hence, the transitions need to be well estimated by the algorithm. These sufficient conditions cannot be transferred directly to E^3 since in E^3 there is not a *single* value function consistent with the policy that would fulfill these upper and lower bounds. In the exploration mode, E^3 plans more aggressively than R-MAX for unknown states from a known state and ignores rewards in the known states, thus potentially underestimating the expected value instead of overestimating it as R-MAX

does. Nonetheless, it appears likely that KWIK-E^3 is PAC-MDP, but a proof has to be made along different lines than for KWIK-R-MAX.

In this subsection, we have established general conditions under which REX is a PAC-MDP reinforcement learning algorithm. In the next subsection, we use these conditions to take a closer look at the exploration efficiency of the concrete REX instance we investigate in our experiments.

3.4 A Model-Based Reinforcement Learner for Relational Domains with Fully Unknown Transition Dynamics

Our relational exploration framework REX is independent of the concrete choices for the transition model representation and learner, the planning algorithm and the relational count function. Here, we propose a concrete instance which we will use in our evaluation. To our knowledge, this instance of REX is the first empirically evaluated relational model-based reinforcement learning algorithm which learns and exploits full-fledged models of transition dynamics. It uses NID rules to learn a transition model $\hat{T}$, plans with the PRADA algorithm and uses the contexts of learned NID rules to estimate the count functions $\kappa_\Phi(s)$ and $\kappa_\Phi(s,a)$. The rule contexts are learned from experience and provide compact descriptions of situation classes. Thus, a state is known if all actions in this state are known. An action is known if there is sufficient confidence in its covering rule. The confidence of a rule depends on the number of experiences in $\mathcal{E}$ it explains, as described above. Our context-based approach to estimate the count function is simple, but empirically effective. We are certain that more elaborate and efficient exploration strategies can be derived from the above principles in the future.

Given these concrete instantiations of the learner, planner and count function components we reconsider briefly the theoretical guarantees. The following corollary shows under which assumptions this REX instance is PAC-MDP.

Corollary 4 *If PRADA uses exact inference and the unknown true transition model T can be modeled by means of NID rules with mutually exclusive contexts, without deictic references and without the special noise outcomes, then there exists a learner $\mathcal{L}$ for such NID rules such that* REX *in the* R-MAX *variant with PRADA, $\mathcal{L}$ and a context-based count function $\kappa_\Phi(s,a)$ is PAC-MDP.*

Proof To learn such NID rules, REX can use a KWIK learner built from a combination of the KWIK learners for the individual rule components proposed by Walsh (2010). If we assume exact inference, PRADA can be adapted to produce ε_P-accurate plans. Then, the corollary follows directly from Corollary 3. ■

Corollary 4 assures us that the exploration strategy of our REX instance is theoretically justified. However, the practicability of our REX instance relies on properties of the planner and learner components which violate the assumptions of the corollary. In relational domains, we easily have to deal with thousands if not millions of random variables. Unfortunately, exact propositional inference has exponential cost in the treewidth of the ground model, making it infeasible for most real-world applications. Even recent advances in lifted exact inference approaches—handling whole sets of indistinguishable variables and factors together—are still extremely complex, generally do not scale to realistic domains, and have only been applied to very small artificial problems. Thus, using approximate inference in PRADA is crucial for the efficiency of planning. While this makes it difficult to provide optimality guarantees, in practice PRADA produces reliably effective plans in difficult noisy settings if its number of action-sequence samples is sufficiently high (Lang and

Toussaint, 2010). Using approximate inference also implies that our REX instance plans through partially unknown states. In general, it is unclear how to efficiently build and exclusively use an MDP of known relational states. However, in each state PRADA, and thus REX, only takes known actions for planning into account: PRADA only considers those actions in a state for which it has a confident unique covering rule.

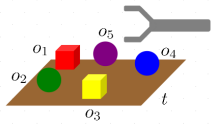
Furthermore, concerning the learning component of REX, deictic references and noise outcomes are expressive language constructs which are essential for learning and modeling the dynamics in practical applications (Pasula et al., 2007). There are, however, no KWIK-learners or other algorithms with theoretical guarantees for learning full NID rules. As noted by Walsh (2010) (p. 64), aside from not covering deictic references, his learning algorithms "would be inappropriate in a schema that allowed for 'noise' or 'miscellaneous' outcomes. [...] In such a situation, a heuristic scoring function that tries to minimize the weights on the noise term is likely a better solution." The heuristic learning algorithm for NID rules by Pasula et al. (2007) which we use in our evaluation cannot guarantee that the learned rules, and thus our learned count functions of known states and actions, converge to near-optimal models. As only contexts of rules with sufficient empirical evidence are used to estimate the count functions, however, the decisions for knowing states and actions are stable in practice. In our investigated domains, independent learning runs always converged to the same or very similar rule-sets and hence to the same learned count functions.

3.5 Illustrative Example

In Table 4, we present an example of an agent using the instance of our REX framework described in Section 3.4 with the E^3 strategy in a robot manipulation domain. In the beginning, the robot is given a relational vocabulary to describe the world on a symbolic level. It has the ability to convert its perceptions into the corresponding symbolic representation. Furthermore, it can execute two different types of motor primitives for grabbing objects and putting them on other objects. It can trigger these motor primitives by the symbolic actions $grab(X)$ and $puton(X)$. These actions are always executed and their effects depend on the context. Thus, in the example scenario with objects $O = \{o_1, o_2, o_3, o_4, o_5, t\}$ the action space of the robot consists of the actions $A = \{$ $puton(t)$, $puton(o_1)$, $puton(o_2)$, $puton(o_3)$, $puton(o_4)$, $puton(o_5)$, $grab(o_1)$, $grab(o_2)$, $grab(o_3)$, $grab(o_4)$, $grab(o_5)$ $\}$. See Toussaint et al. (2010) for a corresponding real-world robotic setup.

The robot is told to build a tower from the objects on the table. The robot does not know, however, how its actions change the state of the world. It has to learn this from experience and use its insights to achieve its goal—a prototypical model-based reinforcement learning situation. The robot will apply relational exploration to learn as much as possible about the dynamics of its world in as little time as possible. It uses NID rules to learn and represent the transition dynamics. Based on the contexts of these NID rules $Q = \{\Phi_i\}_i$ and its experiences $\mathcal{E}$, it estimates the counts of known states and actions as described above. Here, we assume the robot is confident about a rule if this rule explains at least two of its experiences and hence set $\zeta = 2$.

At $\boldsymbol{t = 0}$, the robot starts with zero knowledge about the transition dynamics. As s_0 is unknown, it performs a direct exploration action. All actions are equally unknown so it chooses randomly to grab cube o_1 and learns rule r_1 from the experience (s_0, a_0, s_1). At $\boldsymbol{t = 1}$, although all actions in s_1 and thus s_1 itself are unknown, the robot is less uncertain about the $grab(\cdot)$ actions for the objects lying on the table: these are covered by the rule r_1 which explains its single experience. It

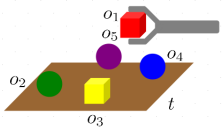


$t = 0$

No rule learned yet.

Direct exploration

Next action: $a_0 = grab(o_1)$



$t = 1$

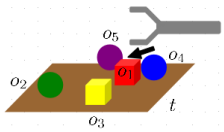
Learned rule r_1

$grab(X)$: $on(X,Y)$

$\rightarrow \begin{cases} 1.0: inhand(X), \neg on(X,Y) \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_1 = puton(o_4)$



$t = 2$

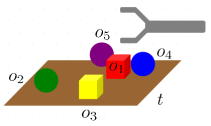
Learned rule r_2

$puton(X)$: $inhand(Y), table(Z)$

$\rightarrow \begin{cases} 1.0: on(Y,Z), \neg inhand(Y) \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_2 = puton(o_3)$



$t = 3$

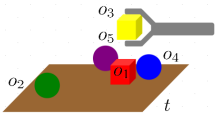
Learned rule r_3

$puton(X)$: $inhandNil()$

$\rightarrow \begin{cases} 1.0: - \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_3 = grab(o_3)$



$t = 4$

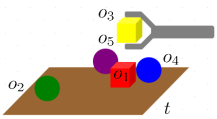
Confirmed rule r_1

$grab(X)$: $on(X,Y)$

$\rightarrow \begin{cases} 1.0: inhand(X), \neg on(X,Y) \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_4 = grab(o_3)$



$t = 5$

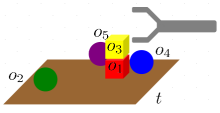
Learned rule r_4

$grab(X)$: $inhand(X)$

$\rightarrow \begin{cases} 1.0: - \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_5 = puton(o_1)$



$t = 6$

Learned rules r_5 and r_6

$puton(X)$: $\neg ball(X), inhand(Y)$

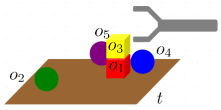
$\rightarrow \begin{cases} 1.0: on(Y,X), \neg inhand(Y) \\ 0.0: \text{noise} \end{cases}$

$puton(X)$: $ball(X), inhand(Y), table(Z)$

$\rightarrow \begin{cases} 1.0: on(Y,Z), \neg inhand(Y) \\ 0.0: \text{noise} \end{cases}$

Direct exploration

Next action: $a_6 = puton(o_2)$



$t = 7$

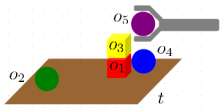
Confirmed rule r_3

$puton(X)$: $inhandNil()$

$\rightarrow \begin{cases} 1.0: - \\ 0.0: \text{noise} \end{cases}$

Planned exploration

Next action: $a_7 = grab(o_5)$



$t = 8$

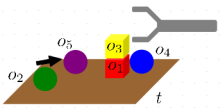
Confirmed rule r_1

$grab(X)$: $on(X,Y)$

$\rightarrow \begin{cases} 1.0: inhand(X), \neg on(X,Y) \\ 0.0: \text{noise} \end{cases}$

Planned exploration

Next action: $a_8 = puton(o_2)$



$t = 9$

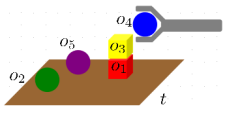
Confirmed rule r_6

$puton(X) : ball(X), inhand(Y), table(Z)$

$\rightarrow \begin{cases} 1.0: on(Y,Z), \neg inhand(Y) \\ 0.0: \text{noise} \end{cases}$

Planned exploration

Next action: $a_9 = grab(o_4)$



$t = 10$

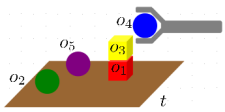
Confirmed rule r_1

$grab(X)$: $on(X,Y)$

$\rightarrow \begin{cases} 1.0: inhand(X), \neg on(X,Y) \\ 0.0: \text{noise} \end{cases}$

Planned exploration

Next action: $a_{10} = grab(o_4)$



$t = 11$

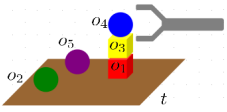
Confirmed rule r_4

$grab(X)$: $inhand(X)$

$\rightarrow \begin{cases} 1.0: - \\ 0.0: \text{noise} \end{cases}$

Planned exploration

Next action: $a_{10} = puton(o_3)$



$t = 12$

Confirmed rule r_5

$puton(X)$: $\neg ball(X), inhand(Y)$

$\rightarrow \begin{cases} 1.0: on(Y,X), \neg inhand(Y) \\ 0.0: \text{noise} \end{cases}$

Done.

Table 4: *Example of Relational E^3*. A robot manipulates objects scattered on a table by means of motor primitives triggered by symbolic actions $puton(\cdot)$ and $grab(\cdot)$. The robot is told to build a tower. It starts with zero knowledge ($\mathcal{E} = \emptyset$) and learns from experience how its actions change the state of the world. The arrows at $t = 2$ and $t = 9$ indicate that the manipulated objects have fallen off balls.

is most uncertain about grabbing the inhand object (which is not covered by r_1) and all $puton(\cdot)$ actions which are not covered by any rule. The robot chooses $puton(o_4)$ randomly among these most unknown actions. As o_4 is a ball, the inhand-held cube o_1 falls from o_4 on the table t, resulting in state s_2. The robot generalizes this experience in form of the rule r_2 that $puton(X)$ will lead to putting the inhand-object on the table. This false generalization is uncertain, however, as it is only covered by one experience.

At $\boldsymbol{t = 2}$, the robot is most uncertain about the $puton(\cdot)$ actions which are not covered by any of its learned rules (the previously learned rule r_2 requires $inhand(Y)$ in its context which is not fulfilled in s_2). Therefore, the robot chooses randomly among them and performs $puton(o_3)$. It observes no effects and learns the rule r_3, predicting *puton* actions in contexts where nothing is held inhand. At $\boldsymbol{t = 3}$, the robot is equally uncertain about all actions (all actions are covered by rules which explain exactly one experience). It chooses randomly $grab(o_3)$. The experience (s_3, a_3, s_4) confirms the rule r_1 about which it is certain by now as r_1 explains two experiences. At $\boldsymbol{t = 4}$, $grab(o_3)$ is the only action which is not covered by any rule and therefore performed. The resulting state s_5 is not different from s_4, resulting in the learned rule r_4.

At $\boldsymbol{t = 5}$, the most uncertain actions, $grab(o_3)$ and all *puton*-actions, are covered by rules with confidence one (that is, explaining one experience). The robot chooses randomly $puton(o_1)$. This provides an insightful experience: by comparing the experiences (s_1, a_1, s_2) and (s_5, a_5, s_6), the robot "understands" that rule r_2 is a false generalization and instead learns r_5 for putting on cubes and the table and r_6 putting on balls. At $\boldsymbol{t = 6}$, all *grab*-actions are known due to the confidence in rule r_1: the robot can predict their effects with confidence. In contrast, it is uncertain about all *puton*-actions as rule r_3 explains only one experience, so it randomly chooses $puton(o_2)$. The resulting experience confirms r_3 about which it is certain then.

At $\boldsymbol{t = 7}$, the robot can predict the effects of all actions with certainty. Therefore, it tries to exploit its knowledge about known states to plan for its goal to build a tower. The rule r_5 (modeling *puton*-actions for cubes) required for tower building, however, is still uncertain and thus the corresponding actions and states are unknown and cannot be considered in planning. Therefore, exploitation fails and the robot performs planned exploration instead: it plans for unknown states in which its rules r_4, r_5 and r_6 can be tested. Such states are reached by *grab*-actions and the robot chooses randomly to perform $grab(o_5)$. At $\boldsymbol{t = 8}$, the planned exploration of the last time-step allows to confirm several rules by performing one of the unknown *puton* actions or the likewise unknown $grab(o_5)$. The robot chooses randomly $puton(o_2)$. The resulting state s_9 confirms r_6 about which it is certain by now. At $\boldsymbol{t = 9}$, like in s_7 the robot performs planned exploration and grabs a random object, namely $grab(o_4)$.

At $\boldsymbol{t = 10}$, the unknown actions are $grab(o_4)$, $puton(o_1)$, $puton(o_3)$ and $puton(t)$ whose covering rules explain only one experience. The robot chooses randomly $grab(o_4)$ whose outcome in s_{11} confirms the rule r_4. At $\boldsymbol{t = 11}$, from the remaining three unknown actions, it chooses randomly $puton(o_3)$, confirming the rule r_5. At $\boldsymbol{t = 12}$, the highest possible tower has been built. Hence, the robot cannot exploit its knowledge to build an even higher tower. Similarly, planned exploration for unknown states fails and the robot concludes that it is done.

In this example, there have been no exploitation steps as in the last time-step achieved by direct and planned exploration, the highest possible reward has already been achieved. If there were more cubes, however, the robot could successfully exploit its knowledge to achieve even more rewarding states from s_{12}.

To keep this illustrative example short, we simplified in some respects: First, the robot makes most often the correct generalizations, even if competing false generalizations have the same statistical evidence. For instance, from experience (s_2, a_2, s_3) the robot could also generalize that $puton(X)$ does not have effects if X is a cube (instead of if its hand is empty). A second simplification in our example is our neglect of noise in the actions of the robot. Note however that our algorithms account for stochastic actions and our experimental evaluation is performed in intrinsically noisy domains. The third simplification is that we determined the "random" choices of the robot to be informative actions. For instance, if the robot had chosen $puton(o_1)$ in s_{11}, it might have had to revise its rule r_5 to incorporate $clear(X)$ and to come up with a rule for putting on objects which are not clear, leading to a more accurate model requiring more exploration steps.

4. Evaluation

We demonstrate that integrating our approach REX with NID rules and the planner PRADA as described in Section 3.4 leads to a practical exploration system for relational RL which outperforms established non-relational techniques on a large number of relevant problems. Our results show that REX explores efficiently worlds with many objects and transfers learned knowledge to new situations, objects and, in contrast to model-free relational RL techniques, even to new tasks. To our knowledge, this is the first evaluation of a model-based reinforcement learning agent which learns both the complete structure as well as the parameters of relational transition models.

We compare five methods inspired by E^3 and R-MAX based on different state and action representations:

- **flat E^3**: learning and exploration on an enumerated representation;
- **factored E^3**: learning and exploration on a factored propositional representation;
- **relational ε-greedy**: learning on a relational representation, ε-greedy exploration;
- **relational E^3**: learning and exploration on a relational representation; and
- **relational R-MAX**: learning and exploration on a relational representation.

It is not our goal to explicitly compare E^3 and R-MAX. We present results for both, relational E^3 and relational R-MAX, to demonstrate that REX is a practical, effective approach with either strategy. For the non-relational baselines, we have arbitrarily chosen E^3, but we might have likewise used R-MAX with similar results to expect.

We use NID rules to learn and represent transition models T in *all* investigated methods for two reasons: *(i)* there is an effective learning algorithm for NID rules, and *(ii)* we can express and learn transition models on all representation levels with NID rules which allows for a consistent comparison. Relational E^3, R-MAX and ε-greedy learn full-fledged abstract NID rules as described in Section 2.4.1. Our factored E^3 learns propositional (that is, grounded) NID rules using a slightly modified learning algorithm; ground literals imitate the propositional state attributes. (The factored E^3 algorithm of Kearns and Koller (1999) cannot learn the model structure and thus cannot be used as an alternative baseline.) Flat E^3 uses pseudo propositional NID rules where the rule context describes a complete ground relational state; hence, a rule is only applicable in a specific state and this approach corresponds to the original E^3. Exemplary rules for all representations are shown in

$$grab(X): \quad on(X,Y),\ ball(X),\ cube(Y),\ table(Z)$$
$$\rightarrow \begin{cases} 0.7 & : & inhand(X),\ \neg on(X,Y) \\ 0.2 & : & on(X,Z),\ \neg on(X,Y) \\ 0.1 & : & \text{noise} \end{cases}$$

(a) Abstract NID rule

$$grab(d): \quad on(d,b)$$
$$\rightarrow \begin{cases} 0.7 & : & inhand(d),\ \neg on(d,b) \\ 0.2 & : & on(d,t),\ \neg on(d,b) \\ 0.1 & : & \text{noise} \end{cases}$$

(b) Factored propositional NID rule

$$grab(d): \quad on(a,t), on(b,t), on(c,a), on(d,b), \neg on(a,b), \neg on(a,c) \ldots, \neg inhand(a), \ldots$$
$$\rightarrow \begin{cases} 0.7 & : & inhand(d),\ \neg on(d,b) \\ 0.2 & : & on(d,t),\ \neg on(d,b) \\ 0.1 & : & \text{noise} \end{cases}$$

(c) Flat "NID rule"

Table 5: *Illustration of NID rules on different representation levels*. (a) The abstract rule uses variables to generalize over objects (same as in Table 3). (b) The factored propositional rule imitates propositional state attributes by using only ground literals. (c) The flat rule specifies a complete state in its context. Note that the propositional rules (b) and (c) can renounce on typing predicates such as $cube(\cdot)$ as they do not abstract from object identities. The presented rules might have been learned from the grab experiences shown in Table 2.

Table 5. In each domain, the parameters for the rule learning algorithm, α and p_{min}, were chosen by a preliminary coarse heuristic search; the rule mutation operators and their order were the same across all experiments and exploration algorithms.

We use our novel exploration approach REX introduced in Section 3.2 and Section 3.4 as a concrete instance of relational E^3 and R-MAX and a factored propositional variant of REX as factored propositional E^3. Thus, we learn (abstract or propositional) NID rules for the last action after each new observation from scratch using the algorithm of Pasula et al. (2007) (appropriately modified for the propositional representations) and employ PRADA (Lang and Toussaint, 2010) for exploitation or planned exploration. The reward function is not learned, but provided to the agent. PRADA plans in the grounded relational representation and therefore can deal with rules on all abstraction levels. PRADA's planning horizon and its number of plan samples were both set to large values across all experiments to ensure that the planning performance of PRADA is reliable and has negligible influence on the reported results. To estimate a count function for known states and actions, REX and similarly its propositional counterparts use the contexts of rules $Q = \{\phi_r\}$. Similarly as done in the evaluation of KWIK-R-MAX (Li, 2009), we set the threshold ζ for known states and actions

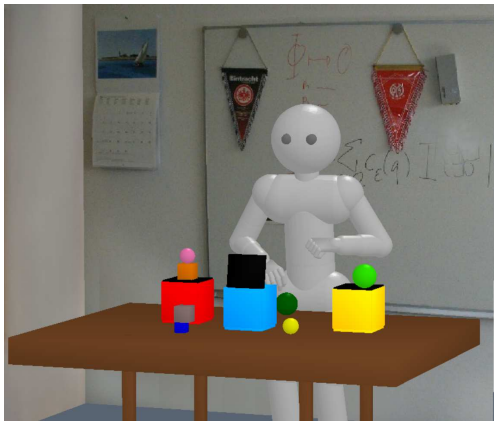
(Section 2.3) heuristically as we cannot derive it from theory due to the heuristic nature of the learning algorithm for NID rules. For all investigated E^3 and R-MAX algorithms, we set $\zeta = 2$; hence, an action is considered known in a state if its covering rule explains two experiences in $\mathcal{E}$. Since our investigated domains have mostly comparable levels of stochasticity, we did not optimize ζ for the individual domains. Although $\zeta = 2$ may not be optimal in each scenario, it allows the agent to explore the environments of our experiments within a reasonable number of actions (< 100), while providing some confidence in the estimate. We have implemented REX and its propositional counterparts, the learning algorithm for NID rules and the planning algorithm PRADA in C++.[1]

We compare our approach REX to relational ε-greedy which is the established exploration technique in relational RL approaches (see the discussion in Section 1.2). Relational ε-greedy is not a simple baseline, but an effective technique which learns abstract relational transition models (in our case NID rules) and uses them for exploitation—thereby employing the same set of known states as REX (relational E^3 and R-MAX). In contrast to REX, it performs a random action for exploration. Relational ε-greedy often profits from its optimism to try to exploit: in contrast to REX, it does not require to fully know a state before it can start exploitation there. In our experiments, we used decaying schemes to set ε dynamically, enabling a high exploration rate in the beginning and more exploitation with an increasing number of time-steps.

We emphasize that we are not aware of any other relational exploration approach for learning full transition models apart from ε-greedy which we could use as a baseline in our evaluation. As discussed in detail in Section 1.2, the conceptual approach of Walsh (2010) based on a relational R-MAX framework has not been demonstrated in practice and is supposedly intractable in large domains like ours. More importantly, it is limited to a relational language which is less expressive than the one required for a large fraction of our investigated scenarios.

Our first test domain, called *Robot manipulation domain*, is a simulated complex 3D desktop environment where a robot manipulates cubes, balls and boxes scattered on a table (Table 6). (In the following, we refer to this simulated robot when speaking of "the robot".) This is a relevant scenario in robotics: "competent pick and place operations may provide a sufficient functional basis for the manipulation requirements of a many of the targeted applications [of modern robotics]" (Christensen, 2009, p. 56). Any autonomous system which wants to deliberately manipulate objects has to master some sort of abstract reasoning at a symbolic level. In current state-of-the-art robotics, however, this problem is largely unresolved and perceived as a bottleneck. Likewise, the ability to generalize from experiences is viewed as a key desired capability of robots: "It is largely perception and machine learning that distinguish a robot from an ordinary machine" (Christensen, 2009, p. 78). We will see that our approach REX provides a practical solution to these problems. We use a 3D rigid-body dynamics simulator (ODE) that enables a realistic behavior of the manipulated objects. For instance, piles of objects may topple over or objects may even fall off the table (in which case they become out of reach for the robot). Depending on their type, objects show different characteristics. For example, it is almost impossible to successfully put an object on top of a ball, and building piles with small objects is more difficult. The robot can grab objects, try to put them on top of other objects, in a box or on the table. Boxes have a lid; special actions may open or close the lid; taking an object out of a box or putting it into it is possible only when the box is opened. *The actions of the robot are affected by noise* so that resulting object piles are not straight-aligned. We

1. The website `http://userpage.fu-berlin.de/tlang/explore/` provides our code of PRADA, the learning algorithm of NID rules and the robot manipulation simulator as well as videos of exemplary rounds in the robot manipulation domain.



Given to the robot:

- Symbolic vocabulary to describe states ($on(X,Y)$, $inhand(X)$, $clear(X)$, ...)
- Ability to convert continuous perceptions into a symbolic representation
- Symbolic actions triggering noisy motor primitives (always executed, effects depend on contexts) ($grab(X)$, $puton(X)$ ...)
- Reward function

Not given to the robot:

- Transition model for symbolic actions
  "In which contexts do the motor primitives have which effects?"

Table 6: In our robot manipulation domain, a simulated robot has to explore a 3D desktop environment with cubes, balls and boxes of different sizes and colors to master various tasks.

assume full observability of triples (s,a,s') that specify how the world changed when an action was executed in a certain state. We represent states with predicates $cube(X)$, $ball(X)$, $box(X)$, $table(X)$, $on(X,Y)$, $contains(X,Y)$, $out(X)$, $inhand(X)$, $upright(X)$, $closed(X)$, $clear(X) \equiv \forall Y.\neg on(Y,X)$, $inhandNil() \equiv \neg\exists X.inhand(X)$ and functions $size(X)$, $color(X)$. These symbols are obtained by querying the state of the simulator and translating it according to simple hand-made guidelines, thereby sidestepping the difficult problem of converting the agent's observations into an internal representation. For instance, $on(a,b)$ holds if a and b exert friction forces on each other and a's z-coordinate is greater than the one of b, while their x- and y-coordinates are similar. If there are o objects and f different object sizes and colors in a world, the state space is huge with $f^{2o}2^{2o^2+8o}$ different states (not excluding states one would classify as "impossible" given some intuition about real world physics); in our scenarios, we usually have much more than 10^{50} states. This points at the potential of using abstract relational knowledge for exploration. We define five different types of actions, denoted by different predicate symbols. These actions correspond to motor primitives whose effects and contexts we want to explore, learn and exploit. The $grab(X)$ action triggers the robot to open its hand, move its hand next to X, let it grab X and raise the robot arm again. The execution of this action is not influenced by any further factors. For example, if a different object Y has been held in the hand before, it will fall down on either the table or a third object just below Y; if there are objects on top of X, these are very likely to fall down. The $puton(X)$ action centers the robot's hand at a certain distance above X, opens it and raises the hand again. For instance, if there is an object Z on X, the object Y that was potentially inhand may end up on Z or Z might fall off X. The $openBox(X)$ and $closeBox(X)$ actions only apply to boxes. $openBox(X)$ triggers the robot to move its arm next to a box X and try to open its lid; this is only successful if there is no object on top of this box. If the box is already open, the position of the lid does not change. Similarly, $closeBox(X)$ triggers the robot to move its arm next to the lid of the box X and close it; if the box is already closed, the lid is not moved. The $doNothing()$ action triggers no movement of the robot's arm. The robot might choose this action if it thinks that any other action could be harmful with

respect to its expected reward. We emphasize again that actions always execute, regardless of the state of the world. Also, actions which are rather unintuitive for humans such as trying to grab the table or to put an object on top of itself are carried out. The robot has to learn by itself the effects and contexts of such motor primitives.

Our second set of test domains, called *IPPC* in the following, are domains taken from the international planning competition in 2008 (IPPC, 2008). While these domains are defined in the probabilistic planning domain definition language (PPDDL), the transition dynamics of many domains can be represented by NID rules (Lang and Toussaint, 2010). In our experiments, we show that these representations can be explored and learned efficiently. These experiments also demonstrate that our exploration approaches perform equally well when using a restricted representation language, namely without deictic references and noise outcomes which are not allowed by PPDDL.

We perform multiple experiments in our test domains where we pursue the same, similar or different tasks over several rounds:

- **Series 1 – Robot Manipulation Domain**:
  - Experiment 1: Simple task in unchanging worlds (Figure 2, p. 3757)
  - Experiment 2: Generalization to new worlds (Figure 3, p. 3758)
  - Experiment 3: Advanced task in unchanging worlds (Figure 4, p. 3759)
  - Experiment 4: Generalization to new tasks (Figure 5, p. 3760)
- **Series 2 – IPPC Domains**:
  - Experiment 5: Exploding Blocksworld (Figure 6, p. 3761)
  - Experiment 6: Triangle-tireworld (Figure 7, p. 3762)
  - Experiment 7: Search-and-rescue (Figure 8, p. 3763)

In all experiments *the robot starts from zero knowledge ($\mathcal{E} = \emptyset$) in the first round* and carries over experiences to the next rounds. Each round is limited by a maximum number of actions (100 in the robot manipulation domain, varying numbers in the IPPC domains). If the task is not solved by then, the round fails.

We report the success rates and the action numbers (failed trials contribute to the action number with the maximum number of actions). Both are direct measures of the goal-directedness and acting performance of an autonomous agent. Likewise these measures also evaluate the learning performance of the agent: the goal-directed performance depends on the learned rules and the active exploration on the learned count functions. Thus, high success rates and low action numbers indicate a good performance in learning rules and count functions—"good" in the sense of enabling goal-directed behavior.

4.1 Series 1 – Robot Manipulation Domain

In this first series of experiments, we compare our relational exploration approach REX in both variants (E^3 and R-MAX) to relational ε-greedy and to flat and factored E^3 approaches in successively more complex problem settings.

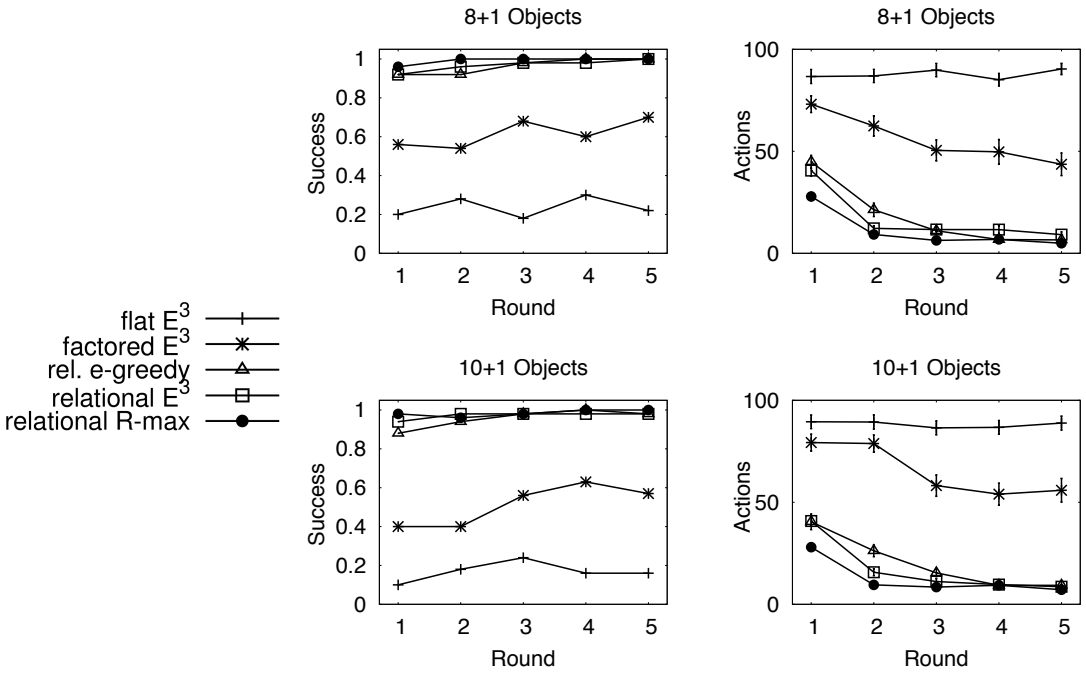


Figure 2: *Experiment 1 (Robot manipulation domain): Simple task in unchanging worlds*. A run consists of 5 subsequent rounds with the same start situations and goal objects. The robot starts with no knowledge in the first round. The success rate and the mean number of actions with standard deviations over 50 runs are shown (5 start situations, 10 seeds).

4.1.1 EXPERIMENT 1: SIMPLE TASK IN UNCHANGING WORLDS

The goal in each round is to stack two specific objects, $on(o_1, o_2)$. To collect statistics we investigate worlds of varying numbers of objects (cubes, balls, boxes and table): for each object number, we create five worlds with different objects. For each such world, we perform 10 independent runs with different random seeds. Each run consists of 5 rounds with the same goal instance and the same start situation. In some worlds, goal objects are put inside boxes in the beginning, necessitating more intense exploration to learn how to deal with boxes. Figure 2 shows that the relational explorers have superior success rates, require significantly fewer actions and reuse their learned knowledge effectively in subsequent rounds. The propositional explorers are overburdened with large numbers of objects—their learned count functions of known states and actions fail to recognize similar situations. In contrast, the relational explorers scale well with increasing numbers of objects. In the larger worlds, our REX approaches, that is relational E^3 and R-MAX, require less rounds than ε-greedy to solve the tasks with few actions: the learned count functions permit effective exploration.

4.1.2 EXPERIMENT 2: GENERALIZATION TO NEW WORLDS

In this series of experiments, the objects, their total numbers and the specific goal instances are different in each round (worlds of 7, 9 and 11 objects). We create 10 problem sequences (each with 5 rounds) and perform 10 trials for each sequence with different random seeds. As Figure 3 shows

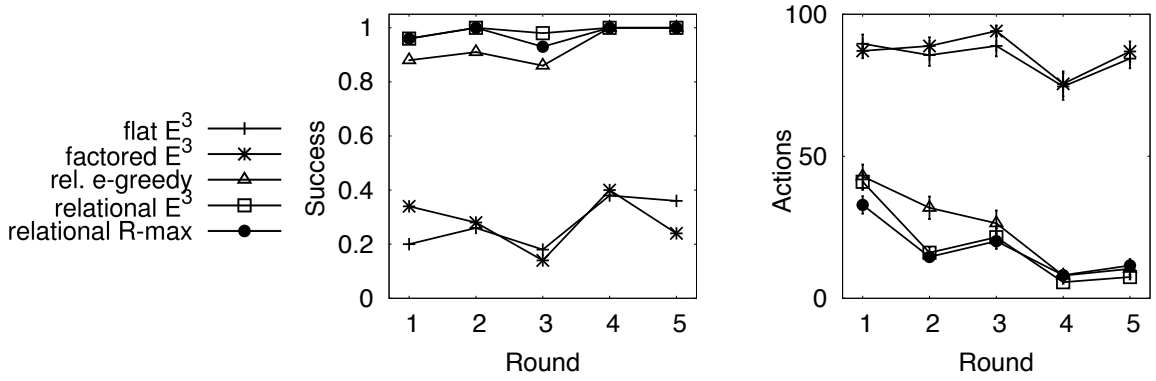


Figure 3: *Experiment 2 (Robot manipulation domain): Generalization to new worlds*. A run consists of a sequence of 5 subsequent problems (corresponding to rounds) with different objects, numbers of objects (6 - 10 cubes/balls/boxes + table) and start situations in each problem. The robot starts with no knowledge in the first round. The success rate and the mean number of actions with standard deviations over 100 runs are shown (10 sequences, 10 seeds).

the performance of the relational explorers is good from the beginning. Their learned rules and count functions for known states and actions enable a stable performance of relational E^3 and R-MAX at a near-optimal level after already 2 rounds, while relational ε-greedy requires 3-4 rounds. This experiment shows that the relational explorers can transfer their learned knowledge to new situations and objects. In contrast, the propositional explorers cannot transfer their knowledge to different worlds due to the limits of their learned rules and count functions and thus neither their success rates nor their action numbers improve in subsequent rounds.

4.1.3 EXPERIMENT 3: ADVANCED TASK IN UNCHANGING WORLDS

The goal in each round is to "clear" the table. 5 movable objects (balls and cubes) of different colors are distributed over the table. To clear up such a movable object, it needs to be put into a box of the same color. We perform 10 independent runs with different random seeds in 5 different start situations. Each run consists of 5 rounds with the same start situation. In start situations, balls and cubes may form piles, lie on top of closed boxes or be contained in boxes of a different color. Figure 4 presents our results. The non-relational exploration approaches fail to solve this task and are omitted in the figure. Relational E^3 and R-MAX outperform relational ε-greedy .

4.1.4 EXPERIMENT 4: GENERALIZATION TO NEW TASKS

Finally, we perform three tasks of increasing difficulty in succession: (i) piling two specific objects in simple worlds with cubes and balls, (ii) in worlds extended by boxes (as in Exp. 1 and 2), and (iii) clearing up the desktop by putting all movable objects into boxes of the same color where the required objects may be partially contained in wrong boxes in the beginning (as in Exp. 3). Each task is performed for three rounds in changing worlds with different goal objects. The results are presented in Figure 5. Non-relational explorers cannot generalize over objects and fail in such

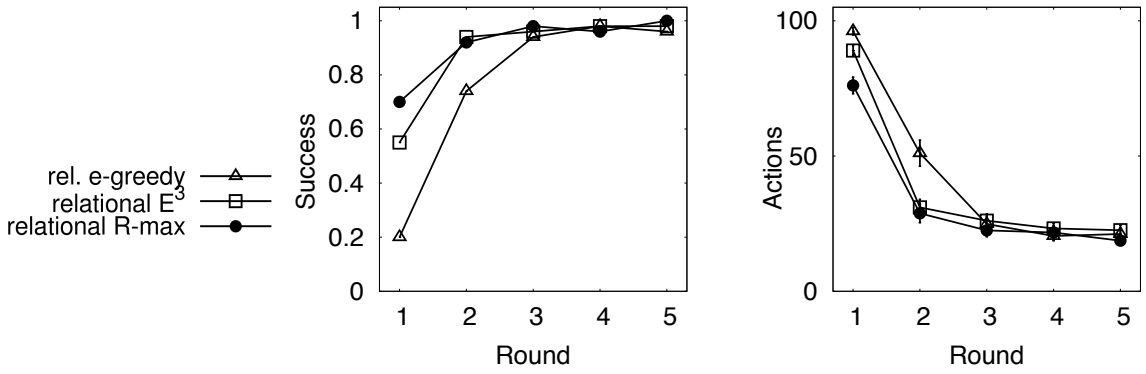


Figure 4: *Experiment 3 (Robot manipulation domain): Advanced task in unchanging worlds*. The goal is to put colored balls and cubes into boxes of the same color. A run consists of 5 subsequent rounds with the same start situation. The robot starts with zero knowledge in the first round. The success rate and the mean estimators of action numbers with standard deviations over 50 runs are shown (5 start situations, 10 seeds).

tasks (see Exp. 2 and 3) so we omit their results here. The results demonstrate that the relational explorers are not only able to generalize over different situations and objects, but transfer the learned knowledge from simple to difficult tasks in the sense of curriculum learning (Bengio et al., 2009). This is shown in the bottom row of Figure 5 where the results of relational E^3 are compared to restarting the learning procedure at the beginning of each new task (that is, in rounds 4 and 7) (the corresponding graphs for relational R-MAX and relational ε-greedy are similar). Furthermore, both relational E^3 and R-MAX benefit from their learned count functions for active exploration and outperform relational ε-greedy clearly.

4.1.5 SUMMARY

The experiments in the robot manipulation domain show that our relational exploration approach REX is a practical and effective full-system approach for exploration in challenging realistic problem settings. Our results confirm the intuitive expectation that relational knowledge improves learning transition models and learning count functions for known states and actions and thus exploration. Experiment 1 shows that relational explorers scale better with the number of objects than propositional explorers. The more challenging Experiments 2 and 3 demonstrate that the principled relational E^3 and R-MAX exploration of REX outperforms the established ε-greedy exploration: the learned count functions permit effective active exploration. Furthermore, these experiments show that REX transfers its knowledge efficiently to new situations and objects—while propositional explorers fail to do so. Finally, Experiment 4 demonstrates that REX is able to transfer its learned knowledge even to new tasks—in contrast to model-free relational approaches.

4.2 Series 2 – IPPC

In the second series of experiments, we demonstrate the effectiveness of REX in domains of the international planning competition. We choose three domains whose transition dynamics can be

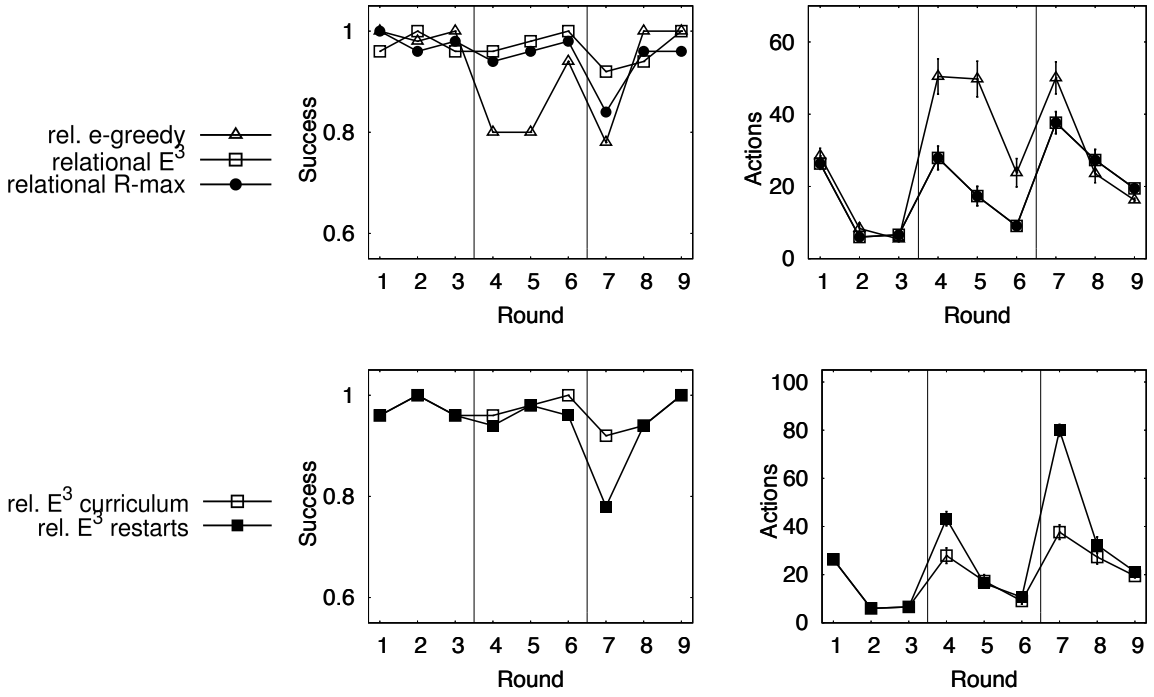


Figure 5: *Experiment 4 (Robot manipulation domain): Generalization to new tasks*. A run consists of a sequence of 9 problems (corresponding to rounds) with different objects, numbers of objects (6 - 10 cubes/balls/boxes + table) and start situations in each problem. The tasks are changed between rounds 3 and 4 and rounds 6 and 7 to more difficult tasks. The robot starts with no knowledge in the first round. The success rate and the mean number of actions with standard deviations over 50 runs are shown (10 sequences, 5 seeds). The *top row* presents the results of different relational exploration approaches. The *bottom row* compares the full curriculum relational E^3 which transfers learned knowledge to new tasks (same as in the top row) with restarting relational E^3 with each new task.

modeled by NID rules. We convert the PPDDL definitions of these domains manually into sets of NID rules and use these as the model of the true world dynamics. The agent tries to estimate the rules from its experiences. As PPDDL does not allow for deictic references and noise outcomes, this series of experiments demonstrates that REX performs also well with restricted relational languages.

For each domain, we present results on representative problem instances. To collect statistics, we perform 50 trials on the same problem instance with different random seeds. Each trial consists of 5 or 10 subsequent rounds (worlds do not change). In the IPPC domains, most actions do not have effects in a given state: in addition to specifying contexts to distinguish different effect sets, the PPDDL action operators define also restrictive action preconditions in the IPPC domains. This is in contrast to the intrinsically noisy robot manipulation domain where actions almost always have an effect. For instance, in the latter domain it is always possible to try to grab an object, be it clear, at the bottom of a pile or in a box. NID rules don't distinguish between contexts and preconditions. To focus on the context learning part, we introduce a restriction for exploration in *all* (relational and

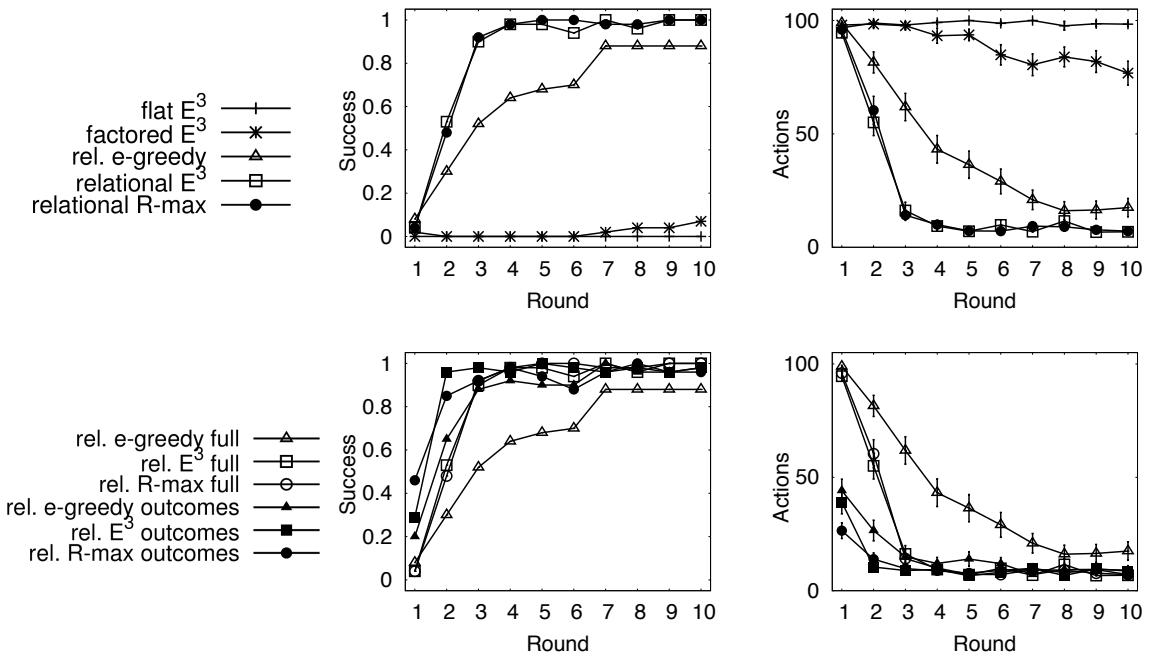


Figure 6: *Experiment 5 (IPPC): Exploding Blocksworld (problem instance 5).* A run consists of 10 subsequent rounds with the same start situation and goal objects. The agent starts with no knowledge in the first round. The success rate and the mean number of actions with standard deviations over 50 runs based on different random seeds are shown. The *top row* presents the results for learning the full transition models. The *bottom row* compares learning the full transition models (same as in the top row) with learning the outcomes and their distributions only when rule contexts are provided a-priori.

non-relational) investigated approaches: actions which have been executed without effects (so that supposedly their preconditions were violated) are forbidden until the next state change.

4.2.1 EXPERIMENT 5: EXPLODING BLOCKSWORLD

The results for problem instance 5 of this domain are displayed in the top row of Figure 6. They show that the propositional explorers almost always fail. Their performance is hampered in particular by the fact that most actions do not have effects in a given state. This is hazardous if one cannot generalize one's experiences over objects, resulting in barely useful estimated count functions for known states and actions: the propositional explorers spend too much time in each state exploring actions without effects. In contrast, the relational explorers learn quickly to solve the task. Both relational E^3 and R-MAX clearly outperform relational ε-greedy in both the success rate as well as the number of required actions, indicating the usefulness of the learned count functions for active exploration.

In the bottom row of Figure 6, the results of the relational approaches are compared to the scenario where the contexts of rules are known a-priori and only the outcomes of rules and their

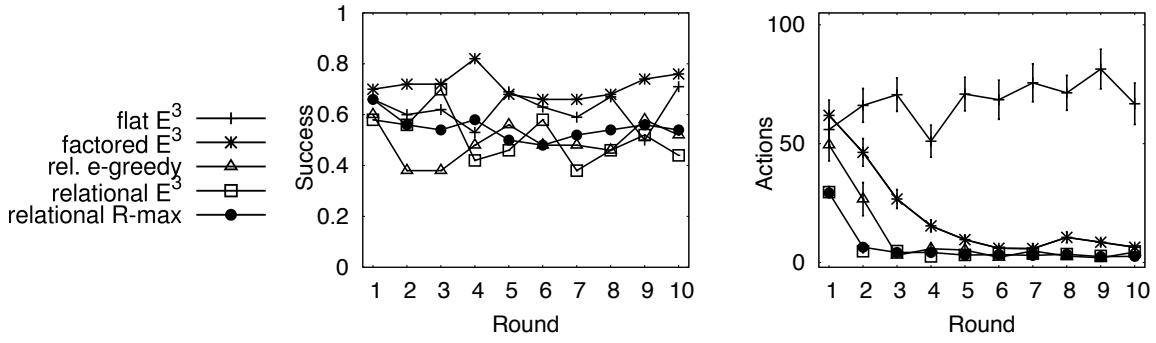


Figure 7: *Experiment 6 (IPPC): Triangle Tireworld (problem instance 1).* A run consists of 10 subsequent rounds with the same start situation and goal specification. The agent starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers with standard deviations over 50 runs based on different random seeds are shown.

probabilities need to be learned. As expected, all relational approaches learn to solve the tasks much faster in this simpler scenario. Relational ε-greedy is still clearly inferior to the principled REX approaches.

4.2.2 EXPERIMENT 6: TRIANGLE TIREWORLD

The results presented in Figure 7 for problem instance 1 show that the success rates of all methods fluctuate in similar value ranges with factored E^3 performing best. In contrast, the smaller action numbers of relational E^3 and R-MAX indicate the advantage of learning and using expressive count functions for active exploration. The difficulty of this domain lies in dead-lock situations (where the agent has a flat tire, but no spare tire is available). The specific contexts for landing in such dead-locks are hard to learn given the limited number of available relevant experiences. Here, our general choice of the threshold for knowing states and actions, $\zeta = 2$, (which we did not optimize for individual domains) is too small: the explorers are too early confident about their learned model. This hurts the relational explorers in particular as their models generalize the most and hence their potentially still inaccurate predictions get applied wrongly more often.

4.2.3 EXPERIMENT 7: SEARCH AND RESCUE

In this domain, the agent can collect intermediate and final rewards in addition to solving the task. We present the total rewards with the success rates and the action numbers for problem instance 8 in Figure 8. Overall, relational R-MAX performs best with respect to all measures; relational E^3 has similar success rates and action numbers, but collects less rewards. While flat E^3 performs worst, here factored E^3 most often outperforms relational ε-greedy. This shows the benefit of a principled exploration strategy based on learned count functions of known states and actions.

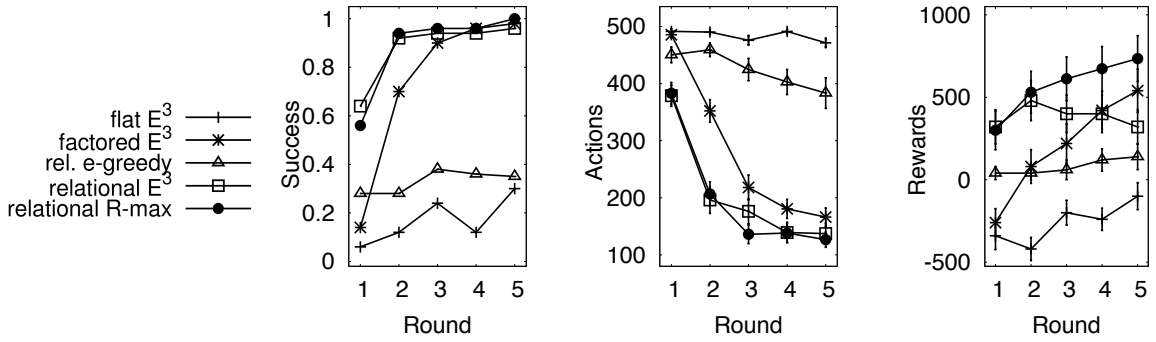


Figure 8: *Experiment 7 (IPPC): Search and Rescue (problem instance 8).* A run consists of 5 subsequent rounds with the same start situation and goal specification. The agent starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers and the rewards with standard deviations over 50 runs based on different random seeds are shown.

4.2.4 SUMMARY

The results in the IPPC domains confirm our experimental findings in the robot manipulation domain: using either E^3 or R-MAX action selection strategies, our exploration approach REX is able to efficiently explore relational domains with fully unknown transition dynamics and different characteristics. All our experiments show that learning relational count functions of known states and actions and exploiting them in a principled exploration strategy outperforms both principled propositional exploration methods as well as the established technique for relational domains, relational ε-greedy.

5. Conclusion

Efficient exploration in relational worlds is an interesting problem that is fundamental to many real-life decision-theoretic planning problems, but has received little attention so far. We have approached this problem by proposing relational exploration strategies that borrow ideas from efficient techniques for propositional representations. The key step in going from propositional to relational representations is a new definition of the concept of the novelty of states and actions. We have introduced a framework of relational count functions to estimate empirical counts of relational data. In general this is a difficult problem since it requires learning from positive data only: the agent wants to generalize the experienced state transitions to other states, but has no negative examples and thus runs the risk of overgeneralization. We have introduced a relational exploration framework called REX where such count functions are learned from experience and drive exploration in relational domains. We have provided guarantees on the exploration efficiency of REX under certain assumptions on the model learner and planner.

We have proposed an instantiation of the REX framework by integrating the relational planner PRADA and a NID rule learner. This combination results in the first relational model-based reinforcement learner with a principled exploration strategy for domains with fully unknown transition

dynamics whose effectiveness has been empirically demonstrated on a large number of difficult tasks in complex environments with many objects. Our experimental results show a significant improvement over established non-relational techniques for solving relevant, difficult and highly stochastic planning tasks in a 3D simulated robot environment and in domains of the international planning competition. Our results demonstrate that relational exploration, driven by estimated relational count functions, does not only improve the exploration performance, but also enables the transfer of learned knowledge to new situations and objects and even in a curriculum learning setting where different tasks have to be solved one after the other.

On a more general level, our work shows that it is promising to investigate the combination of statistical machine learning methods with expressive symbolic representations for developing intelligent agents. On the one hand, the symbolic representation provides abstract relational features which are key for generalization. On the other hand, an autonomous agent needs to adapt to its environment from experience and thus statistical techniques are required to actively learn compact symbolic representations.

5.1 Future Work

There are several interesting avenues for future work. One should start to explore statistical relational reasoning and learning techniques for the relational count function estimation problem implicit in exploring relational worlds. Interesting candidates include relational variants of kernel-based methods (Driessens et al., 2006), regression and cluster trees (Blockeel and de Raedt, 1998) as well as boosted relational dependency networks (Neville and Jensen, 2007; Natarajan et al., 2010) and their extension to the online learning setting. We believe our framework opens the door to a large variety of possible exploration strategies—our specific choices have only served as a first proof of concept. Another avenue of future work is to investigate incremental learning of transition models. For instance, how can a set of probabilistic relational rules be modified efficiently with additional experiences? The resulting algorithms might in turn provide new relational exploration strategies. Future work should also explore the connection between relational exploration and transfer learning. As it is hopeless to explore the whole state of a non-trivial world, investigating the relevance of objects or object classes for the task at hand is a further challenge for future research. Finally, examining our approach in other problem scenarios is appealing, for instance, applying it to large-scale applications such as the web or to geometrical reasoning of robots.

Acknowledgments

We thank the anonymous reviewers whose careful and thorough comments have helped greatly to improve the paper. TL, MT and KK were supported by the German Research Foundation (DFG, TO 409/1-3 and SPP 1527-5). KK was supported by the European Commission under contract number FP7-248258-First-MM and the Fraunhofer ATTRACT Fellowship STREAM.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 41–48, 2009.

Mustafa Bilgic, Lilyana Mihalkova, and Lise Getoor. Active learning for networked data. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2010.

Hendrik Blockeel and Luc de Raedt. Top-down induction of first order local decision trees. *Artificial Intelligence Journal*, 101:185–297, 1998.

Craig Boutilier, Ray Reiter, and Bob Price. Symbolic dynamic programming for first-order MDPs. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 690–700, 2001.

Ronen I. Brafman and Moshe Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 3:213–231, 2002.

Henrik Christensen. From internet to robotics – a roadmap for US robotics, May 2009. `http://www.us-robotics.us/reports/CCC\%20Report.pdf`.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research (JAIR)*, 4(1):129–145, 1996.

Tom Croonenborghs, Jan Ramon, Hendrik Blockeel, and Maurice Bruynooghe. Online learning and exploiting relational models in reinforcement learning. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 726–731, 2007.

Luc de Raedt, P. Frasconi, Kristian Kersting, and S.H. Muggleton, editors. *Probabilistic Inductive Logic Programming*, volume 4911 of *Lecture Notes in Computer Science*. Springer, 2008.

Carlos Diuk. *An Object-Oriented Representation for Efficient Reinforcement Learning*. PhD thesis, Rutgers, The State University of New Jersey, New Brunswick, NJ, 2010.

Carlos Diuk, Andre Cohen, and Michael Littman. An object-oriented representation for efficient reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2008.

Carlos Diuk, Lihong Li, and Bethany R. Leffler. The adaptive k-meteorologists problem and its application to structure learning and feature selection in reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2009.

Kurt Driessens and Sašo Džeroski. Integrating guidance into relational reinforcement learning. *Machine Learning Journal*, 57(3):271–304, 2004.

Kurt Driessens, Jan Ramon, and Thomas Gärtner. Graph kernels and Gaussian processes for relational reinforcement learning. *Machine Learning Journal*, 64(1-3):91–119, 2006.

Sašo Džeroski, L. de Raedt, and Kurt Driessens. Relational reinforcement learning. *Machine Learning Journal*, 43:7–52, 2001.

Arkady Epshteyn, Adam Vogel, and Gerald DeJong. Active reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 296–303, 2008.

Lise Getoor and Ben Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, 2007.

Carlos Guestrin, Relu Patrascu, and Dale Schuurmans. Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 235–242, 2002.

Carlos Guestrin, Daphne Koller, Chris Gearhart, and Neal Kanodia. Generalizing plans to new environments in relational MDPs. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 1003–1010, 2003.

Florian Halbritter and Peter Geibel. Learning models of relational MDPs using graph kernels. In *Proc. of the Mexican Conf. on AI (MICAI)*, pages 409–419, 2007.

Steffen Hölldobler, Eldar Karabaev, and Olga Skvortsova. FluCaP: a heuristic search planner for first-order MDPs. *Journal of Artificial Intelligence Research (JAIR)*, 27:419–439, 2006.

IPPC. Sixth International Planning Competition, Uncertainty Part, 2008. URL `http://ippc-2008.loria.fr/wiki/index.php/Main\_Page`.

Saket Joshi, Kristian Kersting, and Roni Khardon. Self-taught decision theoretic planning with first order decision diagrams. In *Proc. of the Int. Conf. on Automated Planning and Scheduling (ICAPS)*, 2010.

Leslie Pack Kaelbling, Michael Littman, and Andrew Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research (JAIR)*, 4:237–285, 1996.

Sham Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

Sham Kakade, Michael Kearns, and John Langford. Exploration in metric state spaces. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2003.

Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored MDPs. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 740–747, 1999.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning Journal*, 49(2-3):209–232, 2002.

Kristian Kersting and Kurt Driessens. Non–parametric policy gradients: A unified treatment of propositional and relational domains. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2008.

Kristian Kersting, Martijn van Otterlo, and Luc de Raedt. Bellman goes relational. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 465–472, 2004.

J. Zico Kolter and Andrew Ng. Near-Bayesian exploration in polynomial time. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 513–520, 2009.

Tobias Lang. *Planning and Exploration in Stochastic Relational Worlds*. PhD thesis, Fachbereich Mathematik und Informatik, Freie Universität Berlin, 2011.

Tobias Lang and Marc Toussaint. Relevance grounding for planning in relational domains. In *Proc. of the European Conf. on Machine Learning (ECML)*, 2009.

Tobias Lang and Marc Toussaint. Planning with noisy probabilistic relational rules. *Journal of Artificial Intelligence Research (JAIR)*, 39:1–49, 2010.

Lihong Li. *A Unifying Framework for Computational Reinforcement Learning Theory*. PhD thesis, Department of Computer Science, Rutgers University, New Brunswick, NJ, USA, 2009.

Lihong Li, Michael Littman, Thomas Walsh, and Alexander Strehl. Knows what it knows: a framework for self-aware learning. *Machine Learning Journal*, 82(3):568–575, 2011.

Stephen Muggleton. Learning from positive data. In *Selected Papers from the 6th International Workshop on Inductive Logic Programming*, pages 358–376, London, UK, 1997. Springer-Verlag.

Sriraam Natarajan, Tushar Khot, Kristian Kersting, Bernd Gutmann, and Jude Shavlik. Boosting relational dependency networks. In *Proc. of the Int. Conf. on Inductive Logic Programming (ILP)*, 2010.

Jennifer Neville and David Jensen. Relational dependency networks. *Journal of Machine Learning Research (JMLR)*, 8:653–692, 2007.

Shan-Hwei Nienhuys-Cheng and Ronald de Wolf, editors. *Foundations of Inductive Logic Programming*, volume 1228 of *Lecture Notes in Computer Science*. Springer, 1997.

Ali Nouri and Michael L. Littman. Dimension reduction and its application to model-based exploration in continuous spaces. *Machine Learning Journal*, 81:85–98, October 2010.

Hanna M. Pasula, Luke S. Zettlemoyer, and Leslie Pack Kaelbling. Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research (JAIR)*, 29:309–352, 2007.

Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete Bayesian reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 697–704, 2006.

Jan Ramon. *Clustering and Instance-Based Learning in First Order Logic*. PhD thesis, Department of Computer Science, K.U.Leuven, Leuven, Belgium, 2002.

Jan Ramon, Kurt Driessens, and T. Croonenborghs. Transfer learning in reinforcement learning problems through partial policy recycling. In *Proc. of the European Conf. on Machine Learning (ECML)*, pages 699–707, 2007.

Scott Sanner. Simultaneous learning of structure and value in relational reinforcement learning. In *Proc. of the ICML-05 Workshop on "Rich Representations for Relational Reinforcement Learning"*, 2005.

Scott Sanner. Online feature discovery in relational reinforcement learning. In *Proc. of the ICML-06 Workshop on "Open Problems in Statistical Relational Learning"*, 2006.

Scott Sanner and Craig Boutilier. Practical solution techniques for first-order MDPs. *Artificial Intelligence Journal*, 173(5-6):748–788, 2009.

Jürgen Schmidhuber. Curious model-building control systems. In *Proc. of Int. Joint Conf. on Neural Networks*, volume 2, pages 1458–1463, 1991.

Alexander L. Strehl and Michael L. Littman. Online linear regression and its application to model-based reinforcement learning. In *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*, pages 737–744, 2007.

Alexander L. Strehl, Lihong Li, and Michael Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research (JMLR)*, 2009.

Sebastian Thrun. The role of exploration in learning control. In *Handbook for Intelligent Control: Neural, Fuzzy and Adaptive Approaches*. Van Nostrand Reinhold, 1992.

Marc Toussaint, Nils Plath, Tobias Lang, and Nikolay Jetchev. Integrated motor control, planning, grasping and high-level reasoning in a blocks world using probabilistic inference. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

Thomas J. Walsh. *Efficient Learning of Relational Models for Sequential Decision Making*. PhD thesis, Rutgers, The State University of New Jersey, New Brunswick, NJ, 2010.

Thomas J. Walsh and Michael L. Littman. Efficient learning of action schemas and web-service descriptions. In *Proc. of the Nat. Conf. on Artificial Intelligence (AAAI)*, pages 714–719, 2008.

Thomas J. Walsh, Istvan Szita, Carlos Diuk, and Michael L. Littman. Exploring compact reinforcement-learning representations with linear regression. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2009.

Chenggang Wang, Saket Joshi, and Roni Khardon. First order decision diagrams for relational MDPs. *Journal of Artificial Intelligence Research (JAIR)*, 31:431–472, 2008.

David Windridge and Josef Kittler. Perception-action learning as an epistemologically-consistent model for self-updating cognitive representation. *Brain Inspired Cognitive Systems (special issue), Advances in Experimental Medicine and Biology*, 657, 2010.

Zhao Xu, Kristian Kersting, and Thorsten Joachims. Fast active exploration for link–based preference learning using Gaussian processes. In *Proc. of the European Conf. on Machine Learning (ECML)*, 2010.