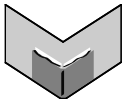
The Journal of Machine Learning Research
Volume 11
Print-Archive Edition

Pages 1–1832



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 11
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 2010.

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 11, 2010

Journal of Machine Learning Research 11 (2010) 1-18 Submitted 6/09; Revised 9/09; Published 1/10

An Efficient Explanation of Individual Classifications using Game Theory

Erik Štrumbelj ERIK.STRUMBELJ@FRI.UNI-LJ.SI
Igor Kononenko IGOR.KONONENKO@FRI.UNI-LJ.SI
Faculty of Computer and Information Science
University of Ljubljana
Tržaska 25, 1000 Ljubljana, Slovenia

Editor: Stefan Wrobel

Abstract

We present a general method for explaining individual predictions of classification models. The method is based on fundamental concepts from coalitional game theory and predictions are explained with contributions of individual feature values. We overcome the method's initial exponential time complexity with a sampling-based approximation. In the experimental part of the paper we use the developed method on models generated by several well-known machine learning algorithms on both synthetic and real-world data sets. The results demonstrate that the method is efficient and that the explanations are intuitive and useful.

Keywords: data postprocessing, classification, explanation, visualization

1. Introduction

Acquisition of knowledge from data is the quintessential task of machine learning. The data are often noisy, inconsistent, and incomplete, so various preprocessing methods are used before the appropriate machine learning algorithm is applied. The knowledge we extract this way might not be suitable for immediate use and one or more *data postprocessing* methods could be applied as well. Data postprocessing includes the integration, filtering, evaluation, and *explanation* of acquired knowledge. The latter is the topic of this paper.

To introduce the reader with some of the concepts used in this paper, we start with a simple illustrative example of an explanation for a model's prediction (see Fig. 1). We use Naive Bayes because its prediction can, due to the assumption of conditional independence, easily be transformed into *contributions of individual feature values* - a vector of numbers, one for each feature value, which indicates how much each feature value contributed to the Naive Bayes model's prediction. This can be done by simply applying the logarithm to the model's equation (see, for example, Kononenko and Kukar, 2007; Becker et al., 1997).

In our example, the contributions of the three feature values can be interpreted as follows. The prior probability of a Titanic passenger's survival is 32% and the model predicts a 67% chance of survival. The fact that this passenger was female is the sole and largest contributor to the increased chance of survival. Being a passenger from the third class and an adult both speak against survival, the latter only slightly. The actual class label for this instance is "yes", so the classification is correct. This is a trivial example, but providing the end-user with such an explanation on top of a prediction, makes the prediction easier to understand and to trust. The latter is crucial in situations

where important and sensitive decisions are made. One such example is medicine, where medical practitioners are known to be reluctant to use machine learning models, despite their often superior performance (Kononenko, 2001). The inherent ability of explaining its decisions is one of the main reasons for the frequent use of the Naive Bayes classifier in medical diagnosis and prognosis (Kononenko, 2001). The approach used to explain the decision in Fig. 1 is specific to Naive Bayes, but *can we design an explanation method which works for any type of classifier?* In this paper we address this question and propose a method for explaining the predictions of classification models, which can be applied to any classifier in a *uniform way*.

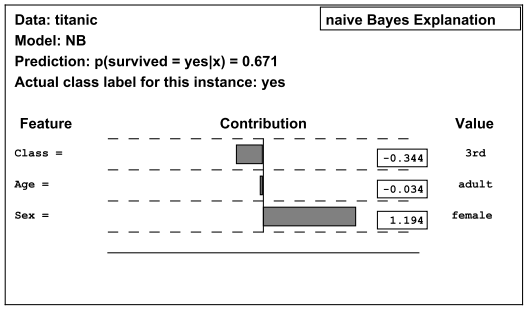


Figure 1: An instance from the well-known Titanic data set with the Naive Bayes model's prediction and an explanation in the form of contributions of individual feature values. A copy of the Titanic data set can be found at http://www.ailab.si/orange/datasets.psp.

1.1 Related Work

Before addressing *general* explanation methods, we list a few *model-specific* methods to emphasize two things. First, most models have model-specific explanation methods. And second, providing an explanation in the form of contributions of feature values is a common approach. Note that many more model-specific explanation methods exist and this is far from being a complete reference. Similar to Naive Bayes, other machine learning models also have an inherent explanation. For example, a decision tree's prediction is made by following a decision rule from the root to the leaf, which contains the instance. Decision rules and Bayesian networks are also examples of transparent classifiers. Nomograms are a way of visualizing contributions of feature values and were applied to Naive Bayes (Možina et al., 2004) and, in a limited way (linear kernel functions), to SVM (Jakulin et al., 2005). Other related work focusses on explaining the SVM model, most recently in the form of visualization (Poulet, 2004; Hamel, 2006) and rule-extraction (Martens et al., 2007). The ExplainD framework (Szafron et al., 2006) provides explanations for additive classifiers in the form of contributions. Breiman provided additional tools for showing how individual features contribute to the predictions of his Random Forests (Breiman, 2001). The explanation and interpretation of artificial neural networks, which are arguably one of the least transparent models, has also received a lot of attention, especially in the form of rule extraction (Towell and Shavlik, 1993; Andrews et al., 1995; Nayak, 2009).

So, *why do we even need a general explanation method?* It is not difficult to think of a reasonable scenario where a general explanation method would be useful. For example, imagine a user using a classifier and a corresponding explanation method. At some point the model might be replaced with a better performing model of a different type, which usually means that the ex-

planation method also has to be modified or replaced. The user then has to invest time and effort into adapting to the new explanation method. This can be avoided by using a general explanation method. Overall, a good general explanation method reduces the dependence between the user-end and the underlying machine learning methods, which makes work with machine learning models more user-friendly. This is especially desirable in commercial applications and applications of machine learning in fields outside of machine learning, such as medicine, marketing, etc. An effective and efficient general explanation method would also be a useful tool for comparing how a model predicts different instances and how different models predict the same instance.

As far as the authors are aware, there exist two other general explanation methods for explaining a model's prediction: the work by Robnik-Šikonja and Kononenko (2008) and the work by Lemaire et al. (2008). While there are several differences between the two methods, both explain a prediction with contributions of feature values and both use the same basic approach. A feature value's contribution is defined as the difference between the model's initial prediction and its average prediction across perturbations of the corresponding feature. In other words, we look at how the prediction would change if we "ignore" the feature value. This myopic approach can lead to serious errors if the feature values are conditionally dependent, which is especially evident when a disjunctive concept (or any other form of redundancy) is present. We can use a simple example to illustrate how these methods work. Imagine we ask someone who is knowledgeable in boolean logic *What will the result of (*1 *OR* 1*) be?*. It will be one, of course. Now we mask the first value and ask again *What will the result of (something OR* 1*) be?*. It will still be one. So, it does not matter if the person knows or does not know the first value - the result does not change. Hence, we conclude that the first value is irrelevant for that persons decision regarding whether the result will be 1 or 0. Symmetrically, we can conclude that the second value is also irrelevant for the persons decision making process. Therefore, both values are irrelevant. This is, of course, an incorrect explanation of how these two values contribute to the persons decision.

Further details and examples of where existing methods would fail can be found in our previous work (Štrumbelj et al., 2009), where we suggest observing the changes across all possible subsets of features values. While this effectively deals with the shortcomings of previous methods, it suffers from an exponential time complexity.

To summarize, we have existing general explanation methods, which sacrifice a part of their effectiveness for efficiency, and we know that generating effective contributions requires observing the power set of all features, which is far from efficient. The contribution of this paper and its improvement over our previous work is twofold. First, we provide a rigorous theoretical analysis of our explanation method and link it with known concepts from game theory, thus formalizing some of its desirable properties. And second, we propose an efficient sampling-based approximation method, which overcomes the exponential time complexity and does not require retraining the classifier.

The remainder of this paper is organized as follows. Section 2 introduces some basic concepts from classification and coalitional game theory. In Section 3 we provide the theoretical foundations, the approximation method, and a simple illustrative example. Section 4 covers the experimental part of our work. With Section 5 we conclude the paper and provide ideas for future work.

2. Preliminaries

First, we introduce some basic concepts from classification and coalitional game theory, which are used in the formal description of our explanation method.

2.1 Classification

In machine learning classification is a form of supervised learning where the objective is to predict the *class label* for unlabelled input instances, each described by feature values from a *feature space*. Predictions are based on background knowledge and knowledge extracted (that is, learned) from a sample of labelled instances (usually in the form of a training set).

Definition 1 *The feature space $\mathcal{A}$ is the cartesian product of n features (represented with the set $N = \{1,2,...,n\}$): $\mathcal{A} = A_1 \times A_2 \times ... \times A_n$, where each feature A_i is a finite set of feature values.*

Remark 1 *With this definition of a feature space we limit ourselves to finite (that is, discrete) features. However, we later show that this restriction does not apply to the approximation method, which can handle both discrete and continuous features.*

To formally describe situations where feature values are ignored, we define a subspace $\mathcal{A}_S = A'_1 \times A'_2 \times ... \times A'_n$, where $A'_i = A_i$ if $i \in S$ and $A'_i = \{\varepsilon\}$ otherwise. Therefore, given a set $S \subset N$, $\mathcal{A}_S$ is a feature subspace, where features not in S are "ignored" ($\mathcal{A}_N = \mathcal{A}$). Instances from a subspace have one or more components unknown as indicated by ε. Now we define a classifier.

Definition 2 *A classifier, f, is a mapping from a feature space to a normalized $|C|$-dimensional space $f : \mathcal{A} \to [0,1]^{|C|}$, where C is a finite set of labels.*

Remark 2 *We use a more general definition of a classifier to include classifiers which assign a rank or score to each class label. However, in practice, we mostly deal with two special cases: classifiers in the traditional sense (for each vector, one of the components is 1 and the rest are 0) and probabilistic classifiers (for each vector, the vector components always add up to 1 and are therefore a probability distribution over the class label state space).*

2.2 Coalitional Game Theory

The following concepts from coalitional game theory are used in the formalization of our method, starting with the definition of a coalitional game.

Definition 3 *A coalitional form game is a tuple $\langle N, v \rangle$, where $N = \{1,2,...,n\}$ is a finite set of n players, and $v : 2^N \to \Re$ is a characteristic function such that $v(\varnothing) = 0$.*

Subsets of N are *coalitions* and N is referred to as the *grand coalition* of all players. Function v describes the *worth* of each coalition. We usually assume that the grand coalition forms and the goal is to split its worth $v(N)$ among the players in a "fair" way. Therefore, the *value* (that is, solution) is an operator ϕ which assigns to $\langle N, v \rangle$ a vector of payoffs $\phi(v) = (\phi_1, ..., \phi_n) \in \Re^n$. For each game with at least one player there are infinitely many solutions, some of which are more "fair" than others. The following four statements are attempts at axiomatizing the notion of "fairness" of a solution ϕ and are key for the axiomatic characterization of the Shapley value.

Axiom 1 $\sum_{i \in N} \phi_i(v) = v(N)$. *(efficiency axiom)*

Axiom 2 *If for two players i and j $v(S \cup \{i\}) = v(S \cup \{j\})$ holds for every S, where $S \subset N$ and $i, j \notin S$, then $\phi_i(v) = \phi_j(v)$). (symmetry axiom)*

Axiom 3 *If $v(S\cup\{i\}) = v(S)$ holds for every S, where $S \subset N$ and $i \notin S$, then $\phi_i(v) = 0$. (dummy axiom)*

Axiom 4 *For any pair of games $v, w : \phi(v+w) = \phi(v) + \phi(w)$, where $(v+w)(S) = v(S) + w(S)$ for all S. (additivity axiom)*

Theorem 1 *For the game $\langle N, v\rangle$ there exists a unique solution ϕ, which satisfies axioms 1 to 4 and it is the Shapley value:*

$$Sh_i(v) = \sum_{S\subseteq N\setminus\{i\}, s=|S|} \frac{(n-s-1)!s!}{n!}(v(S\cup\{i\}) - v(S)), \quad i = 1,...,n.$$

Proof For a detailed proof of this theorem refer to Shapley's paper (1953). ■

The Shapley value has a wide range of applicability as illustrated in a recent survey paper by Moretti and Patrone (2008), which is dedicated entirely to this unique solution concept and its applications. From the few applications of the Shapley value in machine learning, we would like to bring to the readers attention the work of Keinan et al. (2004), who apply the Shapley value to function localization in biological and artificial networks. Their MSA framework is later used and adapted into a method for feature selection (Cohen et al., 2007).

3. Explaining Individual Predictions

In this section we provide the theoretical background. We start with a description of the intuition behind the method and then link it with coalitional game theory.

3.1 Definition of the Explanation Method

Let $N = \{1,2,...,n\}$ be a set representing n features, f a classifier, and $x = (x_1, x_2, ..., x_n) \in \mathcal{A}$ an instance from the feature space. First, we choose a class label. We usually chose the predicted class label, but we may choose any other class label that is of interest to us and explain the prediction from that perspective (for example, in the introductory example in Fig. 1 we could have chosen "survival = no" instead). Let c be the chosen class label and let $f_c(x)$ be the prediction component which corresponds to c. Our goal is to explain how the given feature values contribute to the *prediction difference* between the classifiers prediction for this instance and the expected prediction if no feature values are given (that is, if all feature values are "ignored"). The prediction difference can be generalized to an arbitrary subset of features $S \subseteq N$.

Definition 4 *The prediction difference $\Delta(S)$ when only values of features represented in S are known, is defined as follows:*

$$\Delta(S) = \frac{1}{|\mathcal{A}_{N\setminus S}|}\sum_{y\in\mathcal{A}_{N\setminus S}} f_c(\tau(x,y,S)) - \frac{1}{|\mathcal{A}_N|}\sum_{y\in\mathcal{A}_N} f_c(y), \tag{1}$$

$$\tau(x,y,S) = (z_1, z_2, ..., z_n), \quad z_i = \begin{cases} x_i & ; \quad i \in S \\ y_i & ; \quad i \notin S. \end{cases}$$

Remark 3 *In our previous work (Štrumbelj et al., 2009) we used a different definition:* $\Delta(S) = f_c^*(S) - f_c^*(\varnothing)$*, where* $f^*(W)$ *is obtained by retraining the classifier only on features in* W *and re-classifying the instance (similar to the wrappers approach in feature selection Kohavi and John 1997). With this retraining approach we avoid the combinatorial explosion of going through all possible perturbations of "ignored" feature's values, but introduce other issues. However, the exponential complexity of going through all subsets of* N *still remains.*

The expression $\Delta(S)$ is the difference between the expected prediction when we know only those values of x, whose features are in S, and the expected prediction when no feature values are known. Note that we assume uniform distribution of feature values. Therefore, we make no assumption about the prior relevance of individual feature values. In other words, we are equally interested in each feature value's contribution. The main shortcoming of existing general explanation methods is that they do not take into account all the potential dependencies and interactions between feature values. To avoid this issue, we implicitly define interactions by defining that each prediction difference $\Delta(S)$ is composed of 2^N contributions of interactions (that is, each subset of feature values might contribute something):

$$\Delta(S) = \sum_{W \subseteq S} I(W), \quad S \subseteq N. \tag{2}$$

Assuming $I(\varnothing) = 0$ (that is, an interaction of nothing always contributes 0) yields a recursive definition. Therefore, function I, which describes the interactions, always exists and is uniquely defined for a given N and function Δ:

$$I(S) = \Delta(S) - \sum_{W \subset S} I(W), \quad S \subseteq N. \tag{3}$$

Now we distribute the interaction contributions among the n feature values. For each interaction the involved feature values can be treated as equally responsible for the interaction as the interaction would otherwise not exist. Therefore, we define a feature value's contribution φ_i by assigning it an equal share of each interaction it is involved in

$$\varphi_i(\Delta) = \sum_{W \subseteq N \setminus \{i\}} \frac{I(W \cup \{i\})}{|W \cup \{i\}|}, \quad i = 1,2,...,n. \tag{4}$$

It is not surprising that we manage in some way to uniquely quantify all the possible interactions, because we explore the entire power set of the involved feature values. So, two questions arise: *Can we make this approach computationally feasible?* and *What are the advantages of this approach over other possible divisions of contributions?*. We now address both of these issues, starting with the latter.

Theorem 2 $\langle N = \{1,2,...,n\}, \Delta \rangle$ *is a coalitional form game and* $\varphi(\Delta) = (\varphi_1, \varphi_2, ..., \varphi_n)$ *corresponds to the game's Shapley value* $Sh(\Delta)$.

Proof Following the definition of Δ we get that $\Delta(\varnothing) = 0$, so the explanation of the classifier's prediction can be treated as a coalitional form game $\langle N, \Delta \rangle$. Now we provide an elementary proof that the contributions of individual feature values correspond to the Shapley value for the game

$\langle N, \Delta \rangle$. The recursive definition of I given in Eq. (3) can be transformed into its non-recursive form:

$$I(S) = \sum_{W \subseteq S} ((-1)^{|S|-|W|} \Delta(W)). \tag{5}$$

Eq. (5) can be proven by induction. We combine Eq. (4) and Eq. (5) into the following non-recursive formulation of the contribution of a feature value:

$$\varphi_i(\Delta) = \sum_{W \subseteq N \setminus \{i\}} \frac{\sum_{Q \subseteq (W \cup \{i\})} ((-1)^{|W \cup \{i\}|-|Q|} \Delta(Q))}{|W \cup \{i\}|}. \tag{6}$$

Let us examine the number of times $\Delta(S \cup \{i\})$, $S \subseteq N, i \notin S$, appears on the right-hand side of Eq. (6). Let $M_{\Delta(S \cup \{i\})}$ be the number of all such appearances and $k = n - |S| - 1$. The term $\Delta(S \cup \{i\})$ appears when $S \subseteq W$ and only once for each such W. For W, where $S \subseteq W$ and $|W| = |S| + a$, $\Delta(S \cup \{i\})$ appears with an alternating sign, depending on the parity of a, and there are exactly $\binom{k}{a}$ such W in the sum in Eq. (6), because we can use any combination of a additional elements from the remaining k elements that are not already in the set S. If we write all such terms up to $W = N$ and take into account that each interaction $I(W)$ is divided by $|W|$, we get the following series:

The treatment is similar for $\Delta(S)$, $i \notin S$ where we get $M_{\Delta(S)} = -V(n,k)$. The series $V(n,k)$ can be expressed with the beta function:

$$(1-x)^k = \binom{k}{0} - \binom{k}{1} x + \binom{k}{2} x^2 - \ldots \pm \binom{k}{k} x^k$$

$$\int_0^1 x^{n-k-1} (1-x)^k \, dx = \int_0^1 (\binom{k}{0} x^{n-k-1} - \binom{k}{1} x^{n-k} + \ldots \pm \binom{k}{n-1} x^{n-1}) \, dx$$

$$B(n-k, k+1) = \frac{\binom{k}{0}}{n-k} - \frac{\binom{k}{1}}{n-k+1} + \ldots \pm \frac{\binom{k}{k}}{n} = V(n,k).$$

Using $B(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$, we get $V(n,k) = \frac{(n-k-1)!k!}{n!}$. Therefore:

$$\varphi_i(\Delta) = \sum_{S \subseteq N \setminus \{i\}} V(n, n-s-1) \cdot \Delta(S \cup \{i\}) - \sum_{S \subseteq N \setminus \{i\}} V(n, n-s-1) \cdot \Delta(S) =$$

$$= \sum_{S \subseteq N \setminus \{i\}} \frac{(n-s-1)!s!}{n!} \cdot (\Delta(S \cup \{i\}) - \Delta(S)).$$

■

So, the explanation method can be interpreted as follows. The instance's feature values form a coalition which causes a change in the classifier's prediction. We divide this change amongst the feature values in a way that is fair to their contributions across all possible sub-coalitions. Now that we have established that the contributions correspond to the Shapley value, we take another look at its axiomatization. Axioms 1 to 3 and their interpretation in the context of our explanation method are of particular interest. The 1st axiom corresponds to our decomposition in Eq. (2) - the

sum of all n contributions in an instance's explanation is equal to the difference in prediction $\Delta(N)$. Therefore, the contributions are implicitly normalized, which makes them easier to compare across different instances and different models. According to the 2nd axiom, if two features values have an identical influence on the prediction they are assigned contributions of equal size. The 3rd axiom says that if a feature has no influence on the prediction it is assigned a contribution of 0. When viewed together, these properties ensure that any effect the features might have on the classifiers output will be reflected in the generated contributions, which effectively deals with the issues of previous general explanation methods.

3.1.1 AN ILLUSTRATIVE EXAMPLE

In the introduction we used a simple boolean logic example to illustrate the shortcomings of existing general explanation methods. We concluded that in the expression (1 OR 1) both values are irrelevant and contribute nothing to the result being 1. This error results from not observing all the possible subsets of features. With the same example we illustrate how our explanation method works. We write $N = \{1,2\}$, $\mathcal{A} = \{0,1\} \times \{0,1\}$, and $x = (1,1)$. In other words, we are explaining the classifier's prediction for the expression $(1 \text{ OR } 1)$. Following the steps described in Section 3, we use Eq. (1) to calculate the $\Delta-$terms. Intuitively, $\Delta(S)$ is the difference between the classifiers expected prediction if only values of features in S are known and the expected prediction if no values are known. If the value of at least one of the two features is known, we can predict, with certainty, that the result is 1. If both values are unknown (that is, masked) one can predict that the probability of the result being 1 is $\frac{3}{4}$. Therefore, $\Delta(1) = \Delta(2) = \Delta(1,2) = 1 - \frac{3}{4} = \frac{1}{4}$ and $\Delta(\varnothing) = \frac{3}{4} - \frac{3}{4} = 0$. Now we can calculate the interactions. $I(1) = \Delta(1) = \frac{1}{4}$ and $I(2) = \Delta(2) = \frac{1}{4}$. When observed together, the two features contribute less than their individual contributions would suggest, which results in a negative interaction: $I(1,2) = \Delta(1,2) - (I(1) + I(2)) = -\frac{1}{4}$. Finally, we divide the interactions to get the final contributions: $\varphi_1 = I(1) + \frac{I(1,2)}{2} = \frac{1}{8}$ and $\varphi_2 = I(2) + \frac{I(1,2)}{2} = \frac{1}{8}$. The generated contributions reveal that both features contribute the same amount towards the prediction being 1 and the contributions sum up to the initial difference between the prediction for this instance and the prior belief.

3.2 An Approximation

We have shown that the generated contributions, φ_i, are effective in relating how individual feature values influence the classifier's prediction. Now we provide an efficient approximation. The approximation method is based on a well known alternative formulation of the Shapley value. Let $\pi(N)$ be the set of all ordered permutations of N. Let $Pre^i(O)$ be the set of players which are predecessors of player i in the order $O \in \pi(N)$. A feature value's contribution can now be expressed as:

$$\varphi_i(\Delta) = \frac{1}{n!} \sum_{O \in \pi(N)} \left(\Delta(Pre^i(O) \cup \{i\}) - \Delta(Pre^i(O)) \right), \quad i = 1, ..., n. \tag{7}$$

Eq. (7) is the a well-known alternative formulation of the Shapley value. An algorithm for the computation of the Shapley value, which is based on Eq. (7), was presented by Castro et al. (2008). However, in our case, the exponential time complexity is still hidden in our definition of Δ (see Eq. (1)). If we use the alternative definition used in our previous work (see Remark 3), we can compute function $\Delta(S)$, for a given S, in polynomial time (assuming that the learning algorithm has

a polynomial time complexity). However, this requires retraining the classifier for each $S \subseteq N$, so the method would no longer be independent of the learning algorithm and we would also require the training set that the original classifier was trained on. To avoid this and still achieve an efficient explanation method, we extend the sampling algorithm in the following way. We use a different, but equivalent formulation of Eq. (1). While the sum in this equation redundantly counts each $f(\tau(x,y,S))$ term $|\mathcal{A}_S|$ times (instead of just once) it is equivalent to Eq. (1) and simplifies the sampling procedure:

$$\Delta(S) = \frac{1}{|\mathcal{A}|} \sum_{y \in \mathcal{A}} \left(f(\tau(x,y,S)) - f(y) \right). \tag{8}$$

We replace occurrences of Δ in Eq. (7) with Eq. (8):

$$\varphi_i(\Delta) = \frac{1}{n! \cdot |\mathcal{A}|} \sum_{O \in \pi(N)} \sum_{y \in \mathcal{A}} \left(f(\tau(x,y,Pre^i(O) \cup \{i\})) - f(\tau(x,y,Pre^i(O))) \right).$$

We use the following sampling procedure. Our sampling population is $\pi(N) \times \mathcal{A}$ and each order/instance pair defines one sample $X_{O,y \in \mathcal{A}} = f(\tau(x,y,Pre^i(O) \cup \{i\})) - f(\tau(x,y,Pre^i(O)))$. If some features are continuous, we have an infinite population, but the properties of the sampling procedure do not change. If we draw a sample completely at random then all samples have an equal probability of being drawn ($\frac{1}{n! \cdot |\mathcal{A}|}$) and $E[X_{O,y \in \mathcal{A}}] = \varphi_i$. Now consider the case where m such samples are drawn (with replacement) and observe the random variable $\hat{\varphi}_i = \frac{1}{m} \sum_{j=1}^{m} X_j$, where X_j is the $j-$th sample. According to the central limit theorem, $\hat{\varphi}_i$ is approximately normally distributed with mean φ_i and variance $\frac{\sigma_i^2}{m}$, where σ_i^2 is the population variance for the $i-$th feature. Therefore, $\hat{\varphi}_i$ is an unbiased and consistent estimator of φ_i. The computation is summarized in Algorithm 1.

Algorithm 1 Approximating the contribution of the i-th feature's value, φ_i, for instance $x \in \mathcal{A}$.

determine m, the desired number of samples
$\varphi_i \leftarrow 0$
for $j = 1$ to m **do**
  choose a random permutation of features $O \in \pi(N)$
  choose a random instance $y \in \mathcal{A}$
  $v_1 \leftarrow f(\tau(x,y,Pre^i(O) \cup \{i\}))$
  $v_2 \leftarrow f(\tau(x,y,Pre^i(O)))$
  $\varphi_i \leftarrow \varphi_i + (v_1 - v_2)$
end for
$\varphi_i \leftarrow \frac{\varphi_i}{m}$

$\{v_1$ and v_2 are the classifier's predictions for two instances, which are constructed by taking instance y and then changing the value of each feature which appears before the i-th feature in order O (for v_1 this includes the $i-$th feature) to that feature's value in x. Therefore, these two instances only differ in the value of the $i-$th feature.$\}$

3.2.1 ERROR/SPEED TRADEOFF

We have established an unbiased estimator of the contribution. Now we investigate the relationship between the number of samples we draw and the approximation error. For each φ_i, the number of samples we need to draw to achieve the desired error, depends only on the population variance σ_i^2. In practice, σ_i^2 is usually unknown, but has an upper bound, which is reached if the population is uniformly distributed among its two extreme values. According to Eq. (1), the maximum and minimum value of a single sample are 1 and -1, respectively, so $\sigma_i^2 \leq 1$. Let the tuple $\langle 1-\alpha, e\rangle$ be a description of the desired error restriction and $P(|\varphi_i - \hat{\varphi}_i| < e) = 1-\alpha$ the condition, which has to be fulfilled to satisfy the restriction. For any given $\langle 1-\alpha, e\rangle$, there is a constant number of samples we need to satisfy the restriction: $m_i(\langle 1-\alpha, e\rangle) = \frac{Z_{1-\alpha}^2 \cdot \sigma^2}{e^2}$, where $Z_{1-\alpha}^2$ is the Z-score, which corresponds to the $1-\alpha$ confidence interval. For example, we want 99% of the approximated contributions to be less than 0.01 away from their actual values and we assume worst-case variance $\sigma_i^2 = 1$, for each $i \in N$. Therefore, we have to draw approximately 65000 samples per feature, regardless of how large the feature space is. The variances are much lower in practice, as we show in the next section.

For each feature value, the number of samples $m_i(\langle 1-\alpha, e\rangle)$ is a linear function of the sample variance σ_i^2. The key to minimizing the number of samples is to estimate the sample variance σ_i^2 and draw the appropriate number of samples. This estimation can be done during the sampling process, by providing confidence intervals for the required number of samples, based on our estimation of variance on the samples we already took. While this will improve running times, it will not have any effect on the time complexity of the method, so we delegate this to further work. The optimal (minimal) number of samples we need for the entire explanation is: $m_{min}(\langle 1-\alpha, e\rangle) = n \cdot \frac{Z_{1-\alpha}^2 \cdot \overline{\sigma^2}}{e^2}$, where $\overline{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n} \sigma_i^2$. Therefore, the number $n \cdot \overline{\sigma^2}$, where $\overline{\sigma^2}$ is estimated across several instances, gives a complete description of how complex a model's prediction is to explain (that is, proportional to how many samples we need).

A note on the method's time complexity. When explaining an instance, the sampling process has to be repeated for each of the n feature values. Therefore, for a given error and confidence level, the time complexity of the explanation is $O(n \cdot T(\mathcal{A}))$, where function $T(\mathcal{A})$ describes the instance classification time of the model on $\mathcal{A}$. For most machine learning models $T(\mathcal{A}) \leq n$.

4. Empirical Results

The evaluation of the approximation method is straightforward as we focus only on approximation errors and running times. We use a variety of different classifiers both to illustrate that it is indeed a general explanation method and to investigate how the method behaves with different types of classification models. The following models are used: a decision tree (DT), a Naive Bayes (NB), a SVM with polynomial kernel (SVM), a multi-layer perceptron artificial neural network (ANN), Breiman's random forests algorithm (RF), logistic regression (logREG), and ADABoost boosting with either Naive Bayes (bstNB) or a decision tree (bstDT) as the weak learner. All experiments were done on an off-the-shelf laptop computer (2GHz dual core CPU, 2GB RAM), the explanation method is a straightforward Java implementation of the equations presented in this paper, and the classifiers were imported from the Weka machine learning software (`http://www.cs.waikato.ac.nz/~ml/weka/index.html`).

model	# instances	# features (n)	$\max(\sigma_i^2)$	$\max(\overline{\sigma^2})$	$n\cdot \max(\overline{\sigma^2})$
CondInd	2000	8	0.25	0.06	0.48
Xor	2000	6	0.32	0.16	0.96
Group	2000	4	0.30	0.16	0.64
Cross	2000	4	0.43	0.14	0.92
Chess	2000	4	0.44	0.22	0.88
Sphere	2000	5	0.21	0.13	0.65
Disjunct	2000	5	0.10	0.06	0.30
Random	2000	4	0.19	0.12	0.48
Oncology	849	13	0.16	0.08	1.04
Annealing	798	38	0.08	0.02	0.76
Arrhythymia	452	279	0.03	10^{-3}	0.28
Breast cancer	286	9	0.22	0.10	0.90
Hepatitis	155	19	0.20	0.05	0.95
Ionosphere	351	34	0.20	0.04	1.36
Iris	150	4	0.23	0.10	0.40
Monks1	432	6	0.29	0.12	0.72
Monks2	432	6	0.31	0.27	1.62
Monks3	432	6	0.20	0.07	0.42
Mushroom	8124	22	0.24	0.05	1.10
Nursery	12960	8	0.22	0.03	0.24
Soybean	307	35	0.20	0.01	0.35
Thyroid	7200	21	0.18	0.02	0.42
Zoo	101	17	0.25	0.02	0.14

Table 1: List of data sets used in our experiments. The variance of the most complex feature value and the variance of most complex model to explain are included.

The list of data sets used in our experiments can be found in Table 1. The first 8 data sets are synthetic data sets, designed specifically for testing explanation methods (see Robnik-Šikonja and Kononenko, 2008; Štrumbelj et al., 2009). The synthetic data sets contain the following concepts: conditionally independent features (CondInd), the xor problem (Xor, Cross, Chess), irrelevant features only (Random), disjunction (Disjunct, Sphere), and spatially clustered class values (Group). The Oncology data set is a real-world oncology data set provided by the Institute of Oncology, Ljubljana. To conserve space, we do not provide all the details about this data set, but we do use an instance from it as an illustrative example. Those interested in a more detailed description of this data set and how our previous explanation method is successfully applied in practice can refer to our previous work (Štrumbelj et al., 2009). The remaining 14 data sets are from the UCI machine learning repository (Asuncion and Newman, 2009).

The goal of our first experiment is to illustrate how approximated contributions converge to the actual values. The fact that they do is already evident from the theoretical analysis. However, the reader might find useful this additional information about the behavior of the approximation error. The following procedure is used. For each data set we use half of the instances for training and half for testing the explanation method. For each data set/classifier pair, we train the classifier on the training set and use both the explanation method and its approximation on each test instance. For the approximation method the latter part is repeated several times, each time with a different setting of how many samples are drawn per feature. Only synthetic data sets were used in this procedure, because the smaller number of features allows us to compute the actual contributions.

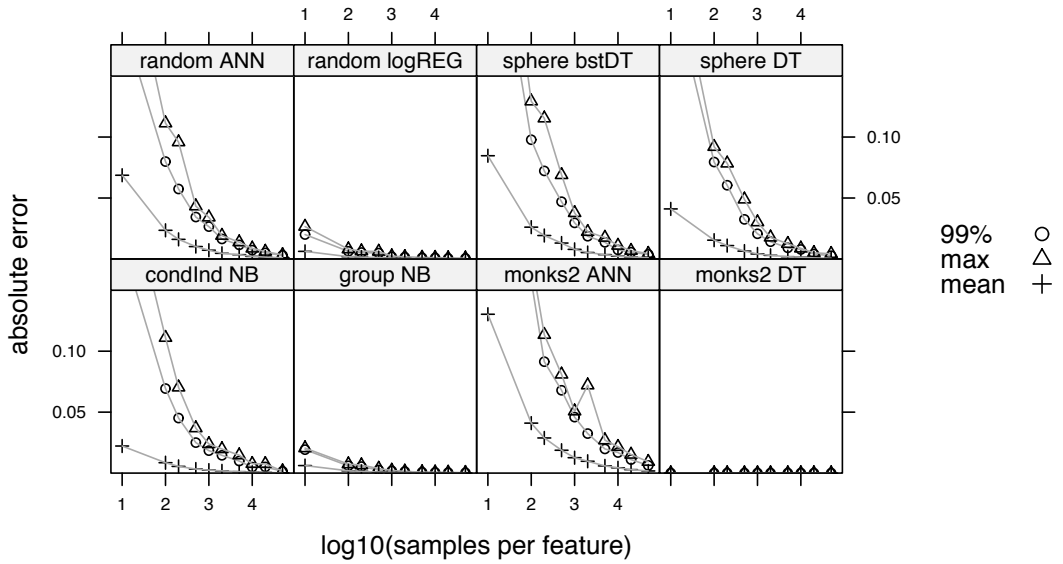


Figure 2: Mean, 99th-percentile, and maximum errors for several data set/classifier pairs and across different settings of how many samples are drawn per feature. Note that the error is the absolute difference between the approximated and actual contribution of a feature's value. The maximum error is the largest such observed difference across all instances.

Some of the results of this experiment are shown in Fig. 2. We can see that it only takes about a thousand of samples per feature to achieve a reasonably low approximation error. When over 10000 samples are drawn, all the contributions across all features and all test instances are very close to the actual contributions. From the discussion of the approximation error, we can see that the number of samples depends on the variance in the model's output, which in turn directly depends on how much the model has learned. Therefore, it takes only a few samples for a good approximation when explaining a model which has acquired little or no knowledge. This might be either due to the model not being able to learn the concepts behind the data set or because there are no concepts to learn. A few such examples are the Naive Bayes model on the Group data set (model's accuracy: 0.32, relative freq. of majority class: 0.33), the Decision Tree on Monks2 (acc. 0.65 , rel. freq. 0.67), and Logistic Regression on the Random data set (acc. 0.5 , rel. freq. 0.5). On the other hand, if a model successfully learns from the data set, it requires more samples to explain. For example, Naive Bayes on CondInd (acc. 0.92 , rel. freq. 0.50) and a Decision Tree on Sphere (acc. 0.80 , rel. freq. 0.50). In some cases a model acquires incorrect knowledge or over-fits the data set. One such example is the ANN model, which was allowed to over-fit the Random data set (acc. 0.5 , rel. freq. 0.5). In this case the method explains what the model has learned, regardless of whether the knowledge is correct or not. And although the explanations would not tell us much about the concepts behind the data set (we conclude from the model's performance, that it's knowledge is useless), they would reveal what the model has learned, which is the purpose of an explanation method.

In our second experiment we measure sample variances and classification running times. These will provide insight into how much time is needed for a good approximation. We use the same procedure as before, on all data sets, using only the approximation method. We draw 50000 samples per feature. Therefore, the total number of samples for each data set/classifier pair is: 50000n times

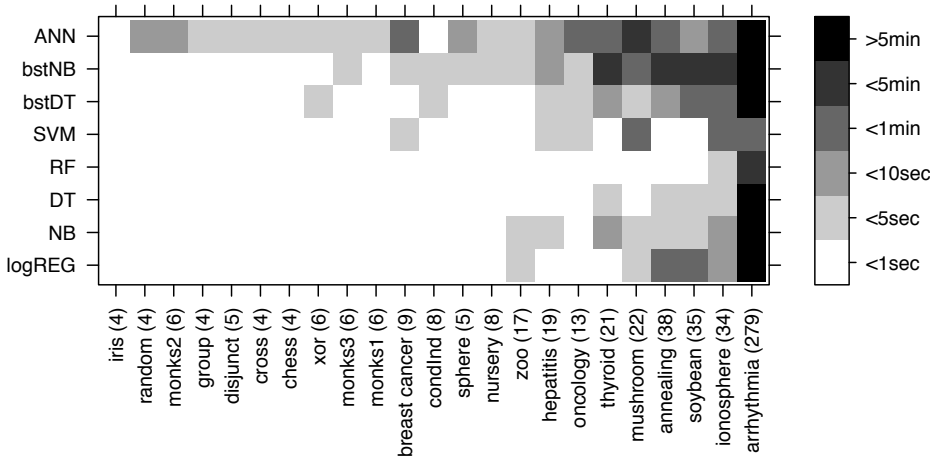


Figure 3: Visualization of explanation running times across all data set/classifier pairs.

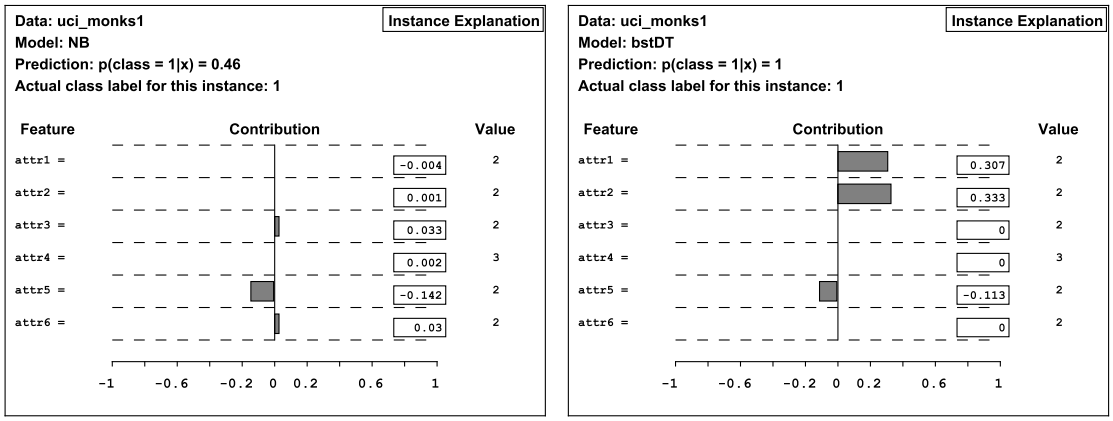
the number of test instances, which is sufficient for a good estimate of classification times and variances. The maximum σ_i^2 in Table 1 reveal that the crisp features of synthetic data sets have higher variance and are more difficult to explain than features from real-world data sets. Explaining the prediction of the ANN model for an instance Monks2 is the most complex explanation (that is, requires the most samples - see Fig. 2), which is understandable given the complexity of the Monks2 concept (class label = 1 iff exactly two feature values are 1). Note that most maximum values are achieved when explaining ANN.

From the data gathered in our second experiment, we generate Fig. 3, which shows the time needed to provide an explanation with the desired error $\langle 99\%, 0.01 \rangle$. For smaller data sets (smaller in the number of features) the explanation is generated almost instantly. For larger data sets, generating an explanation takes less than a minute, with the exception of bstNB on a few data sets and the ANN model on the Mushroom data set. These two models require more time for a single classification.

The Arrhythmia data set, with its 279 features, is an example of a data set, where the explanation can not be generated in some sensible time. For example, it takes more than an hour to generate an explanation for a prediction of the bstNB model. The explanation method is therefore less appropriate for explaining models which are built on several hundred features or more. Arguably, providing a comprehensible explanation involving a hundred or more features is a problem in its own right and even inherently transparend models become less comprehensible with such a large number of features. However, the focus of this paper is on providing an effective and general explanation method, which is computationally feasible on the majority of data sets we encounter. Also note that large data sets are often reduced to a smaller number of features in the preprocessing step of data acquisition before a learning algorithm is applied. Therefore, when considering the number of features the explanation method can still handle, we need not count irrelevant features, which are not included in the final model.

4.1 Example Explanation

Unlike running times and approximation errors, the usefulness and intuitiveness of the generated explanations is a more subjective matter. In this section we try to illustrate the usefulness of the



(a) *NB model* (b) *bstDT model*

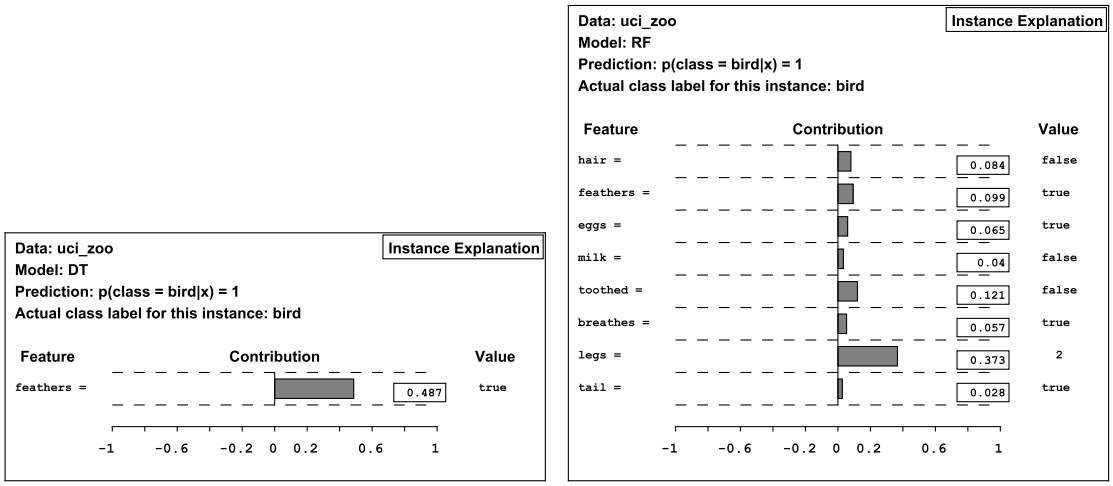
Figure 4: The boosting model correctly learns the concepts of the Monks1 data set, while Naive Bayes does not and misclassifies this instance.

method's explanations with several examples. When interpreting the explanations, we take into account both the magnitude and the sign of the contribution. If a feature-value has a larger contribution than another it has a larger influence on the model's prediction. If a feature-value's contribution has a positive sign, it contributes towards increasing the model's output (probability, score, rank, ...). A negative sign, on the other hand, means that the feature-value contributes towards decreasing the model's output. An additional advantage of the generated contributions is that they sum up to the difference between the model's output prediction and the model's expected output, given no information about the values of the features. Therefore, we can discern how much the model's output moves when given the feature values for the instance, which features are responsible for this change, and the magnitude of an individual feature-value's influence. These examples show how the explanations can be interpreted. They were generated for various classification models and data sets, to show the advantage of having a general explanation method.

The first pair of examples (see Fig. 4) are explanations for an instance from the first of the well-known Monks data sets. For this data set the class label is 1 iff attr1 and attr2 are equal or attr5 equals 1. The other 3 features are irrelevant. The NB model, due to its assumption of conditional independence, does not learn the importance of equality between the first two features and misclassifies the instance. However, both NB and bstDT learn the importance of the fifth feature and explanations reveal that value 2 for the fifth feature speaks against class 1.

The second pair of examples (see Fig. 5) is from the Zoo data set. Both models predict that the instance represents a bird. Why? The explanations reveal that DT predicts this animal is a bird, because it has feathers. The more complex RF model predicts its a bird, because it has two legs, but also because the animal is toothless, with feathers, without hair, etc... These first two pairs of examples illustrate how the explanations reflect what the model has learnt and how we can compare explanations from different classifiers.

In our experiments we are not interested in the prediction quality of the classifiers and do not put much effort into optimizing their performance. Some examples of underfitting and overfitting are actually desirable as they allow us to inspect if the explanation method reveals what the classifier has (or has not) learned. For example, Fig. 6(a) shows the explanation of the logREG model's prediction for an instance from the Xor data set. Logistic regression is unable to learn the exclusive-



(a) *DT model* (b) *RF model*

Figure 5: Explanations for an instance from the Zoo data set. The DT model uses a single feature, while several feature values influence the RT model. Feature values with contributions ≤ 0.01 have been removed for clarity.

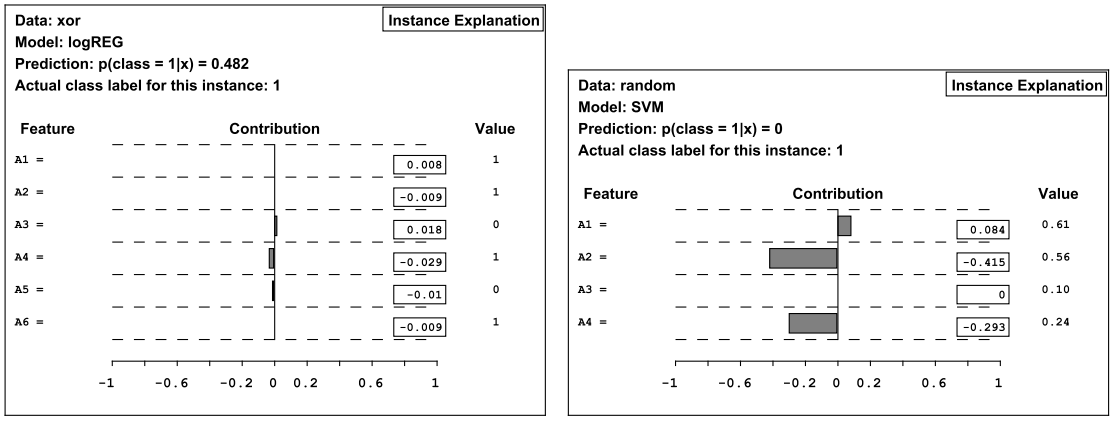
or concept of this data set (for this data set the class label is the odd parity bit for the first three feature values) and the explanation is appropriate. On the other hand, SVM manages to overfit the Random data set and finds concepts where there are none (see Fig. 6(b)).

Fig. 7(a) is an explanation for ANN's prediction for the introductory instance from the Titanic data set (see Fig. 1). Our explanation for the NB model's prediction (Fig. 7(b)) is very similar to the inherent explanation (taking into account that a logarithm is applied in the inherent explanation). The ANN model, on the other hand, predicts a lower chance of survival, because being a passenger from the 3rd class has a much higher negative contribution for ANN.

Our final example illustrates how the method can be used in real-world situations. Fig. 8 is an explanation for RT's prediction regarding whether breast cancer will (class = 1) or will not (class = 2) recur for this patient. According to RF it is more likely that cancer will not recur and the explanation indicates that this is mostly due to a low number of positive lymph nodes (nLymph). The lack of lymphovascular invasion (LVI) or tissue invasion (invasive) also contributes positively. A high ratio of removed lymph nodes was positive (posRatio) has the only significant negative contribution. Oncologists found this type of explanation very useful.

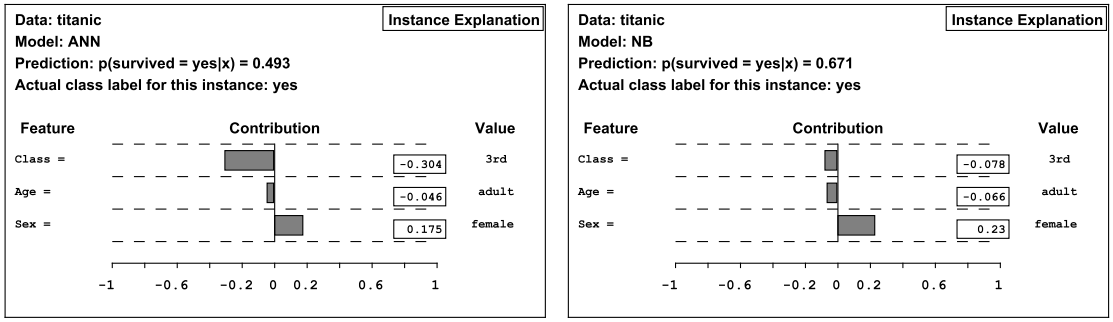
5. Conclusion

In the introductive section, we asked if an efficient and effective general explanation method for classifiers' predictions can be made. In conclusion, we can answer yes. Using only the input and output of a classifier we decompose the changes in its prediction into contributions of individual feature values. These contributions correspond to known concepts from coalitional game theory. Unlike with existing methods, the resulting theoretical properties of the proposed method guarantee that no matter which concepts the classifier learns, the generated contributions will reveal the influence of feature values. Therefore, the method can effectively be used on any classifier. As we show



(a) *logREG model* (b) *SVM model*

Figure 6: The left hand side explanation indicates that the feature values have no significant influence on the logREG model on the Xor data set. The right hand side explanation shows how SVM overfits the Random data set.



(a) *ANN model* (b) *NB model*

Figure 7: Two explanations for the Titanic instance from the introduction. The left hand side explanation is for the ANN model. The right hand side explanation is for the NB model.

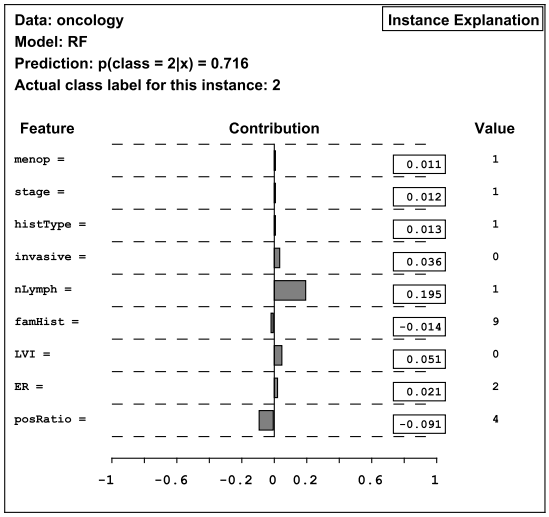


Figure 8: An explanation or the RF model's prediction for a patient from the Oncology data set.

on several examples, the method can be used to visually inspect models' predictions and compare the predictions of different models.

The proposed approximation method successfully deals with the initial exponential time complexity, makes the method efficient, and feasible for practical use. As part of further work we intend to research whether we can efficiently not only compute the contributions, which already reflect the interactions, but also highlight (at least) the most important individual interactions as well. A minor issue left to further work is extending the approximation with an algorithm for optimizing the number of samples we take. It would also be interesting to explore the possibility of applying the same principles to the explanation of regression models.

References

Robert Andrews, Joachim Diederich, and Alan B. Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8:373–389, 1995.

Artur Asuncion and David J. Newman. UCI Machine Learning Repository, `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2009.

Barry Becker, Ron Kohavi, and Dan Sommerfield. Visualizing the simple bayesian classier. In *KDD Workshop on Issues in the Integration of Data Mining and Data Visualization*, 1997.

Leo Breiman. Random forests. *Machine Learning Journal*, 45:5–32, 2001.

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers and Operations Research*, 2008. (in print, doi: 10.1016/j.cor.2008.04.004).

Shay Cohen, Gideon Dror, and Eytan Ruppin. Feature selection via coalitional game theory. *Neural Computation*, 19(7):1939–1961, 2007.

Lutz Hamel. Visualization of support vector machines with unsupervised learning. In *Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8. IEEE, 2006.

Aleks Jakulin, Martin Možina, Janez Demšar, Ivan Bratko, and Blaž Zupan. Nomograms for visualizing support vector machines. In *KDD '05: Proceeding of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 108–117, New York, NY, USA, 2005. ACM. ISBN 1-59593-135-X.

Alon Keinan, Ben Sandbank, Claus C. Hilgetag, Isaac Meilijson, and Eytan Ruppin. Fair attribution of functional contribution in artificial and biological networks. *Neural Computation*, 16(9):1887–1915, 2004.

Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence journal*, 97(1–2):273–324, 1997.

Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23:89–109, 2001.

Igor Kononenko and Matjaz Kukar. *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood publ., 2007.

Vincent Lemaire, Raphaël Fraud, and Nicolas Voisine. Contact personalization using a score understanding method. In *International Joint Conference on Neural Networks (IJCNN)*, 2008.

David Martens, Bart Baesens, Tony Van Gestel, and Jan Vanthienen. Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, 183(3):1466–1476, 2007.

Stefano Moretti and Fioravante Patrone. Transversality of the shapley value. *TOP*, 16(1):1–41, 2008.

Martin Možina, Janez Demšar, Michael Kattan, and Blaž Zupan. Nomograms for visualization of naive bayesian classifier. In *PKDD '04: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 337–348, New York, NY, USA, 2004. Springer-Verlag New York, Inc. ISBN 3-540-23108-0.

Richi Nayak. Generating rules with predicates, terms and variables from the pruned neural networks. *Neural Networks*, 22(4):405–414, 2009.

Francois Poulet. Svm and graphical algorithms: A cooperative approach. In *Proceedings of Fourth IEEE International Conference on Data Mining*, pages 499–502, 2004.

Marko Robnik-Šikonja and Igor Kononenko. Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20:589–600, 2008.

Lloyd S. Shapley. *A Value for n-person Games*, volume II of *Contributions to the Theory of Games*. Princeton University Press, 1953.

Duane Szafron, Brett Poulin, Roman Eisner, Paul Lu, Russ Greiner, David Wishart, Alona Fyshe, Brandon Pearcy, Cam Macdonell, and John Anvik. Visual explanation of evidence in additive classifiers. In *Proceedings of Innovative Applications of Artificial Intelligence*, 2006.

Geoffrey Towell and Jude W. Shavlik. Extracting refined rules from knowledge-based neural networks, machine learning. *Machine Learning*, 13:71–101, 1993.

Erik Štrumbelj, Igor Kononenko, and Marko Robnik Šikonja. Explaining instance classifications with interactions of subsets of feature values. *Data & Knowledge Engineering*, 68(10):886–904, 2009.

Journal of Machine Learning Research 11 (2010) 19-60 Submitted 7/09; Revised 11/09; Published 1/10

Online Learning for Matrix Factorization and Sparse Coding

Julien Mairal JULIEN.MAIRAL@INRIA.FR
Francis Bach FRANCIS.BACH@INRIA.FR
INRIA - WILLOW Project-Team
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548)
23, avenue d'Italie 75214 Paris CEDEX 13, France

Jean Ponce JEAN.PONCE@ENS.FR
Ecole Normale Supérieure - WILLOW Project-Team
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548)
45, rue d'Ulm 75230 Paris CEDEX 05, France

Guillermo Sapiro GUILLE@UMN.EDU
Department of Electrical and Computer Engineering
University of Minnesota
200 Union Street SE, Minneapolis, MN 55455, USA

Editor: Hui Zou

Abstract

Sparse coding—that is, modelling data vectors as sparse linear combinations of basis elements—is widely used in machine learning, neuroscience, signal processing, and statistics. This paper focuses on the large-scale matrix factorization problem that consists of *learning* the basis set in order to adapt it to specific data. Variations of this problem include dictionary learning in signal processing, non-negative matrix factorization and sparse principal component analysis. In this paper, we propose to address these tasks with a new online optimization algorithm, based on stochastic approximations, which scales up gracefully to large data sets with millions of training samples, and extends naturally to various matrix factorization formulations, making it suitable for a wide range of learning problems. A proof of convergence is presented, along with experiments with natural images and genomic data demonstrating that it leads to state-of-the-art performance in terms of speed and optimization for both small and large data sets.

Keywords: basis pursuit, dictionary learning, matrix factorization, online learning, sparse coding, sparse principal component analysis, stochastic approximations, stochastic optimization, non-negative matrix factorization

1. Introduction

The linear decomposition of a signal using a few atoms of a *learned* dictionary instead of a predefined one—based on wavelets (Mallat, 1999) for example—has recently led to state-of-the-art results in numerous low-level signal processing tasks such as image denoising (Elad and Aharon, 2006; Mairal et al., 2008b), texture synthesis (Peyré, 2009) and audio processing (Grosse et al., 2007; Févotte et al., 2009; Zibulevsky and Pearlmutter, 2001), as well as higher-level tasks such as image classification (Raina et al., 2007; Mairal et al., 2008a, 2009b; Bradley and Bagnell, 2009; Yang et al., 2009), showing that sparse learned models are well adapted to natural signals. Unlike decompositions based on principal component analysis and its variants, these models do not im-

pose that the basis vectors be orthogonal, allowing more flexibility to adapt the representation to the data.[1] In machine learning and statistics, slightly different matrix factorization problems are formulated in order to obtain a few *interpretable* basis elements from a set of data vectors. This includes non-negative matrix factorization and its variants (Lee and Seung, 2001; Hoyer, 2002, 2004; Lin, 2007), and sparse principal component analysis (Zou et al., 2006; d'Aspremont et al., 2007, 2008; Witten et al., 2009; Zass and Shashua, 2007). As shown in this paper, these problems have strong similarities; even though we first focus on the problem of dictionary learning, the algorithm we propose is able to address all of them. While learning the dictionary has proven to be critical to achieve (or improve upon) state-of-the-art results in signal and image processing, effectively solving the corresponding optimization problem is a significant computational challenge, particularly in the context of large-scale data sets that may include millions of training samples. Addressing this challenge and designing a generic algorithm which is capable of efficiently handling various matrix factorization problems, is the topic of this paper.

Concretely, consider a signal $\mathbf{x}$ in $\mathbb{R}^m$. We say that it admits a sparse approximation over a *dictionary* $\mathbf{D}$ in $\mathbb{R}^{m\times k}$, with k columns referred to as *atoms*, when one can find a linear combination of a "few" atoms from $\mathbf{D}$ that is "close" to the signal $\mathbf{x}$. Experiments have shown that modelling a signal with such a sparse decomposition (*sparse coding*) is very effective in many signal processing applications (Chen et al., 1999). For natural images, predefined dictionaries based on various types of wavelets (Mallat, 1999) have also been used for this task. However, learning the dictionary instead of using off-the-shelf bases has been shown to dramatically improve signal reconstruction (Elad and Aharon, 2006). Although some of the learned dictionary elements may sometimes "look like" wavelets (or Gabor filters), they are tuned to the input images or signals, leading to much better results in practice.

Most recent algorithms for dictionary learning (Olshausen and Field, 1997; Engan et al., 1999; Lewicki and Sejnowski, 2000; Aharon et al., 2006; Lee et al., 2007) are iterative *batch* procedures, accessing the whole training set at each iteration in order to minimize a cost function under some constraints, and cannot efficiently deal with very large training sets (Bottou and Bousquet, 2008), or dynamic training data changing over time, such as video sequences. To address these issues, we propose an *online* approach that processes the signals, one at a time, or in mini-batches. This is particularly important in the context of image and video processing (Protter and Elad, 2009; Mairal et al., 2008c), where it is common to learn dictionaries adapted to small patches, with training data that may include several millions of these patches (roughly one per pixel and per frame). In this setting, online techniques based on stochastic approximations are an attractive alternative to batch methods (see, e.g., Bottou, 1998; Kushner and Yin, 2003; Shalev-Shwartz et al., 2009). For example, first-order stochastic gradient descent with projections on the constraint set (Kushner and Yin, 2003) is sometimes used for dictionary learning (see Aharon and Elad, 2008; Kavukcuoglu et al., 2008 for instance). We show in this paper that it is possible to go further and exploit the specific structure of sparse coding in the design of an optimization procedure tuned to this problem, with low memory consumption and lower computational cost than classical batch algorithms. As demonstrated by our experiments, it scales up gracefully to large data sets with millions of training samples, is easy to use, and is faster than competitive methods.

The paper is structured as follows: Section 2 presents the dictionary learning problem. The proposed method is introduced in Section 3, with a proof of convergence in Section 4. Section 5

1. Note that the terminology "basis" is slightly abusive here since the elements of the dictionary are not necessarily linearly independent and the set can be overcomplete—that is, have more elements than the signal dimension.

extends our algorithm to various matrix factorization problems that generalize dictionary learning, and Section 6 is devoted to experimental results, demonstrating that our algorithm is suited to a wide class of learning problems.

1.1 Contributions

This paper makes four main contributions:

- We cast in Section 2 the dictionary learning problem as the optimization of a smooth non-convex objective function over a convex set, minimizing the (desired) *expected* cost when the training set size goes to infinity, and propose in Section 3 an iterative online algorithm that solves this problem by efficiently minimizing at each step a quadratic surrogate function of the empirical cost over the set of constraints. This method is shown in Section 4 to converge almost surely to a stationary point of the objective function.

- As shown experimentally in Section 6, our algorithm is significantly faster than previous approaches to dictionary learning on both small and large data sets of natural images. To demonstrate that it is adapted to difficult, large-scale image-processing tasks, we learn a dictionary on a 12-Megapixel photograph and use it for inpainting—that is, filling some holes in the image.

- We show in Sections 5 and 6 that our approach is suitable to large-scale matrix factorization problems such as non-negative matrix factorization and sparse principal component analysis, while being still effective on small data sets.

- To extend our algorithm to several matrix factorization problems, we propose in Appendix B efficient procedures for projecting onto two convex sets, which can be useful for other applications that are beyond the scope of this paper.

1.2 Notation

We define for $p \geq 1$ the ℓ_p norm of a vector $\mathbf{x}$ in $\mathbb{R}^m$ as $||\mathbf{x}||_p \triangleq (\sum_{i=1}^m |\mathbf{x}[i]|^p)^{1/p}$, where $\mathbf{x}[i]$ denotes the i-th coordinate of $\mathbf{x}$ and $||\mathbf{x}||_\infty \triangleq \max_{i=1,\ldots,m} |\mathbf{x}[i]| = \lim_{p\to\infty} ||\mathbf{x}||_p$. We also define the ℓ_0 pseudo-norm as the sparsity measure which counts the number of nonzero elements in a vector:[2] $||\mathbf{x}||_0 \triangleq \#\{i \text{ s.t. } \mathbf{x}[i] \neq 0\} = \lim_{p\to 0^+} (\sum_{i=1}^m |\mathbf{x}[i]|^p)$. We denote the Frobenius norm of a matrix $\mathbf{X}$ in $\mathbb{R}^{m\times n}$ by $||\mathbf{X}||_F \triangleq (\sum_{i=1}^m \sum_{j=1}^n \mathbf{X}[i,j]^2)^{1/2}$. For a sequence of vectors (or matrices) $\mathbf{x}_t$ and scalars u_t, we write $\mathbf{x}_t = O(u_t)$ when there exists a constant $K > 0$ so that for all t, $||\mathbf{x}_t||_2 \leq K u_t$. Note that for finite-dimensional vector spaces, the choice of norm is essentially irrelevant (all norms are equivalent). Given two matrices $\mathbf{A}$ in $\mathbb{R}^{m_1\times n_1}$ and $\mathbf{B}$ in $\mathbb{R}^{m_2\times n_2}$, $\mathbf{A}\otimes\mathbf{B}$ denotes the Kronecker product between $\mathbf{A}$ and $\mathbf{B}$, defined as the matrix in $\mathbb{R}^{m_1 m_2\times n_1 n_2}$, defined by blocks of sizes $m_2 \times n_2$ equal to $\mathbf{A}[i,j]\mathbf{B}$. For more details and properties of the Kronecker product, see Golub and Van Loan (1996), and Magnus and Neudecker (1999).

2. Note that it would be more proper to write $||\mathbf{x}||_0^0$ instead of $||\mathbf{x}||_0$ to be consistent with the traditional notation $||\mathbf{x}||_p$. However, for the sake of simplicity, we will keep this notation unchanged.

2. Problem Statement

Classical dictionary learning techniques for sparse representation (Olshausen and Field, 1997; Engan et al., 1999; Lewicki and Sejnowski, 2000; Aharon et al., 2006; Lee et al., 2007) consider a finite training set of signals $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ in $\mathbb{R}^{m \times n}$ and optimize the empirical cost function

$$f_n(\mathbf{D}) \triangleq \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{x}_i, \mathbf{D}), \tag{1}$$

where $\mathbf{D}$ in $\mathbb{R}^{m \times k}$ is the dictionary, each column representing a basis vector, and ℓ is a loss function such that $\ell(\mathbf{x}, \mathbf{D})$ should be small if $\mathbf{D}$ is "good" at representing the signal $\mathbf{x}$ in a sparse fashion. The number of samples n is usually large, whereas the signal dimension m is relatively small, for example, $m = 100$ for 10×10 image patches, and $n \geq 100{,}000$ for typical image processing applications. In general, we also have $k \ll n$ (e.g., $k = 200$ for $n = 100{,}000$), but each signal only uses a few elements of $\mathbf{D}$ in its representation, say 10 for instance. Note that, in this setting, overcomplete dictionaries with $k > m$ are allowed. As others (see for example Lee et al., 2007), we define $\ell(\mathbf{x}, \mathbf{D})$ as the optimal value of the ℓ_1 *sparse coding* problem:

$$\ell(\mathbf{x}, \mathbf{D}) \triangleq \min_{\alpha \in \mathbb{R}^k} \frac{1}{2} ||\mathbf{x} - \mathbf{D}\alpha||_2^2 + \lambda ||\alpha||_1, \tag{2}$$

where λ is a regularization parameter. This problem is also known as *basis pursuit* (Chen et al., 1999), or the *Lasso* (Tibshirani, 1996).[3] It is well known that ℓ_1 regularization yields a sparse solution for α, but there is no direct analytic link between the value of λ and the corresponding effective sparsity $||\alpha||_0$. To prevent $\mathbf{D}$ from having arbitrarily large values (which would lead to arbitrarily small values of α), it is common to constrain its columns $\mathbf{d}_1, \ldots, \mathbf{d}_k$ to have an ℓ_2-norm less than or equal to one. We will call $\mathcal{C}$ the convex set of matrices verifying this constraint:

$$\mathcal{C} \triangleq \{\mathbf{D} \in \mathbb{R}^{m \times k} \ \text{ s.t. } \ \forall j = 1, \ldots, k, \ \ \mathbf{d}_j^T \mathbf{d}_j \leq 1\}.$$

Note that the problem of minimizing the empirical cost $f_n(\mathbf{D})$ is not convex with respect to $\mathbf{D}$. It can be rewritten as a joint optimization problem with respect to the dictionary $\mathbf{D}$ and the coefficients $\alpha = [\alpha_1, \ldots, \alpha_n]$ in $\mathbb{R}^{k \times n}$ of the sparse decompositions, which is not jointly convex, but convex with respect to each of the two variables $\mathbf{D}$ and α when the other one is fixed:

$$\min_{\mathbf{D} \in \mathcal{C}, \alpha \in \mathbb{R}^{k \times n}} \sum_{i=1}^{n} \left(\frac{1}{2} ||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda ||\alpha_i||_1 \right). \tag{4}$$

This can be rewritten as a *matrix factorization* problem with a sparsity penalty:

$$\min_{\mathbf{D} \in \mathcal{C}, \alpha \in \mathbb{R}^{k \times n}} \frac{1}{2} ||\mathbf{X} - \mathbf{D}\alpha||_F^2 + \lambda ||\alpha||_{1,1},$$

3. To be more precise, the original formulation of the Lasso is a constrained version of Eq. (2), with a constraint on the ℓ_1-norm of α:

$$\min_{\alpha \in \mathbb{R}^k} \frac{1}{2} ||\mathbf{x} - \mathbf{D}\alpha||_2^2 \ \text{ s.t. } \ ||\alpha||_1 \leq T. \tag{3}$$

Both formulations are equivalent in the sense that for every $\lambda > 0$ (respectively every $T > 0$), there exists a scalar T (respectively λ) so that Equations (2) and (3) admit the same solutions.

where, as before, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ is the matrix of data vectors, and $||\alpha||_{1,1}$ denotes the ℓ_1 norm of the matrix α—that is, the sum of the magnitude of its coefficients. A natural approach to solving this problem is to alternate between the two variables, minimizing over one while keeping the other one fixed, as proposed by Lee et al. (2007) (see also Engan et al. 1999 and Aharon et al. 2006, who use ℓ_0 rather than ℓ_1 penalties, or Zou et al. 2006 for the problem of sparse principal component analysis).[4] Since the computation of the coefficients vectors α_i dominates the cost of each iteration in this block-coordinate descent approach, a second-order optimization technique can be used to accurately estimate $\mathbf{D}$ at each step when α is fixed.

As pointed out by Bottou and Bousquet (2008), however, one is usually not interested in the minimization of the *empirical cost* $f_n(\mathbf{D})$ with high precision, but instead in the minimization of the *expected cost*

$$f(\mathbf{D}) \triangleq \mathbb{E}_{\mathbf{x}}[\ell(\mathbf{x}, \mathbf{D})] = \lim_{n \to \infty} f_n(\mathbf{D}) \text{ a.s.},$$

where the expectation (which is supposed finite) is taken relative to the (unknown) probability distribution $p(\mathbf{x})$ of the data.[5] In particular, given a finite training set, one should not spend too much effort on accurately minimizing the empirical cost, since it is only an approximation of the expected cost. An "inaccurate" solution may indeed have the same or better expected cost than a "well-optimized" one. Bottou and Bousquet (2008) further show that stochastic gradient algorithms, whose rate of convergence is very poor in conventional optimization terms, may in fact in certain settings be shown both theoretically and empirically to be faster in reaching a solution with low expected cost than second-order batch methods. With large training sets, the risk of overfitting is lower, but classical optimization techniques may become impractical in terms of speed or memory requirements.

In the case of dictionary learning, the classical projected first-order projected stochastic gradient descent algorithm (as used by Aharon and Elad 2008; Kavukcuoglu et al. 2008 for instance) consists of a sequence of updates of $\mathbf{D}$:

$$\mathbf{D}_t = \Pi_{\mathcal{C}}\Big[\mathbf{D}_{t-1} - \delta_t \nabla_{\mathbf{D}} \ell(\mathbf{x}_t, \mathbf{D}_{t-1})\Big],$$

where $\mathbf{D}_t$ is the estimate of the optimal dictionary at iteration t, δ_t is the gradient step, $\Pi_{\mathcal{C}}$ is the orthogonal projector onto $\mathcal{C}$, and the vectors $\mathbf{x}_t$ are i.i.d. samples of the (unknown) distribution $p(\mathbf{x})$. Even though it is often difficult to obtain such i.i.d. samples, the vectors $\mathbf{x}_t$ are in practice obtained by cycling on a randomly permuted training set. As shown in Section 6, we have observed that this method can be competitive in terms of speed compared to batch methods when the training set is large and when δ_t is carefully chosen. In particular, good results are obtained using a learning rate of the form $\delta_t \triangleq a/(t+b)$, where a and b have to be well chosen in a data set-dependent way. Note that first-order stochastic gradient descent has also been used for other matrix factorization problems (see Koren et al., 2009 and references therein).

The optimization method we present in the next section falls into the class of online algorithms based on stochastic approximations, processing one sample at a time (or a mini-batch), but further exploits the specific structure of the problem to efficiently solve it by sequentially minimizing a quadratic local surrogate of the expected cost. As shown in Section 3.5, it uses second-order information of the cost function, allowing the optimization without any explicit learning rate tuning.

4. In our setting, as in Lee et al. (2007), we have preferred to use the convex ℓ_1 norm, that has empirically proven to be better behaved in general than the ℓ_0 pseudo-norm for dictionary learning.

5. We use "a.s." to denote almost sure convergence.

3. Online Dictionary Learning

We present in this section the basic components of our online algorithm for dictionary learning (Sections 3.1–3.3), as well as a few minor variants which speed up our implementation in practice (Section 3.4) and an interpretation in terms of a Kalman algorithm (Section 3.5).

3.1 Algorithm Outline

Our procedure is summarized in Algorithm 1. Assuming that the training set is composed of i.i.d. samples of a distribution $p(\mathbf{x})$, its inner loop draws one element $\mathbf{x}_t$ at a time, as in stochastic gradient descent, and alternates classical sparse coding steps for computing the decomposition α_t of $\mathbf{x}_t$ over the dictionary $\mathbf{D}_{t-1}$ obtained at the previous iteration, with dictionary update steps where the new dictionary $\mathbf{D}_t$ is computed by minimizing over $\mathcal{C}$ the function

$$\hat{f}_t(\mathbf{D}) \triangleq \frac{1}{t}\sum_{i=1}^{t}\Big(\frac{1}{2}||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda||\alpha_i||_1\Big), \tag{5}$$

and the vectors α_i for $i < t$ have been computed during the previous steps of the algorithm. The motivation behind this approach is twofold:

- The function $\hat{f}_t$, which is quadratic in $\mathbf{D}$, aggregates the past information with a few sufficient statistics obtained during the previous steps of the algorithm, namely the vectors α_i, and it is easy to show that it upperbounds the empirical cost $f_t(\mathbf{D}_t)$ from Eq. (1). One key aspect of our convergence analysis will be to show that $\hat{f}_t(\mathbf{D}_t)$ and $f_t(\mathbf{D}_t)$ converge almost surely to the same limit, and thus that $\hat{f}_t$ acts as a *surrogate* for f_t.
- Since $\hat{f}_t$ is close to $\hat{f}_{t-1}$ for large values of t, so are $\mathbf{D}_t$ and $\mathbf{D}_{t-1}$, under suitable assumptions, which makes it efficient to use $\mathbf{D}_{t-1}$ as warm restart for computing $\mathbf{D}_t$.

3.2 Sparse Coding

The sparse coding problem of Eq. (2) with fixed dictionary is an ℓ_1-regularized linear least-squares problem. A number of recent methods for solving this type of problems are based on coordinate descent with soft thresholding (Fu, 1998; Friedman et al., 2007; Wu and Lange, 2008). When the columns of the dictionary have low correlation, we have observed that these simple methods are very efficient. However, the columns of learned dictionaries are in general highly correlated, and we have empirically observed that these algorithms become much slower in this setting. This has led us to use instead the LARS-Lasso algorithm, a homotopy method (Osborne et al., 2000; Efron et al., 2004) that provides the whole regularization path—that is, the solutions for all possible values of λ. With an efficient Cholesky-based implementation (see Efron et al., 2004; Zou and Hastie, 2005) for brief descriptions of such implementations), it has proven experimentally at least as fast as approaches based on soft thresholding, while providing the solution with a higher accuracy and being more robust as well since it does not require an arbitrary stopping criterion.

3.3 Dictionary Update

Our algorithm for updating the dictionary uses block-coordinate descent with warm restarts (see Bertsekas, 1999). One of its main advantages is that it is parameter free and does not require any

Algorithm 1 Online dictionary learning.

Require: $\mathbf{x} \in \mathbb{R}^m \sim p(\mathbf{x})$ (random variable and an algorithm to draw i.i.d samples of p), $\lambda \in \mathbb{R}$ (regularization parameter), $\mathbf{D}_0 \in \mathbb{R}^{m \times k}$ (initial dictionary), T (number of iterations).

1: $\mathbf{A}_0 \in \mathbb{R}^{k \times k} \leftarrow 0, \mathbf{B}_0 \in \mathbb{R}^{m \times k} \leftarrow 0$ (reset the "past" information).
2: **for** $t = 1$ to T **do**
3: Draw $\mathbf{x}_t$ from $p(\mathbf{x})$.
4: Sparse coding: compute using LARS

$$\alpha_t \triangleq \arg\min_{\alpha \in \mathbb{R}^k} \frac{1}{2}||\mathbf{x}_t - \mathbf{D}_{t-1}\alpha||_2^2 + \lambda||\alpha||_1.$$

5: $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \alpha_t \alpha_t^T$.
6: $\mathbf{B}_t \leftarrow \mathbf{B}_{t-1} + \mathbf{x}_t \alpha_t^T$.
7: Compute $\mathbf{D}_t$ using Algorithm 2, with $\mathbf{D}_{t-1}$ as warm restart, so that

$$\begin{aligned} \mathbf{D}_t &\triangleq \arg\min_{\mathbf{D} \in \mathcal{C}} \frac{1}{t} \sum_{i=1}^{t} \left(\frac{1}{2}||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda||\alpha_i||_1 \right), \\ &= \arg\min_{\mathbf{D} \in \mathcal{C}} \frac{1}{t} \left(\frac{1}{2} \mathrm{Tr}(\mathbf{D}^T \mathbf{D} \mathbf{A}_t) - \mathrm{Tr}(\mathbf{D}^T \mathbf{B}_t) \right). \end{aligned} \tag{6}$$

8: **end for**
9: **Return** $\mathbf{D}_T$ (learned dictionary).

Algorithm 2 Dictionary Update.

Require: $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_k] \in \mathbb{R}^{m \times k}$ (input dictionary),
$\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_k] \in \mathbb{R}^{k \times k}$
$\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k] \in \mathbb{R}^{m \times k}$

1: **repeat**
2: **for** $j = 1$ to k **do**
3: Update the j-th column to optimize for (6):

$$\begin{aligned} \mathbf{u}_j &\leftarrow \frac{1}{\mathbf{A}[j,j]}(\mathbf{b}_j - \mathbf{D}\mathbf{a}_j) + \mathbf{d}_j, \\ \mathbf{d}_j &\leftarrow \frac{1}{\max(||\mathbf{u}_j||_2, 1)} \mathbf{u}_j. \end{aligned} \tag{7}$$

4: **end for**
5: **until convergence**
6: **Return D (updated dictionary).**

learning rate tuning. Moreover, the procedure does not require to store all the vectors $\mathbf{x}_i$ and α_i, but only the matrices $\mathbf{A}_t = \sum_{i=1}^t \alpha_i \alpha_i^T$ in $\mathbb{R}^{k \times k}$ and $\mathbf{B}_t = \sum_{i=1}^t \mathbf{x}_i \alpha_i^T$ in $\mathbb{R}^{m \times k}$. Concretely, Algorithm 2 sequentially updates each column of $\mathbf{D}$. A simple calculation shows that solving (6) with respect to the j-th column $\mathbf{d}_j$, while keeping the other ones fixed under the constraint $\mathbf{d}_j^T \mathbf{d}_j \leq 1$, amounts to an orthogonal projection of the vector $\mathbf{u}_j$ defined in Eq. (7), onto the constraint set, namely

the ℓ_2-ball here, which is solved by Eq. (7). Since the convex optimization problem (6) admits separable constraints in the updated blocks (columns), convergence to a global optimum is guaranteed (Bertsekas, 1999). In practice, the vectors α_i are sparse and the coefficients of the matrix $\mathbf{A}_t$ are often concentrated on the diagonal, which makes the block-coordinate descent more efficient.[6] After a few iterations of our algorithm, using the value of $\mathbf{D}_{t-1}$ as a warm restart for computing $\mathbf{D}_t$ becomes effective, and a single iteration of Algorithm 2 has empirically found to be sufficient to achieve convergence of the dictionary update step. Other approaches have been proposed to update $\mathbf{D}$: For instance, Lee et al. (2007) suggest using a Newton method on the dual of Eq. (6), but this requires inverting a $k \times k$ matrix at each Newton iteration, which is impractical for an online algorithm.

3.4 Optimizing the Algorithm

We have presented so far the basic building blocks of our algorithm. This section discusses a few simple improvements that significantly enhance its performance.

3.4.1 HANDLING FIXED-SIZE DATA SETS

In practice, although it may be very large, the size of the training set often has a predefined finite size (of course this may not be the case when the data must be treated on the fly like a video stream for example). In this situation, the same data points may be examined several times, and it is very common in online algorithms to simulate an i.i.d. sampling of $p(\mathbf{x})$ by cycling over a randomly permuted training set (see Bottou and Bousquet, 2008 and references therein). This method works experimentally well in our setting but, when the training set is small enough, it is possible to further speed up convergence: In Algorithm 1, the matrices $\mathbf{A}_t$ and $\mathbf{B}_t$ carry all the information from the past coefficients $\alpha_1, \ldots, \alpha_t$. Suppose that at time t_0, a signal $\mathbf{x}$ is drawn and the vector α_{t_0} is computed. If the same signal $\mathbf{x}$ is drawn again at time $t > t_0$, then it is natural to replace the "old" information α_{t_0} by the new vector α_t in the matrices $\mathbf{A}_t$ and $\mathbf{B}_t$—that is, $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \alpha_t \alpha_t^T - \alpha_{t_0} \alpha_{t_0}^T$ and $\mathbf{B}_t \leftarrow \mathbf{B}_{t-1} + \mathbf{x}_t \alpha_t^T - \mathbf{x}_t \alpha_{t_0}^T$. In this setting, which requires storing all the past coefficients α_{t_0}, this method amounts to a block-coordinate descent for the problem of minimizing Eq. (4). When dealing with large but finite sized training sets, storing all coefficients α_i is impractical, but it is still possible to partially exploit the same idea, by removing the information from $\mathbf{A}_t$ and $\mathbf{B}_t$ that is older than two *epochs* (cycles through the data), through the use of two auxiliary matrices $\mathbf{A}'_t$ and $\mathbf{B}'_t$ of size $k \times k$ and $m \times k$ respectively. These two matrices should be built with the same rules as $\mathbf{A}_t$ and $\mathbf{B}_t$, except that at the end of an epoch, $\mathbf{A}_t$ and $\mathbf{B}_t$ are respectively replaced by $\mathbf{A}'_t$ and $\mathbf{B}'_t$, while $\mathbf{A}'_t$ and $\mathbf{B}'_t$ are set to 0. Thanks to this strategy, $\mathbf{A}_t$ and $\mathbf{B}_t$ do not carry any coefficients α_i older than two epochs.

3.4.2 SCALING THE "PAST" DATA

At each iteration, the "new" information α_t that is added to the matrices $\mathbf{A}_t$ and $\mathbf{B}_t$ has the same weight as the "old" one. A simple and natural modification to the algorithm is to rescale the "old" information so that newer coefficients α_t have more weight, which is classical in online learning. For instance, Neal and Hinton (1998) present an online algorithm for EM, where sufficient statistics are aggregated over time, and an exponential decay is used to forget out-of-date statistics. In this

6. We have observed that this is true when the columns of $\mathbf{D}$ are not too correlated. When a group of columns in $\mathbf{D}$ are highly correlated, the coefficients of the matrix $\mathbf{A}_t$ concentrate instead on the corresponding principal submatrices of $\mathbf{A}_t$.

paper, we propose to replace lines 5 and 6 of Algorithm 1 by

$$\mathbf{A}_t \leftarrow \beta_t \mathbf{A}_{t-1} + \alpha_t \alpha_t^T,$$
$$\mathbf{B}_t \leftarrow \beta_t \mathbf{B}_{t-1} + \mathbf{x}_t \alpha_t^T,$$

where $\beta_t \triangleq \left(1 - \frac{1}{t}\right)^\rho$, and ρ is a new parameter. In practice, one can apply this strategy after a few iterations, once $\mathbf{A}_t$ is well-conditioned. Tuning ρ improves the convergence rate, when the training sets are large, even though, as shown in Section 6, it is not critical. To understand better the effect of this modification, note that Eq. (6) becomes

$$\begin{aligned}\mathbf{D}_t &\triangleq \arg\min_{\mathbf{D}\in C} \frac{1}{\sum_{j=1}^t (j/t)^\rho} \sum_{i=1}^t \left(\frac{i}{t}\right)^\rho \left(\frac{1}{2}||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda||\alpha_i||_1\right),\\ &= \arg\min_{\mathbf{D}\in C} \frac{1}{\sum_{j=1}^t (j/t)^\rho} \left(\frac{1}{2}\operatorname{Tr}(\mathbf{D}^T\mathbf{D}\mathbf{A}_t) - \operatorname{Tr}(\mathbf{D}^T\mathbf{B}_t)\right).\end{aligned}$$

When $\rho = 0$, we obtain the original version of the algorithm. Of course, different strategies and heuristics could also be investigated. In practice, this parameter ρ is useful for large data sets only ($n \geq 100\,000$). For smaller data sets, we have not observed a better performance when using this extension.

3.4.3 MINI-BATCH EXTENSION

In practice, we can also improve the convergence speed of our algorithm by drawing $\eta > 1$ signals at each iteration instead of a single one, which is a classical heuristic in stochastic gradient descent algorithms. In our case, this is further motivated by the fact that the complexity of computing η vectors α_i is not linear in η. A Cholesky-based implementation of LARS-Lasso for decomposing a single signal has a complexity of $O(kms + ks^2)$, where s is the number of nonzero coefficients. When decomposing η signals, it is possible to pre-compute the Gram matrix $\mathbf{D}_t^T\mathbf{D}_t$ and the total complexity becomes $O(k^2m + \eta(km + ks^2))$, which is much cheaper than η times the previous complexity when η is large enough and s is small. Let us denote by $\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,\eta}$ the signals drawn at iteration t. We can now replace lines 5 and 6 of Algorithm 1 by

$$\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \frac{1}{\eta}\sum_{i=1}^{\eta} \alpha_{t,i}\alpha_{t,i}^T,$$
$$\mathbf{B}_t \leftarrow \mathbf{B}_{t-1} + \frac{1}{\eta}\sum_{i=1}^{\eta} \mathbf{x}_{t,i}\alpha_{t,i}^T.$$

3.4.4 SLOWING DOWN THE FIRST ITERATIONS

As in the case of stochastic gradient descent, the first iterations of our algorithm may update the parameters with large steps, immediately leading to large deviations from the initial dictionary. To prevent this phenomenon, classical implementations of stochastic gradient descent use gradient steps of the form $a/(t+b)$, where b "reduces" the step size. An initialization of the form $\mathbf{A}_0 = t_0\mathbf{I}$ and $\mathbf{B}_0 = t_0\mathbf{D}_0$ with $t_0 \geq 0$ also slows down the first steps of our algorithm by forcing the solution of the dictionary update to stay close to $\mathbf{D}_0$. As shown in Section 6, we have observed that our method does not require this extension to achieve good results in general.

3.4.5 PURGING THE DICTIONARY FROM UNUSED ATOMS

Every dictionary learning technique sometimes encounters situations where some of the dictionary atoms are never (or very seldom) used, which typically happens with a very bad initialization. A common practice is to replace these during the optimization by randomly chosen elements of the training set, which solves in practice the problem in most cases. For more difficult and highly regularized cases, it is also possible to choose a continuation strategy consisting of starting from an easier, less regularized problem, and gradually increasing λ. This continuation method has not been used in this paper.

3.5 Link with Second-order Stochastic Gradient Descent

For unconstrained learning problems with twice differentiable expected cost, the second-order stochastic gradient descent algorithm (see Bottou and Bousquet, 2008 and references therein) improves upon its first-order version, by replacing the learning rate by the inverse of the Hessian. When this matrix can be computed or approximated efficiently, this method usually yields a faster convergence speed and removes the problem of tuning the learning rate. However, it cannot be applied easily to constrained optimization problems and requires at every iteration an inverse of the Hessian. For these two reasons, it cannot be used for the dictionary learning problem, but nevertheless it shares some similarities with our algorithm, which we illustrate with the example of a different problem.

Suppose that two major modifications are brought to our original formulation: (i) the vectors α_t are independent of the dictionary $\mathbf{D}$—that is, they are drawn at the same time as $\mathbf{x}_t$; (ii) the optimization is unconstrained—that is, $\mathcal{C} = \mathbb{R}^{m\times k}$. This setting leads to the least-square estimation problem

$$\min_{\mathbf{D}\in\mathbb{R}^{m\times k}} \mathbb{E}_{(\mathbf{x},\alpha)}\left[||\mathbf{x}-\mathbf{D}\alpha||_2^2\right], \tag{8}$$

which is of course different from the original dictionary learning formulation. Nonetheless, it is possible to address Eq. (8) with our method and show that it amounts to using the recursive formula

$$\mathbf{D}_t \leftarrow \mathbf{D}_{t-1} + (\mathbf{x}_t - \mathbf{D}_{t-1}\alpha_t)\alpha_t^T\Big(\sum_{i=1}^{t}\alpha_i\alpha_i^T\Big)^{-1},$$

which is equivalent to a second-order stochastic gradient descent algorithm: The gradient obtained at $(\mathbf{x}_t,\alpha_t)$ is the term $-(\mathbf{x}_t-\mathbf{D}_{t-1}\alpha_t)\alpha_t^T$, and the sequence $(1/t)\sum_{i=1}^{t}\alpha_i\alpha_i^T$ converges to the Hessian of the objective function. Such sequence of updates admit a fast implementation called Kalman algorithm (see Kushner and Yin, 2003 and references therein).

4. Convergence Analysis

The main tools used in our proofs are the convergence of empirical processes (Van der Vaart, 1998) and, following Bottou (1998), the convergence of quasi-martingales (Fisk, 1965). Our analysis is limited to the basic version of the algorithm, although it can in principle be carried over to the optimized versions discussed in Section 3.4. Before proving our main result, let us first discuss the (reasonable) assumptions under which our analysis holds.

4.1 Assumptions

(A) The data admits a distribution with compact support K. Assuming a compact support for the data is natural in audio, image, and video processing applications, where it is imposed by the data acquisition process.

(B) The quadratic surrogate functions $\hat{f}_t$ are strictly convex with lower-bounded Hessians. We assume that the smallest eigenvalue of the positive semi-definite matrix $\frac{1}{t}\mathbf{A}_t$ defined in Algorithm 1 is greater than or equal to some constant κ_1. As a consequence, $\mathbf{A}_t$ is invertible and $\hat{f}_t$ is strictly convex with Hessian $\mathbf{I} \otimes \frac{2}{t}\mathbf{A}_t$. This hypothesis is in practice verified experimentally after a few iterations of the algorithm when the initial dictionary is reasonable, consisting for example of a few elements from the training set, or any common dictionary, such as DCT (bases of cosines products) or wavelets (Mallat, 1999). Note that it is easy to enforce this assumption by adding a term $\frac{\kappa_1}{2}||\mathbf{D}||_F^2$ to the objective function, which is equivalent to replacing the positive semi-definite matrix $\frac{1}{t}\mathbf{A}_t$ by $\frac{1}{t}\mathbf{A}_t + \kappa_1\mathbf{I}$. We have omitted for simplicity this penalization in our analysis.

(C) A particular sufficient condition for the uniqueness of the sparse coding solution is satisfied. Before presenting this assumption, let us briefly recall classical optimality conditions for the ℓ_1 decomposition problem in Eq. (2) (Fuchs, 2005). For $\mathbf{x}$ in K and $\mathbf{D}$ in $\mathcal{C}$, α in $\mathbb{R}^k$ is a solution of Eq. (2) if and only if

$$\begin{aligned} &\mathbf{d}_j^T(\mathbf{x} - \mathbf{D}\alpha) = \lambda \operatorname{sign}(\alpha[j]) \text{ if } \alpha[j] \neq 0, \\ &|\mathbf{d}_j^T(\mathbf{x} - \mathbf{D}\alpha)| \leq \lambda \text{ otherwise.} \end{aligned} \tag{9}$$

Let $\alpha^\star$ be such a solution. Denoting by Λ the set of indices j such that $|\mathbf{d}_j^T(\mathbf{x} - \mathbf{D}\alpha^\star)| = \lambda$, and $\mathbf{D}_\Lambda$ the matrix composed of the columns from $\mathbf{D}$ restricted to the set Λ, it is easy to see from Eq. (9) that the solution $\alpha^\star$ is necessary unique if $(\mathbf{D}_\Lambda^T\mathbf{D}_\Lambda)$ is invertible and that

$$\alpha_\Lambda^\star = (\mathbf{D}_\Lambda^T\mathbf{D}_\Lambda)^{-1}(\mathbf{D}_\Lambda^T\mathbf{x} - \lambda\varepsilon_\Lambda), \tag{10}$$

where $\alpha_\Lambda^\star$ is the vector containing the values of $\alpha^\star$ corresponding to the set Λ and $\varepsilon_\Lambda[j]$ is equal to the sign of $\alpha_\Lambda^\star[j]$ for all j. With this preliminary uniqueness condition in hand, we can now formulate our assumption: *We assume that there exists* $\kappa_2 > 0$ *such that, for all* $\mathbf{x}$ *in* K *and all dictionaries* $\mathbf{D}$ *in the subset of* $\mathcal{C}$ *considered by our algorithm, the smallest eigenvalue of* $\mathbf{D}_\Lambda^T\mathbf{D}_\Lambda$ *is greater than or equal to* κ_2. This guarantees the invertibility of $(\mathbf{D}_\Lambda^T\mathbf{D}_\Lambda)$ and therefore the uniqueness of the solution of Eq. (2). It is of course easy to build a dictionary $\mathbf{D}$ for which this assumption fails. However, having $\mathbf{D}_\Lambda^T\mathbf{D}_\Lambda$ invertible is a common assumption in linear regression and in methods such as the LARS algorithm aimed at solving Eq. (2) (Efron et al., 2004). It is also possible to enforce this condition using an elastic net penalization (Zou and Hastie, 2005), replacing $||\alpha||_1$ by $||\alpha||_1 + \frac{\kappa_2}{2}||\alpha||_2^2$ and thus improving the numerical stability of homotopy algorithms, which is the choice made by Zou et al. (2006). Again, we have omitted this penalization in our analysis.

4.2 Main Results

Given assumptions **(A)**–**(C)**, let us now show that our algorithm converges to a stationary point of the objective function. Since this paper is dealing with non-convex optimization, neither our algorithm nor any one in the literature is guaranteed to find the global optimum of the optimization problem. However, such stationary points have often been found to be empirically good enough

for practical applications, for example, for image restoration (Elad and Aharon, 2006; Mairal et al., 2008b).

Our first result (Proposition 2 below) states that given **(A)–(C)**, $f(\mathbf{D}_t)$ converges almost surely and $f(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t)$ converges almost surely to 0, meaning that $\hat{f}_t$ acts as a converging surrogate of f. First, we prove a lemma to show that $\mathbf{D}_t - \mathbf{D}_{t-1} = O(1/t)$. It does not ensure the convergence of $\mathbf{D}_t$, but guarantees the convergence of the positive sum $\sum_{t=1}^{\infty} ||\mathbf{D}_t - \mathbf{D}_{t-1}||_F^2$, a classical condition in gradient descent convergence proofs (Bertsekas, 1999).

Lemma 1 [Asymptotic variations of $\mathbf{D}_t$].
Assume **(A)–(C)**. *Then,*

$$\mathbf{D}_{t+1} - \mathbf{D}_t = O\Big(\frac{1}{t}\Big) \ \ a.s.$$

Proof This proof is inspired by Prop 4.32 of Bonnans and Shapiro (2000) on the Lipschitz regularity of solutions of optimization problems. Using assumption **(B)**, for all t, the surrogate $\hat{f}_t$ is strictly convex with a Hessian lower-bounded by κ_1. Then, a short calculation shows that it verifies the *second-order growth condition*

$$\hat{f}_t(\mathbf{D}_{t+1}) - \hat{f}_t(\mathbf{D}_t) \geq \kappa_1 ||\mathbf{D}_{t+1} - \mathbf{D}_t||_F^2. \tag{11}$$

Moreover,

$$\begin{aligned}\hat{f}_t(\mathbf{D}_{t+1}) - \hat{f}_t(\mathbf{D}_t) &= \hat{f}_t(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_{t+1}) + \hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_t) + \hat{f}_{t+1}(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t) \\ &\leq \hat{f}_t(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_{t+1}) + \hat{f}_{t+1}(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t),\end{aligned}$$

where we have used that $\hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_t) \leq 0$ because $\mathbf{D}_{t+1}$ minimizes $\hat{f}_{t+1}$ on $\mathcal{C}$. Since $\hat{f}_t(\mathbf{D}) = \frac{1}{t}(\frac{1}{2}\mathrm{Tr}(\mathbf{D}^T\mathbf{D}\mathbf{A}_t) - \mathrm{Tr}(\mathbf{D}^T\mathbf{B}_t))$, and $||\mathbf{D}||_F \leq \sqrt{k}$, it is possible to show that $\hat{f}_t - \hat{f}_{t+1}$ is Lipschitz with constant $c_t = (1/t)(||\mathbf{B}_{t+1} - \mathbf{B}_t||_F + \sqrt{k}||\mathbf{A}_{t+1} - \mathbf{A}_t||_F)$, which gives

$$\hat{f}_t(\mathbf{D}_{t+1}) - \hat{f}_t(\mathbf{D}_t) \leq c_t ||\mathbf{D}_{t+1} - \mathbf{D}_t||_F. \tag{12}$$

From Eq. (11) and (12), we obtain

$$||\mathbf{D}_{t+1} - \mathbf{D}_t||_F \leq \frac{c_t}{\kappa_1}.$$

Assumptions **(A)**, **(C)** and Eq. (10) ensure that the vectors α_i and $\mathbf{x}_i$ are bounded with probability one and therefore $c_t = O(1/t)$ a.s. ■

We can now state and prove our first proposition, which shows that we are indeed minimizing a smooth function.

Proposition 2 [Regularity of f].
Assume **(A)** *to* **(C)**. *For* $\mathbf{x}$ *in the support K of the probability distribution p, and* $\mathbf{D}$ *in the feasible set $\mathcal{C}$, let us define*

$$\alpha^\star(\mathbf{x}, \mathbf{D}) = \arg\min_{\alpha \in \mathbb{R}^k} \frac{1}{2}||\mathbf{x} - \mathbf{D}\alpha||_2^2 + \lambda||\alpha||_1. \tag{13}$$

Then,

1. the function ℓ defined in Eq. (2) is continuously differentiable and

$$\nabla_{\mathbf{D}}\ell(\mathbf{x},\mathbf{D}) = -(\mathbf{x}-\mathbf{D}\alpha^\star(\mathbf{x},\mathbf{D}))\alpha^\star(\mathbf{x},\mathbf{D})^T.$$

2. f is continuously differentiable and $\nabla f(\mathbf{D}) = \mathbb{E}_{\mathbf{x}}\left[\nabla_{\mathbf{D}}\ell(\mathbf{x},\mathbf{D})\right]$;

3. $\nabla f(\mathbf{D})$ is Lipschitz on $\mathcal{C}$.

Proof Assumption **(A)** ensures that the vectors $\alpha^\star$ are bounded for $\mathbf{x}$ in K and $\mathbf{D}$ in $\mathcal{C}$. Therefore, one can restrict the optimization problem (13) to a compact subset of $\mathbb{R}^k$. Under assumption **(C)**, the solution of Eq. (13) is unique and $\alpha^\star$ is well-defined. Theorem 5 in Appendix A from Bonnans and Shapiro (1998) can be applied and gives us directly the first statement. Since K is compact, and ℓ is continuously differentiable, the second statement follows immediately.

To prove the third claim, we will show that for all $\mathbf{x}$ in K, $\alpha^\star(\mathbf{x},.)$ is Lipschitz with a constant independent of $\mathbf{x}$,[7] which is a sufficient condition for ∇f to be Lipschitz. First, the function optimized in Eq. (13) is continuous in $\alpha, \mathbf{D}, \mathbf{x}$ and has a unique minimum, implying that $\alpha^\star$ is continuous in $\mathbf{x}$ and $\mathbf{D}$.

Consider a matrix $\mathbf{D}$ in $\mathcal{C}$ and $\mathbf{x}$ in K and denote by $\alpha^\star$ the vector $\alpha^\star(\mathbf{x},\mathbf{D})$, and again by Λ the set of indices j such that $|\mathbf{d}_j^T(\mathbf{x}-\mathbf{D}\alpha^\star)| = \lambda$. Since $\mathbf{d}_j^T(\mathbf{x}-\mathbf{D}\alpha^\star)$ is continuous in $\mathbf{D}$ and $\mathbf{x}$, there exists an open neighborhood V around $(\mathbf{x},\mathbf{D})$ such that for all $(\mathbf{x}',\mathbf{D}')$ in V, and $j \notin \Lambda$, $|\mathbf{d}_j^{T'}(\mathbf{x}'-\mathbf{D}'\alpha^{\star\prime})| < \lambda$ and $\alpha^{\star\prime}[j] = 0$, where $\alpha^{\star\prime} = \alpha^\star(\mathbf{x}',\mathbf{D}')$.

Denoting by $\mathbf{U}_\Lambda$ the matrix composed of the columns of a matrix $\mathbf{U}$ corresponding to the index set Λ and similarly by $\mathbf{u}_\Lambda$ the vector composed of the values of a vector $\mathbf{u}$ corresponding to Λ, we consider the function $\tilde{\ell}$

$$\tilde{\ell}(\mathbf{x},\mathbf{D}_\Lambda,\alpha_\Lambda) \triangleq \frac{1}{2}||\mathbf{x}-\mathbf{D}_\Lambda\alpha_\Lambda||_2^2 + \lambda||\alpha_\Lambda||_1,$$

Assumption **(C)** tells us that $\tilde{\ell}(\mathbf{x},\mathbf{D}_\Lambda,.)$ is strictly convex with a Hessian lower-bounded by κ_2. Let us consider $(\mathbf{x}',\mathbf{D}')$ in V. A simple calculation shows that

$$\tilde{\ell}(\mathbf{x},\mathbf{D}_\Lambda,\alpha_\Lambda^{\star\prime}) - \tilde{\ell}(\mathbf{x},\mathbf{D}_\Lambda,\alpha_\Lambda^\star) \geq \kappa_2||\alpha_\Lambda^{\star\prime} - \alpha_\Lambda^\star||_2^2.$$

Moreover, it is easy to show that $\tilde{\ell}(\mathbf{x},\mathbf{D}_\Lambda,.) - \tilde{\ell}(\mathbf{x}',\mathbf{D}'_\Lambda,.)$ is Lipschitz with constant $e_1||\mathbf{D}_\Lambda - \mathbf{D}'_\Lambda||_F + e_2||\mathbf{x}-\mathbf{x}'||_2$, where e_1, e_2 are constants independent of $\mathbf{D},\mathbf{D}',\mathbf{x},\mathbf{x}'$ and then, one can show that

$$||\alpha^{\star\prime} - \alpha^\star||_2 = ||\alpha_\Lambda^{\star\prime} - \alpha_\Lambda^\star||_2 \leq \frac{1}{\kappa_2}\left(e_1||\mathbf{D}-\mathbf{D}'||_F + e_2||\mathbf{x}-\mathbf{x}'||_2\right).$$

Therefore, $\alpha^\star$ is locally Lipschitz. Since $K \times \mathcal{C}$ is compact, $\alpha^\star$ is uniformly Lipschitz on $K \times \mathcal{C}$, which concludes the proof. ■

Now that we have shown that f is a smooth function, we can state our first result showing that the sequence of functions $\hat{f}_t$ acts asymptotically as a surrogate of f and that $f(\mathbf{D}_t)$ converges almost surely in the following proposition.

7. From now on, for a vector $\mathbf{x}$ in $\mathbb{R}^m$, $\alpha^\star(\mathbf{x},.)$ denotes the function that associates to a matrix $\mathbf{D}$ verifying Assumption **(C)**, the optimal solution $\alpha^\star(\mathbf{x},\mathbf{D})$. For simplicity, we will use these slightly abusive notation in the rest of the paper.

Proposition 3 [Convergence of $f(\mathbf{D}_t)$ and of the surrogate function]. *Let $\hat{f}_t$ denote the surrogate function defined in Eq. (5). Assume* **(A)** *to* **(C)**. *Then,*

1. *$\hat{f}_t(\mathbf{D}_t)$ converges almost surely;*
2. *$f(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t)$ converges almost surely to 0;*
3. *$f(\mathbf{D}_t)$ converges almost surely.*

Proof Part of this proof is inspired by Bottou (1998). We prove the convergence of the sequence $\hat{f}_t(\mathbf{D}_t)$ by showing that the stochastic positive process

$$u_t \triangleq \hat{f}_t(\mathbf{D}_t) \geq 0,$$

is a quasi-martingale and use Theorem 6 from Fisk (1965) (see Appendix A), which states that if the sum of the "positive" variations of u_t are bounded, u_t is a quasi-martingale, which converges with probability one (see Theorem 6 for details). Computing the variations of u_t, we obtain

$$\begin{aligned} u_{t+1} - u_t &= \hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_t(\mathbf{D}_t) \\ &= \hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_t) + \hat{f}_{t+1}(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t) \\ &= \hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_t) + \frac{\ell(\mathbf{x}_{t+1}, \mathbf{D}_t) - f_t(\mathbf{D}_t)}{t+1} + \frac{f_t(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t)}{t+1}, \end{aligned} \tag{14}$$

using the fact that $\hat{f}_{t+1}(\mathbf{D}_t) = \frac{1}{t+1}\ell(\mathbf{x}_{t+1}, \mathbf{D}_t) + \frac{t}{t+1}\hat{f}_t(\mathbf{D}_t)$. Since $\mathbf{D}_{t+1}$ minimizes $\hat{f}_{t+1}$ on $\mathcal{C}$ and $\mathbf{D}_t$ is in $\mathcal{C}$, $\hat{f}_{t+1}(\mathbf{D}_{t+1}) - \hat{f}_{t+1}(\mathbf{D}_t) \leq 0$. Since the surrogate $\hat{f}_t$ upperbounds the empirical cost f_t, we also have $f_t(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t) \leq 0$. To use Theorem 6, we consider the filtration of the past information $\mathcal{F}_t$ and take the expectation of Eq. (14) conditioned on $\mathcal{F}_t$, obtaining the following bound

$$\begin{aligned} \mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t] &\leq \frac{\mathbb{E}[\ell(\mathbf{x}_{t+1}, \mathbf{D}_t) | \mathcal{F}_t] - f_t(\mathbf{D}_t)}{t+1} \\ &\leq \frac{f(\mathbf{D}_t) - f_t(\mathbf{D}_t)}{t+1} \\ &\leq \frac{||f - f_t||_\infty}{t+1}, \end{aligned}$$

For a specific matrix $\mathbf{D}$, the central-limit theorem states that $\mathbb{E}[\sqrt{t}(f(\mathbf{D}_t) - f_t(\mathbf{D}_t))]$ is bounded. However, we need here a stronger result on empirical processes to show that $\mathbb{E}[\sqrt{t}||f - f_t||_\infty]$ is bounded. To do so, we use the Lemma 7 in Appendix A, which is a corollary of Donsker theorem (see Van der Vaart, 1998, chap. 19.2). It is easy to show that in our case, all the hypotheses are verified, namely, $\ell(\mathbf{x}, .)$ is uniformly Lipschitz and bounded since it is continuously differentiable on a compact set, the set $\mathcal{C} \subset \mathbb{R}^{m \times k}$ is bounded, and $\mathbb{E}_{\mathbf{x}}[\ell(\mathbf{x}, \mathbf{D})^2]$ exists and is uniformly bounded. Therefore, Lemma 7 applies and there exists a constant $\kappa > 0$ such that

$$\mathbb{E}[\mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t]^+] \leq \frac{\kappa}{t^{\frac{3}{2}}}.$$

Therefore, defining δ_t as in Theorem 6, we have

$$\sum_{t=1}^{\infty} \mathbb{E}[\delta_t (u_{t+1} - u_t)] = \sum_{t=1}^{\infty} \mathbb{E}[\mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t]^+] < +\infty.$$

Thus, we can apply Theorem 6, which proves that u_t converges almost surely and that

$$\sum_{t=1}^{\infty} |\mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t]| < +\infty \text{ a.s.}$$

Using Eq. (14) we can show that it implies the almost sure convergence of the positive sum

$$\sum_{t=1}^{\infty} \frac{\hat{f}_t(\mathbf{D}_t) - f_t(\mathbf{D}_t)}{t+1}.$$

Using Lemma 1 and the fact that the functions f_t and $\hat{f}_t$ are bounded and Lipschitz, with a constant independent of t, it is easy to show that the hypotheses of Lemma 8 in Appendix A are satisfied. Therefore

$$f_t(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t) \underset{t \to +\infty}{\longrightarrow} 0 \text{ a.s.}$$

Since $\hat{f}_t(\mathbf{D}_t)$ converges almost surely, this shows that $f_t(\mathbf{D}_t)$ converges in probability to the same limit. Note that we have in addition $||f_t - f||_\infty \to_{t \to +\infty} 0$ a.s. (see Van der Vaart, 1998, Theorem 19.4 (Glivenko-Cantelli)). Therefore,

$$f(\mathbf{D}_t) - \hat{f}_t(\mathbf{D}_t) \underset{t \to +\infty}{\longrightarrow} 0 \text{ a.s.}$$

and $f(\mathbf{D}_t)$ converges almost surely, which proves the second and third points. ■

With Proposition 3 in hand, we can now prove our final and strongest result, namely that first-order necessary optimality conditions are verified asymptotically with probability one.

Proposition 4 [Convergence to a stationary point]. *Under assumptions* **(A)** *to* **(C)**, *the distance between* $\mathbf{D}_t$ *and the set of stationary points of the dictionary learning problem converges almost surely to* 0 *when* t *tends to infinity.*

Proof Since the sequences of matrices $\mathbf{A}_t, \mathbf{B}_t$ are in a compact set, it is possible to extract converging subsequences. Let us assume for a moment that these sequences converge respectively to two matrices $\mathbf{A}_\infty$ and $\mathbf{B}_\infty$. In that case, $\mathbf{D}_t$ converges to a matrix $\mathbf{D}_\infty$ in $\mathcal{C}$. Let $\mathbf{U}$ be a matrix in $\mathbb{R}^{m \times k}$. Since $\hat{f}_t$ upperbounds f_t on $\mathbb{R}^{m \times k}$, for all t,

$$\hat{f}_t(\mathbf{D}_t + \mathbf{U}) \geq f_t(\mathbf{D}_t + \mathbf{U}).$$

Taking the limit when t tends to infinity,

$$\hat{f}_\infty(\mathbf{D}_\infty + \mathbf{U}) \geq f(\mathbf{D}_\infty + \mathbf{U}).$$

Let $h_t > 0$ be a sequence that converges to 0. Using a first order Taylor expansion, and using the fact that ∇f is Lipschitz and $\hat{f}_\infty(\mathbf{D}_\infty) = f(\mathbf{D}_\infty)$ a.s., we have

$$f(\mathbf{D}_\infty) + \text{Tr}(h_t \mathbf{U}^T \nabla \hat{f}_\infty(\mathbf{D}_\infty)) + o(h_t \mathbf{U}) \geq f(\mathbf{D}_\infty) + \text{Tr}(h_t \mathbf{U}^T \nabla f(\mathbf{D}_\infty)) + o(h_t \mathbf{U}),$$

and it follows that

$$\text{Tr}\Big(\frac{1}{||\mathbf{U}||_F} \mathbf{U}^T \nabla \hat{f}_\infty(\mathbf{D}_\infty)\Big) \geq \text{Tr}\Big(\frac{1}{||\mathbf{U}_t||_F} \mathbf{U}^T \nabla f(\mathbf{D}_\infty)\Big),$$

Since this inequality is true for all $\mathbf{U}, \nabla \hat{f}_\infty(\mathbf{D}_\infty) = \nabla f(\mathbf{D}_\infty)$. A first-order necessary optimality condition for $\mathbf{D}_\infty$ being an optimum of $\hat{f}_\infty$ is that $-\nabla \hat{f}_\infty$ is in the *normal cone* of the set $\mathcal{C}$ at $\mathbf{D}_\infty$ (Borwein and Lewis, 2006). Therefore, this first-order necessary conditions is verified for f at $\mathbf{D}_\infty$ as well. Since $\mathbf{A}_t, \mathbf{B}_t$ are asymptotically close to their accumulation points, $-\nabla f(\mathbf{D}_t)$ is asymptotically close the normal cone at $\mathbf{D}_t$ and these first-order optimality conditions are verified asymptotically with probability one. ■

5. Extensions to Matrix Factorization

In this section, we present variations of the basic online algorithm to address different optimization problems. We first present different possible regularization terms for α and $\mathbf{D}$, which can be used with our algorithm, and then detail some specific cases such as non-negative matrix factorization, sparse principal component analysis, constrained sparse coding, and simultaneous sparse coding.

5.1 Using Different Regularizers for α

In various applications, different priors for the coefficients α may lead to different regularizers $\psi(\alpha)$. As long as the assumptions of Section 4.1 are verified, our algorithm can be used with:

- Positivity constraints on α that are added to the ℓ_1-regularization. The homotopy method presented in Efron et al. (2004) is able to handle such constraints.
- The Tikhonov regularization, $\psi(\alpha) = \frac{\lambda_1}{2}||\alpha||_2^2$, which does not lead to sparse solutions.
- The elastic net (Zou and Hastie, 2005), $\psi(\alpha) = \lambda_1||\alpha||_1 + \frac{\lambda_2}{2}||\alpha||_2^2$, leading to a formulation relatively close to Zou et al. (2006).
- The group Lasso (Yuan and Lin, 2006; Turlach et al., 2005; Bach, 2008), $\psi(\alpha) = \sum_{i=1}^{s} ||\alpha_i||_2$, where α_i is a vector corresponding to a group of variables.

Non-convex regularizers such as the ℓ_0 pseudo-norm, ℓ_p pseudo-norm with $p < 1$ can be used as well. However, as with any classical dictionary learning techniques exploiting non-convex regularizers (e.g., Olshausen and Field, 1997; Engan et al., 1999; Aharon et al., 2006), there is no theoretical convergence results in these cases. Note also that convex smooth approximation of sparse regularizers (Bradley and Bagnell, 2009), or structured sparsity-inducing regularizers (Jenatton et al., 2009a; Jacob et al., 2009) could be used as well even though we have not tested them.

5.2 Using Different Constraint Sets for D

In the previous subsection, we have claimed that our algorithm could be used with different regularization terms on α. For the dictionary learning problem, we have considered an ℓ_2-regularization on $\mathbf{D}$ by forcing its columns to have less than unit ℓ_2-norm. We have shown that with this constraint set, the dictionary update step can be solved efficiently using a block-coordinate descent approach. Updating the j-th column of $\mathbf{D}$, when keeping the other ones fixed is solved by orthogonally projecting the vector $\mathbf{u}_j = \mathbf{d}_j + (1/\mathbf{A}[j,j])(\mathbf{b}_j - \mathbf{D}\mathbf{a}_j)$ on the constraint set $\mathcal{C}$, which in the classical dictionary learning case amounts to a projection of $\mathbf{u}_j$ on the ℓ_2-ball.

It is easy to show that this procedure can be extended to different convex constraint sets $\mathcal{C}'$ as long as the constraints are a union of independent constraints on each column of $\mathbf{D}$ and the orthogonal projections of the vectors $\mathbf{u}_j$ onto the set $\mathcal{C}'$ can be done efficiently. Examples of different sets $\mathcal{C}'$ that we propose as an alternative to $\mathcal{C}$ are

- The "non-negative" constraints:

$$\mathcal{C}' = \{\mathbf{D} \in \mathbb{R}^{m\times k} \text{ s.t. } \forall j = 1,\ldots,k, \ ||\mathbf{d}_j||_2 \leq 1 \text{ and } \mathbf{d}_j \geq 0\}.$$

- The "elastic-net" constraints:

$$\mathcal{C}' \triangleq \{\mathbf{D} \in \mathbb{R}^{m\times k} \text{ s.t. } \forall j = 1,\ldots,k, \ ||\mathbf{d}_j||_2^2 + \gamma||\mathbf{d}_j||_1 \leq 1\}.$$

  These constraints induce sparsity in the dictionary $\mathbf{D}$ (in addition to the sparsity-inducing regularizer on the vectors α_i). By analogy with the regularization proposed by Zou and Hastie (2005), we call these constraints "elastic-net constraints." Here, γ is a new parameter, controlling the sparsity of the dictionary $\mathbf{D}$. Adding a non-negativity constraint is also possible in this case. Note that the presence of the ℓ_2 regularization is important here. It has been shown by Bach et al. (2008) that using the ℓ_1-norm only in such problems lead to trivial solutions when k is large enough. The combination of ℓ_1 and ℓ_2 constraints has also been proposed recently for the problem of matrix factorization by Witten et al. (2009), but in a slightly different setting.

- The "fused lasso" (Tibshirani et al., 2005) constraints. When one is looking for a dictionary whose columns are sparse and piecewise-constant, a fused lasso regularization can be used. For a vector $\mathbf{u}$ in $\mathbb{R}^m$, we consider the ℓ_1-norm of the consecutive differences of $\mathbf{u}$ denoted by

$$\text{FL}(\mathbf{u}) \triangleq \sum_{i=2}^{m} |\mathbf{u}[i] - \mathbf{u}[i-1]|,$$

  and define the "fused lasso" constraint set

$$\mathcal{C}' \triangleq \{\mathbf{D} \in \mathbb{R}^{m\times k} \text{ s.t. } \forall j = 1,\ldots,k, \ ||\mathbf{d}_j||_2^2 + \gamma_1||\mathbf{d}_j||_1 + \gamma_2 \text{FL}(\mathbf{d}_j) \leq 1\}.$$

  This kind of regularization has proven to be useful for exploiting genomic data such as CGH arrays (Tibshirani and Wang, 2008).

In all these settings, replacing the projections of the vectors $\mathbf{u}_j$ onto the ℓ_2-ball by the projections onto the new constraints, our algorithm is still guaranteed to converge and find a stationary point of the optimization problem. The orthogonal projection onto the "non negative" ball is simple (additional thresholding) but the projection onto the two other sets is slightly more involved. In Appendix B, we propose two algorithms for efficiently solving these problems. The first one is presented in Section B.1 and computes the projection of a vector onto the elastic-net constraint in linear time, by extending the efficient projection onto the ℓ_1-ball from Maculan and de Paula (1989) and Duchi et al. (2008). The second one is a homotopy method, which solves the projection on the fused lasso constraint set in $O(ks)$, where s is the number of piecewise-constant parts in the solution. This method also solves efficiently the fused lasso signal approximation problem presented in Friedman et al. (2007):

$$\min_{\mathbf{u}\in\mathbb{R}^n} \frac{1}{2}||\mathbf{b} - \mathbf{u}||_2^2 + \gamma_1||\mathbf{u}||_1 + \gamma_2 \text{FL}(\mathbf{u}) + \gamma_3||\mathbf{u}||_2^2.$$

Being able to solve this problem efficiently has also numerous applications, which are beyond the scope of this paper. For instance, it allows us to use the fast algorithm of Nesterov (2007) for solving the more general fused lasso problem (Tibshirani et al., 2005). Note that the proposed method could be used as well with more complex constraints for the columns of $\mathbf{D}$, which we have not tested in this paper, addressing for instance the problem of structured sparse PCA (Jenatton et al., 2009b).

Now that we have presented a few possible regularizers for α and $\mathbf{D}$, that can be used within our algorithm, we focus on a few classical problems which can be formulated as dictionary learning problems with specific combinations of such regularizers.

5.3 Non Negative Matrix Factorization

Given a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ in $\mathbb{R}^{m \times n}$, Lee and Seung (2001) have proposed the non negative matrix factorization problem (NMF), which consists of minimizing the following cost

$$\min_{\mathbf{D} \in \mathcal{C}, \alpha \in \mathbb{R}^{k \times n}} \sum_{i=1}^{n} \frac{1}{2} ||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 \text{ s.t. } \mathbf{D} \geq 0, \ \forall \, i, \ \ \alpha_i \geq 0.$$

With this formulation, the matrix $\mathbf{D}$ and the vectors α_i are forced to have non negative components, which leads to sparse solutions. When applied to images, such as faces, Lee and Seung (2001) have shown that the learned features are more localized than the ones learned with a classical singular value decomposition. As for dictionary learning, classical approaches for addressing this problem are batch algorithms, such as the multiplicative update rules of Lee and Seung (2001), or the projected gradient descent algorithm of Lin (2007).

Following this line of research, Hoyer (2002, 2004) has proposed non negative sparse coding (NNSC), which extends non-negative matrix factorization by adding a sparsity-inducing penalty to the objective function to further control the sparsity of the vectors α_i:

$$\min_{\mathbf{D} \in \mathcal{C}, \alpha \in \mathbb{R}^{k \times n}} \sum_{i=1}^{n} \left(\frac{1}{2} ||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda \sum_{j=1}^{k} \alpha_i[j] \right) \text{ s.t. } \mathbf{D} \geq 0, \ \forall \, i, \ \ \alpha_i \geq 0.$$

When $\lambda = 0$, this formulation is equivalent to NMF. The only difference with the dictionary learning problem is that non-negativity constraints are imposed on $\mathbf{D}$ and the vectors α_i. A simple modification of our algorithm, presented above, allows us to handle these constraints, while guaranteeing to find a stationary point of the optimization problem. Moreover, our approach can work in the setting when n is large.

5.4 Sparse Principal Component Analysis

Principal component analysis (PCA) is a classical tool for data analysis, which can be interpreted as a method for finding orthogonal directions maximizing the variance of the data, or as a low-rank matrix approximation method. Jolliffe et al. (2003), Zou et al. (2006), d'Aspremont et al. (2007), d'Aspremont et al. (2008), Witten et al. (2009) and Zass and Shashua (2007) have proposed different formulations for sparse principal component analysis (SPCA), which extends PCA by estimating sparse vectors maximizing the variance of the data, some of these formulations enforcing orthogonality between the sparse components, whereas some do not. In this paper, we formulate SPCA as a sparse matrix factorization which is equivalent to the dictionary learning problem with

eventually sparsity constraints on the dictionary—that is, we use the ℓ_1-regularization term for α and the "elastic-net" constraint for $\mathbf{D}$ (as used in a penalty term by Zou et al. 2006):

$$\min_{\alpha\in\mathbb{R}^{k\times n}} \sum_{i=1}^{n} \left(\frac{1}{2}||\mathbf{x}_i - \mathbf{D}\alpha_i||_2^2 + \lambda||\alpha_i||_1\right) \text{ s.t. } \forall j = 1,\ldots,k, \ ||\mathbf{d}_j||_2^2 + \gamma||\mathbf{d}_j||_1 \leq 1.$$

As detailed above, our dictionary update procedure amounts to successive orthogonal projection of the vectors $\mathbf{u}_j$ on the constraint set. More precisely, the update of $\mathbf{d}_j$ becomes

$$\begin{aligned}
\mathbf{u}_j &\leftarrow \frac{1}{\mathbf{A}[j,j]}(\mathbf{b}_j - \mathbf{D}\mathbf{a}_j) + \mathbf{d}_j,\\
\mathbf{d}_j &\leftarrow \arg\min_{\mathbf{d}\in\mathbb{R}^m} ||\mathbf{u}_j - \mathbf{d}||_2^2 \text{ s.t. } ||\mathbf{d}||_2^2 + \gamma||\mathbf{d}||_1 \leq 1,
\end{aligned}$$

which can be solved in linear time using Algorithm 3 presented in Appendix B. In addition to that, our SPCA method can be used with fused Lasso constraints as well.

5.5 Constrained Sparse Coding

Constrained sparse coding problems are often encountered in the literature, and lead to different loss functions such as

$$\ell'(\mathbf{x},\mathbf{D}) = \min_{\alpha\in\mathbb{R}^k} ||\mathbf{x} - \mathbf{D}\alpha||_2^2 \text{ s.t. } ||\alpha||_1 \leq T, \tag{15}$$

or

$$\ell''(\mathbf{x},\mathbf{D}) = \min_{\alpha\in\mathbb{R}^k} ||\alpha||_1 \text{ s.t. } ||\mathbf{x} - \mathbf{D}\alpha||_2^2 \leq \varepsilon, \tag{16}$$

where T and ε are pre-defined thresholds. Even though these loss functions lead to equivalent optimization problems in the sense that for given $\mathbf{x},\mathbf{D}$ and λ, there exist ε and T such that $\ell(\mathbf{x},\mathbf{D})$, $\ell'(\mathbf{x},\mathbf{D})$ and $\ell''(\mathbf{x},\mathbf{D})$ admit the same solution $\alpha^\star$, the problems of learning $\mathbf{D}$ using ℓ, ℓ' of ℓ'' are not equivalent. For instance, using ℓ'' has proven experimentally to be particularly well adapted to image denoising (Elad and Aharon, 2006; Mairal et al., 2008b).

For all T, the same analysis as for ℓ can be carried for ℓ', and the simple modification which consists of computing α_t using Eq. (15) in the sparse coding step leads to the minimization of the expected cost $\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}_{\mathbf{x}}[\ell'(\mathbf{x},\mathbf{D})]$.

Handling the case ℓ'' is a bit different. We propose to use the same strategy as for ℓ'—that is, using our algorithm but computing α_t solving Eq. (16). Even though our analysis does not apply since we do not have a quadratic surrogate of the expected cost, experimental evidence shows that this approach is efficient in practice.

5.6 Simultaneous Sparse Coding

In some situations, the signals $\mathbf{x}_i$ are not i.i.d samples of an unknown probability distribution, but are structured in groups (which are however independent from each other), and one may want to address the problem of simultaneous sparse coding, which appears also in the literature under various names such as group sparsity or grouped variable selection (Cotter et al., 2005; Turlach et al., 2005; Yuan and Lin, 2006; Obozinski et al., 2009, 2008; Zhang et al., 2008; Tropp et al., 2006; Tropp, 2006). Let $\mathbf{X} = [\mathbf{x}_1,\ldots,\mathbf{x}_q] \in \mathbb{R}^{m\times q}$ be a set of signals. Suppose one wants to obtain sparse decompositions of the signals on the dictionary $\mathbf{D}$ that share the same active set (non-zero coefficients).

Let $\alpha = [\alpha_1, \ldots, \alpha_q]$ in $\mathbb{R}^{k \times q}$ be the matrix composed of the coefficients. One way of imposing this *joint sparsity* is to penalize the number of non-zero rows of α. A classical convex relaxation of this joint sparsity measure is to consider the $\ell_{1,2}$-norm on the matrix α

$$||\alpha||_{1,2} \triangleq \sum_{j=1}^{k} ||\alpha^j||_2,$$

where α^j is the j-th row of α. In that setting, the $\ell_{1,2}$-norm of α is the ℓ_1-norm of the ℓ_2-norm of the rows of α.

The problem of jointly decomposing the signals $\mathbf{x}_i$ can be written as a $\ell_{1,2}$-sparse decomposition problem, which is a subcase of the group Lasso (Turlach et al., 2005; Yuan and Lin, 2006; Bach, 2008), by defining the cost function

$$\ell'''(\mathbf{X}, \mathbf{D}) = \min_{\alpha \in \mathbb{R}^{k \times q}} \frac{1}{2} ||\mathbf{X} - \mathbf{D}\alpha||_F^2 + \lambda ||\alpha||_{1,2},$$

which can be computed using a block-coordinate descent approach (Friedman et al., 2007) or an active set method (Roth and Fischer, 2008).

Suppose now that we are able to draw groups of signals $\mathbf{X}_i, i = 1, \ldots, n$ which have bounded size and are independent from each other and identically distributed, one can learn an adapted dictionary by solving the optimization problem

$$\min_{\mathbf{D} \in \mathcal{C}} \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} \ell'''(\mathbf{X}_i, \mathbf{D}).$$

Being able to solve this optimization problem is important for many applications. For instance, in Mairal et al. (2009c), state-of-the-art results in image denoising and demosaicking are achieved with this formulation. The extension of our algorithm to this case is relatively easy, computing at each sparse coding step a matrix of coefficients α, and keeping the updates of $\mathbf{A}_t$ and $\mathbf{B}_t$ unchanged.

All of the variants of this section have been implemented. Next section evaluates some of them experimentally. An efficient C++ implementation with a Matlab interface of these variants is available on the Willow project-team web page `http://www.di.ens.fr/willow/SPAMS/`.

6. Experimental Validation

In this section, we present experiments on natural images and genomic data to demonstrate the efficiency of our method for dictionary learning, non-negative matrix factorization, and sparse principal component analysis.

6.1 Performance Evaluation for Dictionary Learning

For our experiments, we have randomly selected 1.25×10^6 patches from images in the Pascal VOC'06 image database (Everingham et al., 2006), which is composed of varied natural images; 10^6 of these are kept for training, and the rest for testing. We used these patches to create three data sets A, B, and C with increasing patch and dictionary sizes representing various settings which are typical in image processing applications: We have centered and normalized the patches to have unit ℓ_2-norm and used the regularization parameter $\lambda = 1.2/\sqrt{m}$ in all of our experiments. The $1/\sqrt{m}$

Data set	Signal size m	Nb k of atoms	Type
A	$8 \times 8 = 64$	256	b&w
B	$12 \times 12 \times 3 = 432$	512	color
C	$16 \times 16 = 256$	1024	b&w

term is a classical normalization factor (Bickel et al., 2009), and the constant 1.2 has shown to yield about 10 nonzero coefficients for data set A and 40 for data sets B and C in these experiments. We have implemented the proposed algorithm in C++ with a Matlab interface. All the results presented in this section use the refinements from Section 3.4 since this has lead empirically to speed improvements. Although our implementation is multithreaded, our experiments have been run for simplicity on a single-CPU, single-core 2.66Ghz machine.

The first parameter to tune is η, the number of signals drawn at each iteration. Trying different powers of 2 for this variable has shown that $\eta = 512$ was a good choice (lowest objective function values on the training set—empirically, this setting also yields the lowest values on the test set). Even though this parameter is fairly easy to tune since values of $64, 128, 256$ and 1024 have given very similar performances, the difference with the choice $\eta = 1$ is significant.

Our implementation can be used in both the online setting it is intended for, and in a regular batch mode where it uses the entire data set at each iteration. We have also implemented a first-order stochastic gradient descent algorithm that shares most of its code with our algorithm, except for the dictionary update step. This setting allows us to draw meaningful comparisons between our algorithm and its batch and stochastic gradient alternatives, which would have been difficult otherwise. For example, comparing our algorithm to the Matlab implementation of the batch approach from Lee et al. (2007) developed by its authors would have been unfair since our C++ program has a built-in speed advantage.[8] To measure and compare the performances of the three tested methods, we have plotted the value of the objective function on *the test set*, acting as a surrogate of the expected cost, as a function of the corresponding training time.

6.1.1 ONLINE VS. BATCH

The left column of Figure 1 compares the online and batch settings of our implementation. The full training set consists of 10^6 samples. The online version of our algorithm draws samples from the entire set, and we have run its batch version on the full data set as well as subsets of size 10^4 and 10^5 (see Figure 1). The online setting systematically outperforms its batch counterpart for every training set size and desired precision. We use a logarithmic scale for the computation time, which shows that in many situations, the difference in performance can be dramatic. Similar experiments have given similar results on smaller data sets. Our algorithm uses all the speed-ups from Section 3.4. The parameter ρ was chosen by trying the values $0, 5, 10, 15, 20, 25$, and t_0 by trying different powers of 10. We have selected $(t_0 = 0.001, \rho = 15)$, which has given the best performance in terms of objective function evaluated on the training set for the three data sets. We have plotted three curves for our method: OL1 corresponds to the optimal setting $(t_0 = 0.001, \rho = 15)$. Even though tuning two parameters might seem cumbersome, we have plotted two other curves showing that, on the contrary, our method is very easy to use. The curve OL2, corresponding to the setting

8. Both LARS and the feature-sign algorithm (Lee et al., 2007) require a large number of low-level operations which are not well optimized in Matlab. We have indeed observed that our C++ implementation of LARS is up to 50 times faster than the Matlab implementation of the feature-sign algorithm of Lee et al. (2007) for our experiments.

$(t_0 = 0.001, \rho = 10)$, is very difficult to distinguish from the first curve and we have observed a similar behavior with the setting $(t_0 = 0.001, \rho = 20)$. showing that our method is *robust to the choice of the parameter* ρ. We have also observed that the parameter ρ is useful for large data sets only. When using smaller ones ($n \leq 100,000$), it did not bring any benefit.

Moreover, the curve OL3 is obtained without using a tuned parameter t_0—that is, $\rho = 15$ and $t_0 = 0$, and shows that its influence is very limited since very good results are obtained without using it. On the other hand, we have observed that using a parameter t_0 too big, could slightly slow down our algorithm during the first epoch (cycle on the training set).

6.1.2 COMPARISON WITH STOCHASTIC GRADIENT DESCENT

Our experiments have shown that obtaining good performance with stochastic gradient descent requires using both the mini-batch heuristic *and* carefully choosing a learning rate of the form $a/(\eta t + b)$. To give the fairest comparison possible, we have thus optimized these parameters. As for our algorithm, sampling η values among powers of 2 (as before) has shown that $\eta = 512$ was a good value and gives a significant better performance than $\eta = 1$.

In an earlier version of this work (Mairal et al., 2009a), we have proposed a strategy for our method which does not require any parameter tuning except the mini-batch η and compared it with the stochastic gradient descent algorithm (SGD) with a learning rate of the form $a/(\eta t)$. While our method has improved in performance using the new parameter ρ, SGD has also proven to provide much better results when using a learning rate of the form $a/(\eta t + b)$ instead of $a/(\eta t)$, at the cost of an extra parameter b to tune. Using the learning rate $a/(\eta t)$ with a high value for a results indeed in too large initial steps of the algorithm increasing dramatically the value of the objective function, and a small value of a leads to bad asymptotic results, while a learning rate of the form $a/(\eta t + b)$ is a good compromise.

We have tried different powers of 10 for a and b. First selected the couple $(a = 100,000, b = 100,000)$ and then refined it, trying the values $100,000 \times 2^i$ for $i = -3, \ldots, 3$. Finally, we have selected $(a = 200,000, b = 400,000)$. As shown on the right column of Figure 1, this setting represented by the curve SG1 leads to similar results as our method. The curve SG2 corresponds to the parameters $(a = 400,000, b = 400,000)$ and shows that increasing slightly the parameter a makes the curves worse than the others during the first iterations (see for instance the curve between 1 and 10^2 seconds for data set A), but still lead to good asymptotic results. The curve SG3 corresponds to a situation where a and b are slightly too small $(a = 50,000, b = 100,000)$. It is as good as SG1 for data sets A and B, but asymptotically slightly below the others for data set C. All the curves are obtained as the average of three experiments with different initializations. Interestingly, even though the problem is not convex, the different initializations have led to very similar values of the objective function and the variance of the experiments was always insignificant after 10 seconds of computations.

6.2 Non Negative Matrix Factorization and Non Negative Sparse Coding

In this section, we compare our method with the classical algorithm of Lee and Seung (2001) for NMF and the non-negative sparse coding algorithm of Hoyer (2002) for NNSC. The experiments have been carried out on three data sets with different sizes:

- Data set D is composed of $n = 2,429$ face images of size $m = 19 \times 19$ pixels from the the MIT-CBCL Face Database #1 (Sung, 1996).

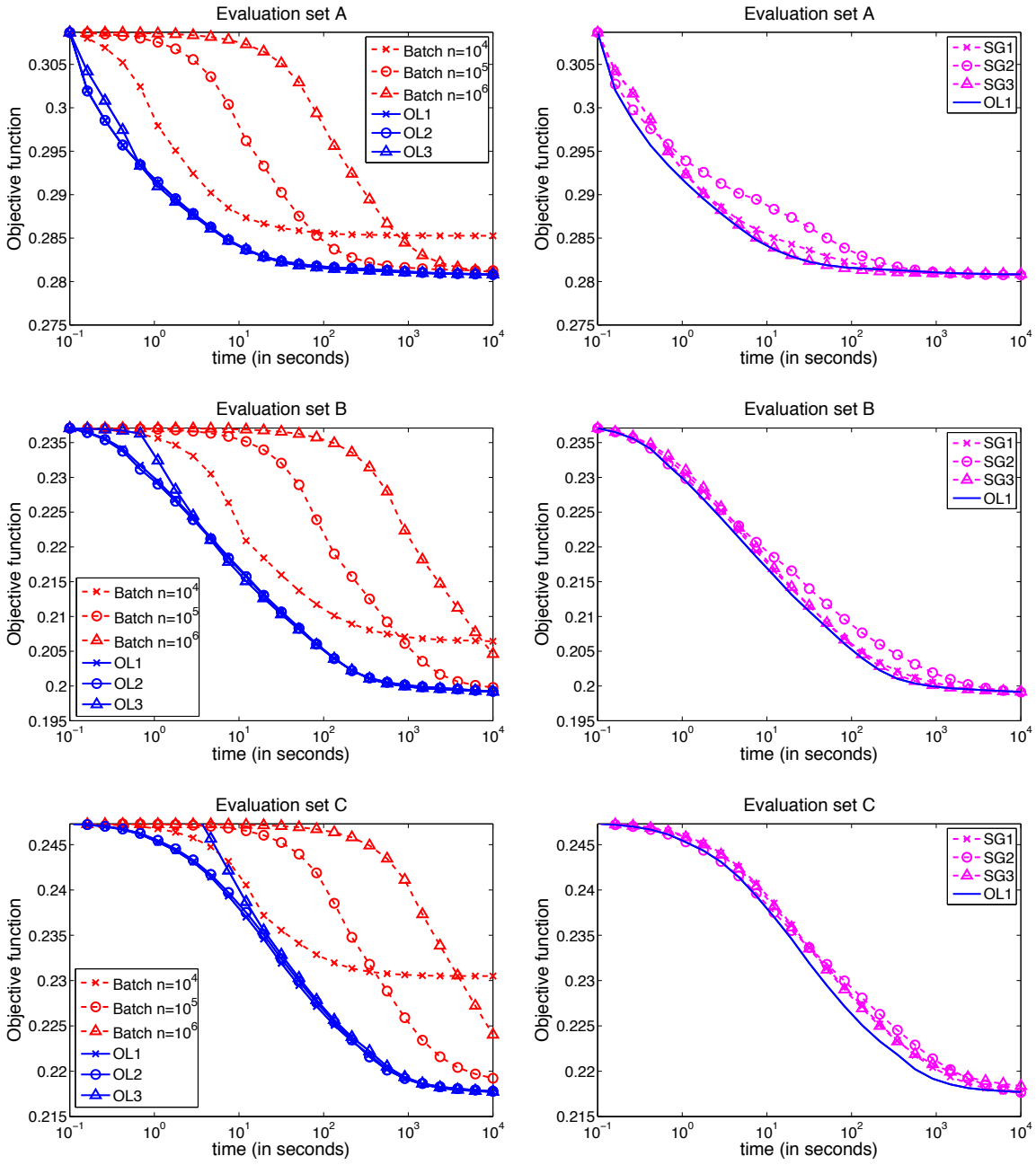


Figure 1: Left: Comparison between our method and the batch approach for dictionary learning. Right: Comparison between our method and stochastic gradient descent. The results are reported for three data sets as a function of computation time on a logarithmic scale. Note that the times of computation that are less than $0.1s$ are not reported. See text for details.

- Data set E is composed of $n = 2,414$ face images of size $m = 192 \times 168$ pixels from the Extended Yale B Database (Georghiades et al., 2001; Lee et al., 2005).
- Data set F is composed of $n = 100,000$ natural image patches of size $m = 16 \times 16$ pixels from the Pascal VOC'06 image database (Everingham et al., 2006).

We have used the Matlab implementations of NMF and NNSC of P. Hoyer, which are freely available at `http://www.cs.helsinki.fi/u/phoyer/software.html`. Even though our C++ implementation has a built-in advantage in terms of speed over these Matlab implementations, most of the computational time of NMF and NNSC is spent on large matrix multiplications, which are typically well optimized in Matlab. All the experiments have been run for simplicity on a single-CPU, single-core 2.4GHz machine, without using the parameters ρ and t_0 presented in Section 3.4—that is, $\rho = 0$ and $t_0 = 0$. As in Section 6.1, a minibatch of size $\eta = 512$ is chosen. Following the original experiment of Lee and Seung (2001) on data set D, we have chosen to learn $k = 49$ basis vectors for the face images data sets D and E, and we have chosen $k = 64$ for data set F. Each input vector is normalized to have unit ℓ_2-norm.

The experiments we present in this section compare the value of the objective function on the data sets obtained with the different algorithms as a function of the computation time. Since our algorithm learns the matrix $\mathbf{D}$ but does not provide the matrix α, the computation times reported for our approach include two steps: First, we run our algorithm to obtain $\mathbf{D}$. Second, we run one sparse coding step over all the input vectors to obtain α. Figure 2 presents the results for NMF and NNSC. The gradient step for the algorithm of Hoyer (2002) was optimized for the best performance and λ was set to $\frac{1}{\sqrt{m}}$. Both $\mathbf{D}$ and α were initialized randomly. The values reported are those obtained for more than 0.1s of computation. Since the random initialization provides an objective value which is by far greater than the value obtained at convergence, the curves are all truncated to present significant objective values. All the results are obtained using the average of 3 experiments with different initializations. As shown on Figure 2, our algorithm provides a significant improvement in terms of speed compared to the other tested methods, even though the results for NMF and NNSC could be improved a bit using a C++ implementation.

6.3 Sparse Principal Component Analysis

We present here the application of our method addressing SPCA with various types of data: faces, natural image patches, and genomic data.

6.3.1 FACES AND NATURAL PATCHES

In this section, we compare qualitatively the results obtained by PCA, NMF, our dictionary learning and our sparse principal component analysis algorithm on the data sets used in Section 6.2. For dictionary learning, PCA and SPCA, the input vectors are first centered and normalized to have a unit norm. Visual results are presented on figures 3, 4 and 5, respectively for the data sets D, E and F. The parameter λ for dictionary learning and SPCA was set so that the decomposition of each input signal has approximately 10 nonzero coefficients. The results for SPCA are presented for various values of the parameter γ, yielding different levels of sparsity. The scalar τ indicates the percentage of nonzero values of the dictionary.

Each image is composed of k small images each representing one learned feature vector. Negative values are blue, positive values are red and the zero values are represented in white. Confirming

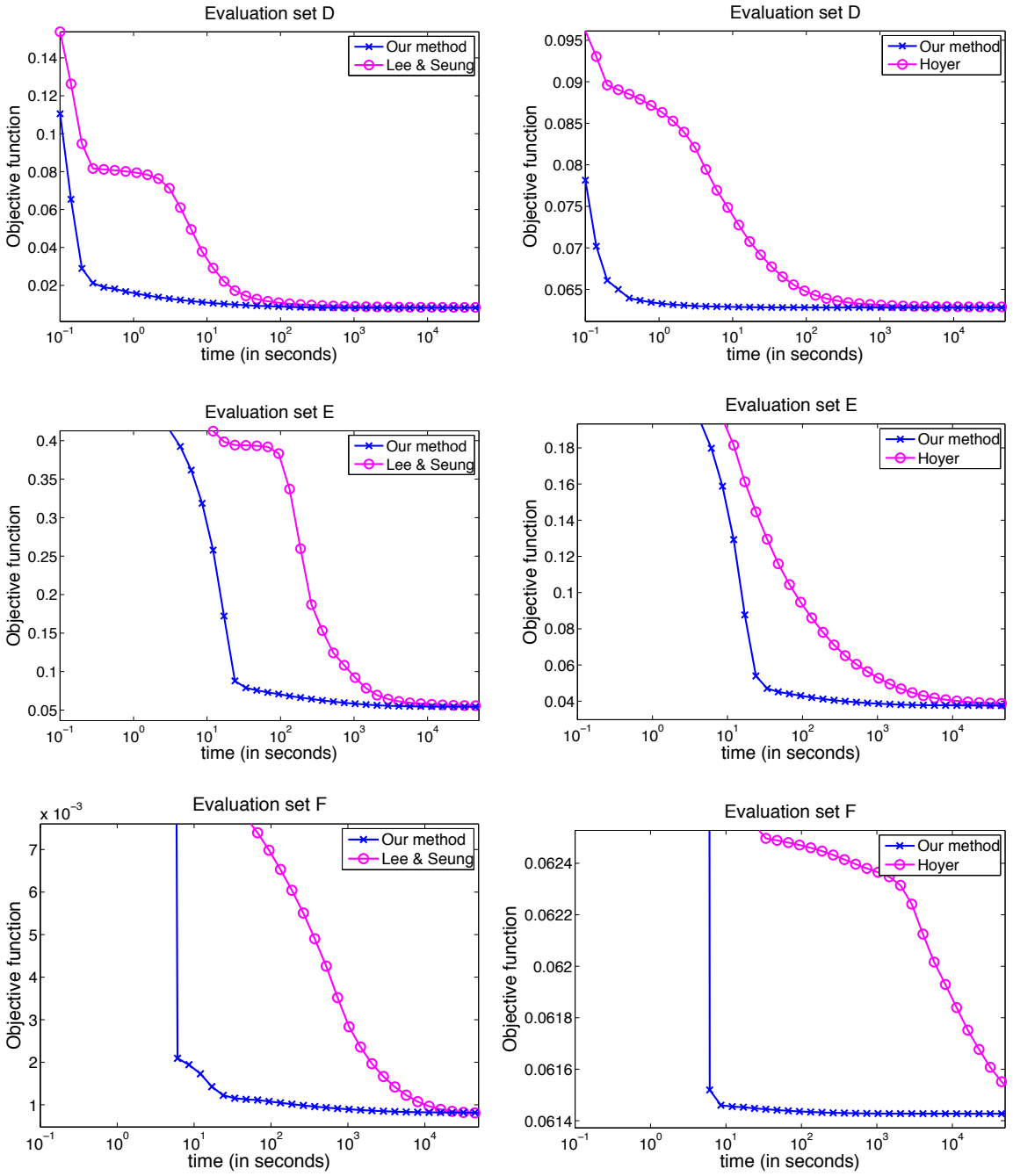


Figure 2: Left: Comparison between our method and the approach of Lee and Seung (2001) for NMF. Right: Comparison between our method and the approach of Hoyer (2002) for NNSC. The value of the objective function is reported for three data sets as a function of computation time on a logarithmic scale.

earlier observations from Lee and Seung (2001), PCA systematically produces features spread out over the images, whereas NMF produces more localized features on the face databases D and E. However, neither PCA, nor NMF are able to learn localized features on the set of natural patches F. On the other hand, the dictionary learning technique is able to learn localized features on data set F, and SPCA is the only tested method that allows controlling the level of sparsity among the learned matrices.

6.3.2 GENOMIC DATA

This experiment follows Witten et al. (2009) and demonstrates that our matrix decomposition technique can be used for analyzing genomic data. Gene expression measurements and DNA copy number changes (comparative genomic hybridization CGH) are two popular types of data in genomic research, which can be used to characterize a set of abnormal tissue samples for instance. When these two types of data are available, a recent line of research tries to analyze the correlation between them—that is, to determine sets of expression genes which are correlated with sets of chromosomal gains or losses (see Witten et al., 2009 and references therein). Let us suppose that for n tissue samples, we have a matrix $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ of gene expression measurements and a matrix $\mathbf{Y}$ in $\mathbb{R}^{n\times q}$ of CGH measurements. In order to analyze the correlation between these two sets of data, recent works have suggested the use of canonical correlation analysis (Hotelling, 1936), which solves[9]

$$\min_{\mathbf{u}\in\mathbb{R}^p,\mathbf{v}\in\mathbb{R}^q} \text{cov}(\mathbf{Xu},\mathbf{Yv}) \;\text{ s.t. }\; ||\mathbf{Xu}||_2 \leq 1 \;\text{ and }\; ||\mathbf{Yv}||_2 \leq 1.$$

When $\mathbf{X}$ and $\mathbf{Y}$ are centered and normalized, it has been further shown that with this type of data, good results can be obtained by treating the covariance matrices $\mathbf{X}^T\mathbf{X}$ and $\mathbf{Y}^T\mathbf{Y}$ as diagonal, leading to a rank-one matrix decomposition problem

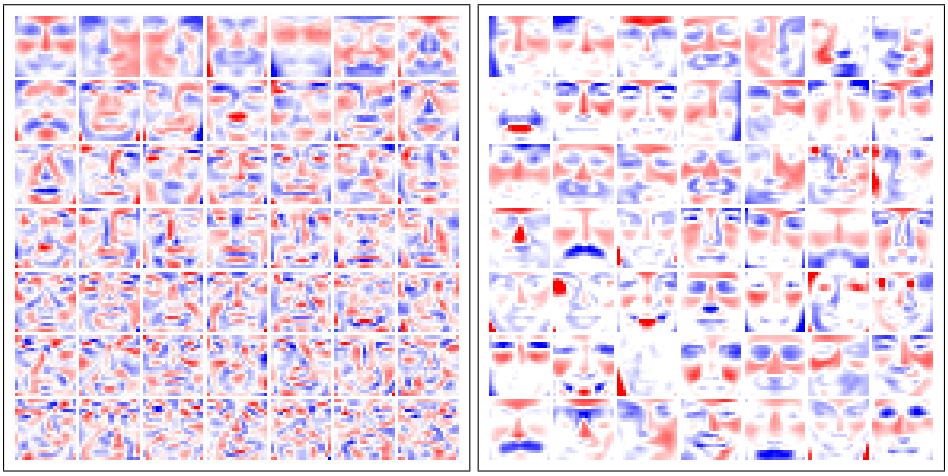
$$\min_{\mathbf{u}\in\mathbb{R}^p,\mathbf{v}\in\mathbb{R}^q} ||\mathbf{X}^T\mathbf{Y}-\mathbf{uv}^T||_F^2 \;\text{ s.t. }\; ||\mathbf{u}||_2 \leq 1, \;\text{ and }\; ||\mathbf{v}||_2 \leq 1.$$

Furthermore, as shown by Witten et al. (2009), this method can benefit from sparse regularizers such as the ℓ_1 norm for the gene expression measurements and a fused lasso for the CGH arrays, which are classical choices used for these data. The formulation we have chosen is slightly different from the one used by Witten et al. (2009) and can be addressed using our algorithm:

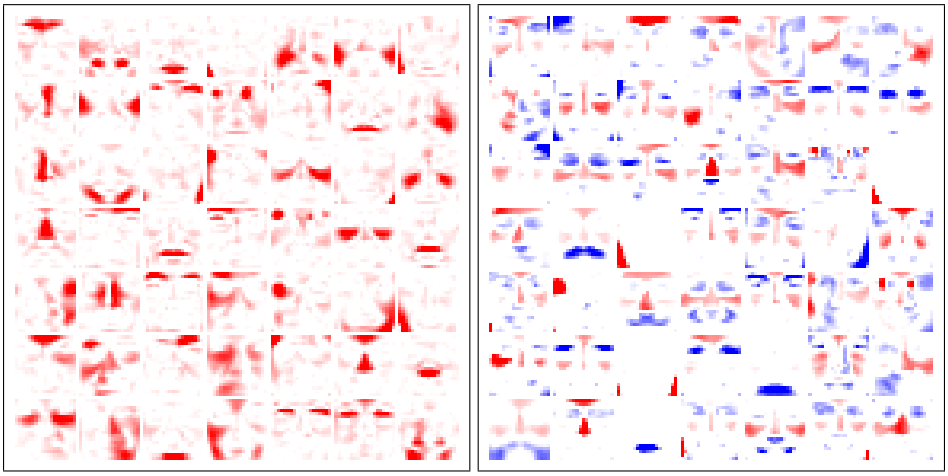
$$\min_{\mathbf{u}\in\mathbb{R}^p,\mathbf{v}\in\mathbb{R}^q} ||\mathbf{Y}^T\mathbf{X}-\mathbf{vu}^T||_F^2+\lambda||\mathbf{u}||_2 \;\text{ s.t. }\; ||\mathbf{v}||_2^2+\gamma_1||\mathbf{v}||_1+\gamma_2\,\text{FL}(\mathbf{v}) \leq 1. \tag{17}$$

In order to assess the effectivity of our method, we have conducted the same experiment as Witten et al. (2009) using the breast cancer data set described by Chin et al. (2006), consisting of $q = 2,148$ gene expression measurements and $p = 16,962$ CGH measurements for $n = 89$ tissue samples. The matrix decomposition problem of Eq. (17) was addressed once for each of the 23 chromosomes, using each time the CGH data available for the corresponding chromosome, and the gene expression of all genes. Following the original choice of Witten et al. (2009), we have selected a regularization parameter λ resulting in about 25 non-zero coefficients in $\mathbf{u}$, and selected $\gamma_1 = \gamma_2 = 1$, which results in sparse and piecewise-constant vectors $\mathbf{v}$. The original matrices $(\mathbf{X},\mathbf{Y})$ are divided into a training set $(\mathbf{X}_{tr},\mathbf{Y}_{tr})$ formed with $3/4$ of the n samples, keeping the rest $(\mathbf{X}_{te},\mathbf{Y}_{te})$ for testing. This

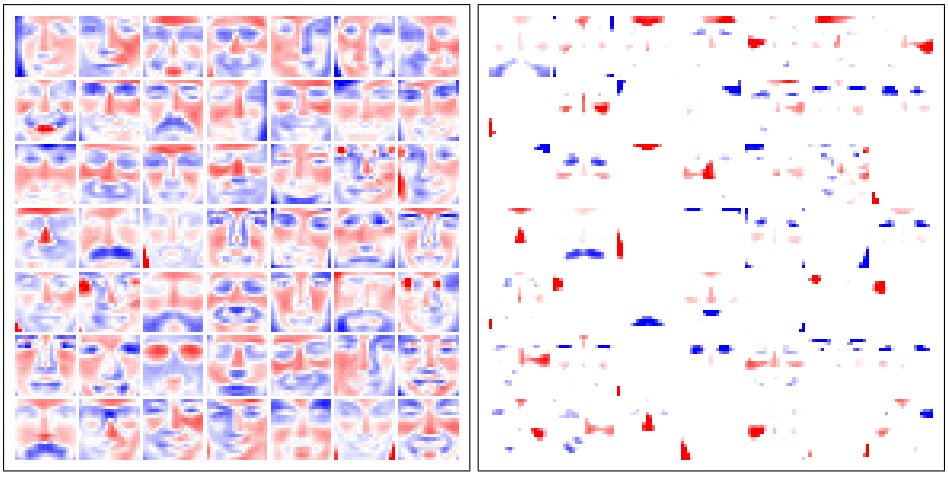
9. Note that when more than one couple of factors are needed, two sequences $\mathbf{u}_1,\mathbf{u}_2,\ldots$ and $\mathbf{v}_1,\mathbf{v}_2,\ldots$ of factors can be obtained recursively subject to orthogonality constraints of the sequences $\mathbf{Xu}_1,\mathbf{Xu}_2,\ldots$ and $\mathbf{Yv}_1,\mathbf{Yv}_2,\ldots$.



(a) PCA (b) SPCA, $\tau = 70\%$

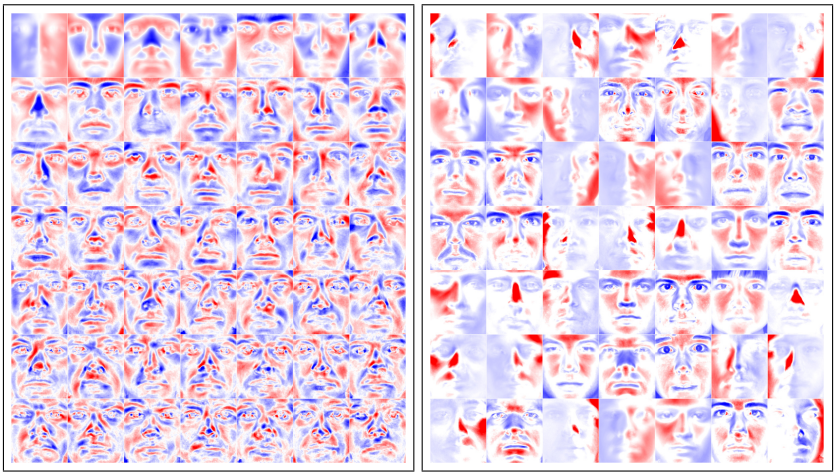


(c) NMF (d) SPCA, $\tau = 30\%$

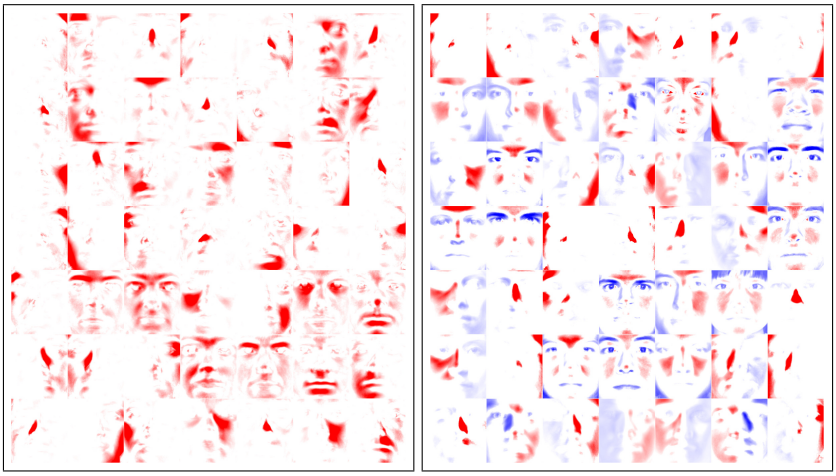


(e) Dictionary Learning (f) SPCA, $\tau = 10\%$

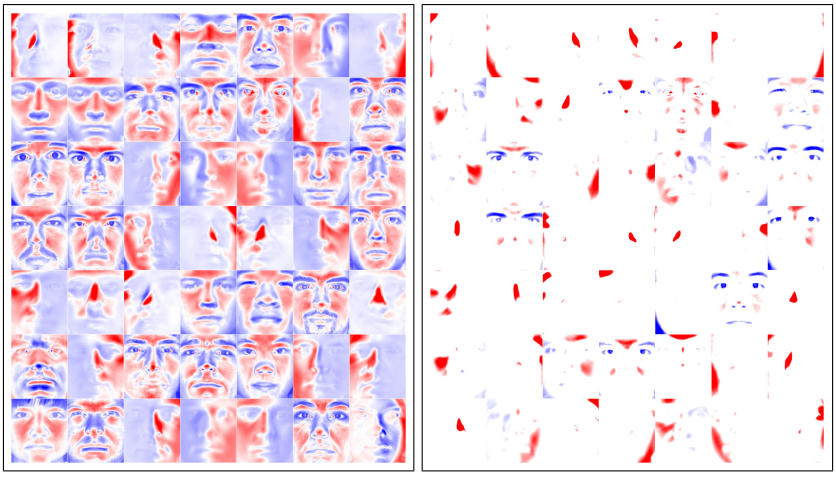
Figure 3: Results obtained by PCA, NMF, dictionary learning, SPCA for data set D.



(a) PCA (b) SPCA, $\tau = 70\%$

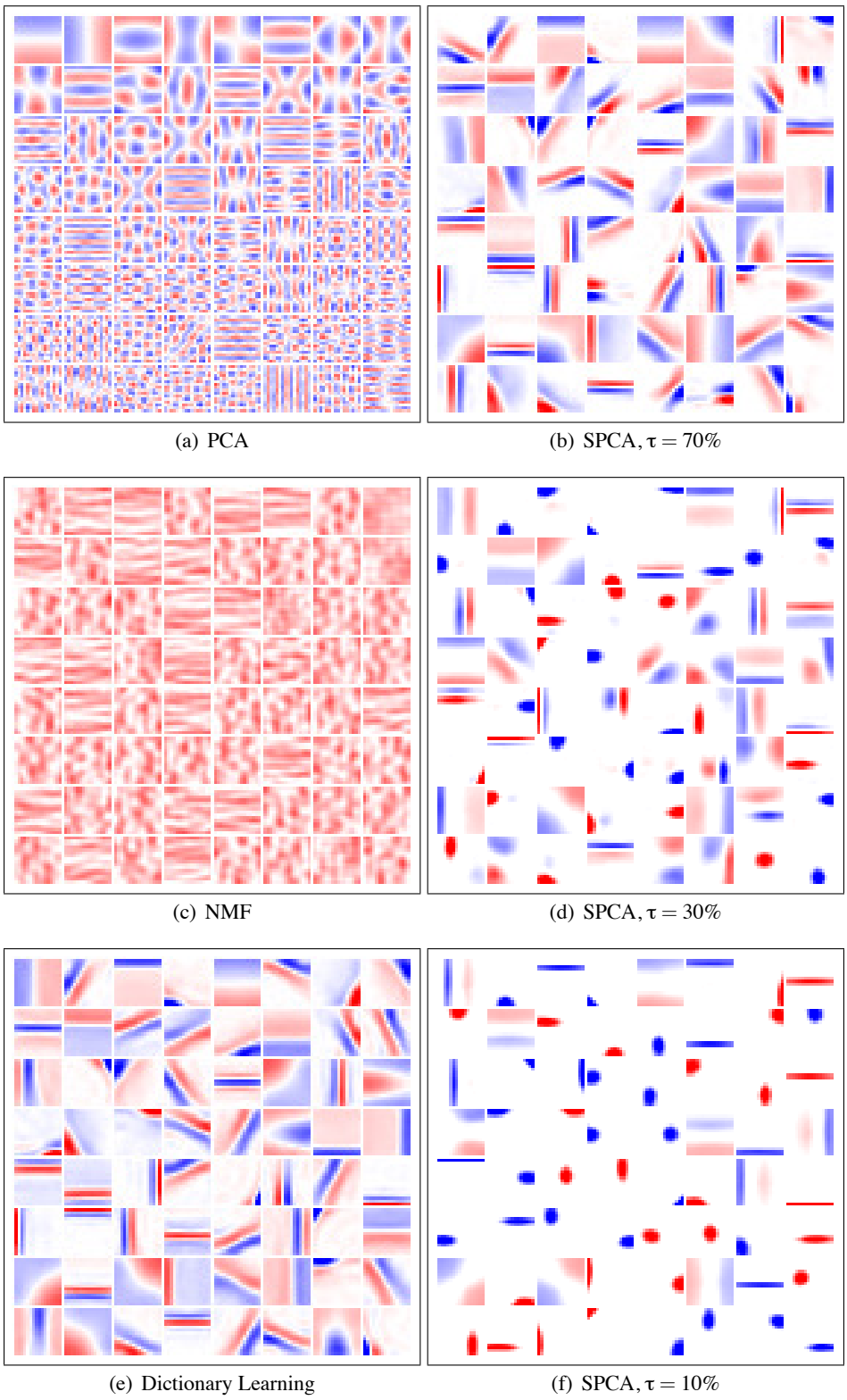


(c) NMF (d) SPCA, $\tau = 30\%$



(e) Dictionary Learning (f) SPCA, $\tau = 10\%$

Figure 4: Results obtained by PCA, NMF, dictionary learning, SPCA for data set E.



(a) PCA (b) SPCA, $\tau = 70\%$

(c) NMF (d) SPCA, $\tau = 30\%$

(e) Dictionary Learning (f) SPCA, $\tau = 10\%$

Figure 5: Results obtained by PCA, NMF, dictionary learning, SPCA for data set F.

experiment is repeated for 10 random splits, for each chromosome a couple of factors $(\mathbf{u}, \mathbf{v})$ are computed, and the correlations $\text{corr}(\mathbf{X}_{tr}\mathbf{u}, \mathbf{Y}_{tr}\mathbf{v})$ and $\text{corr}(\mathbf{X}_{te}\mathbf{u}, \mathbf{Y}_{te}\mathbf{v})$ are reported on Figure 6. The average standard deviation of the experiments results was 0.0339 for the training set and 0.1391 for the test set.

Comparing with the original curves reported by Witten et al. (2009) for their penalized matrix decomposition (PMD) algorithm, our method exhibits in general a performance similar as PMD.[10] Nevertheless, the purpose of this section is more to demonstrate that our method can be used with genomic data than comparing it carefully with PMD. To draw substantial conclusions about the performance of both methods, more experiments would of course be needed.

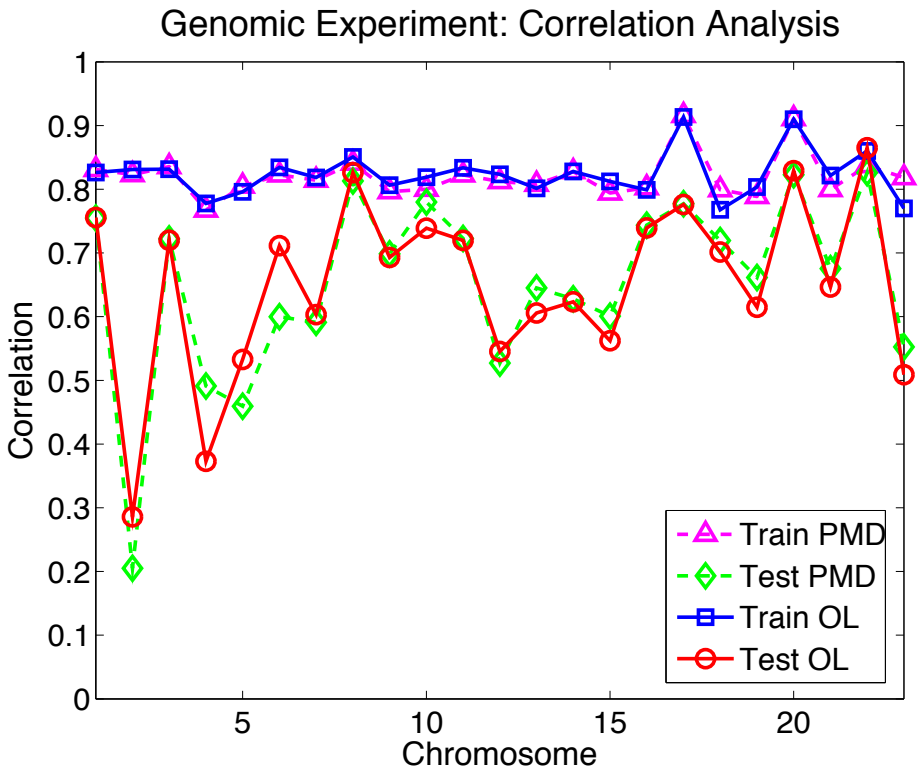


Figure 6: SPCA was applied to the covariance matrix obtained from the breast cancer data (Chin et al., 2006). A fused lasso regularization is used for the CGH data. $3/4$ of the n samples are used as a training set, keeping the rest for testing. Average correlations from 10 random splits are reported for each of the 23 chromosomes, for PMD (Witten et al., 2009) and our method denoted by OL.

10. The curves for PMD were generated with the R software package available at `http://cran.r-project.org/web/packages/PMA/index.html` and a script provided by Witten et al. (2009).

Figure 7: Inpainting example on a 12-Megapixel image. Top: Damaged and restored images. Bottom: Zooming on the damaged and restored images. Note that the pictures presented here have been scaled down for display. (Best seen in color).

6.4 Application to Large-Scale Image Processing

We demonstrate in this section that our algorithm can be used for a difficult large-scale image processing task, namely, removing the text (*inpainting*) from the damaged 12-Megapixel image of Figure 7. Using a multi-threaded version of our implementation, we have learned a dictionary with 256 elements from the roughly 7×10^6 undamaged 12×12 color patches in the image with two epochs in about 8 minutes on a 2.4GHz machine with eight cores. Once the dictionary has been learned, the text is removed using the sparse coding technique for inpainting of Mairal et al. (2008b). Our intent here is of course *not* to evaluate our learning procedure in inpainting tasks, which would require a thorough comparison with state-the-art techniques on standard data sets. Instead, we just wish to demonstrate that it can indeed be applied to a realistic, non-trivial image processing task on a large image. Indeed, to the best of our knowledge, this is the first time that dictionary learning is used for image restoration on such large-scale data. For comparison, the dictionaries used for inpainting in Mairal et al. (2008b) are learned (in batch mode) on 200,000 patches only.

7. Conclusion

We have introduced in this paper a new stochastic online algorithm for learning dictionaries adapted to sparse coding tasks, and proven its convergence. Experiments demonstrate that it is significantly faster than batch alternatives such as Engan et al. (1999), Aharon et al. (2006) and Lee et al. (2007) on large data sets that may contain millions of training examples, yet it does not require a careful learning rate tuning like regular stochastic gradient descent methods. Moreover, we have extended it to other matrix factorization problems such as non negative matrix factorization, and we have proposed a formulation for sparse principal component analysis which can be solved efficiently using our method. Our approach has already shown to be useful for image restoration tasks such as denoising (Mairal et al., 2009c); more experiments are of course needed to better assess its promise in bioinformatics and signal processing. Beyond this, we plan to use the proposed learning framework for sparse coding in computationally demanding video restoration tasks (Protter and Elad, 2009), with dynamic data sets whose size is not fixed, and extending this framework to different loss functions (Mairal et al., 2009b) to address discriminative tasks such as image classification, which are more sensitive to overfitting than reconstructive ones.

Acknowledgments

This paper was supported in part by ANR under grant MGA ANR-07-BLAN-0311. The work of Guillermo Sapiro is partially supported by ONR, NGA, NSF, ARO, and DARPA. The authors would like to like to thank Sylvain Arlot, Léon Bottou, Jean-Philippe Vert, and the members of the Willow project-team for helpful discussions, and Daniela Witten for providing us with her code to generate the curves of Figure 6.

Appendix A. Theorems and Useful Lemmas

We provide in this section a few theorems and lemmas from the optimization and probability literature, which are used in this paper.

Theorem 5 [Corollary of Theorem 4.1 from Bonnans and Shapiro (1998), due to Danskin (1967)].
Let $f : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$. Suppose that for all $\mathbf{x} \in \mathbb{R}^p$ the function $f(\mathbf{x},.)$ is differentiable, and that f and $\nabla_{\mathbf{u}} f(\mathbf{x},\mathbf{u})$ the derivative of $f(\mathbf{x},.)$ are continuous on $\mathbb{R}^p \times \mathbb{R}^q$. Let $v(\mathbf{u})$ be the optimal value function $v(\mathbf{u}) = \min_{\mathbf{x} \in C} f(\mathbf{x},\mathbf{u})$, where C is a compact subset of $\mathbb{R}^p$. Then $v(\mathbf{u})$ is directionally differentiable. Furthermore, if for $\mathbf{u}_0 \in \mathbb{R}^q$, $f(.,\mathbf{u}_0)$ has a unique minimizer $\mathbf{x}_0$ then $v(\mathbf{u})$ is differentiable in $\mathbf{u}_0$ and $\nabla_{\mathbf{u}} v(\mathbf{u}_0) = \nabla_{\mathbf{u}} f(\mathbf{x}_0,\mathbf{u}_0)$.

Theorem 6 [Sufficient condition of convergence for a stochastic process, see Bottou (1998) and references therein (Métivier, 1983; Fisk, 1965)].
Let $(\Omega,\mathcal{F},P)$ be a measurable probability space, u_t, for $t \geq 0$, be the realization of a stochastic process and $\mathcal{F}_t$ be the filtration determined by the past information at time t. Let

$$\delta_t = \begin{cases} 1 & \text{if } \mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t] > 0, \\ 0 & \text{otherwise.} \end{cases}$$

If for all t, $u_t \geq 0$ and $\sum_{t=1}^{\infty} \mathbb{E}[\delta_t(u_{t+1} - u_t)] < \infty$, then u_t is a quasi-martingale and converges almost surely. Moreover,

$$\sum_{t=1}^{\infty} |\mathbb{E}[u_{t+1} - u_t | \mathcal{F}_t]| < +\infty \ \ a.s.$$

Lemma 7 [A corollary of Donsker theorem see Van der Vaart, 1998, chap. 19.2, lemma 19.36 and example 19.7].
Let $F = \{f_\theta : \chi \to \mathbb{R}, \theta \in \Theta\}$ be a set of measurable functions indexed by a bounded subset Θ of $\mathbb{R}^d$. Suppose that there exists a constant K such that

$$|f_{\theta_1}(x) - f_{\theta_2}(x)| \leq K||\theta_1 - \theta_2||_2,$$

for every θ_1 and θ_2 in Θ and x in χ. Then, F is P-Donsker (see Van der Vaart, 1998, chap. 19.2). For any f in F, Let us define $\mathbb{P}_n f$, $\mathbb{P}f$ and $\mathbb{G}_n f$ as

$$\mathbb{P}_n f = \frac{1}{n}\sum_{i=1}^{n} f(X_i), \quad \mathbb{P}f = \mathbb{E}_X[f(X)], \quad \mathbb{G}_n f = \sqrt{n}(\mathbb{P}_n f - \mathbb{P}f).$$

Let us also suppose that for all f, $\mathbb{P}f^2 < \delta^2$ and $||f||_\infty < M$ and that the random elements $X_1, X_2, \ldots$ are Borel-measurable. Then, we have

$$\mathbb{E}_P||\mathbb{G}_n||_F = O(1),$$

where $||\mathbb{G}_n||_F = \sup_{f \in F} |\mathbb{G}_n f|$. For a more general variant of this lemma and additional explanations and examples, see Van der Vaart (1998).

Lemma 8 [A simple lemma on positive converging sums].
Let a_n, b_n be two real sequences such that for all n, $a_n \geq 0, b_n \geq 0$, $\sum_{n=1}^{\infty} a_n = \infty$, $\sum_{n=1}^{\infty} a_n b_n < \infty$, $\exists K > 0 \ \ s.t. \ |b_{n+1} - b_n| < K a_n$. Then, $\lim_{n \to +\infty} b_n = 0$.

Proof The proof is similar to Bertsekas (1999, prop 1.2.4). ■

Appendix B. Efficient Projections Algorithms

In this section, we address the problem of efficiently projecting a vector onto two sets of constraints, which allows us to extend our algorithm to various other formulations.

B.1 A Linear-time Projection on the Elastic-Net Constraint

Let $\mathbf{b}$ be a vector of $\mathbb{R}^m$. We consider the problem of projecting this vector onto the elastic-net constraint set:

$$\min_{\mathbf{u} \in \mathbb{R}^m} \frac{1}{2}||\mathbf{b} - \mathbf{u}||_2^2 \ \text{ s.t. } \ ||\mathbf{u}||_1 + \frac{\gamma}{2}||\mathbf{u}||_2^2 \leq \tau. \tag{18}$$

To solve efficiently the case $\gamma > 0$, we propose Algorithm 3, which extends Maculan and de Paula (1989) and Duchi et al. (2008), and the following lemma which shows that it solves our problem.

Lemma 9 [Projection onto the elastic-net constraint set].
For $\mathbf{b}$ in $\mathbb{R}^m$, $\gamma \geq 0$ and $\tau > 0$, Algorithm 3 solves Eq. (18).

Proof First, if $\mathbf{b}$ is a feasible point of (18), then $\mathbf{b}$ is a solution. We suppose therefore that it is not the case—that is,$||\mathbf{b}||_1 + \frac{\gamma}{2}||\mathbf{b}||_2^2 > \tau$. Let us define the Lagrangian of (18)

$$\mathcal{L}(\mathbf{u},\lambda) = \frac{1}{2}||\mathbf{b}-\mathbf{u}||_2^2 + \lambda\big(||\mathbf{u}||_1 + \frac{\gamma}{2}||\mathbf{u}||_2^2 - \tau\big).$$

For a fixed λ, minimizing the Lagrangian with respect to $\mathbf{u}$ admits a closed-form solution $\mathbf{u}^\star(\lambda)$, and a simple calculation shows that, for all j,

$$\mathbf{u}^\star(\lambda)[j] = \frac{\text{sign}(\mathbf{b}[j])(|\mathbf{b}[j]| - \lambda)^+}{1+\lambda\gamma}.$$

Eq. (18) is a convex optimization problem. Since Slater's conditions are verified and strong duality holds, it is equivalent to the dual problem

$$\max_{\lambda \geq 0} \mathcal{L}(\mathbf{u}^\star(\lambda),\lambda).$$

Since $\lambda = 0$ is not a solution, denoting by $\lambda^\star$ the solution, the complementary slackness condition implies that

$$||\mathbf{u}^\star(\lambda^\star)||_1 + \frac{\gamma}{2}||\mathbf{u}^\star(\lambda^\star)||_2^2 = \tau. \tag{19}$$

Using the closed form of $\mathbf{u}^\star(\lambda)$ is possible to show that the function $\lambda \rightarrow ||\mathbf{u}^\star(\lambda)||_1 + \frac{\gamma}{2}||\mathbf{u}^\star(\lambda)||_2^2$, is strictly decreasing with λ and thus Eq. (19) is a necessary and sufficient condition of optimality for λ. After a short calculation, one can show that this optimality condition is equivalent to

$$\frac{1}{(1+\lambda\gamma)^2}\sum_{j \in S(\lambda)}\Big(|\mathbf{b}[j]| + \frac{\gamma}{2}|\mathbf{b}[j]|^2 - \lambda\big(1+\frac{\gamma\lambda}{2}\big)\Big) = \tau,$$

where $S(\lambda) = \{j \text{ s.t. } |\mathbf{b}[j]| \geq \lambda\}$. Suppose that $S(\lambda^\star)$ is known, then $\lambda^\star$ can be computed in closed-form. To find $S(\lambda^\star)$, it is then sufficient to find the index k such that $S(\lambda^\star) = S(|\mathbf{b}[k]|)$, which is the solution of

$$\max_{k \in \{1,\ldots,m\}} |\mathbf{b}[k]| \text{ s.t. } \frac{1}{(1+|\mathbf{b}[k]|\gamma)^2}\sum_{j \in S(|\mathbf{b}[k]|)}\Big(|\mathbf{b}[j]| + \frac{\gamma}{2}|\mathbf{b}[j]|^2 - |\mathbf{b}[k]|\big(1+\frac{\gamma|\mathbf{b}[k]|}{2}\big)\Big) < \tau.$$

Lines 4 to 14 of Algorithm 3 are a modification of Duchi et al. (2008) to address this problem. A similar proof as Duchi et al. (2008) shows the convergence to the solution of this optimization problem in $O(m)$ in the average case, and lines 15 to 18 of Algorithm 3) compute $\lambda^\star$ after that $S(\lambda^\star)$ has been identified. Note that setting γ to 0 leads exactly to the algorithm of Duchi et al. (2008). ■

As for the dictionary learning problem, a simple modification to Algorithm 3 allows us to handle the non-negative case, replacing the scalars $|\mathbf{b}[j]|$ by $\max(\mathbf{b}[j],0)$ in the algorithm.

B.2 A Homotopy Method for Solving the Fused Lasso Signal Approximation

Let $\mathbf{b}$ be a vector of $\mathbb{R}^m$. We define, following Friedman et al. (2007), the fused lasso signal approximation problem $\mathcal{P}(\gamma_1,\gamma_2,\gamma_3)$:

$$\min_{\mathbf{u} \in \mathbb{R}^m} \frac{1}{2}||\mathbf{b}-\mathbf{u}||_2^2 + \gamma_1||\mathbf{u}||_1 + \gamma_2 \text{FL}(\mathbf{u}) + \frac{\gamma_3}{2}||\mathbf{u}||_2^2, \tag{20}$$

Algorithm 3 Efficient projection on the elastic-net constraint.

Require: $\tau \in \mathbb{R}$; $\gamma \in \mathbb{R}$; $\mathbf{b} \in \mathbb{R}^m$;

if $||\mathbf{b}||_1 + \frac{\gamma}{2}||\mathbf{b}||_2^2 \leq \tau$ **then**
  Return $\mathbf{u} \leftarrow \mathbf{b}$.
else
  $U \leftarrow \{1,\ldots,m\}$; $s \leftarrow 0$; $\rho \leftarrow 0$.
  while $U \neq \emptyset$ **do**
    Pick $k \in U$ at random.
    Partition U:

$$G = \{j \in U \text{ s.t. } |\mathbf{b}[j]| \geq |\mathbf{b}[k]|\},$$
$$L = \{j \in U \text{ s.t. } |\mathbf{b}[j]| < |\mathbf{b}[k]|\}.$$

    $\Delta\rho \leftarrow |G|$; $\Delta s \leftarrow \sum_{j \in G} |\mathbf{b}[j]| + \frac{\gamma}{2}|\mathbf{b}[j]|^2$.
    if $s + \Delta s - (\rho + \Delta\rho)(1 + \frac{\gamma}{2}|\mathbf{b}[k]|)|\mathbf{b}[k]| < \tau(1 + \gamma|\mathbf{b}[k]|)^2$ **then**
      $s \leftarrow s + \Delta s$; $\rho \leftarrow \Delta\rho$; $U \leftarrow L$.
    else
      $U \leftarrow G \setminus \{k\}$.
    end if
  end while
  $a \leftarrow \gamma^2\tau + \frac{\gamma}{2}\rho$,
  $b \leftarrow 2\gamma\tau + \rho$,
  $c \leftarrow \tau - s$,
  $\lambda \leftarrow \frac{-b+\sqrt{b^2-4ac}}{2a}$

$$\forall j = 1,\ldots,n, \mathbf{u}[j] \leftarrow \frac{\text{sign}(\mathbf{b}[j])(|\mathbf{b}[j]| - \lambda)^+}{1 + \lambda\gamma}$$

  Return $\mathbf{u}$.
end if

the only difference with Friedman et al. (2007) being the addition of the last quadratic term. The method we propose to this problem is a homotopy, which solves $\mathcal{P}(\tau\gamma_1, \tau\gamma_2, \tau\gamma_3)$ for all possible values of τ. In particular, for all ε, it provides the solution of the constrained problem

$$\min_{\mathbf{u} \in \mathbb{R}^m} \frac{1}{2}||\mathbf{b} - \mathbf{u}||_2^2 \text{ s.t. } \gamma_1||\mathbf{u}||_1 + \gamma_2 \text{FL}(\mathbf{u}) + \frac{\gamma_3}{2}||\mathbf{u}||_2^2 \leq \varepsilon. \tag{21}$$

The algorithm relies on the following lemma

Lemma 10 *Let* $\mathbf{u}^\star(\gamma_1,\gamma_2,\gamma_3)$ *be the solution of Eq. (20), for specific values of* $\gamma_1,\gamma_2,\gamma_3$. *Then*

- $\mathbf{u}^\star(\gamma_1,\gamma_2,\gamma_3) = \frac{1}{1+\gamma_3}\mathbf{u}^\star(\gamma_1,\gamma_2,0)$.
- *For all* i, $\mathbf{u}^\star(\gamma_1,\gamma_2,0)[i] = \text{sign}(\mathbf{u}^\star(0,\gamma_2,0)[i])\max(|\mathbf{u}^\star(0,\gamma_2,0)[i]| - \lambda_1, 0)$ *—that is,* $\mathbf{u}^\star(\gamma_1,\gamma_2,0)$ *can be obtained by soft thresholding of* $\mathbf{u}^\star(0,\gamma_2,0)$.

The first point can be shown by short calculation. The second one is proven in Friedman et al. (2007) by considering subgradient optimality conditions. This lemma shows that if one knows the solution of $\mathcal{P}(0,\gamma_2,0)$, then $\mathcal{P}(\gamma_1,\gamma_2,\gamma_3)$ can be obtained in linear time.

It is therefore natural to consider the simplified problem

$$\min_{\mathbf{u}\in\mathbb{R}^m} \frac{1}{2}||\mathbf{b}-\mathbf{u}||_2^2+\gamma_2 \mathrm{FL}(\mathbf{u}). \tag{22}$$

With the change of variable $\mathbf{v}[1]=\mathbf{u}[1]$ and $\mathbf{v}[i]=\mathbf{u}[i]-\mathbf{u}[i-1]$ for $i>1$, this problem can be recast as a weighted Lasso

$$\min_{\mathbf{v}\in\mathbb{R}^m} \frac{1}{2}||\mathbf{b}-\mathbf{D}\mathbf{v}||_2^2+\sum_{i=1}^{m} w_i|\mathbf{v}[i]|, \tag{23}$$

where $w_1=0$ and $w_i=\gamma_2$ for $i>1$, and $\mathbf{D}[i,j]=1$ if $i\geq j$ and 0 otherwise. We propose to use LARS (Efron et al., 2004) and exploit the specific structure of the matrix $\mathbf{D}$ to make this approach efficient, by noticing that:

- For a vector $\mathbf{w}$ in $\mathbb{R}^m$, computing $\mathbf{e}=\mathbf{D}\mathbf{w}$ requires $O(m)$ operations instead of $O(m^2)$, by using the recursive formula $\mathbf{e}[1]=\mathbf{w}[1]$, $\mathbf{e}[i+1]=\mathbf{w}[i]+\mathbf{e}[i]$.
- For a vector $\mathbf{w}$ in $\mathbb{R}^n$, computing $\mathbf{e}=\mathbf{D}^T\mathbf{w}$ requires $O(m)$ operations instead of $O(m^2)$, by using the recursive formula $\mathbf{e}[n]=\mathbf{w}[n]$, $\mathbf{e}[i-1]=\mathbf{w}[i-1]+\mathbf{e}[i]$.
- Let $\Gamma=\{a_1,\ldots,a_p\}$ be an active set and suppose $a_1<\ldots<a_p$. Then $(\mathbf{D}_\Gamma^T\mathbf{D}_\Gamma)^{-1}$ admits the closed form value

$$(\mathbf{D}_\Gamma^T\mathbf{D}_\Gamma)^{-1}=\begin{pmatrix} c_1 & -c_1 & 0 & \ldots & 0 & 0\\ -c_1 & c_1+c_2 & -c_2 & \ldots & 0 & 0\\ 0 & -c_2 & c_2+c_3 & \ldots & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 0 & 0 & \ldots & c_{p-2}+c_{p-1} & -c_{p-1}\\ 0 & 0 & 0 & \ldots & -c_{p-1} & c_{p-1}+c_p \end{pmatrix},$$

  where $c_p=\frac{1}{n+1-a_p}$ and $c_i=\frac{1}{a_{i+1}-a_i}$ for $i<p$.

This allows the implementation of this homotopy method without using matrix inversion or Cholesky factorization, solving Eq. (23) in $O(ms)$ operations, where s is the number of non-zero values of the optimal solution $\mathbf{v}$.[11]

Adapting this method for solving Eq. (21) requires following the regularization path of the problems $\mathcal{P}(0,\tau\gamma_2,0)$ for all values of τ, which provides as well the regularization path of the problem $\mathcal{P}(\tau\lambda_1,\tau\lambda_2,\tau\lambda_3)$ and stops whenever the constraint becomes unsatisfied. This procedure still requires $O(ms)$ operations.

Note that in the case $\gamma_1=0$ and $\gamma_3=0$, when only the fused-lasso term is present in Eq (20), the same approach has been proposed in a previous work by Harchaoui and Lévy-Leduc (2008), and Harchaoui (2008) to solve Eq. (22), with the same tricks for improving the efficiency of the procedure.

11. To be more precise, s is the number of kinks of the regularization path. In practice, s is roughly the same as the number of non-zero values of the optimal solution $\mathbf{v}$.

References

M. Aharon and M. Elad. Sparse and redundant modeling of image content using an image-signature-dictionary. *SIAM Journal on Imaging Sciences*, 1(3):228–247, July 2008.

M. Aharon, M. Elad, and A. M. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Transactions on Signal Processing*, 54(11): 4311–4322, November 2006.

F. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1224, 2008.

F. Bach, J. Mairal, and J. Ponce. Convex sparse matrix factorizations. Technical report, 2008. Preprint arXiv:0812.1869.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific Belmont, 1999.

P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of statistics*, 37(4):1705–1732, 2009.

J. F. Bonnans and A. Shapiro. Optimization problems with perturbations: A guided tour. *SIAM Review*, 40(2):202–227, 1998.

J. F. Bonnans and A. Shapiro. *Perturbation Analysis of Optimization Problems*. Springer, 2000.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Springer, 2006.

L. Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. 1998.

L. Bottou and O. Bousquet. The trade-offs of large scale learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. MIT Press, 2008.

D. M. Bradley and J. A. Bagnell. Differentiable sparse coding. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 113–120. 2009.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1999.

K. Chin, S. DeVries, J. Fridlyand, P.T. Spellman, R. Roydasgupta, W. L. Kuo, A. Lapuk, R. M. Neve, Z. Qian, T. Ryder, et al. Genomic and transcriptional aberrations linked to breast cancer pathophysiologies. *Cancer Cell*, 10(6):529–541, 2006.

S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Transactions on Signal Processing*, 53(7):2477–2488, 2005.

J. M. Danskin. The theory of max-min, and its application to weapons allocation problems. *Ökonometrie und Unternehmensforschung*, 1967.

A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

A. d'Aspremont, F. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–499, 2004.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 54(12):3736–3745, December 2006.

K. Engan, S. O. Aase, and J. H. Husoy. Frame based signal compression using method of optimal directions (MOD). In *Proceedings of the 1999 IEEE International Symposium on Circuits Systems*, volume 4, 1999.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results, 2006.

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.

D. L. Fisk. Quasi-martingales. *Transactions of the American Mathematical Society*, 120(3):359–388, 1965.

J. Friedman, T. Hastie, H. Hölfling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

W. J. Fu. Penalized regressions: The bridge versus the Lasso. *Journal of Computational and Graphical Statistics*, 7:397–416, 1998.

J. J. Fuchs. Recovery of exact sparse representations in the presence of bounded noise. *IEEE Transactions on Information Theory*, 51(10):3601–3608, 2005.

A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

G. H. Golub and C. F. Van Loan. *Matrix computations*. John Hopkins University Press, 1996.

R. Grosse, R. Raina, H. Kwong, and A. Y. Ng. Shift-invariant sparse coding for audio classification. In *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

Z. Harchaoui. *Méthodes à Noyaux pour la Détection*. PhD thesis, Télécom ParisTech, 2008.

Z. Harchaoui and C. Lévy-Leduc. Catching change-points with Lasso. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. MIT Press, 2008.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

P. O. Hoyer. Non-negative sparse coding. In *Proc. IEEE Workshop on Neural Networks for Signal Processing*, 2002.

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

L. Jacob, G. Obozinski, and J.-P. Vert. Group Lasso with overlap and graph Lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

R. Jenatton, J-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, 2009a. Preprint arXiv:0904.3523v1.

R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. Technical report, 2009b. Preprint arXiv:0909.1440v1.

I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the Lasso. *Journal of Computational and Graphical Statistics*, 12(3):531–547, 2003.

K. Kavukcuoglu, M. Ranzato, and Y. LeCun. Fast inference in sparse coding algorithms with applications to object recognition. Technical report, Computational and Biological Learning Lab, Courant Institute, NYU, 2008.

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

H. J. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, pages 556–562, 2001.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 801–808. MIT Press, 2007.

K. C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):684–698, 2005.

M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural Computation*, 12(2):337–365, 2000.

C.J. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 19 (10):2756–2779, 2007.

N. Maculan and J. R. G. Galdino de Paula. A linear-time median-finding algorithm for projecting a vector on the simplex of Rn. *Operations Research Letters*, 8(4):219–222, 1989.

J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics, revised edition*. John Wiley, Chichester, 1999.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008a.

J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, 17(1):53–69, January 2008b.

J. Mairal, G. Sapiro, and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modelling and Simulation*, 7(1):214–241, April 2008c.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009a.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 1033–1040. MIT Press, 2009b.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009c.

S. Mallat. *A Wavelet Tour of Signal Processing, Second Edition*. Academic Press, New York, September 1999.

M. Métivier. *Semi-martingales*. Walter de Gruyter, 1983.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models*, 89:355–368, 1998.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, 2007.

G. Obozinski, M. J. Wainwright, and M. I. Jordan. Union support recovery in high-dimensional multivariate regression. *UC Berkeley Technical Report 761*, August 2008.

G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 2009. Published online.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

M. R. Osborne, B. Presnell, and B. A. Turlach. A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20(3):389–403, 2000.

G. Peyré. Sparse modeling of textures. *Journal of Mathematical Imaging and Vision*, 34(1):17–31, May 2009.

M. Protter and M. Elad. Image sequence denoising via sparse and redundant representations. *IEEE Transactions on Image Processing*, 18(1):27–36, 2009.

R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2007.

V. Roth and B. Fischer. The Group-Lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *22nd Annual Conference on Learning Theory (COLT)*, 2009.

K.-K. Sung. *Learning and Example Selection for Object and Pattern Recognition*. PhD thesis, MIT, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, 1996.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B*, 58(1):267–288, 1996.

R. Tibshirani and P. Wang. Spatial smoothing and hot spot detection for CGH data using the fused Lasso. *Biostatistics*, 9(1):18–29, 2008.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, 67(1):91–108, 2005.

J. A. Tropp. Algorithms for simultaneous sparse approximation. part ii: Convex relaxation. *Signal Processing, Special Issue "Sparse Approximations in Signal and Image Processing"*, 86:589–602, April 2006.

J. A. Tropp, A. C. Gilbert, and M. J. Strauss. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing, Special Issue "Sparse Approximations in Signal and Image Processing"*, 86:572–588, April 2006.

B. A. Turlach, W. N. Venables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005.

A. W. Van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

T. T. Wu and K. Lange. Coordinate descent algorithms for Lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B*, 68:49–67, 2006.

R. Zass and A. Shashua. Nonnegative sparse PCA. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 1561–1568. MIT Press, 2007.

H. H. Zhang, Y. Liu, Y. Wu, and J. Zhu. Selection for the multicategory svm via adaptive sup-norm regularization. *Electronic Journal of Statistics*, 2:149–167, 2008.

M. Zibulevsky and B. A. Pearlmutter. Blind source separation by sparse decomposition in a signal dictionary. *Neural Computation*, 13(4):863–882, 2001.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.

Journal of Machine Learning Research 11 (2010) 61-87 Submitted 11/09; Published 1/10

Model Selection: Beyond the Bayesian/Frequentist Divide

Isabelle Guyon GUYON@CLOPINET.COM
ClopiNet
955 Creston Road
Berkeley, CA 94708, USA

Amir Saffari SAFFARI@ICG.TUGRAZ.AT
Institute for Computer Graphics and Vision
Graz University of Technology
Inffeldgasse 16
A-8010 Graz, Austria

Gideon Dror GIDEON@MTA.AC.IL
The Academic College of Tel-Aviv-Yaffo
2 Rabeinu Yerucham St., Jaffa
Tel-Aviv 61083, Israel

Gavin Cawley GCC@CMP.UEA.AC.UK
School of Computing Sciences
University of East Anglia
Norwich, NR4 7TJ, U.K.

Editor: Lawrence Saul

Abstract

The principle of parsimony also known as "Ockham's razor" has inspired many theories of model selection. Yet such theories, all making arguments in favor of parsimony, are based on very different premises and have developed distinct methodologies to derive algorithms. We have organized challenges and edited a special issue of JMLR and several conference proceedings around the theme of model selection. In this editorial, we revisit the problem of avoiding overfitting in light of the latest results. We note the remarkable convergence of theories as different as Bayesian theory, Minimum Description Length, bias/variance tradeoff, Structural Risk Minimization, and regularization, in some approaches. We also present new and interesting examples of the complementarity of theories leading to hybrid algorithms, neither frequentist, nor Bayesian, or perhaps both frequentist and Bayesian!

Keywords: model selection, ensemble methods, multilevel inference, multilevel optimization, performance prediction, bias-variance tradeoff, Bayesian priors, structural risk minimization, guaranteed risk minimization, over-fitting, regularization, minimum description length

1. Introduction

The problem of learning is often decomposed into the tasks of fitting parameters to some training data, and then selecting the best model using heuristic or principled methods, collectively referred to as *model selection* methods. Model selection methods range from simple yet powerful cross-validation based methods to the optimization of cost functions penalized for model complexity, derived from performance bounds or Bayesian priors.

This paper is not intended as a general review of the state-of-the-art in model selection nor a tutorial; instead it is a synthesis of the collection of papers that we have assembled. It also provides a unifying perspective on Bayesian and frequentist methodologies used in various model selection methods. We highlight a new trend in research on model selection that blends these approaches.

The reader is expected to have some basic knowledge of familiar learning machines (linear models, neural networks, tree classifiers and kernel methods) and elementary notions of learning theory (bias/variance tradeoff, model capacity or complexity, performance bounds). Novice readers are directed to the companion paper (Guyon, 2009), which reviews basic learning machines, common model selection techniques, and provides elements of learning theory.

When we started organizing workshops and competitions around the problem of model selection (of which this collection of papers is the product), both theoreticians and practitioners welcomed us with some scepticism; model selection being often viewed as somewhat "old hat". Some think that the problem is solved, others that it is not a problem at all! For Bayesian theoreticians, the problem of model selection is circumvented by averaging all models over the posterior distribution. For risk minimization theoreticians (called "frequentists" by the Bayesians) the problem is solved by minimizing performance bounds. For practitioners, the problem is solved using cross-validation. However, looking more closely, most theoretically grounded methods of solving or circumventing model selection have at least one hyper-parameter left somewhere, which ends up being optimized by cross-validation. Cross-validation seems to be the universally accepted ultimate remedy. But it has its dark sides: (a) there is no consensus on how to choose the fraction of examples reserved training and for validation; (b) the overall learning problem may be prone to over-fitting the cross-validation error (Cawley and Talbot, 2009). Therefore, from our point of view, the problem of optimally dividing the learning problem into multiple levels of inference and optimally allocating training data to these various levels remains unsolved, motivating our efforts. From the novel contributions we have gathered, we are pleased to see that researchers are going beyond the usual Bayesian/frequentist divide to provide new creative solutions to those problems: we see the emergence of multi-level optimization methods, which are both Bayesian and frequentist. How can that be? Read on!

After explaining in Section 2 our notational conventions, we briefly review a range of different Bayesian and frequentist approaches to model selection in Section 3, which we then unify in Section 4 under the framework of multi-level optimization. Section 5 then presents the advances made by the authors of papers that we have edited. In Section 6, we open a discussion on advanced topics and open problems. To facilitate reading, a glossary is appended; throughout the paper, words found in the glossary are indicated in boldface.

2. Notations and Conventions

In its broadest sense, *model selection* designates an ensemble of techniques used to select a model, that best explains some data or phenomena, or best predicts future data, observations or the consequences of actions. This broad definition encompasses both scientific and statistical modeling. In this paper, we address only the problem of statistical modeling and are mostly concerned with **supervised learning** from independently and identically distributed (*i.i.d.*) data. Extensions to **unsupervised learning** and non *i.i.d.* cases will be discussed in Section 6.

The goal of supervised learning is to predict a target variable $y \in \mathcal{Y}$, which may be continuous (regression) or categorical or binary (classification). The predictions are made using observations

$\mathbf{x}$ from a domain $\mathcal{X}$, often a vectorial space of dimension n, the number of features. The data pairs $\{\mathbf{x}, y\}$ are independently and identically distributed according to an unknown (but fixed) distribution $P(\mathbf{x}, y)$. A number m of pairs drawn from that distribution are given, forming the training data $D = \{(\mathbf{x}_k, y_k), k = 1, ...m\}$. We will denote by $X = [x_{ki}]$, $k = 1, ...m, i = 1, ...n$, the matrix of dimensions (m, n) whose rows are the training patterns and whose columns are the features. Finally, we denote by $\mathbf{y}$ the column vector of dimensions $(m, 1)$ containing the target values y_k.

There are several formulations of the supervised learning problem:

- **Function approximation** (induction) methods seek a function f (called *model* or *learning machine*) belonging to a model class $\mathcal{F}$, which minimizes a specified risk functional (or maximizes a certain utility). The goal is to minimize an *expected risk* $R[f] = \int \mathcal{L}(f(\mathbf{x}), y)\, dP(\mathbf{x}, y)$, also called *generalization error*, where $\mathcal{L}(f(\mathbf{x}), y)$ is a loss function (often a negative log likelihood) measuring the discrepancy between $f(\mathbf{x})$ and y. Since $P(\mathbf{x}, y)$ is unknown, only estimates of $R[f]$ can be computed, which we call *evaluation functions* or *estimators*. Function approximation methods differ in the choice of evaluation function and optimization algorithm and include **risk minimization**, **PAC leaning**, **maximum likelihood optimization**, and **MAP learning**.

- **Bayesian and ensemble methods** make predictions according to model averages that are convex combinations of models $f \in \mathcal{F}$, that is, which belong to the convex closure of the model class $\mathcal{F}^*$. Such methods differ in the type of model averaging performed. **Bayesian learning** methods approximate $E_f(y|\mathbf{x}) = \int_{f \in \mathcal{F}} f(\mathbf{x})\, dP(f)$, an expectation taken over a class of models $\mathcal{F}$, using an unknown probability distribution $P(f)$ over the models. Starting from a "prior", our knowledge of this distribution is refined into a "posterior" when we see some data. **Bagging** ensemble methods approximate $E_D(f(\mathbf{x}, D))$, where $f(\mathbf{x}, D)$ is a function from the model class $\mathcal{F}$, trained with m examples and $E_D(\cdot)$ is the mathematical expectation over all training sets of size m. The key point in these methods is to generate a diverse set of functions, each providing a different perspective over the problem at hand, the ensemble thus forming a concensus view.

- **Transduction** methods make direct predictions of y given $\mathbf{x}$ and X, bypassing the modeling step. We do not address such methods in this paper.

The desired properties of the chosen predictor include: good generalization performance, fast training/prediction, and ease of interpretation of the predictions. Even though all of these aspects are important in practice, we will essentially focus on the first aspect: obtaining the best possible generalization performance. Some of the other aspects of model selection will be discussed in Section 6.

The parametrization of f differentiates the problem of *model selection* from the general machine learning problem. Instead of parameterizing f with one set of parameters, the model selection framework distinguishes between *parameters* and *hyper-parameters*. We adopt the simplified notation $f(\mathbf{x}; \boldsymbol{\alpha}, \boldsymbol{\theta})$ for a model of parameters $\boldsymbol{\alpha}$ and hyper-parameters $\boldsymbol{\theta}$. It should be understood that different models may be parameterized differently. Hence by $f(\mathbf{x}; \boldsymbol{\alpha}\boldsymbol{\theta})$ we really mean $f(\mathbf{x}; \boldsymbol{\alpha}(\boldsymbol{\theta}), \boldsymbol{\theta})$ or $f_{\boldsymbol{\theta}}(\mathbf{x}; \boldsymbol{\alpha})$. For instance, for a linear model $f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x}$, $\boldsymbol{\alpha} = \mathbf{w}$; for a kernel method $f(\mathbf{x}, \boldsymbol{\alpha}) = \sum_k \alpha_k K(\mathbf{x}, \mathbf{x}_k)$, $\boldsymbol{\alpha} = [\alpha_k]$. The hyper-parameters may include indicators of presence or absence of features, choice of preprocessing methods,, choice of algorithm or model sub-class (e.g., linear models, neural networks, kernel methods, etc.), algorithm or model sub-class

parameters (e.g., number of layers and units per layer in a neural network, maximum degree of a polynomial, bandwidth of a kernel), choice of post-processing, etc. We also refer to the parameters of the prior $P(f)$ in **Bayesian/MAP learning** and the parameters of the **regularizer** $\Omega[f]$ in **risk minimization** as hyper-parameters even if the resulting predictor is not an explicit function of those parameters, because they are used in the process of learning. In what follows, we relate the problem of model selection to that of *hyper-parameter selection*, taken in is broadest sense and encompassing all the cases mentioned above.

We refer to the adjustment of the model parameters $\boldsymbol{\alpha}$ as the *first level of inference*. When data are split in several subsets for the purpose of training and evaluating models, we call m_{tr} the number of *training examples* used to adjust $\boldsymbol{\alpha}$. If the hyper-parameters $\boldsymbol{\theta}$ are adjusted from a subset of data of size m_{va}, we call the examples used to adjust them at this *second level of inference* the *"validation sample"*. Finally we call m_{te} the number of test examples used to evaluate the final model. The corresponding empirical estimates of the *expected risk* $R[f]$, denoted $R_{tr}[f]$, $R_{va}[f]$, and $R_{te}[f]$, will be called respectively *training error*, *validation error*, and *test error*.

3. The Many Faces of Model Selection

In this section, we track *model selection* from various angles to finally reduce it to the unified view of *multilevel inference*.

3.1 Is Model Selection "Really" a Problem?

It is legitimate to first question whether the distinction between parameters and hyper-parameters is relevant. Splitting the learning problem into two levels of inference may be convenient for conducting experiments. For example, combinations of preprocessing, feature selection, and post-processing are easily performed by fixing $\boldsymbol{\theta}$ and training $\boldsymbol{\alpha}$ with off-the-shelf programs. But, the distinction between parameters and hyper-parameters is more fundamental. For instance, in the model class of kernel methods $f(\mathbf{x}) = \sum_k \boldsymbol{\alpha}_k K(\mathbf{x}, \mathbf{x}_k; \boldsymbol{\theta})$, why couldn't we treat both $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ as regular parameters?

One common argument is that, for fixed values of $\boldsymbol{\theta}$, the problem of learning $\boldsymbol{\alpha}$ can be formulated as a convex optimization problem, with a single unique solution, for which powerful mathematical programming packages are available, while the overall optimization of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ in non-convex. Another compelling argument is that, splitting the learning problem into several levels might also benefit to the performance of the learning machine by "alleviating" (but not eliminating) the problem of **over-fitting**. Consider for example the Gaussian redial basis function kernel $K(\mathbf{x}, \mathbf{x}_k; \theta) = \exp(-\|\mathbf{x} - \mathbf{x}_k\|^2/\theta^2)$. The function $f(\mathbf{x}) = \sum_{k=1}^{m} \alpha_k K(\mathbf{x}, \mathbf{x}_k; \theta)$ is a universal approximator if θ is let to vary and if the sum runs over the training examples. If both $\boldsymbol{\alpha}$ and θ are optimized simultaneously, solutions with a small value of θ^2 might be picked, having zero training error but possibly very poor generalization performance. The model class $\mathcal{F}$ to which f belongs has infinite capacity $C(\mathcal{F})$. In contrast, for a fixed value of the hyper-parameter θ^o, the model $f(\mathbf{x}) = \sum_{k=1}^{m} \boldsymbol{\alpha}_k K(\mathbf{x}, \mathbf{x}_k; \theta^o)$ is linear in its parameters $\boldsymbol{\alpha}_k$ and has a finite capacity, bounded by m. In addition, the capacity of $f(\mathbf{x}) = \sum_{k=1}^{m} \boldsymbol{\alpha}_k^o K(\mathbf{x}, \mathbf{x}_k^o; \theta)$ of parameter θ for fixed values $\boldsymbol{\alpha}_k^o$ and x_k^o is very low (to see that, note that very few examples can be learned without error by just varying the kernel width, given fixed vectors x_k^o and fixed parameters $\boldsymbol{\alpha}_k^o$). Hence, *using multiple levels of inference may reduce over-fitting, while still searching for solutions in a model class of universal approximators*.

This last idea has been taken one step further in the method of *structural risk minimization* (Vapnik, 1979), by introducing new hyper-parameters in learning problems, which initially did not have any. Consider for instance the class of linear models $f(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i$. It is possible to introduce hyper-parameters by imposing a structure in parameter space. A classical example is the structure $\|\mathbf{w}\|^2 \leq A$, where $\|\mathbf{w}\|$ denotes the Euclidean norm and A is a positive hyper-parameter. For increasing values of A the space of parameters is organized in nested subsets. Vapnik (1998) proves for Support Vector Machines (SVM) and Bartlett (1997) for neural networks that tighter performance bounds are obtained by increasing A. The newly introduced parameter allows us to monitor the **bias/variance tradeoff**. Using a Lagrange multiplier, the problem may be replaced by that of minimizing a regularized risk functional $R_{reg} = R_{tr} + \gamma\|\mathbf{w}\|^2$, $\gamma > 0$, where the training loss function is the so-called "hinge loss" (see *e.g.,* Hastie et al., 2000). The same regularizer $\|\mathbf{w}\|^2$ is used in ridge regression (Hoerl, 1962), "weight decay" neural networks (Werbos, 1988), regularized radial-basis function networks (Poggio and Girosi, 1990), Gaussian processes (MacKay, 1992), together with the square loss function. Instead of the Euclidean norm or 2-norm, the 1-norm regularizer $\|\mathbf{w}\|_1 = \sum_i |w_i|$ is used in LASSO (Tibshirani, 1994) and 1-norm versions of SVMs (see *e.g.,* Zhu et al., 2003), logistic regression (Friedman et al., 2009), and Boosting (Rosset et al., 2004). Weston et al. (2003) have proposed a 0-norm regularizer $\|\mathbf{w}\|_0 = \sum_i \mathbf{1}(w_i)$, where $\mathbf{1}(x) = 1$, if $x \neq 0$ and 0 otherwise.

Interestingly, each method stems from a different theoretical justification (some are Bayesian, some are frequentist and some a a little bit of both like PAC-Bayesian bounds, see, for example, Seeger, 2003, for a review), showing a beautiful example of theory convergence (Guyon, 2009). Either way, for a fixed value of the hyper-parameter A or γ the complexity of the learning problem is lower than that of the original problem. We can optimize A or γ at a second level of inference, for instance by cross-validation.

3.2 Bayesian Model Selection

In the Bayesian framework, there is no model selection *per se*, since learning does not involve searching for an optimum function, but averaging over a posterior distribution. For example, if the model class $\mathcal{F}$ consists of models $f(\mathbf{x};\boldsymbol{\alpha},\boldsymbol{\theta})$, the Bayesian assumption is that the parameters $\boldsymbol{\alpha}$ and hyper-parameters $\boldsymbol{\theta}$ of the model used to generate the data are drawn from a prior $P(\boldsymbol{\alpha},\boldsymbol{\theta})$. After observing some data D the predictions should be made according to:

$$E_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y|\mathbf{x},D) = \int\int f(\mathbf{x};\boldsymbol{\alpha},\boldsymbol{\theta})\, P(\boldsymbol{\alpha},\boldsymbol{\theta}|D)\, d\boldsymbol{\alpha}\, d\boldsymbol{\theta}\ .$$

Hence there is no selection of a single model, but a summation over models in the model class $\mathcal{F}$, weighed by $P(\boldsymbol{\alpha},\boldsymbol{\theta}|D)$. The problem is to integrate over $P(\boldsymbol{\alpha},\boldsymbol{\theta}|D)$.[1] A two-level decomposition can be made by factorizing $P(\boldsymbol{\alpha},\boldsymbol{\theta}|D)$ as $P(\boldsymbol{\alpha},\boldsymbol{\theta}|D) = P(\boldsymbol{\alpha}|\boldsymbol{\theta},D)P(\boldsymbol{\theta}|D)$:

$$E_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y|\mathbf{x},D) = \int \left(\int f(\mathbf{x};\boldsymbol{\alpha},\boldsymbol{\theta}) P(\boldsymbol{\alpha}|\boldsymbol{\theta},D)\, d\boldsymbol{\alpha} \right) P(\boldsymbol{\theta}|D)\, d\boldsymbol{\theta}\ . \tag{1}$$

Bayesian model selection decomposes the prior $P(\boldsymbol{\alpha},\boldsymbol{\theta})$ into parameter prior $P(\boldsymbol{\alpha}|\boldsymbol{\theta})$ and a "hyper-prior" $P(\boldsymbol{\theta})$. In **MAP learning**, the type-II likelihood (also called the "evidence") $P(D|\boldsymbol{\theta}) =$

1. The calculation of the integral in closed form may be impossible to carry out; in this case, variational approximations are made or numerical simulations are performed, sampling from $P(\boldsymbol{\alpha},\boldsymbol{\theta}|D)$, and replacing the integral by the summation over a finite number of models.

$\sum_{\boldsymbol{\alpha}} P(D|\boldsymbol{\alpha},\boldsymbol{\theta})P(\boldsymbol{\alpha}|\boldsymbol{\theta})$ is maximized with respect to the hyper-parameters $\boldsymbol{\theta}$ (therefore assuming a flat prior for $\boldsymbol{\theta}$), while the "regular" parameters $\boldsymbol{\alpha}$ are obtained by maximizing the posterior $\boldsymbol{\alpha}^* = \text{argmax}_{\boldsymbol{\alpha}} P(\boldsymbol{\alpha}|\boldsymbol{\theta},D) = \text{argmax}_{\boldsymbol{\alpha}} P(D|\boldsymbol{\alpha},\boldsymbol{\theta})P(\boldsymbol{\alpha}|\boldsymbol{\theta})$.[2]

3.3 Frequentist Model Selection

While Bayesians view probabilities as being realized in the idea of "prior" and "posterior" knowledge of distributions, frequentists define probability in terms of *frequencies of occurrence of events*. In this section, the "frequentist" approach is equated with risk minimization.

There are obvious ties between the problem of *model selection* and that of *performance prediction*. *Performance prediction* is the problem of estimating the *expected risk* or *generalization error* $R[f]$. *Model selection* is the problem of adjusting the capacity or complexity of the models to the available amount of training data to avoid either **under-fitting** or **over-fitting**. Solving the *performance prediction* problem would also solve the *model selection* problem, but model selection is an easier problem. If we find an ordering index $r[f]$ such that for all pairs of functions $r[f_1] < r[f_2] \Rightarrow R[f_1] < R[f_2]$, then the index allows us to correctly carry out *model selection*. Theoretical performance bounds providing a *guaranteed risk* have been proposed as ranking indices (Vapnik, 1998). Arguably, the tightness of the bound is of secondary importance in obtaining a good ranking index. Bounds of the form $r[f] = R_{tr}[f] + \varepsilon(C/m_{tr})$, where C characterizes the capacity or complexity of the model class, penalizes complex models, but the penalty vanishes as $m_{tr} \to \infty$. Some learning algorithms, for example, SVMs (Boser et al., 1992) or boosting (Freund and Schapire, 1996), optimize a *guaranteed risk* rather than the *empirical risk* $R_{tr}[f]$, and therefore provide some guarantee of good *generalization*. Algorithms derived in this way have an embedded model selection mechanism. Other closely related penalty-based methods include Bayesian **MAP learning** and **regularization**.

Many models (and particularly *compound models* including feature selection, preprocessing, learning machine, and post-processing) are not associated with known performance bounds. Common practice among frequentists is to split available training data into m_{tr} training examples to adjust parameters and m_{va} validation examples to adjust hyper-parameters. In an effort to reduce variance, the validation error $R_{va}[f]$ may be averaged over many data splits, leading to a cross-validation (CV) estimator $R_{CV}[f]$. The most widely used CV method is *K-fold cross-validation*. It consists in partitioning training data into $K \simeq (m_{tr} + m_{va})/m_{va}$ disjoint subsets of roughly equal sizes (up to rounding errors), each corresponding to one validation set (the complement being used as training set). In stratified cross-validation, the class proportions of the full data sets are respected in all subsets. The variance of the results may be reduced by performing Q times K-fold cross-validation and averaging the results of the Q runs. Another popular method consists in holding out a single example at a time for validation purposes. The resulting cross-validation error is referred to as "leave-one-out" error $R_{LOO}[f]$. Some preliminary study design is necessary to determine the sufficient amount of test data to obtain a good estimate of the generalization error (Langford, 2005), the sufficient amount of training data to attain desired generalization performances, and an adequate split of the training data between training and validation set. See Guyon (2009) for a discussion of these issues.

2. In some Bayesian formulations of multi-layer Perceptrons, the evidence framework maximizes over $\boldsymbol{\theta}$ but marginalises over the weights, rather than maximizing, so in this case the MAP can apply to the parameters or the hyper-parameters or both.

4. Multi-level Inference: A Unifying View of Model Selection

What is common among the various views of model selection is the idea of multiple levels of inference, each level corresponding to one set of parameters or hyper-parameters. Consider a two-level case for a model class $f(\mathbf{x};\boldsymbol{\alpha},\boldsymbol{\theta})$ parameterized by one set of parameters $\boldsymbol{\alpha}$ and one set of hyper-parameters $\boldsymbol{\theta}$. From the frequentist (risk minimization) point of view, instead of jointly optimizing a risk functional with respect to all parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$, one creates a hierarchy of optimization problems:[3]

$$f^{**} = \text{argmin}_{\boldsymbol{\theta}} R_2[f^*, D] \ , \ \text{ such that } \ f^* = \text{argmin}_{\boldsymbol{\alpha}} R_1[f, D] \tag{2}$$

where R_1 and R_2 are first and second level risk functionals.

From the Bayesian point of view, the goal is to estimate the integral of Equation 1. There are striking similarities between the two approaches. To make the similarity more obvious, we can rewrite Equation 1 to make it look more like Equation 2, using the notation f^{**} for $E_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y|\mathbf{x},D)$:

$$f^{**} = \int f^* \ e^{-R_2} \ d\boldsymbol{\theta} \ , \ \text{ such that } \ f^* = \int f \ e^{-R_1} \ d\boldsymbol{\alpha} \tag{3}$$

where $R_1 = -\ln P(\boldsymbol{\alpha}|\boldsymbol{\theta},D)$ and $R_2 = -\ln P(\boldsymbol{\theta}|D)$. Note that in Bayesian multi-level inference f^* and f^{**} do not belong to $\mathcal{F}$ but to $\mathcal{F}^*$, the closure of $\mathcal{F}$ under convex combinations.

More generally, we define a *multi-level inference problem* as a learning problem organized into a hierarchy of learning problems. Formally, consider a machine learning toolkit which includes a choice of learning machines $\mathcal{A}[\mathcal{B},R]$, where $\mathcal{B}$ is a model space of functions $f(\mathbf{x};\boldsymbol{\theta})$, of parameters $\boldsymbol{\theta}$ and R is an evaluation function (e.g., a risk functional or a negative log posterior). We think of $\mathcal{A}[\mathcal{B},R]$ not as a procedure, but as an "object", in the sense of object oriented programming, equipped with a method "train", which processes data according to a training algorithm:[4]

$$f^{**} = \text{train}(\mathcal{A}[\mathcal{B},R_2],D); \tag{4}$$

This framework embodies the second level of inference of both Equations 2 and 3. The solution f^{**} belongs to $\mathcal{B}^*$, the convex closure of $\mathcal{B}$. To implement the first level of inference, we will consider that $\mathcal{B}$ is itself a learning machine and not just a model space. Its model space $\mathcal{F}$ includes functions $f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\alpha})$ of variable parameters $\boldsymbol{\alpha}$ ($\boldsymbol{\theta}$ is fixed), which are adjusted by the "train" method of $\mathcal{B}$:

$$f^* = \text{train}(\mathcal{B}[\mathcal{F},R_1],D); \tag{5}$$

The solution f^* belongs to $\mathcal{F}^*$, the convex closure of $\mathcal{F}$. The method "train" of $\mathcal{A}$ should call the method "train" of $\mathcal{B}$ as a subroutine, because of the nested nature of the learning problems of Equations 2 and 3. Notice that it is possible that different subsets of the data D are used at the different levels of inference.

We easily see two obvious extensions:

(i) *Multi-level inference:* Equation 4 and 5 are formally equivalent, so this formalism can be extended to more than two levels of inference.

3. It would be more correct if the argmin was assigned to parameters not functions, since the search domain is over parameters, and write $\boldsymbol{\theta}^{**} = \arg\min_{\boldsymbol{\theta}} R_2[f^*,D]$, such that $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} R_1[f,D], f^* = f(\mathbf{x},\boldsymbol{\alpha}^*)$, but we adopt a shorthand to emphasize the similarities between the frequentist and Bayesian approaches.

4. We adopt a Matlab-style notation: the first argument is the object of which the function is a method; the function "train" is overloaded, there is one for each algorithm. The notations are inspired and adapted from the conventions of the Spider package and the CLOP packages (Saffari and Guyon, 2006).

(ii) *Ensemble methods:* The method "train" returns either a single model or a linear combination of models, so the formalism can include all ensemble methods.

We propose in the next section a new classification of *multi-level inference* methods, orthogonal to the classical Bayesian versus frequentist divide, referring to the way in which data are processed rather than the means by which they are processed.

5. Advances in Multi-level Inference

We dedicate this section to reviewing the methods proposed in the collection of papers that we have edited. We categorize multi-level inference modules, each implementing one level of inference, into *filter*, *wrapper*, and *embedded methods*, borrowing from the conventional classification of feature selection methods (Kohavi and John, 1997; Blum and Langley, 1997; Guyon et al., 2006a). *Filters* are methods for narrowing down the model space, without training the learning machine. Such methods include preprocessing, feature construction, kernel design, architecture design, choice of prior or regularizers, choice of a noise model, and filter methods for feature selection. They constitute the highest level of inference[5]. *Wrapper methods* consider the learning machine as a black-box capable of learning from examples and making predictions once trained. They operate with a search algorithm in hyper-parameter space (for example grid search or stochastic search) and an evaluation function assessing the trained learning machine performances (for example the cross-validation error or the Bayesian evidence). They are the *middle-ware* of multi-level inference. *Embedded* methods are similar to wrappers, but they exploit the knowledge of the learning machine algorithm to make the search more efficient and eventually jointly optimize parameters and hyper-parameters, using multi-level optimization algorithms. They are usually used at the lowest level of inference.

5.1 Filters

Filter methods include a broad class of techniques aiming to reduce the model space $\mathcal{F}$ prior to training the learning machine. Such techniques may use "prior knowledge" or "domain knowledge", data from prior studies or from R&R (repeatability and reproducibility) studies, and even the training data themselves. But they do not produce the final model used to make predictions. Several examples of filter methods are found in the collection of papers we have edited:

Preprocessing and feature construction. An important part of machine learning is to find a good data representation, but choosing an appropriate data representation is very domain dependent. In benchmark experiments, it has often been found that generating a large number of low-level features yields better result than hand-crafting a few features incorporating a lot of expert knowledge (Guyon et al., 2007). The feature set can then be pruned by feature selection. In the challenges we have organized (Clopinet, 2004-2009) the data were generally already preprocessed to facilitate the work of the participants. However, additional normalizations, space dimensionality reduction and discretization were often performed by the participants. Of all space dimensionality reduction methods *Principal Component Analysis* (PCA) remains the most widely used. Several top-ranking participants to challenges we organized used PCA, including Neal and Zhang (2006), winners of the NIPS 2003 feature selection challenge, and Lutz (2006), winner of the WCCI 2006 performance prediction

5. Preprocessing is often thought of as a "low-level" operation. However, with respect to model selection, the selection of preprocessing happens generally in the "outer loop" of selection, hence it is at the highest level.

challenge. *Clustering* is also a popular preprocessing method of dimensionality reduction, championed by Saeed (2009) who used a Bernoulli mixture model as an input to an artificial neural network. In his paper on data grid models Boullé (2009) proposes a new method of *data discretization*. It can be used directly as part of a learning machine based on data grids (stepwise constant predictors) or as a preprocessing to other learning machines, such as the Naïve Bayes classifier. Of particular interest in this paper is the use of *data dependent priors*.

Designing kernels and model architectures. Special purpose neural network architectures implementing the idea of "weight sharing" such as Time Delay Neural Networks (Waibel, 1988) or two-dimensional convolutional networks (LeCun et al., 1989) have proved to be very effective in speech and image processing. More recently a wide variety of special purpose kernels have been proposed to incorporate domain knowledge in kernel learning algorithms. Examples include kernels invariant under various transforms (Simard et al., 1993; Pozdnoukhov and Bengio, 2006), string matching kernels (Watkins, 2000), and other sequence and tree kernels (Vishwanathan and Smola, 2003). Along these lines, in our collection of papers, Chloé Agathe Azencott and Pierre Baldi have proposed two-dimensional kernels for high-thoughput screening (Azencott and Baldi, 2009). Design effort has also be put into general purpose kernels. For instance, in the paper of Adankon and Cheriet (2009) , the SVM regularization hyper-parameter C (box-constraint) is incorporated in the kernel function. This facilitates the task of multi-level inference algorithms.

Defining regularizers or priors. Designing priors $P(f)$ or **regularizers** $\Omega[f]$ or structuring parameter space into parameters and several levels of hyper-parameters can also be thought of as a filter method. Most priors commonly used do not embed domain knowledge, they just enforce Ockham's razor by favoring simple (smooth) functions or eliminating irrelevant features. Priors are also often chosen out of convenience to facilitate the closed-form calculation of Bayesian integrals (for instance the use of so-called "conjugate priors", see *e.g.,* Neal and Zhang, 2006). The 2-norm regularizer $\Omega[f] = \|f\|^2_{\mathcal{H}}$ for kernel ridge regression, Support Vector Machines (SVM) and Least-Square Support Vector Machines (LSSVM) have been applied with success by many top-ranking participants of the challenges we organized. Gavin Cawley was co-winner of the WCCI 2006 performance prediction challenge using LSSVMs (Cawley, 2006). Another very successful regularizer is the **Automatic Relevance Determination** (ARD) prior. This regularizer was used in the winning entry of Radford Neal in the NIPS 2003 feature selection challenge (Neal and Zhang, 2006). Gavin Cawley also made top ranking reference entries in the IJCNN 2007 ALvsPK challenge (Cawley and Talbot, 2007b) using a similar ARD prior. For linear models, the 1-norm regularizer $\|\mathbf{w}\|$ is also popular (see *e.g.,* Pranckeviciene and Somorjai, 2009), but this has not been quite as successful in challenges as the 2-norm regularizer or the ARD prior.

Noise modeling. While the prior (or the regularizer) embeds our prior or domain knowledge of the model class, the likelihood (or the loss function) embeds our prior knowledge of the noise model on the predicted variable y. In regression, the square loss corresponds to Gaussian noise model, but other choices are possible. For instance, recently, Gavin Cawley and Nicola Talbot implemented Poisson regression for kernel machines (Cawley et al., 2007). For classification, the many loss functions proposed do not necessarily correspond to a noise model, they are often just bounding the 0/1 loss and are used for computational convenience. In the

Bayesian framework, an sigmoidal function is often used (like the logistic or probit functions) to map the output of a discriminant function $f(\mathbf{x}_k)$ to probabilities p_k. Assuming target values $y_k \in \{0,1\}$, the likelihood $\Pi_k p_k^{y_k}(1-p_k)^{1-y_k}$ corresponds to the cross-entropy cost function $\sum_k y_k \ln p_k + (1-y_k)\ln(1-p_k)$. A clever piece-wise S-shaped function, flat on the asymptotes, was used in Chu et al. (2006) to implement sparsity for a Bayesian SVM algorithm. Noise modeling is not limited to noise models for the target y, it also concerns modeling noise on the input variables $\mathbf{x}$. Many authors have incorporated noise models on $\mathbf{x}$ as part of the kernel design, for example, by enforcing invariance (Simard et al., 1993; Pozdnoukhov and Bengio, 2006). A simple but effective means of using a noise model is to generate additional training data by distorting given training examples. Additional "unsupervised" data is often useful to fit a noise model on the input variables $\mathbf{x}$. Repeatability and reproducibility (*R&R*) studies may also provide data to fit a noise model.

Feature selection filters. Feature selection, as a filter method, allows us to reduce the dimensionality of the feature space, to ease the computations performed by learning machines. This is often a necessary step for computationally expensive algorithms such as neural networks. Radford Neal for instance, used filters based on univariate statistical tests to prune the feature space before applying his Bayesian neural network algorithm (Neal and Zhang, 2006). Univariate filters were also widely used in the KDD cup 2009, which involved classification tasks on a very large database, to cut down computations (Guyon et al., 2009b). Feature selection filters are not limited to univariate filters. Markov blanket methods, for instance, provide powerful feature selection filters (Aliferis et al., 2003). A review of filters for feature selection can be found in Guyon et al. (2006a, Chapter 3).

5.2 Wrappers

Wrapper methods consider learning machines as *black boxes* capable of internally adjusting their parameters $\boldsymbol{\alpha}$ given some data D and some hyper-parameter values $\boldsymbol{\theta}$. No knowledge either of the architecture, of the learning machines, or of their learning algorithm should be required to use a wrapper. Wrappers are applicable to selecting a classifier from amongst a finite set of learning machines ($\boldsymbol{\theta}$ is then a discrete index), or an infinite set (for continuous values of $\boldsymbol{\theta}$). Wrappers can also be used to build ensembles of learning machines, including Bayesian ensembles. Wrappers use a *search algorithm* or a *sampling algorithm* to explore hyper-parameter space and an *evaluation function* (a risk functional $R_D[f(\boldsymbol{\theta})]$, a posterior probability $P(f(\boldsymbol{\theta})|D)$, or any model selection index $r[f(\boldsymbol{\theta})]$) to assess the performance of the sample of trained learning machines , and, either select one single best machine or create an ensemble of machine voting to make predictions.

Search and sampling algorithms. Because the learning machines in the wrapper setting are "black boxes", we cannot sample directly from the posterior distribution $P(f(\boldsymbol{\theta})|D)$ (or according to $\exp -R_D[f(\boldsymbol{\theta})]$ or $\exp -r[f(\boldsymbol{\theta})]$). We can only compute the evaluation function for given values of $\boldsymbol{\theta}$ for which we run the learning algorithm of $f(\boldsymbol{\theta})$, which internally adjusts its parameters $\boldsymbol{\alpha}$. A **search strategy** defines which hyper-parameter values will be considered and in which order (in case a halting criterion ends the search prematurely). Gavin Cawley, in his challenge winning entries, used the Nelder-Mead simplex algorithm (Cawley and Talbot, 2007a). **Monte-Carlo Markov Chain MCMC methods** are used in Bayesian modeling to sample the posterior probability and have given good results in challenges (Neal

and Zhang, 2006). The resulting ensemble is a simple average of the sampled functions $F(\mathbf{x}) = (1/s)\sum_{i=1}^{s} f(\mathbf{x}|\boldsymbol{\theta}_k)$. Wrappers for *feature selection* use all sort of techniques, but sequential *forward selection* or *backward elimination* methods are most popular (Guyon et al., 2006a, Chapter 4). Other stochastic search methods include biologically inspired methods such as *genetic algorithms* and *particle swarm optimization*. Good results have been obtained with this last method in challenges (H. J. Escalante, 2009), showing that extensive search does not necessarily yield over-fit solutions, if some regularization mechanism is used. The authors of that paper rely for that purpose on weight decay and early stopping. Frequentist ensemble methods, including Random Forests (Breiman, 2001) and Logitboost (Friedman et al., 2000) also gave good results in challenges (Lutz, 2006; Tuv et al., 2009; Dahinden, 2009).

Evaluation functions. For Bayesian approaches, the standard evaluation function is the "evidence", that is the marginal likelihood (also called type-II likelihood) (Neal and Zhang, 2006), or, in other words, the likelihood at the second level of inference. For frequentist approaches, the most frequently used evaluation function is the cross-validation estimator. Specifically, K-fold cross-validation is most often used (H. J. Escalante, 2009; Dahinden, 2009; Lutz, 2006; Reunanen, 2007). The values $K = 10$ or $K = 5$ are typically used by practitioners regardless of the difficulty of the problem (error rate, number of examples, number of variables). Computational considerations motivate this choice, but the authors report a relative insensitivity of the result in that range of values of K. The leave-one-out (LOO) estimator is also used, but due to its high variance, it should rather be avoided, except for computational reasons (see in Section 5.3 cases in which the LOO error is inexpensive to compute). These estimators may be poor predictors of the actual learning machine performances, but they are decent model selection indices, provided that the same data splits are used to compute the evaluation function for all models. For **bagging** methods (like Random Forests, Breiman, 2001), the bootstrap estimator is a natural choice: the "out-of-bag" samples, which are those samples not used for training, are used to predict performance. Using empirical estimators at the second level on inference poses the problem of possibly over-fitting them. Some authors advocate using evaluation functions based on prediction risk bounds: Koo and Kil (2008) and Claeskens et al. (2008) derive in this way information criteria for regression models (respectively called "modulus of continuity information criterion" or MCIC and "kernel regression information criterion" or KRIC) and Claeskens et al. (2008) and Pranckeviciene and Somorjai (2009) propose information criteria for classification problems (respectively called "support vector machine information criterion" SVMIC and "transvariation intensity"). The effectiveness of these new criteria is compared empirically in the papers to the classical "Akaike information criterion" or AIC (Akaike, 1973) and the "Bayesian information criterion" or BIC (Schwarz, 1978).

5.3 Embedded Methods

Embedded methods are similar to wrappers. They need an evaluation function and a search strategy to explore hyper-parameter space. But, unlike wrapper methods, they exploit specific features of the learning machine architecture and/or learning algorithm to perform multi-level inference. It is easy to appreciate that knowledge of the nature and structure of a learning machine can allow us to search hyper-parameter space in a more efficient way. For instance, the function $f(\mathbf{x};\boldsymbol{\alpha},\boldsymbol{\theta})$ may be differentiable with respect to hyper-parameters $\boldsymbol{\theta}$ and it may be possible to use *gradient descent* to

optimize an evaluation function $r[f]$. Embedded methods have been attracting substantial attention within the machine learning community in the past few years because of the mathematical elegance of some of the new proposed methods.

Bayesian embedded methods. In the Bayesian framework, the embedded search, sampling or summation over parameters and hyper-parameters is handled in an elegant and consistent way by defining priors both for parameters and hyper-parameters, and computing the posterior, perhaps in two steps, as indicated in Equation 3. Of course, it is more easily said than done and the art is to find methods to carry out this integration, particularly when it is analytically intractable. Variational methods are often used to tackle that problem. Variational methods convert a complex problem into a simpler problem, but the simplification introduces additional "variational" parameters, which must then be optimized, hence introducing another level of inference. Typically, the posterior is bounded from above by a family of functions parameterized by given variational parameters. Optimizing the variational parameters yields the best approximation of the posterior (see *e.g.,* Seeger, 2008). Bayesian pragmatists optimize the evidence (also called type-II likelihood or marginal likelihood) at the second level of inference, but non-purists sometimes have a last recourse to cross-validation. The contributions of Boullé (2007, 2009) stand out in that respect because they propose model selection methods for classification and regression, which have no last recourse to cross-validation, yet performed well in recent benchmarks (Guyon et al., 2008a, 2009b). Such methods have been recently extended to the less studied problem of rank regression (Hue and Boullé, 2007). The methods used are Bayesian in spirit, but make use of original data-dependent priors.

Regularized functionals. In the frequentist framework, the choice of a prior is replaced by the choice of a regularized functional. Those are two-part evaluation functions including the empirical risk (or the negative log-likelihood) and a regularizer (or a prior). For kernel methods, a 2-norm regularizer is often used, yielding the classical penalized functional $R_{reg}[f] = R_{emp}[f] + \gamma \|f\|^2_{\mathcal{F}}$. Pranckeviciene and Somorjai (2009) explore the possibilities offered by a 1-norm regularizer. Such approaches provide an embedded method of feature selection, since the constraints thus imposed on the weight vector drive some weights to exactly zero. We emphasized in the introduction that, in some cases, decomposing the inference problem into multiple levels allows us to conveniently regain the convexity of the optimization problem involved in learning. Ye et al. (2008) propose a multiple kernel learning (MKL) method, in which the optimal kernel matrix is obtained as a linear combination of pre-specified kernel matrices, which can be brought back to a convex program. Few approaches are fully embedded and a wrapper is often used at the last level of inference. For instance, in kernel methods, the kernel parameters may be optimized by gradient descent on the regularized functional, but then the regularization parameter is selected by cross-validation. One approach is to use a bound on the generalization error at the second level of inference. For instance, Guermeur (2007) proposes such a bound for the multi-class SVM, which can be used to choose the values of the "soft margin parameter" C and the kernel parameters. Cross-validation may be preferred by practitioners because it has performed consistently well in benchmarks (Guyon et al., 2006b). This motivated Kunapuli et al. (2009) to integrate the search for optimal parameters and hyper-parameters into a multi-level optimization program, using a regularized functional at the lower level, and cross-validation at the upper level. Another way of integrating a second level of inference performed by cross-validation and the optimization of

a regularized functional at the first level of inference is to use a closed-form expression of the leave-one-out error (or a bound) and optimize it by gradient descent or another classical optimization algorithm. Such *virtual leave-one-out* estimators, requiring training a single classifier on all the data (see *e.g.*, Cawley and Talbot, 2007a; Debruyne et al., 2–8, in the collection of papers we have assembled).

6. Advanced Topics and Open Problems

We have left aside many important aspects of model selection, which, space permitting, would deserve a longer treatment. We briefly discuss them in this section.

6.1 Ensemble Methods

In Section 4, we have made an argument in favor of unifying *model selection* and *ensemble methods*, stemming either from a Bayesian or frequentist perspective, in the common framework of *multi-level optimization*. In Sections 5.1, 5.2, and 5.3, we have given examples of model selection and ensemble methods following *filter*, *wrapper* or *embedded* strategies. While this categorization has the advantage of erasing the dogmatic origins of algorithms, it blurs some of the important differences between model selection and ensemble methods. Ensemble methods can be thought of as a way of circumventing model selection by voting among models rather than choosing a single model. Recent challenges results have proved their effectiveness (Guyon et al., 2009b). Arguably, model selection algorithms will remain important in applications where model simplicity and data understanding prevail, but ever increasing computer power has brought ensemble methods to the forefront of multi-level inference techniques. For that reason, we would like to single out those papers of our collection that have proposed or applied ensemble methods:

Lutz (2006) used boosted shallow decision trees for his winning entries in two consecutive challenges. Boosted decision trees have often ended up among the top ranking methods in other challenges (Guyon et al., 2006a, 2009b). The particular implementation of Lutz of the Logitboost algorithm (Friedman et al., 2000) use a "shrinkage" regularization hyper-parameter, which seems to be key to attain good performance, and is adjusted by cross-validation as well as the total number of *base learners*. Dahinden (2009) successfully applied the Random Forest (RF) algorithm (Breiman, 2001) in the performance prediction challenge (Guyon et al., 2006b). She demonstrated that with minor adaptations (adjustment of the bias value for improved handling of unbalanced classes), the RF algorithm can be applied without requiring user intervention. RF continues to be a popular and successful method in challenges (Guyon et al., 2009b). The top ranking models use very large ensembles of hundreds of trees. One of the unique features of RF algorithms is that they subsample both the training examples and the features to build base learners. Using random subsets of features seems to be a winning strategy, which was applied by others to ensembles of trees using both boosting and bagging (Tuv et al., 2009) and to other base learners (Nikulin, 2009). Boullé (2007) also adopts the idea of creating ensembles using base learners constructed with different subsets of features. Their base learner is the naïve Bayes classifier and, instead of using random subsets, they select subsets with a forward-backward method, using a maximum A Posteriori (MAP) evaluation function (hence not requiring cross-validation). The base learners are then combined with an weighting scheme based on an information theoretic criterion, instead on weighting the models with the posterior probability as in Bayesian model averaging. This basically boils down to using the logarithm of the posterior probabilities instead of the posterior probabilities themselves

for weighting the models. The weights have an interpretation in terms of model compressibility. The authors show that this strategy outperforms Bayesian model averaging on several benchmark data sets. This can be understood by the observation that when the posterior distribution is sharply peaked around the posterior mode, averaging is almost the same as selecting the MAP model. Robustness is introduced by performing a more balanced weighting of the base learners. In contrast with the methods we just mentioned, which choose identical base learners (trees of naïve Bayes), other successful challenge participants have built heterogeneous ensembles of learning machines (including, for example, linear models, kernel methods, trees, naïve Bayes, and neural networks), using cross-validation to evaluate their candidates for inclusion in the ensemble (Wichard, 2007; IBM team, 2009). While Wichard (2007) evaluates classifiers independently, IBM team (2009) uses a forward selection method, adding a new candidate in the ensemble based on the new performance of the ensemble.

6.2 PAC Bayes Approaches

Unifying Bayesian and frequentist model selection procedures under the umbrella of *multi-level inference* may shed new light on correspondences between methods and have a practical impact on the design of toolboxes incorporating model selection algorithms. But there are yet more synergies to be exploited between the Bayesian and the frequentist framework. In this section, we would like to capture the spirit of the PAC Bayes approach and outline possible fruitful directions of research.

The PAC learning framework (Probably Approximately Correct), introduced by Valiant (1984) and later recognized to closely resemble the approach of the Russian school popularized in the US by Vapnik (1979), has become the beacon of frequentist learning theoretic approaches. It quantifies the generalization performance (the *Correct* aspect) of a learning machine via performance bounds (the *Approximate* aspect) holding in probability (the *Probable* aspect):

$$Prob\Big[(R[f] - R_{emp}[f]) \leq \varepsilon(\delta)\Big] \geq (1-\delta) \ ,$$

In this equation, the confidence interval $\varepsilon(\delta)$ (*Approximate* aspect) bounds, with probability $(1-\delta)$ (*Probable* aspect),the difference between the *expected risk* or generalization error $R[f]$ and the *empirical risk*[6] $R_{emp}[f]$ (*Correct* aspect). Recently, many bounds have been proposed to quantify the generalization performance of algorithms (see *e.g.,* Langford, 2005, for a review). The idea of deriving new algorithms, which optimize a bound $\varepsilon(\delta)$ (*guaranteed risk* optimization) has been popularized by the success of SVMs (Boser et al., 1992) and boosting (Freund and Schapire, 1996).

The PAC framework is rooted in the frequentist philosophy of defining probability in terms of *frequencies of occurrence of events* and bounding differences between mathematical expectations and frequencies of events, which vanish with increasingly large sample sizes (law of large numbers). Yet, since the pioneering work of Haussler et al. (1994), many authors have proposed so-called PAC-Bayes bounds. Such bounds assess the performance of existing Bayesian algorithms (see *e.g.,* Seeger, 2003), or are used to derive new Bayesian algorithms optimizing a guaranteed risk functional (see Germain et al. 2009 and references therein).

This is an important paradigm shift, which bridges the gap between the frequentist *structural risk minimization* approach to model selection (Vapnik, 1998) and the *Bayesian prior* approach.

6. at the first level of inference, this would be the training error $R_{tr}[f]$; at the second level of inference this may be the validation error $R_{va}[f]$

It erases the need for assuming that *the model used to fit the data comes from a concept space of functions that generated the data*. Instead, priors may be used to provide a "structure" on a chosen model space (called *hypothesis space* to distinguish it from the *concept space*), which does not necessarily coincide with the *concept space*, of which we often know nothing. Reciprocally, we can interpret structures imposed on a hypothesis space as our prior belief that certain models are going to perform better than others (see, for instance, the examples at the end of Section 3.1).

This opens the door to also regularizing the second level of inference by using performance bounds on the cross-validation error, as was done for instance in Cawley and Talbot (2007a) and Guyon (2009).

6.3 Open Problems

- **Domain knowledge:** From the earliest embodiments of Okcham's razor using the number of free parameters to modern techniques of regularization and bi-level optimization, model selection has come a long way. The problem of finding the right structure remains, the rights prior or the right regularizer. Hence know-how and domain knowledge are still required. But in a recent challenge we organized called "agnostic learning *vs*. prior knowledge" (Guyon et al., 2008b) it appeared that the relatively small incremental improvements gained with prior knowledge came at the expense of important human effort. In many domains, collecting more data is less costly than hiring a domain expert. Hence there is pressure towards improving machine learning toolboxes and, in particular equipping them with model selection tools. For the competitions we organized (Clopinet, 2004-2009), we made a toolbox available with state-of-the-art models (Saffari and Guyon, 2006), which we progressively augmented with the best performing methods. The Particle Swarm Optimization (PSO) model selection method can find the best models in the toolbox and reproduce the results of the challenges (H. J. Escalante, 2009). Much remains to be done to incorporate filter and wrapper model selection algorithms in machine learning toolboxes.

- **Unsupervised learning:** Multi-level optimization and model selection are also central problems for **unsupervised learning**. When no target variable is available as "teaching signal" one can still define regularized risk functionals and multi-level optimization problems (Smola et al., 2001). Hyper-parameters (e.g., "number of clusters") can be adjusted by optimizing a second level objective such as model stability (Ben-Hur et al., 2002), which is an erzatz of cross-validation. The primary difficulty with model selection for unsupervised learning is to validate the selected model. To this day, there is no consensus on how to benchmark methods, hence it is very difficult to quantify progress in this field. This is why we have so far shied away from evaluating unsupervised learning algorithms, but this remains on our agenda.

- **Semi-supervised learning:** Very little has been done for model selection in **semi-supervised learning** problems, in which only some training instances come with target values. Semi-supervised tasks can be challenging for traditional model selection methods, such as cross-validation, because the number of labeled data is often very small. Schuurmans and Southey (2001) used the unlabeled data to test the consistency of a model, by defining a metric over the hypothesis space. Similarly, Madani et al. (2005) introduced the co-validation method, which uses the disagreement of various models on the predictions over the unlabeled data as a model selection tool. In some cases there is no performance gain by using the unlabeled

data for training (Singh et al., 2008). Deciding whether all or part of the unlabeled data should be used for training (*data selection*) may also be considered a model selection problem.

- **Non *i.i.d.* data:** The problem of non *i.i.d.* data raises a number of other questions because if there are significant differences between the distribution of the training and the test data, the cross-validation estimator may be worthless. For instance, in causal discovery problems, training data come from a "natural" distribution while test data come from a different "manipulated" distribution (resulting from some manipulations of the system by an external agent, like clamping a given variable to given values). Several causal graphs may be consistent with the "natural distribution" (not just with the training data, with the true unknown distribution), but yield very different predictions of manipulated data. Rather selecting a single model, it make more sense to select a model class. We have started a program of data exchange and benchmarks to evaluate solutions to such problems (Guyon et al., 2008a, 2009a).

- **Computational considerations:** The selection of the model best suited to a given application is a multi-dimensional problem in which prediction performance is only one of the dimensions. Speed of model building and processing efficiency of deployed models are also important considerations. Model selection algorithms (or ensemble methods) which often require many models to be trained (e.g., wrapper methods with extensive search strategies and using cross-validation to validate models) may be unable to build solutions in a timely manner. At the expense of some acceptable loss in prediction performance, methods using *greedy search strategies* (like forward selection methods) and *single-pass evaluation functions* (requiring the training of only a single model to evaluate a given hyper-parameter choice), may considerably cut the training time. Greedy search methods include forward selection and backward elimination methods. Single-pass evaluation functions include penalized training error functionals (regularized functionals, MAP estimates) and virtual-leave-one-out estimators. The latter allows users to compute the leave-one-out-error at almost no additional computational expense than training a single predictor on all the training data (see *e.g.,* Guyon et al., 2006a, Chapter 2, for a review). Other tricks-of-the-trade include following *regularization paths* to sample the hyper-parameter space more effectively (Rosset and Zhu, 2006; Hastie et al., 2004). For some models, the evaluation function is piecewise linear between a few discontinuous changes occurring for a few finite hyper-parameter values. The whole path can be reconstructed from only the values of the evaluation function at those given points. Finally, Reunanen (2007) proposed clever ways of organizing nested cross-validation evaluations in multiple level of inference model selection using cross-indexing. The author also explored the idea of spending more time to refine the evaluation of the most promising models. Further work needs to be put into model selection methods, which simultaneously address multiple objectives, including optimizing prediction performance and computational cost.

7. Conclusion

In the past twenty years, much effort has been expended towards finding the best regularized functionals. The many embodiments of Ockham's razor in machine learning have converged towards similar regularizers. Yet, the problem of model selection remains because we need to optimize the regularization parameter(s) and often we need to select among various preprocessings, learning machines, and post-processings. In the proceedings of three of the challenges we organized around the

problem of model selection, we have collected a large number of papers, which testify to the vivid activity of the field. Several researchers do not hesitate to propose heretic approaches transcending the usual "frequentist" or Bayesian dogma. We have seen the idea of using the Bayesian machinery to design regularizers with "data-dependent priors" emerge (Boullé, 2007, 2009), much like a few years ago data-dependent performance bounds (Bartlett, 1997; Vapnik, 1998) and PAC-Bayes bounds (Haussler et al., 1994; Seeger, 2003) revolutionized the "frequentist" camp, up to then very fond of uniform convergence bounds and the VC-dimension (Vapnik and Chervonenkis, 1971). We have also seen the introduction of regularization of cross-validation estimators using Bayesian priors (Cawley and Talbot, 2007a). Ensemble methods may be thought of as a way of circumventing model selection. Rather, we think of model selection and ensemble methods as two options to perform multi-level inference, which can be formalized in a unified way.

Within this general framework, we have categorized approaches into *filter*, *wrapper* and *embedded* methods. These methods complement each other and we hope that in a not too distant future, they will be integrated into a consistent methodology: Filters first can prune model space; Wrappers can perform an outer level model selection to select pre/post processings and feature subsets; Embedded methods can perform an inner level hyper-parameter selection integrated within a bi-level optimization program. We conclude that we are moving towards a unified framework for model selection and there is a beneficial synergy between methods, both from a theoretical and from a practical perspective.

Acknowledgments

This project was supported by the National Science Foundation under Grant N0. ECS-0424142. Amir Saffari was supported by the Austrian Joint Research Project Cognitive Vision under projects S9103-N04 and S9104-N04. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Appendix A. Glossary

Automatic Relevance Determination (ARD) prior. The ARD prior was invented for neural networks (MacKay, 1992): all network input variables and all neuron outputs (internal features) are weighed by a scaling factor κ_i, before being independently weighted by the network connections. A hyper-prior must be chosen to favor small values of the κ_i, which makes the influence of irrelevant variables or features naturally fade away. For kernel methods, ARD falls under the same framework as the $\|f\|^2_{\mathcal{H}}$ regularizer, for a special class of kernels using variable (feature) scaling factors. For instance, the ARD prior is implemented by defining the Gaussian kernel (for positive hyper-parameters κ_i):

$$K(\mathbf{x}_h, \mathbf{x}_k) = \exp\left\{ -\sum_{j=1}^{n} \kappa_i (x_{h,j} - x_{k,j})^2 \right\}$$

instead of the regular Gaussian kernel $K(\mathbf{x}_h, \mathbf{x}_k) = \exp\left\{-\kappa \|\mathbf{x}_h - \mathbf{x}_k\|^2\right\}$.

Base learner. In an ensemble method, the individual learning machines that are part of the ensemble.

Bagging. Bagging stands for bootstrap aggregating. Bagging is a parallel ensemble method (all **base learners** are built independently from training data subsets). Several data subsets of size m are drawn independently with replacement from the training set of m examples. On average each subset thus built contains approximately 2/3 of the training examples. The ensemble predictions are made by averaging the predictions of the baser learners. The ensemble approximates $E_D(f(\mathbf{x},D))$, where $f(\mathbf{x},D)$ is a function from the model class F, trained with m examples and $E_D(.)$ is the mathematical expectation over all training sets of size m. The rationale comes from the **bias/variance decomposition** of the **generalization error**. The "out-of-bag" samples (samples not used for learning for each data subset drawn for training) may be used to create a bootstrap prediction of performance.

Bayesian learning. Under the Bayesian framework, it is assumed that the data were generated from a double random process: (1) a model is first drawn according to a **prior** distribution in a **concept space**; (2) data are produced using the model. In the particular case of supervised learning, as for **maximum likelihood learning**, a three-part data generative model is assumed: $P(\mathbf{x})$, $f \in \mathcal{F}$, and a zero-mean noise model. But, it is also assumed that the function f was drawn according to a prior distribution $P(f)$. This allows us to compute the probability of an output y given an input $\mathbf{x}$, $P(y|\mathbf{x}) = \int_{f\in\mathcal{F}} P(y|\mathbf{x},f)dP(f)$, or its mathematical expectation $E(y|\mathbf{x}) = \int_{f\in\mathcal{F}} f(\mathbf{x})dP(f)$, averaging out the noise. After training data D are observed, the prior $P(f)$ is replaced by the **posterior** $P(f|D)$. The mathematical expectation of y given $\mathbf{x}$ is estimated as: $E(y|\mathbf{x},D) = \int_{f\in\mathcal{F}} f(\mathbf{x})dP(f|D)$. Hence, learning consists of calculating the posterior distribution $P(f|D)$ and integrating over it. The predictions are made according to $E(y|\mathbf{x},D)$, a function not necessarily belonging to $\mathcal{F}$. In the case of classification, $E(y|\mathbf{x},D)$ does not take values in $\mathcal{Y}$ (although thresholding the output just takes care of the problem). If we want a model in $\mathcal{F}$, we can use the Gibbs algorithm, which picks one sample in $\mathcal{F}$ according to the posterior distribution $P(f|D)$, or use the **MAP learning** approach. In Bayesian learning, analytically integrating over the posterior distribution is often impossible and the integral may be approximated by finite sum of models, weighted by positive coefficients (see **variational methods**) or by sampling models from the posterior distribution (see **Weighted majority algorithm** and **Monte-Carlo Markov Chain** or MCMC). The resulting estimators of $\hat{E}(y|\mathbf{x},D)$ are convex combinations of functions in $\mathcal{F}$ and, in that sense, Bayesian learning is similar to **ensemble methods**.

Bias/variance decomposition. In the case of a least-square loss, the bias/variance decomposition is given by $E_D[(f(\mathbf{x};D)-E[y|\mathbf{x}])^2] = (E_D[f(\mathbf{x};D)]-E(y|\mathbf{x}))^2+E_D[(f(\mathbf{x};D)-E_D[f(\mathbf{x};D)])^2]$. The second term (the "variance" of the estimator $f(\mathbf{x},D)$) vanishes if $f(\mathbf{x};D)$ equals $E_D[f(\mathbf{x};D)$. The motivates the idea of using an approximation of $E_D[f(\mathbf{x};D)$ as a predictor. In **bagging** the approximation is obtained by averaging over functions trained from m examples drawn at random with replacement from the training set D (bootstrap method). The method works best if $\mathcal{F}$ is not biased (i.e., contains $E(y|\mathbf{x})$). Most models with low bias have a high variance and vice versa, hence the well-known bias/variance tradeoff.

Concept space. A space of data generative models from which the data are drawn. Not to be confused with **model space** or **hypothesis space**.

Empirical risk. An estimator of the **expected risk** that is the average of the loss over a finite number of examples drawn according to $P(\mathbf{x},y)$: $R_{emp} = (1/m)\sum_{k=1}^{m} \mathcal{L}(f(\mathbf{x}_k),y_k)$.

Ensemble methods. Methods of building predictors using multiple **base learners**, which vote to make predictions. Predictions of y are made using a convex combination of functions $f_j \in \mathcal{F}$: $F(\mathbf{x}) = \sum_j p_j f_j(\mathbf{x})$, where p_j are positive coefficients. The two most prominent ensemble methods are bagging (Breiman, 1996) and boosting (Freund and Schapire, 1996). Bagging is a parallel ensemble method (all trees are built independently from training data subsets), while boosting is a serial ensemble method (trees complementing each other are progressively added to decrease the residual error). Random Forests (RF) (Breiman, 2001) are a variant of bagging methods in which both features and examples are subsampled. Boosting methods come in various flavors including Adaboost, Gentleboost, and Logitboost. The original algorithm builds successive models (called "weak learners") by resampling data in a way that emphasizes examples that have proved hardest to learn. Newer versions use a weighting scheme instead of resampling (Friedman, 2000).

Expected risk. The mathematical expectation of a risk functional over the unknown probability distribution $P(\mathbf{x},y)$: $R[f] = \int \mathcal{L}(f(\mathbf{x}),y)\, dP(\mathbf{x},y)$. Also called **generalization error**.

Generalization error. See **expected risk**.

Greedy search strategy. A search strategy, which does not revisit partial decisions already made, is called "greedy". Examples include forward selection and backward elimination in feature selection.

Guaranteed risk. A bound on the **expected risk**. See **PAC learning** and **Structural Risk Minimization** (SRM).

Hypothesis space. A space of models, which are fit to data, not necessarily identical to the **concept space** (which is often unknown).

Loss function. A function $\mathcal{L}(f(\mathbf{x}),y)$, which measures the discrepancy between target values y and model predictions $f(\mathbf{x})$. Examples include the square loss $(y - f(\mathbf{x}))^2$ for regression of the 0/1 loss $\mathbf{1}[f(\mathbf{x}) \neq y]$ for classification).

MAP learning. Maximum a posteriori (MAP) learning shares the same framework as Bayesian learning, but it is further assumed that the posterior $P(f|D)$ is concentrated and that $E(y|\mathbf{x},D)$ can be approximated by $f^*(\mathbf{x})$, with $f^* = \operatorname{argmax}_f P(f|D) = \operatorname{argmax}_f P(D|f)P(f) = \operatorname{argmin}_f -\ln P(D|f) - \ln P(f)$. If we assume a uniform prior, we are brought back to maximum likelihood learning. If both $P(D|f)$ and $P(f)$ are exponentially distributed ($P(y|\mathbf{x},f) = exp - \mathcal{L}(f(\mathbf{x}),y)$ and $P(f) = exp - \Omega[f]$), then MAP learning is equivalent to the minimization of a regularized risk functional.

Maximum likelihood learning. It is assumed that the data were generated by an input distribution $P(\mathbf{x})$, a function f from a **model space** $\mathcal{F}$ coinciding with the **concept space**, and a zero-mean **noise model**. For regression, for instance, if Gaussian noise $\varepsilon \sim \mathcal{N}(0,\sigma^2)$ is assumed, y is distributed according to $P(y|\mathbf{x},f) = \mathcal{N}(f(\mathbf{x}),\sigma^2)$. In the simplest case, $P(\mathbf{x})$ and the noise model are not subject to training (the values of $\mathbf{x}$ are fixed and the noise model is known). Learning then consists in searching for the function f^*, which maximizes the likelihood $P(D|f)$, or equivalently (since $P(\mathbf{x})$ is not subject to training) $f^* = \operatorname{argmax}_f P(\mathbf{y}|X,f) = \operatorname{argmin}_f - \ln P(\mathbf{y}|X,f)$. With the *i.i.d.* assumption, $f^* = \operatorname{argmax}_f \Pi_{k=1}^m P(y_k|\mathbf{x}_k,f) = \operatorname{argmin}_f \sum_{k=1}^m -$

$\ln P(y_k|\mathbf{x}_k, f)$. For distributions belonging to the exponential family $P(y|\mathbf{x}, f) = \exp\{-\mathcal{L}(f(\mathbf{x}), y)\}$, the maximum likelihood method is equivalent to the method of minimizing the **empirical risk**. In the case of Gaussian noise, this corresponds to the method of least squares.

Model space. A space of predictive models, which are fit to data. Synonym of **hypothesis space**. For Bayesian models, also generally coincides with the **concept space**, but not for frequentists.

Monte-Carlo Markov Chain (MCMC) method. To approximate Bayesian integrals one can sample from the posterior distribution $P(f|D)$ following a Monte-Carlo Markov chain (MCMC), then make predictions according to $\hat{E}(y|\mathbf{x}, D) = \sum_j f_j(\mathbf{x})$. In a MCMC, at each step new candidate models $f_j \in \mathcal{F}$ are considered, in a local neighborhood of the model selected at the previous step. The new model is accepted if it provides a better fit to the data according to the posterior distribution or, if not, a random decision is made to accept it, following the Gibbs distribution (better models having a greater chance of acceptance).

Over-fitting avoidance. Model selection is traditionally associated with the so-called problem of **over-fitting** avoidance. Over-fitting means fitting the training examples well (i.e., obtaining large model likelihood or low empirical risk values, computed from training data), but generalizing poorly on new test examples. Over-fitting is usually blamed on too large a large number of free parameters to be estimated, relative to the available number of training examples. The most basic model selection strategy is therefore to restrict the number of free parameters according to "strict necessity". This heuristic strategy is usually traced back in history to the principle known as Ockham's razor "Plurilitas non est ponenda sin necessitate" (William of Ockham, 14^{th} century). In other words, of two theories providing similarly good predictions, the simplest one should be preferred, that is, shave off unnecessary parameters. Most modern model selection strategies claim some affiliation with Ockham's razor, but the number of free parameters is replaced by a measure of **capacity** or **complexity** of the model class, $C[\mathcal{F}]$. Intuitively, model classes with large $C[\mathcal{F}]$ may include the correct model, but it is hard to find. In this case, even models with a low training error may have a large generalization error (high "variance"; over-fitting problem). Conversely, model classes with small $C[\mathcal{F}]$ may yield "biased" models, that is, with both high training and generalization error (under-fitting). See **bias/variance decomposition.**.

PAC learning. The "probably approximately correct" (PAC) learning procedures, seek a function minimizing a **guaranteed risk** $R_{gua}[f] = R_{emp}[f] + \varepsilon(C, \delta)$ such that with (high) probability $(1-\delta)$, $R[f] \leq R_{gua}[f]$. C is a measure of capacity or complexity.

Regularizers and regularization. The regularization method consists of replacing the minimization of the **empirical risk** $R_{emp}[f]$ by that of $R_{reg}[f] = R_{emp} + \Omega[f]$. A regularizer $\Omega[f]$ is a functional penalizing "complex" functions. If both $R_{tr}[f]$ and $\Omega[f]$ are convex, there is a unique minimum of $R_{reg}[f]$ with respect to f. In **MAP learning**, $-\ln P(f)$ can be thought of as a **regularizer**. One particularly successful regularizer is the 2-norm regularizer $\|f\|^2_{\mathcal{H}}$ for model functions $f(\mathbf{x}) = \sum_{k=1}^{m} \alpha_k K(\mathbf{x}, \mathbf{x}_k)$ belonging to a Reproducing Kernel Hilbert Space $\mathcal{H}$ (kernel methods). In the particular case of the linear model $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x}$,

we have $\|f\|^2_{\mathcal{H}} = \|\mathbf{w}\|^2$, a commonly used regularized found in many algorithms including ridge regression (Hoerl, 1962) and SVMs (Boser et al., 1992). In the general case, $\|f\|^2_{\mathcal{H}} = \mathbf{f}K^{-1}\mathbf{f} = \boldsymbol{\alpha}^T K\boldsymbol{\alpha}$, where $\mathbf{f} = [f(\mathbf{x}_k)]_{k=1}^m$ is the vector of predictions of the training examples, $\boldsymbol{\alpha} = [\alpha_k]_{k=1}^m$ and $K = [K(\mathbf{x}_h, \mathbf{x}_k]$, $h = 1, ...m$ $k = 1, ...m$. Due to the duality between RKHS and stochastic processes (Wahba, 1990), the functions in the RKHS can also be explained as a family of random variables in a Gaussian process, assuming a prior $P(f)$ proportional to $\exp(-\gamma\|f\|_{\mathcal{H}}) = \exp(-\gamma\mathbf{f}K^{-1}\mathbf{f})$ and the kernel matrix K is interpreted as a covariance matrix $K(\mathbf{x}_h, \mathbf{x}_k)$ =cov$[f(x_k), f(x_k)]$.

Risk minimization. Given a **model space** or **hypothesis space** $\mathcal{F}$ of functions $y = f(\mathbf{x})$, and a **loss function** $\mathcal{L}(f(\mathbf{x}), y)$, we want to find the function $f^* \in \mathcal{F}$ that minimizes the **expected risk** $R[f] = \int \mathcal{L}(f(\mathbf{x}), y)\, dP(\mathbf{x}, y)$. Since $P(\mathbf{x}, y)$ is unknown, only estimations of $R[f]$ can be computed. The simplest estimator is the average of the loss over a finite number of examples drawn according to $P(\mathbf{x}, y)$ called the **empirical risk**: $R_{emp} = (1/m)\sum_{k=1}^m \mathcal{L}(f(\mathbf{x}_k), y_k)$. The minimization of the empirical risk is the basis of many machine learning approaches to selecting f^*, but minimizing regularized risk functionals is often preferred. See **regularization**. Also, related are the **PAC learning** procedures and the method of **Structural Risk Minimization** (SRM).

Search strategy. There are *optimal search strategies*, which guarantee that the optimum of the evaluation function will be found, including the *exhaustive search* method, for discrete hyper-parameter spaces. The popular *grid search* method for continuous hyper-parameter spaces performs an exhaustive search, up to a certain precision. A related *stochastic search* method is *uniform sampling*. Uniformly sampling parameter space may be computationally expensive and inefficient. If we use a non-uniform distribution $G(\boldsymbol{\theta})$ to sample hyper-parameter space, which resembles $P(f(\boldsymbol{\theta})|D)$, the search can be made more efficient. This idea is exploited in *rejection sampling* and *importance sampling*: according to these methods a Bayesian ensemble $F(\mathbf{x}) = \sum_k w_k f(\mathbf{x}; \boldsymbol{\theta}_k)$ would use weight w_k proportional to $P(f(\boldsymbol{\theta})|D)/G(\boldsymbol{\theta})$. Because of the computational burden of (near) optimum strategies, other strategies are often employed, usually yielding only a *local optimum*. These include *sequential search strategies* such as *coordinate ascent* or descent (making small steps along coordinate axes) or *pattern search* (Momma and Bennett, 2002) (making local steps according to a certain pattern), which, by accepting only moves that improve the evaluation function, find the local optimum nearest to the starting point. Some stochastic search methods accept moves not necessarily improving the value of the evaluation function, like simulated annealing or **Markov chain Monte Carlo (MCMC) methods**. Both methods accept all moves improving the evaluation function and some moves that do not, for example, with probability $\exp -\Delta r/T$, where T is a positive parameter (T=1 for MCMC and progressively diminishes for simulated annealing). Such stochastic methods search hyper-parameter space more intensively and do not become stuck in the nearest local optimum of the evaluation function.

Semi-supervised learning. In semi-supervised learning, in addition to the labeled data, the learning machine is given a (possibly large) set of unlabeled data. Such unlabeled data may be used for training or model selection.

Structural Risk Minimization. The method of Structural Risk Minimization (SRM) provides aeans of building regularized risk functionals (see **Regularization**), using the idea of **guaranteed**

risk minimization, but not requiring the calculation of the model class capacity or complexity, which is often unknown or hard to compute. In the risk minimization framework, it is not assumed that the model space includes a function or "concept", which generated the data (see **concept space** and **hypothesis space**).

Supervised learning. Learning with teaching signal or target y.

Under-fitting. While **over-fitting** is the problem of learning the training data too well the expense of a large **generalization error**, under-fitting is the problem of having a too weak model not even capable of learning the training data and also generalizing poorly.

Unsupervised learning. Learning in the absence of teaching signal or target y.

Weighted majority algorithm. To approximate Bayesian integrals one can draw samples f_j uniformly from the model space of functions $\mathcal{F}$ and make predictions according to $\hat{E}(y|\mathbf{x},D) = \sum_j P(f_j|D) f_j(\mathbf{x})$.

References

M. Adankon and M. Cheriet. Unified framework for SVM model selection. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

H. Akaike. Information theory and an extension of the maximum likelihood principle. In B.N. Petrov and F. Csaki, editors, *2nd International Symposium on Information Theory*, pages 267–281. Akademia Kiado, Budapest, 1973.

C. F. Aliferis, I. Tsamardinos, and A. Statnikov. HITON, a novel Markov blanket algorithm for optimal variable selection. In *2003 American Medical Informatics Association (AMIA) Annual Symposium*, pages 21–25, 2003.

C.-A. Azencott and P. Baldi. Virtual high-throughput screening with two-dimensional kernels. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

P. L. Bartlett. For valid generalization the size of the weights is more important than the size of the network. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 134, Cambridge, MA, 1997. MIT Press.

A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, pages 6–17, 2002.

A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, December 1997.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

M. Boullé. Compression-based averaging of selective naive bayes classifiers. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 8, pages 1659–1685, Jul 2007. URL `http://www.jmlr.org/papers/volume8/boulle07a/boulle07a.pdf`.

M. Boullé. Data grid models for preparation and modeling in supervised learning. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

G. Cawley. Leave-one-out cross-validation based model selection criteria for weighted ls-svms. In *IJCNN*, pages 1661–1668, 2006.

G. Cawley and N. Talbot. Over-fitting in model selection and subsequent selection bias in performance evaluation. *JMLR, submitted*, 2009.

G. Cawley and N. Talbot. Preventing over-fitting during model selection via Bayesian regularisation of the hyper-parameters. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 8, pages 841–861, Apr 2007a. URL `http://www.jmlr.org/papers/volume8/cawley07a/cawley07a.pdf`.

G. C. Cawley and N. L. C. Talbot. Agnostic learning versus prior knowledge in the design of kernel machines. In *Proc. IJCNN07*, Orlando, Florida, Aug 2007b. INNS/IEEE.

G.C. Cawley, G.J. Janacek, and N.L.C. Talbot. Generalised kernel machines. In *International Joint Conference on Neural Networks*, pages 1720 – 1725. IEEE, August 2007.

W. Chu, S. Keerthi, C. J. Ong, and Z. Ghahramani. Bayesian Support Vector Machines for feature ranking and selection. In I. Guyon, et al., editor, *Feature Extraction, Foundations and Applications*, 2006.

G. Claeskens, C. Croux, and J. Van Kerckhoven. An information criterion for variable selection in Support Vector Machines. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 9, pages 541–558, Mar 2008. URL `http://www.jmlr.org/papers/volume9/claeskens08a/claeskens08a.pdf`.

Clopinet. Challenges in machine learning, 2004-2009. URL `http://clopinet.com/challenges`.

C. Dahinden. An improved Random Forests approach with application to the performance prediction challenge datasets. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

M. Debruyne, M. Hubert, and J. Suykens. Model selection in kernel based regression using the influence function. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 9, pages 2377–2400, Oct 2–8. URL `http://www.jmlr.org/papers/volume9/debruyne08a/debruyne08a.pdf`.

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proc. 13th International Conference on Machine Learning*, pages 148–146. Morgan Kaufmann, 1996.

J. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression, a statistical view of boosting. *Annals of Statistics*, 28:337374, 2000.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software (to appear)*, 2009.

P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. PAC-Bayesian learning of linear classifiers. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 353–360, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

Y. Guermeur. VC theory of large margin multi-category classifiers. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 8, pages 2551–2594, Nov 2007. URL `http://www.jmlr.org/papers/volume8/guermeur07a/guermeur07a.pdf`.

I. Guyon. A practical guide to model selection. In J. Marie, editor, *Machine Learning Summer School*. Springer, to appear, 2009.

I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, Editors. *Feature Extraction, Foundations and Applications*. Studies in Fuzziness and Soft Computing. With data, results and sample code for the NIPS 2003 feature selection challenge. Physica-Verlag, Springer, 2006a. URL `http://clopinet.com/fextract-book/`.

I. Guyon, A. Saffari, G. Dror, and J. Buhmann. Performance prediction challenge. In *IEEE/INNS conference IJCNN 2006*, Vancouver, Canada, July 16-21 2006b.

I. Guyon, A. Saffari, G. Dror, and G. Cawley. Agnostic learning vs. prior knowledge challenge. In *IEEE/INNS conference IJCNN 2007*, Orlando, Florida, August 12-17 2007.

I. Guyon, C. Aliferis, G. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A. Statnikov. Design and analysis of the causation and prediction challenge. In *JMLR W&CP*, volume 3, pages 1–33, WCCI2008 workshop on causality, Hong Kong, June 3-4 2008a. URL `http://jmlr.csail.mit.edu/papers/topic/causality.html`.

I. Guyon, A. Saffari, G. Dror, and G. Cawley. Analysis of the IJCNN 2007 agnostic learning vs. prior knowledge challenge. In *Neural Networks*, volume 21, pages 544–550, Orlando, Florida, March 2008b.

I. Guyon, D. Janzing, and B. Schölkopf. Causality: objectives and assessment. In *NIPS 2008 workshop on causality*, volume 7. JMLR W&CP, in press, 2009a.

I. Guyon, V. Lemaire, M. Boullé, Gideon Dror, and David Vogel. Analysis of the KDD cup 2009: Fast scoring on a large orange customer database. In *KDD cup 2009, in press*, volume 8. JMLR W&CP, 2009b.

L. E. Sucar H. J. Escalante, M. Montes. Particle swarm model selection. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 10, pages 405–440, Feb 2009. URL `http://www.jmlr.org/papers/volume10/escalante09a/escalante09a.pdf`.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning, Data Mining, Inference and Prediction*. Springer Verlag, 2000.

T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *JMLR*, 5:1391–1415, 2004. URL http://jmlr.csail.mit.edu/papers/volume5/hastie04a/hastie04a.pdf.

D. Haussler, M. Kearns, and R. Schapire. Bounds on the sample complexity of Bayesian learning using information theory and the vc dimension. *Machine Learning*, 14(1):83–113, 1994. ISSN 0885-6125.

A. E. Hoerl. Application of ridge analysis to regression problems. *Chemical Engineering Progress*, 58:54–59, 1962.

C. Hue and M. Boullé. A new probabilistic approach in rank regression with optimal Bayesian partitioning. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 8, pages 2727–2754, Dec 2007. URL http://www.jmlr.org/papers/volume8/hue07a/hue07a.pdf.

IBM team. Winning the KDD cup orange challenge with ensemble selection. In *KDD cup 2009, in press*, volume 8. JMLR W&CP, 2009.

R. Kohavi and G. John. Wrappers for feature selection. *Artificial Intelligence*, 97(1-2):273–324, December 1997.

I. Koo and R. M. Kil. Model selection for regression with continuous kernel functions using the modulus of continuity. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 9, pages 2607–2633, Nov 2008. URL http://www.jmlr.org/papers/volume9/koo08b/koo08b.pdf.

G. Kunapuli, J.-S. Pang, and K. Bennett. Bilevel cross-validation-based model selection. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

J. Langford. Tutorial on practical prediction theory for classification. *JMLR*, 6:273–306, Mar 2005. URL http://jmlr.csail.mit.edu/papers/volume6/langford05a/langford05a.pdf.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. J. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541 – 551, 1989.

R. W. Lutz. Logitboost with trees applied to the WCCI 2006 performance prediction challenge datasets. In *Proc. IJCNN06*, pages 2966–2969, Vancouver, Canada, July 2006. INNS/IEEE.

D. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4:448–472, 1992.

O. Madani, D. M. Pennock, and G. W. Flake. Co-validation: Using model disagreement to validate classification algorithms. In *NIPS*, 2005.

M. Momma and K. Bennett. A pattern search method for model selection of Support Vector Regression. In *In Proceedings of the SIAM International Conference on Data Mining*. SIAM, 2002.

R. Neal and J. Zhang. High dimensional classification with Bayesian neural networks and dirichlet diffusion trees. In I. Guyon, et al., editor, *Feature Extraction, Foundations and Applications*, 2006.

V. Nikulin. Classification with random sets, boosting and distance-based clustering. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 247(4945):978 – 982, February 1990.

A. Pozdnoukhov and S. Bengio. Invariances in kernel methods: From samples to objects. *Pattern Recogn. Lett.*, 27(10):1087–1097, 2006. ISSN 0167-8655.

E. Pranckeviciene and R. Somorjai. Liknon feature selection: Behind the scenes. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

J. Reunanen. Model selection and assessment using cross-indexing. In *Proc. IJCNN07*, Orlando, Florida, Aug 2007. INNS/IEEE.

S. Rosset and J. Zhu. Sparse, flexible and efficient modeling using L1 regularization. In I. Guyon, et al., editor, *Feature Extraction, Foundations and Applications*, 2006.

S. Rosset, J. Zhu, and T. Hastie. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5:941–973, 2004.

M. Saeed. Hybrid learning using mixture models and artificial neural networks. In I. Guyon, et al., editor, *Hands on Pattern Recognition*. Microtome, 2009.

A. Saffari and I. Guyon. Quick start guide for CLOP. Technical report, Graz University of Technology and Clopinet, May 2006. URL `http://clopinet.com/CLOP/`.

D. Schuurmans and F. Southey. Metric-based methods for adaptive model selection and regularization. *Machine Learning, Special Issue on New Methods for Model Selection and Model Combination*, 48:51–84, 2001.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

M. Seeger. PAC-Bayesian generalisation error bounds for Gaussian process classification. *JMLR*, 3:233–269, 2003. URL `http://jmlr.csail.mit.edu/papers/volume3/seeger02a/seeger02a.pdf`.

M. Seeger. Bayesian inference and optimal design for the sparse linear model. *JMLR*, 9:759–813, 2008. ISSN 1533-7928.

P. Simard, Y. LeCun, and J. Denker. Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 50–58, San Mateo, CA, 1993. Morgan Kaufmann.

A. Singh, R. Nowak, and X. Zhu. Unlabeled data: Now it helps, now it doesn't. In *NIPS*, 2008.

A. Smola, S. Mika, B. Schölkopf, and R. Williamson. Regularized principal manifolds. *JMLR*, 1:179–209, 2001. URL http://jmlr.csail.mit.edu/papers/volume1/smola01a/smola01a.pdf.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

E. Tuv, A. Borisov, G. Runger, and K. Torkkola. Feature selection with ensembles, artificial variables, and redundancy elimination. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 10, pages 1341–1366, Jul 2009. URL http://www.jmlr.org/papers/volume10/tuv09a/tuv09a.pdf.

L. Valiant. A theory of the learnable. *Communications of the ACM,*, 27(11):1134–1142, 1984.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, N.Y., 1998.

V. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian]*. Nauka, Moscow, 1979. (English translation: Springer Verlag, New York, 1982).

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab. Appl.*, 16:264–180, 1971.

S. Vishwanathan and A. Smola. Fast kernels for string and tree matching. In *Advances in Neural Information Processing Systems 15*, pages 569–576. MIT Press, 2003. URL http://books.nips.cc/papers/files/nips15/AA11.pdf.

G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1990.

A. Waibel. Consonant recognition by modular construction of large phonemic time-delay neural networks. In *NIPS*, pages 215–223, 1988.

C. Watkins. Dynamic alignment kernels. In A.J. Smola, P.L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 39–50, Cambridge, MA, 2000. MIT Press. URL http://www.cs.rhul.ac.uk/home/chrisw/dynk.ps.gz.

P. Werbos. Backpropagation: Past and future. In *International Conference on Neural Networks*, pages 343–353. IEEE, IEEE press, 1988.

J. Weston, A. Elisseff, B. Schoelkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *JMLR*, 3:1439–1461, 2003.

J. Wichard. Agnostic learning with ensembles of classifiers. In *Proc. IJCNN07*, Orlando, Florida, Aug 2007. INNS/IEEE.

J. Ye, S. Ji, and J. Chen. Multi-class discriminant kernel learning via convex programming. In I. Guyon and A. Saffari, editors, *JMLR, Special Topic on Model Selection*, volume 9, pages 719–758, Apr 2008. URL http://www.jmlr.org/papers/volume9/ye08b/ye08b.pdf.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *NIPS*, 2003.

Journal of Machine Learning Research 11 (2010) 89-109 Submitted 3/09; Revised 10/09; Published 1/10

On-Line Sequential Bin Packing

András György GYA@SZIT.BME.HU
Machine Learning Research Group
Computer and Automation Research Institute
of the Hungarian Academy of Sciences
Kende u. 13–17, 1111 Budapest, Hungary

Gábor Lugosi GABOR.LUGOSI@GMAIL.COM
ICREA and Department of Economics
Universitat Pompeu Fabra
Ramon Trias Fargas 25–27
08005 Barcelona, Spain

György Ottucsák OTTUCSAK@GMAIL.COM
Department of Computer Science and Information Theory
Budapest University of Technology and Economics
Magyar Tudósok Körútja 2
1117 Budapest, Hungary

Editor: Shie Mannor

Abstract

We consider a sequential version of the classical bin packing problem in which items are received one by one. Before the size of the next item is revealed, the decision maker needs to decide whether the next item is packed in the currently open bin or the bin is closed and a new bin is opened. If the new item does not fit, it is lost. If a bin is closed, the remaining free space in the bin accounts for a loss. The goal of the decision maker is to minimize the loss accumulated over n periods. We present an algorithm that has a cumulative loss not much larger than any strategy in a finite class of reference strategies for any sequence of items. Special attention is payed to reference strategies that use a fixed threshold at each step to decide whether a new bin is opened. Some positive and negative results are presented for this case.

Keywords: bin packing, on-line learning, prediction with expert advice

1. Introduction

In the classical *off-line* bin packing problem, an algorithm receives *items* (also called *pieces*) of size $x_1, x_2, \ldots, x_n \in (0,1]$. We have an infinite number of bins, each with capacity 1, and every item is to be assigned to a bin. Further, the sum of the sizes of the items (also denoted by x_t) assigned to any bin cannot exceed its capacity. A bin is empty if no item is assigned to it, otherwise, it is used. The goal of the algorithm is to minimize the number of used bins. This is one of the classical NP-hard problems and heuristic and approximation algorithms have been investigated thoroughly, see, for example, Coffman et al. (1997).

Another well-studied version of the problem is the so-called *on-line* bin packing problem. Here items arrive one by one and each item x_t must be assigned to a bin (with free space at least x_t)

immediately, without any knowledge of the next pieces. In this setting the goal is the same as in the off-line problem, that is, the number of used bins is to be minimized, see, for example, Seiden (2002).

In both the off-line and on-line problems the algorithm has access to the bins in arbitrary order. In this paper we abandon this assumption and introduce a more restricted version that we call *sequential bin packing*. In this setting items arrive one by one (just like in the on-line problem) but in each round the algorithm has only two possible choices: assign the given item to the (only) open bin or to the "next" empty bin (in this case this will be the new open bin), and items cannot be assigned anymore to closed bins. An algorithm thus determines a sequence of binary decisions $i_1, \ldots, i_n$ where $i_t = 0$ means that the next item is assigned to the open bin and $i_t = 1$ means that a new bin is opened and the next item is assigned to that bin. Of course, if $i_t = 0$, then it may happen that the item x_t does not fit in the open bin. In that case the item is "lost." If the decision is $i_t = 1$ then the remaining empty space in the last closed bin is counted as a loss. The measure of performance we use is the total sum of all lost items and wasted empty space.

Just as in the original bin packing problem, we may distinguish off-line and on-line versions of the sequential bin packing problem. In the *off-line sequential* bin packing problem the entire sequence $x_1, \ldots, x_n$ is known to the algorithm at the outset. Note that unlike in the classical bin packing problem, the order of the items is relevant. This problem turns out to be computationally significantly easier than its non-sequential counterpart. In Section 3 we present a simple algorithm with running time of $O(n^2)$ that minimizes the total loss in the off-line sequential bin packing problem.

Much more interesting is the on-line variant of the sequential bin packing problem. Here the items x_t are revealed one by one, *after* the corresponding decision i_t has been made. In other words, each decision has to be made without any knowledge on the size of the item. Formulated this way, the problem is reminiscent of an on-line *prediction problem*, see Cesa-Bianchi and Lugosi (2006). However, unlike in standard formulations of on-line prediction, here the loss the predictor suffers depends not only on the outcome x_t and decision i_t but also on the "state" defined by the fullness of the open bin.

Our goal is to extend the usual bin packing problems to situations in which one can handle only one bin at a time, and items must be processed immediately so they cannot wait for bin changes. To motivate the on-line sequential model, one may imagine a simple revenue management problem in which a decision maker has a unit storage capacity at his disposal. A certain product arrives in packages of different size and after each arrival, it has to be decided whether the stored packages are shipped or not. (Storage of the product is costly.) If the stored goods are shipped, the entire storage capacity becomes available again. If they are not shipped one waits for the arrival of the next package. However, if the next package is too large to fit in the remaining open space, it is lost (it will be stored in another warehouse).

In another example of application, a sensor collects measurements that can be compressed to variable size (these are the items). The sensor communicates its measurements by sending frames of some fixed size (bins). Since it has limited memory, it cannot store more data than one frame. To save energy, the sensor must maximize its throughput (the proportion of useful data in each frame) and at the same time minimize data loss (this trade-off is reflected in the definition of the loss function).

Just like in on-line prediction, we compare the performance of an algorithm with the best in a pool of reference algorithms (experts). Given a set of N reference strategies, we construct a

randomized algorithm for the sequential on-line bin packing problem that achieves a cumulative loss (measured as the sum of the total wasted capacity and lost items) that is less than the total loss of the best strategy in the class (determined in hindsight) plus a quantity of the order of $n^{2/3}\ln^{1/3} N$.

Arguably the most natural comparison class contains all algorithms that use a fixed threshold to decide whether a new bin is opened. In other words, reference predictors are parameterized by a real number $p \in (0,1]$. An expert with parameter p simply decides to open a new bin whenever the remaining free space in the open bin is less than p. We call such an expert a *constant-threshold* strategy. First we point out that obtaining uniform regret bounds for this class is difficult. We present some impossibility results in relation to this problem. We also offer some data-dependent bounds for an algorithm designed to compete with the best of all constant-threshold strategies, and show that if item sizes are jittered with a certain noise then a uniform regret bound of the order of $n^{2/3}\ln^{1/3} n$ may be achieved .

The principal difficulty of the problem lies in the fact that each action of the decision maker takes the problem in a new "state" (determined by the remaining empty space in the open bin) which has an effect on future losses. Moreover, the state of the algorithm is typically different from the state of the experts which makes comparison difficult. In related work, Merhav et al. (2002) considered a similar setup in which the loss function has a "memory," that is, the loss of a predictor depends on the loss of past actions. Furthermore, Even-Dar et al. (2005) and Yu et al. (2009) considered the MDP case where the adversarial reward function changes according to some fixed stochastic dynamics. However, there are several main additional difficulties in the present case. First, unlike in Merhav et al. (2002), but similarly to Even-Dar et al. (2005) and Yu et al. (2009), the loss function has an unbounded memory as the state may depend on an arbitrarily long sequence of past predictions. Second, the state space is infinite (the $[0,1)$ interval) and the class of experts we compare to is also infinite, in contrast to both of the above papers. However, the special properties of the bin packing problem make it possible to design a prediction strategy with small regret.

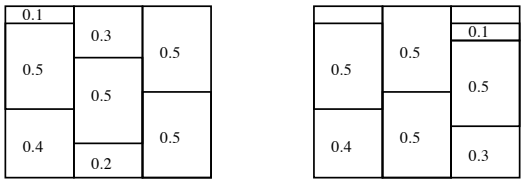
Note that the MDP setting of Even-Dar et al. (2005) and Yu et al. (2009) would be a too pessimistic approach to our problem, as in our case there is a strong connection between the rewards in different states, thus the absolute adversarial reward function results in an overestimated worst case. Also, in the present case, state transitions are deterministically given by the outcome, the previous state, and the action of the decision maker, while in the setup of Even-Dar et al. (2005) and Yu et al. (2009) transitions are stochastic and depend only on the state and the decision of the algorithm, but not on the reward (or on the underlying individual sequence generating the reward).

We also mention here the similar *on-line bin packing with rejection* problem where the algorithm has an opportunity to reject some items and the loss function is the sum of the number of the used bins and the "costs" of the rejected items, see He and Dósa (2005).[1] However, instead of the number of used bins, we use the sum of idle capacities (missed or free spaces) in the used bins to measure the loss.

The following example may help explain the difference between various versions of the problem.

Example 1 *Let the sequence of the items be* $\langle 0.4, 0.5, 0.2, 0.5, 0.5, 0.3, 0.5, 0.1\rangle$. *Then the cumulative loss of the optimal off-line bin packing is* 0 *and it is* 0.4 *in the case of sequential off-line bin packing (see Figure 1). In the sequential case the third item (0.2) has been rejected.*

1. In sequential bin packing we assume that the cost of the items coincides with their size. In this case the optimal solution of bin-packing with rejection is trivially to reject all items.



a) off-line b) sequential off-line

Figure 1: The difference between the optimal solutions for the off-line and sequential off-line problems.

The rest of the paper is organized as follows. In Section 2 the problem is defined formally. In Section 3 the complexity of the off-line sequential bin packing problem is analyzed. The main results of the paper are presented in Sections 4 and 5.

2. Setup

We use a terminology borrowed from the theory of on-line prediction with expert advice. Thus, we call the sequential decisions of the on-line algorithm *predictions* and we use *forecaster* as a synonym for algorithm.

We denote by $I_t \in \{0,1\}$ the action of the forecaster at time t (i.e., when $t-1$ items have been received). Action 0 means that the next item will be assigned to the open bin and action 1 represents the fact that a new bin is opened and the next item is assigned to the next empty bin. Note that we assume that we start with an open empty bin, thus for any reasonable algorithm, $I_1 = 0$, and we will restrict our attention to such algorithms. The sequence of decisions up to time t is denoted by $\mathbf{I}_t \in \{0,1\}^t$.

Denote by $\widehat{s}_t \in [0,1)$ the free space in the open (last) bin at time $t \geq 1$, that is, after having placed the items $x_1, x_2, \ldots, x_t$ according to the sequence $\mathbf{I}_t$ of actions. This is the *state* of the forecaster. More precisely, the state of the forecaster is defined, recursively, as follows: As at the beginning we have an empty bin, $\widehat{s}_0 = 1$. For $t = 1, 2, \ldots, n$,

- $\widehat{s}_t = 1 - x_t$, when the algorithm assigns the item to the next empty bin (i.e., $I_t = 1$);
- $\widehat{s}_t = \widehat{s}_{t-1}$, when the assigned item does not fit in the open bin (i.e., $I_t = 0$ and $\widehat{s}_{t-1} < x_t$);
- $\widehat{s}_t = \widehat{s}_{t-1} - x_t$, when the assigned item fits in the open bin (i.e., $I_t = 0$ and $\widehat{s}_{t-1} \geq x_t$).

This may be written in a more compact form:

$$\begin{aligned} \widehat{s}_t &= \widehat{s}_t(I_t, x_t, \widehat{s}_{t-1}) \\ &= I_t(1-x_t) + (1-I_t)(\widehat{s}_{t-1} - \mathbb{I}_{\{\widehat{s}_{t-1} \geq x_t\}} x_t) \end{aligned}$$

where $\mathbb{I}_{\{\cdot\}}$ denotes the indicator function of the event in brackets, that is, it equals 1 if the event is true and 0 otherwise. The loss suffered by the forecaster at round t is

$$\ell(I_t, x_t \mid \widehat{s}_{t-1}),$$

where the loss function ℓ is defined by

$$\ell(0,x \mid s) = \begin{cases} 0, & \text{if } s \geq x; \\ x, & \text{otherwise} \end{cases} \tag{1}$$

and

$$\ell(1,x \mid s) = s\,. \tag{2}$$

The goal of the forecaster is to minimize its cumulative loss defined by

$$\widehat{L}_t = L_{\mathbf{I}_t,t} = \sum_{s=1}^{t} \ell(I_s, x_s \mid \widehat{s}_{s-1})\,.$$

In the off-line version of the problem, the entire sequence $x_1,\ldots,x_n$ is given and the solution is the optimal sequence $\mathbf{I}_n^*$ of actions

$$\mathbf{I}_n^* = \operatorname*{argmin}_{\mathbf{I}_n \in \{0,1\}^n} L_{\mathbf{I}_n,n}\,.$$

In the on-line version of the problem the forecaster does not know the size of the next items, and the sequence of items can be completely arbitrary. We allow the forecaster to randomize its decisions, that is, at each time instance t, I_t is allowed to depend on a random variable U_t where $U_1,\ldots,U_n$ are i.i.d. uniformly distributed random variables in $[0,1]$.

Since we allow the forecaster to randomize, it is important to clarify that the entire sequence of items are determined *before* the forecaster starts making decisions, that is, $x_1,\ldots,x_n \in (0,1]$ are fixed and cannot depend on the randomizing variables. (This is the so-called *oblivious adversary* model known in the theory of sequential prediction, see, for example, Cesa-Bianchi and Lugosi 2006.)

The performance of a sequential on-line algorithm is measured by its cumulative loss. It is natural to compare it to the cumulative loss of the off-line solution $\mathbf{I}_n^*$. However, it is easy to see that in general it is impossible to achieve an on-line performance that is comparable to the optimal solution. (This is in contrast with the non-sequential counterpart of the bin packing problem in which there exist on-line algorithms for which the number of used bins is within a constant factor of that of the optimal solution, see Seiden 2002.)

So in order to measure the performance of a sequential on-line algorithm in a meaningful way, we adopt an approach extensively used in on-line prediction (the so-called "experts" framework). We define a set of reference forecasters, the so-called *experts*. The performance of the algorithm is evaluated relative to this set of experts, and the goal is to perform asymptotically as well as the best expert from the reference class.

Formally, let $f_{E,t} \in \{0,1\}$ be the decision of an expert E at round t, where $E \in \mathcal{E}$ and $\mathcal{E}$ is the set of the experts. This set may be finite or infinite, we consider both cases below. Similarly, we denote the state of expert E with $s_{E,t}$ after the t-th item has been revealed. Then the loss of expert E at round t is

$$\ell(f_{E,t}, x_t \mid s_{E,t-1})$$

and the cumulative loss of expert E is

$$L_{E,n} = \sum_{t=1}^{n} \ell(f_{E,t}, x_t \mid s_{E,t-1}).$$

SEQUENTIAL ON-LINE BIN PACKING PROBLEM WITH EXPERT ADVICE

Parameters: set $\mathcal{E}$ of experts, state space $\mathcal{S} = [0,1)$, action space $\mathcal{A} = \{0,1\}$, non-negative loss function $\ell : (\mathcal{A} \times (0,1] | \mathcal{S}) \to [0,1)$, number n of items.
Initialization: $\widehat{s}_0 = 1$ and $s_{E,0} = 1$ for all $E \in \mathcal{E}$.
For each round $t = 1, \ldots, n$,

(a) each expert forms its action $f_{E,t} \in \mathcal{A}$;

(b) the forecaster observes the actions of the experts and forms its own decision $I_t \in \mathcal{A}$;

(c) the next item $x_t \in (0,1]$ is revealed;

(d) the algorithm incurs loss $\ell(I_t, x_t \mid \widehat{s}_{t-1})$ and each expert $E \in \mathcal{E}$ incurs loss $\ell(f_{E,t}, x_t \mid s_{E,t-1})$. The states of the experts and the algorithm are updated.

Figure 2: Sequential on-line bin packing problem with expert advice.

The goal of the algorithm is to perform almost as well as the best expert from the reference class $\mathcal{E}$ (determined in hindsight). Ideally, the normalized difference of the cumulative losses (the so-called *regret*) should vanish as n grows, that is, one wishes to achieve

$$\limsup_{n\to\infty} \frac{1}{n}(\widehat{L}_n - \inf_{E\in\mathcal{E}} L_{E,n}) \le 0$$

with probability one, regardless of the sequence of items. This property is called *Hannan consistency*, see Hannan (1957). The model of sequential on-line bin packing with expert advice is given in Figure 2.

In Sections 4 and 5 we design sequential on-line bin packing algorithms. In Section 4 we assume that the class $\mathcal{E}$ of experts is finite. For this case we establish a uniform regret bound, regardless of the class and the sequence of items. In Section 5 we consider the (infinite) class of experts defined by constant-threshold strategies. This case turns out to be considerably more difficult. We show that algorithms, similar (in some sense) to the one developed for the finite expert classes, cannot work in general in this situation. We provide a data-dependent regret bound for a generalization of the finite-expert algorithm of Section 4, which, in accordance with the previous result, does not guarantee consistency in general. However, we show that if the item sizes are jittered with certain noise, the regret of the algorithm vanishes uniformly regardless of the original sequence of items.

But before turning to the on-line problem, we show how the off-line problem can be solved by a simple quadratic-time algorithm.

3. Sequential Off-line Bin Packing

As it is well known, most variants of the bin packing problem are NP-hard, including bin packing with rejection, see He and Dósa (2005), and maximum resource bin packing, see Boyar et al. (2006).

In this section we show that the sequential bin packing problem is significantly easier. Indeed, we offer an algorithm to find the optimal sequential strategy with time complexity $O(n^2)$ where n is the number of the items.

The key property is that after the t-th item has been received, the 2^t possible sequences of decisions cannot lead to more than t different states.

Lemma 1 *For any fixed sequence of items $x_1, x_2, \ldots, x_n$ and for every $1 \leq t \leq n$,*

$$|\mathcal{S}_t| \leq t,$$

where

$$\mathcal{S}_t = \{s : s = s_{\mathbf{I}_t,t}, \mathbf{I}_t \in \{0,1\}^t\}$$

and $s_{\mathbf{I}_t,t}$ is the state reached after receiving items $x_1, \ldots, x_t$ with the decision sequence $\mathbf{I}_t$.

Proof The proof goes by induction. Note that since $I_1 = 0$, we always have $s_{\mathbf{I}_1,1} = 1 - x_1$, and therefore $|\mathcal{S}_1| = 1$. Now assume that $|\mathcal{S}_{t-1}| \leq t-1$. At time t, the state of every sequence of decisions with $I_t = 0$ belongs to the set $\mathcal{S}'_t = \{s' : s' = s - \mathbb{I}_{\{s \geq x_t\}} x_t, s \in \mathcal{S}_{t-1}\}$ and the state of those with $I_t = 1$ becomes $1 - x_t$. Therefore,

$$|\mathcal{S}_t| \leq |\mathcal{S}'_t| + 1 \leq |\mathcal{S}_{t-1}| + 1 \leq t$$

as desired. ■

To describe a computationally efficient algorithm to compute $\mathbf{I}^*_n$, we set up a graph with the set of possible states as a vertex set (there are $O(n^2)$ of them by Lemma 1) and we show that the shortest path on this graph yields the optimal solution of the sequential off-line bin packing problem.

To formalize the problem, consider a finite directed acyclic graph with a set of vertices $V = \{v_1, \ldots, v_{|V|}\}$ and a set of edges $E = \{e_1, \ldots, e_{|E|}\}$. Each vertex $v_k = v(s_k, t_k)$ of the graph is defined by a time index t_k and a state $s_k \in \mathcal{S}_{t_k}$ and corresponds to state s_k reachable after t_k steps. To show the latter dependence, we will write $v_k \in \mathcal{S}_{t_k}$. Two vertices (v_i, v_j) are connected by an edge if and only if $v_i \in \mathcal{S}_{t-1}$, $v_j \in \mathcal{S}_t$ and state v_j is reachable from state v_i. That is, by choosing either action 0 or action 1 in state v_i, the new state becomes v_j after item x_t has been placed. Each edge has a label and a weight: the label corresponds to the action (zero or one) and the weight equals the loss, depending on the initial state, the action, and the size of the item. Figure 3 shows the proposed graph. Moreover a sink vertex $v_{|V|}$ is introduced that is connected with all vertices in $\mathcal{S}_n$. These edges have weight equal to the loss of the final states. These losses only depend on the initial state of the edges. More precisely, for $(v_i, v_{|V|})$ the loss is $1 - s_i$, where $v_i \in \mathcal{S}_n$.

Notice that there is a one to one correspondence between paths from v_1 to $v_{|V|}$ and possible sequences of actions of length n. Furthermore, the total weight of each path (calculated as the sum of the weights on the edges of the path) is equal to the loss of the corresponding sequence of actions. Thus, if we find a path with minimal total weight from v_1 to $v_{|V|}$, we also find the optimal sequence of actions for the off-line bin packing problem. It is well known that this can be done in $O(|V| + |E|)$ time.[2]

Now by Lemma 1, $|V| \leq n(n+1)/2 + 1$, where the additional vertex accounts for the sink. Moreover it is easy to see that $|E| \leq n(n-1) + n = n^2$. Hence the total time complexity of finding the off-line solution is $O(n^2)$.

2. Here we assume the simplified computational model that referring to each vertex (and edge) requires a constant number of operations. In a more refined computational model this may be scaled with an extra $\log |V|$ factor.

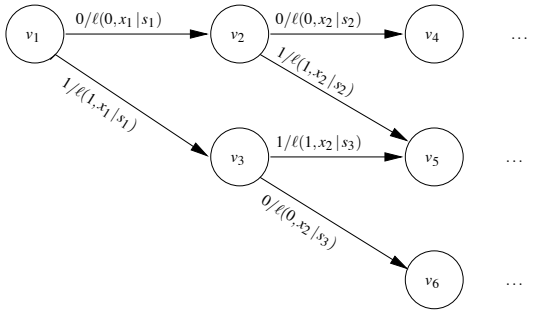


Figure 3: The graph corresponding to the off-line sequential bin packing problem.

4. Sequential On-line Bin Packing

In this section we study the sequential on-line bin packing problem with expert advice, as described in Section 2. We deal with two special cases. First we consider finite classes of experts (i.e., reference algorithms) without any assumption on the form or structure of the experts. We construct a randomized algorithm that, with large probability, achieves a cumulative loss not larger than that of the best expert plus $O(n^{2/3}\ln^{1/3}N)$ where $N=|\mathcal{E}|$ is the number of experts.

The following simple lemma is a key ingredient of the results of this section. It shows that in sequential on-line bin packing the cumulative loss is not sensitive to the initial states in the sense that the cumulative loss depends on the initial state in a minor way.

Lemma 2 *Let $i_1,\ldots,i_m\in\{0,1\}$ be a fixed sequence of decisions and let $x_1,\ldots,x_m\in(0,1]$ be a sequence of items. Let $s_0,s_0'\in[0,1)$ be two different initial states. Finally, let $s_0,\ldots,s_m$ and $s_0',\ldots,s_m'$ denote the sequences of states generated by $i_1,\ldots,i_m$ and $x_1,\ldots,x_m$ starting from initial states s_0 and s_0', respectively. Then*

$$\left|\sum_{t=1}^{m}\ell(i_t,x_t\mid s_{t-1}')-\sum_{t=1}^{m}\ell(i_t,x_t\mid s_{t-1})\right|\le s_0'+s_0\le 2\,.$$

Proof Let m' denote the smallest index for which $i_{m'}=1$. Note that $s_{t-1}=s_{t-1}'$ for all $t>m'$. Therefore, we have

$$\begin{aligned}
&\sum_{t=1}^{m}\ell(i_t,x_t\mid s_{t-1}')-\sum_{t=1}^{m}\ell(i_t,x_t\mid s_{t-1})\\
&\quad=\sum_{t=1}^{m'}\ell(i_t,x_t\mid s_{t-1}')-\sum_{t=1}^{m'}\ell(i_t,x_t\mid s_{t-1})\\
&\quad=\sum_{t=1}^{m'-1}\ell(0,x_t\mid s_{t-1}')-\sum_{t=1}^{m'-1}\ell(0,x_t\mid s_{t-1})+\ell(1,x_{m'}\mid s_{m'-1}')-\ell(1,x_{m'}\mid s_{m'-1})\,.
\end{aligned}$$

Now using the definition of the loss (see Equations 1 and 2), we write

$$\begin{aligned}
&\sum_{t=1}^{m} \ell(i_t, x_t \mid s'_{t-1}) - \sum_{t=1}^{m} \ell(i_t, x_t \mid s_{t-1}) \\
&= \sum_{t=1}^{m'-1} x_t (\mathbb{I}_{\{s'_{t-1} < x_t\}} - \mathbb{I}_{\{s_{t-1} < x_t\}}) + s'_{m'-1} - s_{m'-1} \\
&\le \sum_{t=1}^{m'-1} x_t (1 - \mathbb{I}_{\{s_{t-1} < x_t\}}) + s'_{m'-1} - s_{m'-1} \\
&\le \sum_{t=1}^{m'-1} x_t (1 - \mathbb{I}_{\{s_{t-1} < x_t\}}) + s'_0 \\
&\le s_0 + s'_0
\end{aligned}$$

where the next-to-last inequality holds because $s'_{m'-1} \le s'_0$ and $s_{m'-1} \ge 0$, and the last inequality follows from the fact that

$$\begin{aligned}
0 \le s_{m'-1} &= s_{m'-2} - \mathbb{I}_{\{s_{m'-2} \ge x_{m'-1}\}} x_{m'-1} \\
&= s_{m'-3} - \mathbb{I}_{\{s_{m'-3} \ge x_{m'-2}\}} x_{m'-2} - \mathbb{I}_{\{s_{m'-2} \ge x_{m'-1}\}} x_{m'-1} \\
&= s_0 - \sum_{t=1}^{m'-1} \mathbb{I}_{\{s_{t-1} \ge x_t\}} x_t \ .
\end{aligned}$$

Similarly,

$$\sum_{t=1}^{m} \ell(i_t, x_t \mid s_{t-1}) - \sum_{t=1}^{m} \ell(i_t, x_t \mid s'_{t-1}) \le s'_0 + s_0$$

and the statement follows. ■

The following example shows that the upper bound of the lemma is tight.

Example 2 *Let $x_1 = s_0$, $s'_0 < s_0$, and $m' = 2$. Then*

$$\begin{aligned}
&\sum_{t=1}^{m} \ell(i_t, x_t \mid s'_{t-1}) - \sum_{t=1}^{m} \ell(i_t, x_t \mid s_{t-1}) \\
&= \ell(0, x_1 \mid s'_0) + \ell(1, x_2 \mid s'_1) - \big(\ell(0, x_1 \mid s_0) + \ell(1, x_2 \mid s_1)\big) \\
&= \ell(0, s_0 \mid s'_0) + \ell(1, x_2 \mid s'_0) - \big(\ell(0, s_0 \mid s_0) + \ell(1, x_2 \mid 0)\big) \\
&= s_0 + s'_0 - (0 + 0) \ .
\end{aligned}$$

Now we consider the on-line sequential bin packing problem when the goal of the algorithm is to keep its cumulative loss close to the best in a finite set of experts. In other words, we assume that the class of experts is finite, say $|\mathcal{E}| = N$, but we do not assume any additional structure of the experts. The ideas presented here will be used in Section 5 when we consider the infinite class of constant-threshold experts.

The proposed algorithm partitions the time period $t = 1, \ldots, n$ into segments of length m where $m < n$ is a positive integer whose value will be specified later. This way we obtain $n' = \lfloor n/m \rfloor$

segments of length m, and, if m does not divide n, an extra segment of length less than m. At the beginning of each segment, the algorithm selects an expert randomly, according to an exponentially weighted average distribution. During the entire segment, the algorithm follows the advice of the selected expert. By changing actions so rarely, the algorithm achieves a certain synchronization with the chosen expert, since the effect of the difference in the initial states is minor, according to Lemma 2. (A similar idea was used in Merhav et al. (2002) in a different context.) The algorithm is described in Figure 4. Recall that each expert $E \in \mathcal{E}$ recommends an action $f_{E,t} \in \{0,1\}$ at every time instance $t = 1,\dots,n$. Since we have N experts, we may identify $\mathcal{E}$ with the set $\{1,\dots,N\}$. Thus, experts will be indexed by the positive integers $i \in \{1,\dots,N\}$. At the beginning of each segment, the algorithm chooses expert i randomly, with probability $p_{i,t}$, where the distribution $\mathbf{p}_t = (p_{1,t},\dots,p_{N,t})$ is specified in the algorithm. The random selection is made independently for each segment.

The following theorem establishes a performance bound of the algorithm. Recall that $\widehat{L}_n$ denotes the cumulative loss of the algorithm while $L_{i,n}$ is that of expert i.

Theorem 3 *Let n, $N \geq 1$, $\eta > 0$, $1 \leq m \leq n$, and $\delta \in (0,1)$. For any sequence $x_1,\dots,x_n \in (0,1]$ of items, the cumulative loss $\widehat{L}_n$ of the randomized strategy defined in Figure 4 satisfies for all $i = 1,\dots,N$, with probability at least $1-\delta$,*

$$\widehat{L}_n \leq L_{i,n} + \frac{m}{\eta}\ln\frac{1}{w_{i,0}} + \frac{m\eta}{8} + \sqrt{\frac{nm}{2}\ln\frac{1}{\delta}} + \frac{2n}{m} + 2m.$$

In particular, choosing $w_{i,0} = 1/N$ for all $i = 1,\dots,N$, $m = (16n/\ln(N/\delta))^{1/3}$ and $\eta = \sqrt{8m\ln N/n}$, one has

$$\widehat{L}_n - \min_{i=1,\dots,N} L_{i,n} \leq \frac{3}{\sqrt[3]{2}} n^{2/3}\ln^{1/3}\frac{N}{\delta} + 4\left(\frac{2n}{\ln(N/\delta)}\right)^{1/3}.$$

Proof We introduce an auxiliary quantity, the so-called *hypothetical loss*, defined as the loss the algorithm would suffer if it had been in the same state as the selected expert. This hypothetical loss does not depend on previous decisions of the algorithm. More precisely, the true loss of the algorithm at time instance t is $\ell(I_t, x_t \mid \widehat{s}_t)$ and its hypothetic loss is $\ell(I_t, x_t \mid s_{J_t,t})$. Introducing the notation

$$\ell_{i,t} = \ell(f_{i,t}, x_t \mid s_{i,t}),$$

the hypothetical loss of the algorithm is just

$$\ell(I_t, x_t \mid s_{J_t,t}) = \ell(f_{J_t,t}, x_t \mid s_{J_t,t}) = \ell_{J_t,t}.$$

Now it follows by a well-known result of randomized on-line prediction (see, e.g., Lemma 5.1 and Corollary 4.2 in Cesa-Bianchi and Lugosi, 2006) that the hypothetical loss of the sequential on-line bin packing algorithm satisfies simultaneously for all $i = 1,\dots,N$, with probability at least $1-\delta$,

$$\sum_{t=1}^{n}\ell_{J_t,t} \leq \sum_{t=1}^{n}\ell_{i,t} + m\left(\frac{1}{\eta}\ln\frac{1}{w_{i,0}} + \frac{n'\eta}{8} + \sqrt{\frac{n'}{2}\ln\frac{1}{\delta}}\right) + m, \tag{3}$$

where $n' = \lfloor \frac{n}{m} \rfloor$ and the last m term comes from bounding the difference on the last, not necessarily complete segment. Now we may decompose the regret relative to expert i as follows:

$$\widehat{L}_n - L_{i,n} = \left(\widehat{L}_n - \sum_{t=1}^{n}\ell_{J_t,t}\right) + \left(\sum_{t=1}^{n}\ell_{J_t,t} - L_{i,n}\right).$$

SEQUENTIAL ON-LINE BIN PACKING ALGORITHM

Parameters: Real number $\eta > 0$ and $m \in \mathbb{N}^+$.
Initialization: $\widehat{s}_0 = 1$, $s_{i,0} = 1$ and $w_{i,0} > 0$ are set arbitrarily for $i = 1,\ldots,N$ such that $w_{1,0} + w_{2,0} + \cdots + w_{N,0} = 1$.
For each round $t = 1,\ldots,n$,

(a) If $((t-1) \bmod m) = 0$ then

- calculate the updated probability distribution

$$p_{i,t} = \frac{w_{i,t-1}}{\sum_{j=1}^{N} w_{j,t-1}}$$

for $i = 1,\ldots,N$;

- randomly select an expert $J_t \in \{1,\ldots,N\}$ according to the probability distribution $\mathbf{p}_t = (p_{1,t},\ldots,p_{N,t})$;

otherwise, let $J_t = J_{t-1}$.

(b) Follow the chosen expert: $I_t = f_{J_t,t}$.

(c) The size of next item $x_t \in (0,1]$ is revealed.

(d) The algorithm incurs loss

$$\ell(I_t, x_t \mid \widehat{s}_{t-1})$$

and each expert i incurs loss $\ell(f_{i,t}, x_t \mid s_{i,t-1})$. The states of the experts and the algorithm are changed.

(e) Update the weights

$$w_{i,t} = w_{i,t-1} e^{-\eta \ell(f_{i,t}, x_t \mid s_{i,t-1})}$$

for all $i \in \{1,\ldots,N\}$.

Figure 4: Sequential on-line bin packing algorithm.

The second term on the right-hand side is bounded using (3). To bound the first term, observe that by Lemma 2,

$$\begin{aligned}
\widehat{L}_n - \sum_{t=1}^{n} \ell_{J_t,t} &= \sum_{t=1}^{n} \ell(I_t, x_t \mid \widehat{s}_{t-1}) - \sum_{t=1}^{n} \ell(I_t, x_t \mid s_{J_{t-1},t-1}) \\
&\le m + \sum_{s=0}^{n'-1} \sum_{t=1}^{m} \left(\ell(I_{sm+t}, x_{sm+t} \mid \widehat{s}_{sm+t-1}) - \ell(I_{sm+t}, x_{sm+t} \mid s_{J_{sm+t-1},sm+t-1}) \right) \\
&\le m + 2n'
\end{aligned}$$

where in the first inequality we bounded the difference on the last segment separately. ■

5. Constant-threshold Experts

In this section we address the sequential on-line bin packing problem when the goal is to perform almost as well as the best in the class of all constant-threshold strategies. Recall that a constant-threshold strategy is parameterized by a number $p \in (0,1]$ and it opens a new bin if and only if the remaining empty space in the bin is less than p. More precisely, if the state of the algorithm defined by expert with parameter p is $s_{p,t-1}$, then at time t the expert's advice is $\mathbb{I}_{\{s_{p,t-1}<p\}}$. To simplify notation, we will refer to each expert with its parameter, and, similarly to the previous section, $f_{p,t}$ and $s_{p,t}$ will denote the decision of expert p at time t, and its state after the decision, respectively.

The difficulty in this setup is that there are uncountably many constant-threshold experts. The simplest possibility is to discretize the class. For example, by considering the set of constant-threshold experts with values of p in the set $\{1/N, 2/N, \ldots, 1\}$ and using the randomized algorithm described in the previous section, we immediately obtain that the cumulative regret of the algorithm, when compared to the best constant-threshold expert with p in this set is bounded by $O(n^{2/3}\ln^{1/3} N)$ with high probability. It is natural to suspect that if N is large, the loss of the best discretized constant-threshold expert is not much larger than that corresponding to the best (unrestricted) value of $p \in (0,1]$. However, this is not true in general. The next lemma shows that any such discretization is doomed to failure, at least in the worst-case sense. We denote by $L_{p,n}$ the cumulative loss of the constant-threshold expert indexed by $p \in (0,1]$.

Lemma 4 *For all n such that $n/4$ is a positive integer and $1/2 < a < b \le 1$ there exists a sequence $x_1, \ldots, x_n$ of items such that*

$$\sup_{p \in (a,b]} L_{p,n} < \inf_{p \notin (a,b]} L_{p,n} - \frac{n}{4} + 3$$

for any values of the initial states $s_{p,0} \in [p,1], p \in (0,1]$.[3]

Proof Given $1/2 \le a < b \le 1$, we construct a sequence with the announced property. The first fourth of the sequence is defined by $x_1 = 1-a$ and $x_2 = \cdots = x_{n/4} = 1$. If an expert asks for a new bin after the first item then it suffers no loss for $t = 2, \ldots, n/4$, thus the cumulative loss up to time $n/4$ is bounded as $L_{p,n/4} \le 1$. Note that any expert with parameter $p > a$ is such, as the first item always fits the actual bin, as by the conditions of the lemma $1-a \le a < p \le s_{p,0}$, but then the empty space becomes $s_{0,p} - (1-a) \le a < p$, and so expert p opens a new bin. In case of an expert with parameter $q \le a$, it depends on the initial state if the expert opens a new bin. If the actual bin is left open after the first item then the expert suffers loss $L_{q,n/4} = n/4 - 1$. In particular, if $s_{q,0} = 1$ then after the first item expert q moves to state $s_{q,1} = a$ and leaves the bin open. Thus, after time $n/4$ an expert either suffers loss at least $n/4-1$ (then the parameter of the expert is at most a), or it suffers loss at most 1, but then it is in the state $s_{p,n/4} = 1$. Now for the second forth of the sequence repeat the first one, that is, let $x_{n/4+1} = 1-a, x_{n/4+2} = \cdots = x_{n/2} = 1$. By the above argument we can see that if an expert with parameter $q \le a$ does not suffer large loss up to time $n/4$ then it starts with an empty bin and suffers a large loss in the second fourth of the segment. Thus, $L_{q,n/2} \ge n/4 - 1$ for any $q \le a$. On the other hand, for any expert $p > a$ we have $L_{p,n/2} < 2$ and $s_{p,n/2} = 1$.

3. Note that for any expert $p \in (0,1]$, $s_{p,t} \in [p,1]$ for all $t \ge 1$ regardless of the initial state, and so it is natural to restrict the initial state to $[p,1]$, as well.

After this point of time, let $x_{n/2+1} = 1-b$, $x_{n/2+2} = b$ and repeat this pair of items $n/4$ times. After receiving $x_{n/2+1} = 1-b$, every expert with parameter $p \in (a,b]$ keeps the bin open and therefore does not suffer any loss after receiving the next item. On the other hand, experts with parameter $r > b$ close the bin, suffer loss b, and after $x_{n/2+2} = b$ is received, once again they close the bin and suffer loss $1-b$ (here we used the fact that $r > 1-b$ since we assumed $b > 1/2$. Thus, between periods $n/2+1$ and n, all experts with $p \in (a,b]$ suffer zero loss while experts with parameter $r > b$ suffer loss $n/4$.

Summarizing, for the sequence

$$1-a, \underbrace{1,1,\ldots,1}_{n/4-1 \text{ periods}}, 1-a, \underbrace{1,1,\ldots,1}_{n/4-1 \text{ periods}}, \underbrace{1-b,b,1-b,b\ldots,1-b,b}_{n/2 \text{ periods}},$$

we have

$$L_{p,n} \begin{cases} < 2 & \text{if } p \in (a,b] \\ \geq n/4-1 & \text{if } p \leq a \\ \geq n/4 & \text{if } p > b. \end{cases}$$

■

Lemma 4 implies that one cannot expect a small regret with respect to all possible constant-threshold experts. This is true for any algorithm that, as the one proposed in the previous section, divides time into segments and on each segment chooses a constant-threshold expert and acts as the chosen expert during the following segment. Recall that this segmentation was necessary to make sure that the state of the algorithm gets synchronized with the chosen one. The statement is formalized below.

Theorem 5 *Consider any sequential on-line bin packing algorithm that divides time into segments of lengths $m_1, m_2, \ldots, m_k \geq 3$ (where $\sum_{i=1}^k m_i = n$) such that, at the beginning of each segment m_i, the algorithm chooses (in a possibly randomized way) a parameter $p_i \in (0,1]$ and follows this expert during the segment, that is, $I_t = \mathbb{I}_{\{\widehat{s}_{t-1} < p_i\}}$ for all $t = \sum_{j=1}^{i-1} m_j + 1, \ldots, \sum_{j=1}^{i} m_j$. Then there exists a sequence of items $x_1, \ldots, x_n$ such that the loss of the algorithm satisfies, with probability at least $1/2$,*

$$\widehat{L}_n \geq \inf_{p \in (0,1]} L_{p,n} + \frac{n}{4} - 6k\,.$$

Proof We construct the sequence of items using the sequence shown in the proof of Lemma 4 as a building block. At time 1, divide the interval $(0,1]$ into $2k$ subintervals of equal length and choose one of these intervals uniformly at random. Denote the end points of this interval by $(A_1, B_1]$. Then during the first segment we define the items by

$$1-A_1, \underbrace{1,1,\ldots,1}_{\lfloor m_1/4 \rfloor - 1 \text{ periods}}, 1-A_1, \underbrace{1,1,\ldots,1}_{\lfloor m_1/4 \rfloor - 1 \text{ periods}}, \underbrace{1-B_1,B_1,1-B_1,B_1\ldots,1-B_1,B_1}_{\lfloor m_1/2 \rfloor \text{ periods}}\,.$$

If m_1 is not divisible by 4, we may define the remaining (at most three) items arbitrarily. Then, according to Lemma 4, if the algorithm does not choose an expert to follow from the interval $(A_1, B_1]$ then its loss is larger by at least $\frac{m_1}{4} - 6$ than that of any expert in $(A_1, B_1]$. (The extra 3 come from

the possibility that m_1 is not divisible by 4.) However, no matter how the algorithm chooses the expert to follow, the probability that it finds the correct subinterval is $1/(2k)$.

To continue the construction, we now divide the interval $(A_1, B_1]$ into $2k$ intervals of equal length and choose one at random, say $(A_2, B_2]$. We define the next items similarly to the first segment, but now we make sure that the optimal constant-threshold expert falls in the interval $(A_2, B_2]$, that is, the items of the second segment are defined by

$$1-A_2, \underbrace{1,1,\ldots,1}_{\lfloor m_2/4\rfloor-1 \text{ periods}}, 1-A_2, \underbrace{1,1,\ldots,1}_{\lfloor m_2/4\rfloor-1 \text{ periods}}, \underbrace{1-B_2, B_2, 1-B_2, B_2 \ldots, 1-B_2, B_2}_{\lfloor m_2/2\rfloor \text{ periods}}.$$

As before, if m_2 is not divisible by 4, we may define the remaining (at most three) items arbitrarily. Once again, the excess loss of the algorithm, when compared to the best constant-threshold expert, is at least $\frac{m_2}{4} - 6$ with probability $1/(2k)$.

We may continue the same randomized construction of the item sizes in the same manner, always dividing the previously chosen interval into $2k$ equal pieces, choosing one at random, and constructing the item sequence so that experts in the chosen interval are significantly better than any other expert.

By the union bound, the probability that the forecaster never chooses the correct interval is at least $1/2$, so with probability at least $1/2$,

$$\widehat{L}_n - \inf_{p\in(0,1]} L_{p,n} \geq \sum_{i=1}^{k} \left(\frac{m_i}{4} - 6\right) = \frac{n}{4} - 6k$$

as desired. ■

The theorem above shows that if one uses a segmentation for synchronization purposes, one cannot expect nontrivial regret bounds that hold uniformly over all possible sequences of items and for all constant-threshold experts, unless the number of segments is proportional to n. It seems unlikely that without such synchronization one may achieve $o(n)$ regret. Unfortunately, we do not have a formal proof for arbitrary algorithms (that do not divide time into segments).

However, one may still obtain meaningful regret bounds that depend on the data. We derive such a bound next. We also show that under some natural restrictions on the item sizes, this result allows us to derive regret bounds that hold uniformly over all constant-threshold experts.

In order to understand the structure of the problem of constant-threshold experts, it is important to observe that on any sequence of n items, experts can exhibit only a finite number of different behaviors. In a sense, the "effective" number of experts is not too large and this fact may be exploited by an algorithm.

For $t = 1,\ldots,n$ we call two experts *t-indistinguishable* (with respect to the sequence of items $x_1,\ldots,x_{t-1}$) if their decision sequences are identical up to time t (note that any two experts are 1-indistinguishable, as all experts p start with a decision $f_{p,1} = 0$). This property defines a natural partitioning of the class of experts into maximal t-indistinguishable sets, where any two experts that belong to the same set are t-indistinguishable, and experts from different sets are not t-indistinguishable. Obviously, there are no more than 2^t maximal t-indistinguishable sets. This bound, although finite, is still too large to be useful. However, it turns out that the number of maximal t-indistinguishable sets only grows at most quadratically with t.

The first step in proving this fact is the next lemma that shows that the maximal t-indistinguishable expert sets are intervals.

Lemma 6 *Let $1 \geq p > r > 0$ be such that expert p and expert r are t-indistinguishable. Then for any $p > q > r$ expert q is t-indistinguishable from both experts p and r. Thus, the maximal t-indistinguishable expert sets form subintervals of $(0,1]$.*

Proof By the assumption of the lemma the decision sequences of experts p and r coincide, that is,

$$f_{p,u} = f_{r,u} \qquad \text{and} \qquad s_{p,u} = s_{r,u}$$

for all $u = 1,2,\ldots,t$. Let $t_1,t_2,\ldots$ denote the time instances when expert p (or expert r) assigns the next item to the next empty bin (i.e., $f_{p,u} = 1$ for $u = t_1,t_2,\ldots$). If expert q also decides 1 at time t_k for some k, then it will decide 0 for $t = t_k+1,\ldots,t_{k+1}-1$ since so does expert p and $p > q$, and will decide 1 at time t_{k+1} as $q > r$. Thus the decision sequence of expert q coincides with that of expert p and r for time instances $t_k+1,\ldots,t_{k+1}$ in this case. Since all experts start with the empty bin at time 0, the statement of the lemma follows by induction. ■

Based on the lemma we can identify the t-indistinguishable sets by their end points. Let $Q_t = \{q_{1,t},\ldots,q_{N_t,t}\}$ denote the set of the end points after receiving $t-1$ items, where $N_t = |Q_t|$ is the number of maximal t-indistinguishable sets, and $q_{0,t} = 0 < q_{1,t} < q_{2,t} < \cdots < q_{N_t,t} = 1$. Then the t-indistinguishable sets are $(q_{k-1,t},q_{k,t}]$ for $k = 1,\ldots,N_t$. The next result shows that the number of maximal t-indistinguishable sets cannot grow too fast.

Lemma 7 *The number of the maximal t-indistinguishable sets is at most quadratic in the number of the items t. More precisely, $N_t \leq 1+t(t-1)/2$ for any $1 \leq t \leq n$.*

Proof The proof is by induction. First, $N_1 = 1$ (and $Q_1 = \{1\}$) since the first decision of each expert is 1. Now assume that $N_t \leq 1+t(t-1)/2$ for some $1 \leq t \leq n-1$. When the next item x_t arrives, an expert p with state s decides 1 in the next step if and only if $0 \leq s - x_t < p$. Therefore, as each expert belonging to the same indistinguishable set has the same state, the k-th maximal $(t-1)$-indistinguishable interval with state s is split into two subintervals if and only if $q_{k-1,t-1} < s - x_t \leq q_{k,t-1}$ (experts in this interval with parameters larger than $s-x_t$ will form one subset, and the ones with parameter at most $s-x_t$ will form the other one). As the number of possible states after t decisions (the number of different possible values of $s-x_t$) is at most t by Lemma 1, it follows that at most t intervals can be split, and so $N_{t+1} \leq N_t + t \leq 1+t(t+1)/2$, where the second inequality holds by the induction hypothesis. ■

Lemma 7 shows that the "effective" number of constant-threshold experts is not too large. This fact makes it possible to apply our earlier algorithm for the case of finite expert classes with reasonable computational complexity. However, note that the number of "distinguishable" experts, that is, the number of the maximal indistinguishable sets, constantly grows with time, and each indistinguishable set contains a continuum number of experts. Therefore we need to redefine the algorithm carefully. This may be done by a two-level random choice of the experts: first we choose an indistinguishable expert set, then we pick one expert from this set randomly. The resulting algorithm is given in Figure 5.

SEQUENTIAL ON-LINE BIN PACKING ALGORITHM WITH CONSTANT-THRESHOLD EXPERTS

Parameters: $\eta > 0$ and $m \in \mathbb{N}^+$.
Initialization: $w_{0,1} = 1, N_1 = 1, Q_1 = \{1\}, s_{1,0} = 1$ and $\widehat{s}_0 = 1$.
For each round $t = 1, \ldots, n$,

(a) If $((t-1) \bmod m) = 0$ then

- for $i = 1, \ldots, N_t$, compute the probabilities

$$p_{i,t} = \frac{w_{i,t-1}}{\sum_{j=1}^{N_t} w_{j,t-1}};$$

- randomly select an interval $J_t \in \{1, \ldots, N_t\}$ according to the probability distribution $\mathbf{p}_t = (p_{1,t}, \ldots, p_{N_t,t})$;
- choose an expert p_t uniformly from the interval $(q_{J_t-1,t}, q_{J_t,t}]$;

otherwise, let $p_t = p_{t-1}$.

(b) Follow the decision of expert p_t: $I_t = f_{p_t,t}$.

(c) $x_t \in (0,1]$, the size of the next item is revealed.

(d) The algorithm incurs loss $\ell(I_t, x_t \mid \widehat{s}_{t-1})$ and each expert $p \in (0,1]$ incurs loss $\ell(f_{p,t}, x_t \mid s_{p,t-1})$, where $p \in [0,1)$.

(e) Compute the state $\widehat{s}_t$ of the algorithm by (1), and calculate the auxiliary weights and states of the expert sets for all $i = 1, \ldots, N_t$ by

$$\begin{aligned} \tilde{w}_{i,t} &= w_{i,t-1} e^{-\eta \ell(f_{i,t}, x_t \mid s_{i,t-1})} \\ \tilde{s}_{i,t} &= f_{i,t}(1-x_t) + (1-f_{i,t})(s_{i,t} - \mathbb{I}_{\{s_{i,t} \geq x_t\}} x_t). \end{aligned}$$

(f) Update the end points of the intervals:

$$Q_{t+1} = Q_t \cup \bigcup_{i=1}^{N_t} \{\tilde{s}_{i,t} : q_{i-1,t} < \tilde{s}_{i,t} \leq q_{i,t}\}$$

and $N_{t+1} = |Q_{t+1}|$.

(g) Assign the new states and weights to the $(t+1)$-indistinguishable sets

$$s_{i,t+1} = \tilde{s}_{j,t} \quad \text{and} \quad w_{i,t+1} = \tilde{w}_{j,t} \frac{q_{i,t+1} - q_{i-1,t+1}}{q_{j,t} - q_{j-1,t}}$$

for all $i = 1, \ldots, N_{t+1}$ and $j = 1, \ldots, N_t$ such that $q_{j-1,t} < q_{i,t+1} \leq q_{j,t}$.

Figure 5: Sequential on-line bin packing algorithm with constant-threshold experts.

Up to step (e) the algorithm is essentially the same as in the case of finitely many experts. The two-level random choice of the expert is performed in step (a). In step (f) we update the t-indistinguishable sets, and usually introduce new indistinguishable expert sets. Because of these new expert sets, the update of the weights $w_{i,t}$ and the states $s_{i,t}$ are performed in two steps, (e) and (g), where the actual update is made in step (e), and reordering of these quantities according to the new indistinguishable sets is performed in step (g) together with the introduction of the weights and states for the newly formed expert sets. (Note that in step (g) the factor $(q_{i,t+1} - q_{i-1,t+1})/(q_{j,t} - q_{j-1,t})$ is the proportion of the lengths of the indistinguishable intervals expert $q_{i,t+1}$ belongs to at times $t+1$ and t.)

The performance and complexity of the algorithm is given in the next theorem.

Theorem 8 *Let $n \geq 1$, $\eta > 0$, $1 \leq m \leq n$, and $\delta \in (0,1)$. For any sequence $x_1, \ldots, x_n \in (0,1]$ of items, the cumulative loss $\widehat{L}_n$ of the randomized strategy defined above satisfies for all $p \in (0,1]$, with probability at least $1-\delta$,*

$$\widehat{L}_n \leq L_{p,n} + \frac{m}{\eta} \ln \frac{1}{l_{p,n}} + \frac{n\eta}{8} + \sqrt{\frac{nm}{2} \ln \frac{1}{\delta}} + \frac{2n}{m} + 2m$$

where $l_{p,n}$ is the length of the maximal n-indistinguishable interval that contains p. Moreover, the algorithm can be implemented with time complexity $O(n^3)$ and space complexity $O(n^2)$.

Remark 9 *(i) By choosing $m \sim n^{1/3}$ and $\eta \sim n^{-1/3}$, the regret bound is of the order of $n^{2/3} \ln(1/l_{p,n})$. Note that the constant $\ln(1/l_{p,n})$ reflects the difficulty of the problem (similarly to, for example, the notion of margin in classification, $l_{p,n}$ measures the freedom in choosing an optimal decision boundary, that is, an optimal threshold). If the indistinguishable interval containing the optimal experts is small, then the problem is hard (and the corresponding penalty term in the bound is large). On the other hand, as $N_n \leq 1 + n(n-1)/2$, if the classes of indistinguishable experts are more or less of uniform size, then the corresponding term in the bound is of the order of $\ln n$. We show below that this is always the case if there is a certain randomness in the item sizes.*

(ii) The way of splitting the weight between new maximal indistinguishable classes in step (g) could be modified in many different ways. For example, instead of assigning weights proportionally to the length of the new intervals, one could simply give half of the weight to both new classes. In this case, instead of the term $\ln(1/l_{p^*,n})$ for the optimal expert p^*, we would get in the bound the number of splits performed until reaching the optimal maximal n-indistinguishable class. The hardness of the problem comes from the fact that the partitioning of the experts into maximal indistinguishable classes is not known in advance. If we knew it, we could just simply apply the algorithm of Theorem 3 to the resulting N_n experts (as in Theorem 4.1 of Cesa-Bianchi and Lugosi, 2006) to obtain a uniformly good bound over all constant-threshold experts.

Proof It is easy to see that the two-level choice of the expert p_t ensures that the algorithm is the same as for the finite expert class with the experts defined by Q_n with initial weights $w_{i,0} = l_{q_{i,n},n} = q_{i,n} - q_{i-1,n}$ for the n-indistinguishable expert class containing $q_{i,n}$. Thus, Theorem 3 can be used to bound the regret, where the number of experts is N_t.

For the second part note that the algorithm has to store the states, the intervals, the weights and the probabilities, each on the order of $O(n^2)$ based on Lemma 7. Concerning time complexity, the algorithm has to update the weights and states in each round (requiring $O(n^2)$ computations per

round), and has to compute the probabilities once in every m step, which requires $O(n^3/m)$ computations. Thus the time complexity of the algorithm is $O(n^3)$. ■

Next we use Theorem 8 to show that, for many natural sequences of items, the algorithm above guarantees a small regret uniformly for all constant-threshold experts. In particular, we show that if item sizes are jittered by random noise, then the algorithm shown above has a small regret with respect to all constant-threshold experts (it is well-known that, for general systems, introducing such random perturbations often reduces the sensitivity, and hence results in a more uniform performance, for different values of the input). To this end, we simply need to show that n-indistinguishable intervals cannot be too short. We consider a simple model when the item sizes are noisy versions of an arbitrary fixed sequence. For simplicity we assume that the noise is uniformly distributed but the result remains true under more general circumstances. For illustration purposes the simplified model is sufficient.

Theorem 10 *Let $y_1,\dots,y_n \in (0,1]$ be arbitrary and define the item sizes by*

$$x_t = \begin{cases} y_t+\sigma_t & \text{if } y_t+\sigma_t \in (0,1] \\ 1 & \text{if } y_t+\sigma_t > 1 \\ 0 & \text{if } y_t+\sigma_t \le 0 \end{cases}$$

where $\sigma_1,\dots,\sigma_n$ are independent random variables, uniformly distributed on the interval $[-\varepsilon,\varepsilon]$ for some $\varepsilon > 0$. If the algorithm of Figure 5 is used with parameters $m = (16n/\ln(n^5/\varepsilon\delta))^{1/3}$ and $\eta = \sqrt{8m\ln(n^5/\varepsilon)/n}$, then with probability at least $1-\delta-1/(4n)$, one has

$$\widehat{L}_n - \min_{p\in(0,1]} L_{p,n} \le \frac{3}{\sqrt[3]{2}} n^{2/3} \ln^{1/3}\frac{n^5}{\varepsilon\delta} + 4\left(\frac{2n}{\ln(n^5/\varepsilon\delta)}\right)^{1/3}. \tag{4}$$

Proof The result follows directly from Theorem 8 if we show that the length of the shortest maximal n-indistinguishable interval is at most ε/n^5 with probability at least $1-1/(4n)$ (with respect to the distribution of the random noise). A very crude bounding suffices to show this. Simply recall from the proof of Lemma 7 that, at time t, a maximal t-indistinguishable interval (p,q) is split if and only if $x_t \in (s+p, s+q)$ where s denotes the state of a corresponding constant-threshold expert. Note that $(s+p,s+q) \subseteq (0,1)$, since $x_t = 0$ or $x_t = 1$ cannot split any maximal t-indistinguishable interval, but any such interval can be split by an appropriately chosen x_t. At time t there are at most $t^2/2$ different maximal t-indistinguishable intervals and at most t different states, so by the union bound, the probability that there exists a maximal t-indistinguishable interval of length at most ε/n^5 that is split at time t is bounded by $t^3/2$ times the probability that $x_t \in (s+p,s+q)$ for a fixed interval with $q-p \le \varepsilon/n^5$. Because of the assumption on how x_t is generated, the latter probability is bounded by $(q-p)/(2\varepsilon) \le 1/(2n^5)$ (the truncation of x_t at 0 and 1 has no effect, because $(s+p,s+q) \subseteq (0,1)$). Hence, the probability that there exists a maximal t-indistinguishable interval of length at most ε/n^5 that is split at time t is no more than $t^3/2 \cdot 1/(2n^5) \le 1/(4n^2)$. Thus, using the union bound again, the probability that during the n rounds of the game there exists any maximal t-indistinguishable interval of length at most ε/n^5 that is split is at most $1/(4n)$, and therefore, with probability at least $1-1/(4n)$, all maximal n-indistinguishable intervals have length at least ε/n^5, as desired. ■

Remark 11 *(i) The theorem above shows that, for example, if* $\varepsilon = \Omega(n^{-a})$ *for some* $a > 0$ *(i.e., if the noise level is not too small), then the regret with respect to the best constant-threshold expert is* $O(n^{2/3}\ln^{1/3} n)$.

(ii) A similar model can be obtained, if, instead of having perturbed item sizes, the experts observe the free space in their bins with some noise. Thus, instead of $s_{p,t-1}$*, expert* p *observes* $s_{p,t-1} + \sigma_{p,t}$ *truncated to the interval* $[0,1]$*, and makes decision* $f_{p,t}$ *based on this value. As in the case of Theorem 10, we assume that the noise is independent over time, that is, the random ensembles* $\{\sigma_{p,t}\}_{p\in(0,1]}$ *are independent for all* t*. If each component is identical, that is,* $\sigma_{p,t} = \sigma_t$ *for all* $p \in (0,1]$*, then essentially the same argument applies as in the previous theorem, and so* (4) *holds if the sequence* $\sigma_1, \ldots, \sigma_n$ *satisfies the assumptions of Theorem 10. On the other hand, if the components of the vectors are also independent, then the problem becomes more difficult, as the* t*-indistinguishable classes may not be disjoint intervals anymore. An intermediate assumption on the noise that still guarantees that* (4) *holds for this scenario is that* $\sigma_{p,t} = \sigma_{q,t}$ *if* p *and* q *are* t*-indistinguishable. Then the same argument as in Theorem 10 works with the only difference (omitting the effects of truncation to* $[0,1]$*) that here we have to estimate the probability that* $x_t \in (s+p+\sigma_{t,q}, s+q+\sigma_{t,q})$ *for a fixed* x_t *instead of estimating the probability that* $x_t \in (s+p, s+q)$ *with a randomized* x_t*. However, it is easy to see that the same bound holds in both cases.*

Finally, we present a simple example that reveals that the loss of the best expert can be arbitrarily far from that of the optimal sequential off-line packing.

Example 3 *Let the sequence of items be*

$$\langle \underbrace{\varepsilon, 1{-}\varepsilon, \varepsilon, 1{-}\varepsilon, \ldots, \varepsilon, 1{-}\varepsilon}_{2k}, \varepsilon, \underbrace{1, 1, \ldots, 1}_{k} \rangle,$$

where the number of items is $n = 3k+1$ *and* $0 < \varepsilon < 1/2$*. An optimal sequential off-line packing is achieved if we drop any of the* ε *terms; then the total loss is* ε*. In contrast to this, the loss of any constant-threshold expert is* $1-\varepsilon+k$ *independently of the choice of the parameter* p*. Namely, if* $p \le 1-\varepsilon$ *then the loss is* 0 *for the first* $2k$ *items, but after the algorithm is stuck and suffers* $k+1-\varepsilon$ *loss. If* $p > 1-\varepsilon$*, then the loss is* k *for the first* $2k$ *items and after that* $1-\varepsilon$ *for the rest of the sequence.*

6. Conclusions

In this paper we provide an extension of the classical bin packing problems to an on-line sequential scenario. In this setting items are received one by one, and before the size of the next item is revealed, the decision maker needs to decide whether the next item is packed in the currently open bin or the bin is closed and a new bin is opened. If the new item does not fit, it is lost. If a bin is closed, the remaining free space in the bin accounts for a loss. The goal of the decision maker is to minimize the loss accumulated over n periods.

We give an algorithm that has a cumulative loss not much larger than any finite set of reference algorithms. We also study in detail the case when the class of reference strategies contains all constant-threshold experts. We prove some negative results, showing that it is hard to compete with the overall best constant-threshold expert if no assumption is imposed on the item sizes. We also derive data-dependent regret bounds and show that under some mild assumptions on the data the

cumulative loss can be made not much larger than that of any strategy that uses a fixed threshold at each step to decide whether a new bin is opened. An interesting aspect of the problem is that the loss function has an (unbounded) memory. The presented solutions rely on the fact that one can "synchronize" the loss function in the sense that no matter in what state an algorithm is started, its loss may change only by a small additive constant. The result for constant-threshold experts is obtained by a covering of the uncountable set of constant-threshold experts such that the cardinality of the chosen finite set of experts grows only quadratically with the sequence length. The approach in the paper can easily be extended to any control problem where the loss function has such a synchronizable property.

Acknowledgments

The authors would like to thank Robert Kleinberg for pointing out a crucial mistake in the earlier version (György et al., 2008) of this paper. In that paper we erroneously claimed that it was possible to construct a sequential on-line bin packing algorithm that has a small regret with respect to all constant-threshold experts for all possible item sequences.

The authors acknowledge support by the Hungarian Scientific Research Fund (OTKA F60787), by the Mobile Innovation Center of Hungary, by the Spanish Ministry of Science and Technology grant MTM2009-09063, and by the PASCAL2 Network of Excellence under EC grant no. 216886.

References

J. Boyar, L. Epstein, L. M. Favrholdt, J. S. Kohrt, K. S. Larsen, M. M. Pedersen, and S. Wøhlk. The maximum resource bin packing problem. *Theoretical Computer Science*, 362:127–139, 2006.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, to appear, Cambridge, 2006.

E. G. Coffman, M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: a survey. In *Approximation Algorithms for NP-hard Problems*, pages 46–93. PWS Publishing Co., Boston, MA, 1997.

E. Even-Dar, S. M. Kakade, and Y. Mansour. Experts in a markov decision process. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 401–408, Cambridge, MA, 2005. MIT Press.

A. György, G. Lugosi, and Gy. Ottucsák. On-line sequential bin packing. In *Proceedings of the 21st Annual Conference on Learning Theory, COLT 2008*, pages 447–454, Helsinki, Finland, July 2008.

J. Hannan. Approximation to bayes risk in repeated plays. In M. Dresher, A. Tucker, and P. Wolfe, editors, *Contributions to the Theory of Games*, volume 3, pages 97–139. Princeton University Press, 1957.

Y. He and Gy. Dósa. Bin packing and covering problems with rejection. In *Proceedings of the 11th Annual International Conference on Computing and Combinatorics, COCOON 2005*, volume

3595 of *Lecture Notes in Computer Science*, pages 885–894, Kunming, China, August 2005. Springer Berlin/Heidelberg.

N. Merhav, E. Ordentlich, G. Seroussi, and M. J. Weinberger. On sequential strategies for loss functions with memory. *IEEE Transactions on Information Theory*, 48:1947–1958, 2002.

S. S. Seiden. On the online bin packing problem. *Journal of the ACM*, 49(5):640–671, 2002.

J. Y. Yu, S. Mannor, and N. Shimkin. Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research*, 34:737–757, 2009.

Journal of Machine Learning Research 11 (2010) 111-130 Submitted 2/09; Revised 11/09; Published 1/10

Classification Methods with Reject Option Based on Convex Risk Minimization

Ming Yuan MYUAN@ISYE.GATECH.EDU
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205, USA

Marten Wegkamp WEGKAMP@STAT.FSU.EDU
Department of Statistics
Florida State University
Tallahassee, FL 32306, USA

Editor: Bin Yu

Abstract

In this paper, we investigate the problem of binary classification with a reject option in which one can withhold the decision of classifying an observation at a cost lower than that of misclassification. Since the natural loss function is non-convex so that empirical risk minimization easily becomes infeasible, the paper proposes minimizing convex risks based on surrogate convex loss functions. A necessary and sufficient condition for infinite sample consistency (both risks share the same minimizer) is provided. Moreover, we show that the excess risk can be bounded through the excess surrogate risk under appropriate conditions. These bounds can be tightened by a generalized margin condition. The impact of the results is illustrated on several commonly used surrogate loss functions.

Keywords: classification, convex surrogate loss, empirical risk minimization, generalized margin condition, reject option

1. Introduction

In binary classification, one observes independent realizations $(X_1,Y_1),\ldots,(X_n,Y_n)$ of the random pair (X,Y) where $X \in \mathcal{X}$ and $Y \in \mathcal{Y} = \{-1,1\}$. The goal is to learn from these training data a classification rule $g : \mathcal{X} \mapsto \mathcal{Y}$ that classifies an observation X into the two classes. It is recognized that in many applications the consequences of misclassification can be substantial. In such situations, a less specific response that reserves the right of not making a decision, sometimes referred to as a reject option (see, e.g., Herbei and Wegkamp, 2006), may even be more preferable than risking misclassification. This, for example, is typical in medical studies where screening of a certain disease can be done based on relatively inexpensive clinical measures. If the classification based on these measurements are satisfactory, nothing further needs to be done. But in the event that there are ambiguities, it would be more desirable to take a rejection option and seek more expensive studies to identify a subject's disease status. Similar approaches are often adopted in DNA sequencing or genotyping applications, where the rejection option is commonly referred to as a "no-call". Similar problems have attracted much attention in various application fields and also received increasing

amount of interest more recently in machine learning literature. See Ripley (1996) and Bartlett and Wegkamp (2008) and references therein.

To accommodate the reject option, we now seek a classification rule $g : \mathcal{X} \to \tilde{\mathcal{Y}}$ where $\tilde{\mathcal{Y}} = \{-1, 1, 0\}$ is an augmented response space and $g(X) = 0$ indicates that no definitive classification will be made for X or a reject option is taken. To measure the performance of a classification rule, we employ the following loss function that generalizes the usual 0-1 loss to account for reject option:

$$\ell[g(X), Y] = \begin{cases} 1 & \text{if } g(X) \neq Y \text{ and } g(X) \neq 0 \\ d & \text{if } g(X) = 0 \\ 0 & \text{if } g(X) = Y \end{cases} .$$

In other words, an ambiguous response ($g(X) = 0$) incurs a loss of d whereas misclassification incurs a loss of 1. Note that d is necessarily smaller than $1/2$. Otherwise, rather than taking a rejection option with a loss d, we can always flip a fair coin to randomly assign ± 1 as the value of $g(X)$, which incurs an average loss of $1/2 \leq d$. For this reason, we shall assume that $d < 1/2$ in what follows.

For any classification rule $g : \mathcal{X} \to \tilde{\mathcal{Y}}$, the risk function is then given by $R(g) = \mathbb{E}(\ell[g(X), Y])$ where the expectation is taken over the joint distribution of X and Y. It is not hard to show that the optimal classification rule $g^* := \arg\min R(g)$ is given by (see, e.g., Bartlett and Wegkamp, 2008)

$$g^*(X) = \begin{cases} 1 & \text{if } \eta(X) > 1 - d \\ 0 & \text{if } d \leq \eta(X) \leq 1 - d \\ -1 & \text{if } \eta(X) < d \end{cases}$$

where $\eta(X) = \mathbb{P}(Y = 1|X)$. The corresponding risk is

$$R^* := \inf R(g) = R(g^*) = \mathbb{E}\left(\min\{\eta(X), 1 - \eta(X), d\}\right).$$

Thus, the performance of any classification rule $g : \mathcal{X} \to \tilde{\mathcal{Y}}$ can be measured by the excess risk $\Delta R(g) := R(g) - R^*$.

Appealing to the general empirical risk minimization strategy, one could attempt to derive a classification rule from the training data by minimizing the empirical risk

$$R_n(g) = \frac{1}{n} \sum_{i=1}^{n} \ell[g(X_i), Y_i].$$

Similar to the usual 0-1 loss, however, ℓ is not convex in g; and direct minimization of R_n is typically an NP-hard problem. A common remedy is to consider a surrogate convex loss function. To this end, let $\phi : \mathbb{R} \mapsto \mathbb{R}$ be a convex function. Denote by

$$Q(f) = \mathbb{E}[\phi(Y f(X))]$$

the corresponding risk for a discriminant function $f : \mathcal{X} \mapsto \mathbb{R}$. Let $\hat{f}_n$ be the minimizer of

$$Q_n(f) = \frac{1}{n} \sum_{i=1}^{n} \phi(Y_i f(X_i))$$

over a certain functional space $\mathcal{F}$ consisting of functions that map from $\mathcal{X}$ to $\mathbb{R}$. $\hat{f}_n$ can be conveniently converted to a classification rule $C(\hat{f}_n;\delta)$ as follows:

$$C(f(X);\delta) = \begin{cases} 1 & \text{if } f(X) > \delta \\ 0 & \text{if } |f(X)| \le \delta \\ -1 & \text{if } f(X) < -\delta \end{cases}$$

where $\delta > 0$ is a parameter that as we shall see plays a critical role in determining the performance of $C(\hat{f}_n,\delta)$.

In this paper, we investigate the statistical properties of this general convex risk minimization technique. To what extent $C(\hat{f}_n,\delta)$ mimics the optimal classification rule g^* plays a critical role in the success of this technique. Let f_ϕ^* be the minimizer of $Q(f)$. We shall assume throughout the paper that f_ϕ^* is uniquely defined. Typically f_ϕ^* reflects the limiting behavior of $\hat{f}_n$ when $\mathcal{F}$ is rich enough and there are infinitely many training data. Therefore the first question is whether or not f_ϕ^* can be used to recover the optimal rule g^*. A surrogate loss function ϕ that satisfies this property is often called infinite sample consistent (see, e.g., Zhang, 2004) or classification calibrated (see, e.g., Bartlett, Jordan and McAuliffe, 2006). A second question further concerns the relationship between the excess risk $\Delta R[C(f,\delta)]$ and the excess ϕ risk $\Delta Q(f) = Q(f) - \inf Q(f)$: Can we find an increasing function $\rho : \mathbb{R} \mapsto \mathbb{R}$ such that for all f,

$$\Delta R[C(f,\delta)] \le \rho\left(\Delta Q(f)\right) ? \tag{1}$$

Clearly the infinite sample consistency of ϕ implies that $\rho(0) = 0$. Such a bound on the excess risk provides useful tools in bounding the excess risk of $\hat{f}_n$. In particular, (1) indicates that

$$\Delta R[C(\hat{f}_n,\delta)] \le \rho\left(\Delta Q(\hat{f}_n)\right) = \rho\left[\Delta Q(\bar{f}) + \left(Q(\hat{f}_n) - Q(\bar{f})\right)\right],$$

where $\bar{f} = \arg\min_{f\in\mathcal{F}} Q(f)$. The first term $\Delta Q(\bar{f})$ on the right-hand side exhibits the approximation error of functional class $\mathcal{F}$ whereas the second term $Q(\hat{f}_n) - Q(\bar{f})$ is the estimation error.

In the case when there is no reject option, these problems have been well investigated in recent years (Lin, 2002; Zhang, 2004; Bartlett, Jordan and McAuliffe, 2006). In this paper, we establish similar results when there is a reject option. The most significant difference between the two situations, with or without the reject option, is the role of δ. As we shall see, for some loss functions such as least squares, exponential or logistic, a good choice of δ yields classifiers that are infinite sample consistent. For other loss functions, however, such as the hinge loss, no matter how δ is chosen, the classification rule $C(f,\delta)$ cannot be infinite sample consistent.

The remainder of the paper is organized as follows. We first examine in Section 2 the infinite sample consistency for classification with reject option. After establishing a general result, we consider its implication on several commonly used loss functions. In Section 3, we establish bounds on the excess risk in the form of (1), followed by applications to the popular loss functions. We also show that under an additional assumption on the behavior of $\eta(X)$ near d and $1-d$ as in Herbei and Wegkamp (2006), generalizing the condition in the case of $d = 1/2$ of Mammen and Tsybakov (1999) and Tsybakov (2004), the bound (1) can be tightened considerably. Section 4 discusses rates of convergence of the empirical risk minimizer $\widehat{f}_n$ that minimizes the empirical risk $Q_n(f)$ over a bounded class $\mathcal{F}$. Section 5 considers extension to asymmetric loss where one type of misclassification may be more costly than the other. All proofs are relegated to Section 6.

2. Infinite Sample Consistency

We first give a general result on the infinite sample consistency of the classification rule $C(f_\phi^*, \delta)$.

Theorem 1 *Assume that ϕ is convex. Then the classification rule $C(f_\phi^*, \delta)$ for some $\delta > 0$ is infinite sample consistent, that is, $C(f_\phi^*, \delta) = g^*$ if and only if $\phi'(\delta)$ and $\phi'(-\delta)$ both exist, $\phi'(\delta) < 0$, and*

$$\frac{\phi'(\delta)}{\phi'(\delta) + \phi'(-\delta)} = d. \tag{2}$$

When there is no reject option, it is known that the necessary and sufficient condition for the infinite sample consistency is that ϕ is differentiable at 0 and $\phi'(0) < 0$ (see, e.g., Bartlett, Jordan and McAuliffe, 2006). As indicated by Theorem 1, the differentiability of ϕ at $\pm\delta$ plays a more prominent role in the general case when there is a reject option.

From Theorem 1 it is also evident that the infinite sample consistency depends on both ϕ and the choice of thresholding parameter δ. Observe that for any $\delta_1 < \delta_2$,

$$\phi'(-\delta_2) \le \phi'(-\delta_1) \le \phi'(\delta_1) \le \phi'(\delta_2),$$

which implies that the left-hand side of (2) is a decreasing function of δ. If ϕ is strictly convex, then it is strictly decreasing; and therefore there is at most one value of δ that satisfies (2). In other words, for strictly convex ϕ, there is at most one thresholding parameter δ such that $C(f_\phi^*, \delta) = g^*$. On the other hand, if ϕ is twice differentiable such that $\phi'(0) < 0$ and $\phi'(z) \ge 0$ as $z \to +\infty$, then for any $d < 1/2$, there always exists a $\delta > 0$ such that (2) holds. This is because the left-hand side of (2) is a decreasing function of δ, which approaches its supremum $1/2$ when $\delta \downarrow 0$ and 0 when δ increases. Moreover, the twice differentiability of ϕ ensures that the left-hand side of (2) is also a continuous function of δ. The following is therefore a direct consequence of Theorem 1:

Corollary 2 *If ϕ is strictly convex, then either there is a unique $\delta > 0$ such that $C(f_\phi^*, \delta)$ is infinite sample consistent; or $C(f_\phi^*, \delta)$ is not infinite sample consistent for any $\delta > 0$. In addition to convexity, if ϕ is twice differentiable such that $\phi'(0) < 0$ and $\phi'(z) \ge 0$ as $z \to +\infty$, then there always exists a $\delta > 0$ such that $C(f_\phi^*, \delta)$ is infinite sample consistent.*

Theorem 1 provides a general guideline on how to choose δ for common choices of convex losses. Below we look at several concrete examples.

2.1 Least Squares Loss

We first examine the least squares loss $\phi(z) = (1-z)^2$. Observe that

$$\frac{\phi'(\delta)}{\phi'(-\delta) + \phi'(\delta)} = \frac{1-\delta}{2}.$$

All conditions of Theorem 1 are met if and only if $\delta = 1 - 2d$.

Corollary 3 *For the least squares loss,*

$$C(f_\phi^*, 1-2d) = g^*.$$

2.2 Exponential Loss

Exponential loss, $\phi(z) = \exp(-z)$, is connected with boosting (Friedman, Hastie and Tibshirani, 2000). Because

$$\frac{\phi'(\delta)}{\phi'(-\delta)+\phi'(\delta)} = \frac{1}{1+\exp(2\delta)},$$

Therefore all conditions of Theorem 1 are met if and only if

$$\delta = \frac{1}{2}\log\left(\frac{1}{d}-1\right).$$

Corollary 4 *For the exponential loss,*

$$C\left(f_\phi^*, \frac{1}{2}\log\left(\frac{1}{d}-1\right)\right) = g^*.$$

2.3 Logistic Loss

Logisitic regression employs loss $\phi(z) = \ln(1+\exp(-z))$. Similar to before,

$$\frac{\phi'(\delta)}{\phi'(-\delta)+\phi'(\delta)} = \frac{1}{1+\exp(\delta)},$$

which suggests that all conditions of Theorem 1 are met if

$$\delta = \log\left(\frac{1}{d}-1\right).$$

Corollary 5 *For the logistic loss,*

$$C\left(f_\phi^*, \log\left(\frac{1}{d}-1\right)\right) = g^*.$$

2.4 Squared Hinge Loss

Squared hinge loss, $\phi(z) = (1-z)_+^2$, is another popular choice for which

$$\frac{\phi'(\delta)}{\phi'(-\delta)+\phi'(\delta)} = \frac{1-\delta}{2}.$$

Similar to the least squares loss, we have the following corollary.

Corollary 6 *For the squared hinge loss,*

$$C\left(f_\phi^*, 1-2d\right) = g^*.$$

2.5 Distance Weighted Discrimination

Marron, Todd and Ahn (2007) recently introduced the so-called distance weighted discrimination method where the following loss function (see, e.g., Bartlett, Jordan and McAuliffe, 2006) is used

$$\phi(z) = \begin{cases} \frac{1}{z} & \text{if } z \geq \gamma \\ \frac{1}{\gamma}\left(2 - \frac{z}{\gamma}\right) & \text{if } z < \gamma \end{cases}, \tag{3}$$

where $\gamma > 0$ is a constant. It is not hard to see that ϕ is convex. Moreover,

$$\phi'(z) = \begin{cases} -1/z^2 & \text{if } z \geq \gamma \\ -1/\gamma^2 & \text{if } z < \gamma \end{cases}.$$

Thus,

$$\frac{\phi'(\delta)}{\phi'(-\delta) + \phi'(\delta)} = \begin{cases} 1/2 & \text{if } \delta < \gamma \\ \frac{1/\delta^2}{1/\delta^2 + 1/\gamma^2} & \text{if } \delta > \gamma \end{cases}.$$

In other words, we have the following result for the distance weighted discrimination loss.

Corollary 7 *For the loss (3),*

$$C\left(f_\phi^*, [(1-d)/d]^{1/2}\gamma\right) = g^*.$$

2.6 Hinge Loss

The popular support vector machine employs the hinge loss, $\phi(z) = (1-z)_+$. The hinge loss is differentiable everywhere except 1. Therefore

$$\frac{\phi'(\delta)}{\phi'(-\delta) + \phi'(\delta)} = \begin{cases} \frac{1}{2} & \text{if } 0 < \delta < 1 \\ 0 & \text{if } \delta > 1 \end{cases}.$$

Because $0 < d < 1/2$, there does not exist a δ such that all conditions of Theorem 1 are met. As a matter of fact, for any $\delta > 0$, $C(f_\phi^*, \delta) \neq g^*$. Motivated by this observation, Bartlett and Wegkamp (2008) introduce the following modification to the hinge loss:

$$\phi(z) = \begin{cases} 1 - az & \text{if } z \leq 0 \\ 1 - z & \text{if } 0 < z \leq 1 \\ 0 & \text{if } z > 1 \end{cases}, \tag{4}$$

where $a > 1$. Note that with this modification,

$$\frac{\phi'(\delta)}{\phi'(-\delta) + \phi'(\delta)} = \begin{cases} 1/(a+1) & \text{if } 0 < \delta < 1 \\ 0 & \text{if } \delta > 1 \end{cases}.$$

Therefore, we have the following corollary.

Corollary 8 *For the modified hinge loss (4) and any $\delta < 1$, if $a = (1-d)/d$, then*

$$C\left(f_\phi^*, \delta\right) = g^*.$$

It is interesting to note that for the examples we considered previously, a specific choice of δ is needed to ensure the infinite sample consistent. Whereas for the modified hinge loss, a range of choice of δ can serve the same purpose. However, as we shall see in the next section, different choices of δ for the modified hinge loss may result in slightly different bound on the excess risk with $\delta = 1/2$ appearing to be more preferable in that it yields the smallest upper bound of the excess risk.

3. Excess Risk

We now turn to the excess risk $\Delta R[C(f,\delta)]$ and show how it can be bounded through the excess ϕ risk

$$\Delta Q(f) := Q(f) - Q(f_\phi^*).$$

Recall that the infinite sample consistency established in the previous section means that $\Delta Q(f) = 0$ implies throughout this section that $\Delta R(C(f,\delta)) = 0$. For brevity, we shall assume implicitly that δ is chosen in accordance with Theorem 1 to ensure infinite sample consistency. Write

$$Q_{\eta(X)}(z) = \eta(X)\phi(z) + (1-\eta(X))\phi(-z).$$

By definition,

$$Q_{\eta(X)}(f_\phi^*(X)) = \inf_z Q_{\eta(X)}(z).$$

Denote

$$\Delta Q_\eta(f) = Q_\eta(f) - Q_\eta(f_\phi^*)$$

where we suppress the dependence of η, f and f_ϕ^* on X for brevity.

Theorem 9 *Assume that ϕ is convex, $\phi'(\delta)$ and $\phi'(-\delta)$ both exist, $\phi'(\delta) < 0$, and (2) holds. In addition, suppose that there exist constants $C > 0$ and $s \geq 1$ such that*

$$\begin{aligned} |\eta - d|^s &\leq C^s \Delta Q_\eta(-\delta); \\ |(1-\eta) - d|^s &\leq C^s \Delta Q_\eta(\delta). \end{aligned}$$

Then

$$\Delta R[C(f,\delta)] \leq 2C\left[\Delta Q(f)\right]^{1/s}. \tag{5}$$

It is immediate from Theorem 9 that $\Delta Q(\hat{f}_n) \to_p 0$ implies $\Delta R(\hat{f}_n) \to_p 0$. In other words, consistency in terms of ϕ risk implies the consistency in terms of loss ℓ. It is worth noting that the constant in the upper bound can be tightened under stronger conditions.

Theorem 10 *In addition to the assumptions of Theorem 9, assume that*

$$\begin{aligned} (2\eta - 1)_+^s &\leq C^s \Delta Q_\eta(-\delta); \\ (1 - 2\eta)_+^s &\leq C^s \Delta Q_\eta(\delta). \end{aligned}$$

Then

$$\Delta R[C(f,\delta)] \leq C\left[\Delta Q(f)\right]^{1/s}.$$

We can improve the bounds even further by the following margin condition. Assume that for some $\alpha \geq 0$ and $A \geq 1$

$$\mathbb{P}\{|\eta(X) - z| \leq t\} \leq A t^\alpha \tag{6}$$

for all $0 \leq t < d$ at $z = d$ and $z = 1 - d$. This assumption was introduced in Herbei and Wegkamp (2006) and generalizes the margin condition of Mammen and Tsybakov (1999) and Tsybakov (2004). It is always met for $\alpha = 0$ and $A = 1$. The other extreme is for $\alpha \to +\infty$ - the case where $\eta(X)$ stays away from d and $1 - d$ with probability one.

Theorem 11 *In addition to the assumptions of Theorem 9, assume that (6) holds for some $\alpha \geq 0$ and $A \geq 1$. Then, for some K depending on A and α,*

$$\Delta R\left[C(f,\delta)\right] \leq K\left[\Delta Q(f)\right]^{1/(s+\beta-\beta s)}, \tag{7}$$

where $\beta = \alpha/(1+\alpha)$.

In case $\alpha = 0$, the exponent $1/(s+\beta-\beta s)$ is $1/s$ on the right hand side in (7) above, and the situation is as in Theorem 9. For $\alpha \to +\infty$, the bound (7) improves upon the one in Theorem 9 as the exponent $1/(s+\beta-\beta s)$ converges to 1.

We now examine the consequences of Theorems 9, 10 and 11 on several common loss functions.

3.1 Least Squares

Note that for the least squares loss

$$\Delta Q_\eta(f) = (2\eta - 1 - f)^2.$$

Simple algebraic manipulations show that

$$\begin{aligned} \Delta Q_\eta(-\delta) &= 4|\eta - d|^2; \\ \Delta Q_\eta(\delta) &= 4|(1-\eta) - d|^2. \end{aligned}$$

Therefore, by Theorems 9 and 11,

Corollary 12 *For the least squares loss,*

$$\Delta R[C(f,1-2d)] \leq \left[\Delta Q(f)\right]^{1/2}.$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R[C(f,1-2d)] \leq K\left[\Delta Q(f)\right]^{\frac{1+\alpha}{2+\alpha}}$$

for some constant $K > 0$.

3.2 Exponential Loss

An application of Taylor expansion yields (see, e.g., Zhang, 2004)

$$\Delta Q_\eta(f) \geq 2\left(\eta - \frac{1}{1+\exp(-2f)}\right)^2.$$

Therefore,

$$\begin{aligned} \Delta Q_\eta(-\delta) &\geq 2|\eta - d|^2; \\ \Delta Q_\eta(\delta) &\geq 2|(1-\eta) - d|^2. \end{aligned}$$

Therefore, by Theorems 9 and 11,

Corollary 13 *For the exponential loss,*

$$\Delta R\left[C\left(f,\frac{1}{2}\log\left(\frac{1}{d}-1\right)\right)\right] \leq \sqrt{2}\left[\Delta Q(f)\right]^{1/2}.$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R\left[C\left(f,\frac{1}{2}\log\left(\frac{1}{d}-1\right)\right)\right] \leq K\left[\Delta Q(f)\right]^{\frac{1+\alpha}{2+\alpha}}$$

for some constant $K > 0$.

3.3 Logistic Loss

Similar to exponential loss, an application of Taylor expansion yields

$$\Delta Q_\eta(f) \geq 2\left(\eta - \frac{1}{1+\exp(-f)}\right)^2.$$

Therefore,

$$\begin{aligned}\Delta Q_\eta(-\delta) &\geq 2|\eta - d|^2;\\ \Delta Q_\eta(\delta) &\geq 2|(1-\eta) - d|^2.\end{aligned}$$

Therefore, by Theorems 9 and 11,

Corollary 14 *For the logistic loss,*

$$\Delta R\left[C\left(f,\log\left(\frac{1}{d}-1\right)\right)\right] \leq \sqrt{2}\left[\Delta Q(f)\right]^{1/2}.$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R\left[C\left(f,\log\left(\frac{1}{d}-1\right)\right)\right] \leq K\left[\Delta Q(f)\right]^{\frac{1+\alpha}{2+\alpha}}$$

for some constant $K > 0$.

3.4 Squared Hinge Loss

Simple algebraic derivation shows

$$\Delta Q_\eta(f) = (2\eta - 1 - f)^2 - \eta(f-1)_+^2 - (1-\eta)(f+1)_-^2.$$

Therefore,

$$\begin{aligned}\Delta Q_\eta(-\delta) &= 4|\eta - d|^2;\\ \Delta Q_\eta(\delta) &= 4|(1-\eta) - d|^2.\end{aligned}$$

By Theorems 9 and 11,

Corollary 15 *For the squared hinge loss,*

$$\Delta R\left[C(f,1-2d)\right] \leq \left[\Delta Q(f)\right]^{1/2}.$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R[C(f,1-2d)] \leq K\left[\Delta Q(f)\right]^{\frac{1+\alpha}{2+\alpha}}$$

for some constant $K > 0$.

3.5 Distance Weighted Discrimination

Observe that

$$Q_\eta(z) = \begin{cases} \frac{\eta}{z} + \frac{(1-\eta)z}{\gamma^2} + \frac{2(1-\eta)}{\gamma} & \text{if } z \geq \gamma \\ \frac{2}{\gamma} + \frac{z}{\gamma^2}(1-2\eta) & \text{if } |z| < \gamma \\ \frac{2\eta}{\gamma} - \frac{\eta z}{\gamma^2} - \frac{1-\eta}{z} & \text{if } z \leq -\gamma \end{cases}.$$

Hence

$$\inf Q_\eta(z) = \frac{2}{\gamma}\left(\sqrt{\eta(1-\eta)} + \min\{\eta, 1-\eta\}\right)$$

and

$$f_\phi^* = \begin{cases} (\eta/(1-\eta))^{1/2}\gamma & \text{if } \eta > 1/2 \\ \text{any value in } [-\gamma,\gamma] & \text{if } \eta = 1/2 \\ ((1-\eta)/\eta)^{1/2}\gamma & \text{if } \eta < 1/2 \end{cases}.$$

Recall that $\delta = ((1-d)/d)^{1/2}\gamma$. Then

$$\begin{aligned} \Delta Q_\eta(\delta) &\geq \left(\frac{\eta}{\delta} + \frac{(1-\eta)\delta}{\gamma^2} - 2\sqrt{\eta(1-\eta)}/\gamma\right) \\ &= \left(\left(\frac{\eta}{\delta}\right)^{1/2} - \left(\frac{(1-\eta)\delta}{\gamma^2}\right)^{1/2}\right)^2 \\ &= \frac{\eta\delta}{\gamma^2}\left(\left(\frac{d}{1-d}\right)^{1/2} - \left(\frac{1-\eta}{\eta}\right)^{1/2}\right)^2 \\ &= \frac{\eta\delta}{\gamma^2}\left(\left(\frac{d}{1-d}\right)^{1/2} + \left(\frac{1-\eta}{\eta}\right)^{1/2}\right)^{-2}\left(\frac{d}{1-d} - \frac{1-\eta}{\eta}\right)^2 \\ &= \frac{\delta}{\gamma^2(1-d)^2}\left[\left(\frac{d}{1-d}\right)^{1/2}\eta^{1/2} + (1-\eta)^{1/2}\right]^{-2}(1-\eta-d)^2. \end{aligned}$$

Observe that

$$\left(\frac{d}{1-d}\right)^{1/2}\eta^{1/2} + (1-\eta)^{1/2} \leq (1-d)^{-1/2}.$$

Thus,

$$(1-\eta-d)^2 \leq \gamma(1-d)^{1/2}d^{1/2}\Delta Q_\eta(\delta).$$

Similarly,

$$(\eta-d)^2 \leq \gamma(1-d)^{1/2}d^{1/2}\Delta Q_\eta(-\delta).$$

From Theorems 9 and 11, we conclude that

Corollary 16 *For the distance weighted discrimination loss,*

$$\Delta R\left[C(f,((1-d)/d)^{1/2}\gamma)\right] \leq \gamma^{1/2}(1-d)^{1/4}d^{1/4}[\Delta Q(f)]^{1/2}.$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R\left[C(f,((1-d)/d)^{1/2}\gamma)\right] \leq K\,[\Delta Q(f)]^{\frac{1+\alpha}{2+\alpha}}$$

for some constant $K > 0$.

3.6 Hinge Loss with Rejection Option

As shown by Bartlett and Wegkamp (2008), for the modified hinge loss (4),

$$\arg\min_z Q_\eta(z) = \begin{cases} -1 & \text{if } \eta \le d \\ 0 & \text{if } d < \eta < 1-d \\ 1 & \text{if } \eta > 1-d \end{cases} .$$

Simple algebraic manipulations lead to

$$\Delta Q_\eta(-\delta) = \begin{cases} (1-\delta)(d-\eta)/d & \text{if } \eta \le d \\ (\eta-d)\delta/d & \text{if } d < \eta < 1-d \\ 1-(1-\eta)/d+(\eta-d)\delta/d & \text{if } \eta > 1-d \end{cases} ,$$

and

$$\Delta Q_\eta(\delta) = \begin{cases} 1-\eta/d+(1-\eta-d)\delta/d & \text{if } \eta \le d \\ (1-\eta-d)\delta/d & \text{if } d < \eta < 1-d \\ (\delta-1)(1-\eta-d)/d & \text{if } \eta > 1-d \end{cases} .$$

Therefore,

$$\begin{aligned} \frac{\min\{\delta,1-\delta\}}{d}|\eta-d| &\le \Delta Q_\eta(-\delta); \\ \frac{\min\{\delta,1-\delta\}}{d}|(1-\eta)-d| &\le \Delta Q_\eta(\delta). \end{aligned}$$

Furthermore,

$$\begin{aligned} \frac{\min\{\delta,1-\delta\}}{d}(2\eta-1)_+ &\le \Delta Q_\eta(-\delta); \\ \frac{\min\{\delta,1-\delta\}}{d}(1-2\eta)_+ &\le \Delta Q_\eta(\delta). \end{aligned}$$

From Theorems 10 and 11, we conclude that

Corollary 17 *For the modified hinge loss and any* $\delta < 1$,

$$\Delta R[C(f,\delta)] \le \frac{d}{\min\{\delta,1-\delta\}}\Delta Q(f). \tag{8}$$

Furthermore, if the margin condition (6) holds, then

$$\Delta R[C(f,\delta)] \le K\Delta Q(f) \tag{9}$$

for some constant $K > 0$.

Notice that the corollary also suggests that $\delta = 1/2$ yields the best constant $2d$ in the upper bound. A similar result has also been recently established by Bartlett and Wegkamp (2008). It is also interesting to see that (8) cannot be further improved by the generalized margin condition (6) as the bounds (8) and (9) only differ by a constant.

4. Rates of Convergence for Empirical Risk Minimizers

In this section we briefly review the possible rates of convergence for minimizers of the empirical risk $Q_n(f) = (1/n)\sum_{i=1}^n \phi(Y_i f(X_i))$ over a convex class of discriminant functions $\mathcal{F}$; and show the implications of the excess risk bounds obtained in the previous section. The analysis of the generalized hinge loss is complicated and is treated in detail in Wegkamp (2007) and Bartlett and Wegkamp (2008). The other loss functions ϕ considered in this paper have in common that the modulus of convexity of Q,

$$\delta(\varepsilon) = \inf\left\{ \frac{Q(f)+Q(g)}{2} - Q\left(\frac{f+g}{2}\right) : \ \mathbb{E}[(f-g)^2(X)] \geq \varepsilon^2 \right\}$$

satisfies $\delta(\varepsilon) \geq c\varepsilon^2$ for some $c > 0$ and that, for some $L < \infty$,

$$|\phi(x) - \phi(x')| \leq L|x - x'| \text{ for all } x, x' \in \mathbb{R}.$$

We have the following result that imposes a restriction on the $1/n$-covering number $N_n = N(1/n, L_\infty, \mathcal{F})$, the cardinality of the set of closed balls with radius $1/n$ in L_∞ needed to cover $\mathcal{F}$.

Theorem 18 *Assume that $|f| \leq B$ for all $f \in \mathcal{F}$ and let $0 < \gamma < 1$. With probability at least $1-\gamma$,*

$$Q(\hat{f}_n) \quad \leq \quad \inf_{f\in\mathcal{F}} Q(f) + \frac{3L}{n} + 8\left(\frac{L^2}{2c} + \frac{B}{6}\right)\frac{\log(N_n/\gamma)}{n}$$

Together with the excess risk bounds from Theorems 9 and 11, we have

Corollary 19 *Under the assumptions of Theorems 9 and 18, we have, with probability at least $1-\gamma$,*

$$\Delta R(C(\hat{f}_n, \delta)) \quad \leq \quad 2C\left\{ \inf_{f\in\mathcal{F}} \Delta Q(f) + \frac{3L}{n} + 8\left(\frac{L^2}{2c} + \frac{LB}{3}\right)\frac{\log(N_n/\gamma)}{n} \right\}^{1/s}.$$

Furthermore, if the generalized margin condition (6) holds, then with probability at least $1-\gamma$,

$$\Delta R(C(\hat{f}_n, \delta)) \quad \leq \quad K\left\{ \inf_{f\in\mathcal{F}} \Delta Q(f) + \frac{3L}{n} + 8\left(\frac{L^2}{2c} + \frac{LB}{3}\right)\frac{\log(N_n/\gamma)}{n} \right\}^{1/(s+\beta-\beta s)}$$

for some constant $K > 0$.

In the special case where $\mathcal{F}$ consists of linear combinations

$$\mathsf{f}_\lambda(x) = \sum_{j=1}^M \lambda_j f_j(x)$$

of simple discriminant functions (decision stumps) $f_1, \ldots, f_M$ with $\sum_{j=1}^M |\lambda_j| \leq B$ and $|f_j| \leq 1$, we obtain the rate $(M \log n/n)^{1/(s+\beta-\beta s)}$. We can view B as a tuning parameter here, and if the functions f_j are near orthogonal in the sense that

$$\max_{1\leq i\neq j\leq M} \frac{\mathbb{E}[f_i(X)f_j(X)]}{\sqrt{\mathbb{E}[f_i^2(X)]\mathbb{E}[f_j^2(X)]}} \leq \frac{c}{|\lambda_0|_0}$$

for some small $c > 0$, a small modification of Theorem 1 in Wegkamp (2007) shows that we also adapt to the unknown sparsity of the minimizer λ_0 of $Q(\mathsf{f}_\lambda)$ over λ in that the rate becomes $(|\lambda_0|_0 \log n/n)^{1/(s+\beta-\beta s)}$ for suitably chosen $B = B(n)$.

5. Asymmetric Loss

We have focused thus far on the case where misclassifying from one class to the other, either $g(X) = 1$ while $Y = -1$ or $g(X) = -1$ while $Y = 1$, is assigned the same loss. In many applications, however, one type of misclassification may incur a heavier loss than the other. Such situations naturally arise in risk management or medical diagonsis. To this end, the following loss function can be adopted in place of ℓ:

$$\ell_\theta[g(X),Y] = \begin{cases} 1 & \text{if } g(X) = -1 \text{ and } Y = 1 \\ \theta & \text{if } g(X) = 1 \text{ and } Y = -1 \\ d & \text{if } g(X) = 0 \\ 0 & \text{if } g(X) = Y \end{cases}.$$

We shall assume that $\theta < 1$ without loss of generality. It can be shown that the rejection option is only available if $d < \theta/(1+\theta)$ (see, e.g., Herbei and Wegkamp, 2006), which we shall assume throughout the section. When this holds, the corresponding Bayes rule is given by (see, e.g., Herbei and Wegkamp, 2006)

$$g^*_\theta(X) = \begin{cases} 1 & \text{if } \eta(X) > 1 - d/\theta \\ 0 & \text{if } d \le \eta(X) \le 1 - d/\theta \\ -1 & \text{if } \eta(X) < d \end{cases}.$$

Instead of $C(\hat{f}_n, \delta)$, an asymmetrically truncated classification rule, $\hat{f}_n, C(\hat{f}_n; \delta_1, \delta_2)$, can be used for our purpose here where

$$C(f(X); \delta_1, \delta_2) = \begin{cases} 1 & \text{if } f(X) > \delta_1 \\ 0 & \text{if } -\delta_2 \le f(X) \le \delta_1 \\ -1 & \text{if } f(X) < -\delta_2 \end{cases}.$$

The behavior of the asymmetically truncated classification rule $C(\hat{f}_n; \delta_1, \delta_2)$ can be studied in a similar fashion as before. In particular, we have the following results in parallel to Theorems 1 and 9.

Theorem 20 *Assume that ϕ is convex. Then $C(f^*_\phi, \delta_1, \delta_2)$ for some $\delta_1, \delta_2 > 0$ is infinite sample consistent, that is, $C(f^*_\phi, \delta_1, \delta_2) = g^*_\theta$ if and only if $\phi'(\pm\delta_1)$ and $\phi'(\pm\delta_2)$ exist; $\phi'(\delta_1), \phi'(\delta_2) < 0$; and*

$$\begin{aligned} \frac{\phi'(\delta_1)}{\phi'(-\delta_1) + \phi'(\delta_1)} &= \frac{d}{\theta}; \\ \frac{\phi'(\delta_2)}{\phi'(-\delta_2) + \phi'(\delta_2)} &= d. \end{aligned}$$

Furthermore, if $C(f^*_\phi, \delta_1, \delta_2)$ is infinite sample consistent and

$$\begin{aligned} |\theta(1-\eta) - d|^s &\le C^s \Delta Q_\eta(\delta_1); \\ |\eta - d|^s &\le C^s \Delta Q_\eta(-\delta_2), \end{aligned}$$

then

$$\Delta R_\theta[C(f, \delta_1, \delta_2)] \le 2C[\Delta Q(f)]^{1/s},$$

where $\Delta R_\theta(g) = R_\theta(g) - R_\theta(g^*_\theta)$ and $R_\theta(g) = \mathbb{E}[\ell_\theta(g(X), Y)]$.

Theorem 20 can be proved in the same fashion as Theorems 1 and 9 and is therefore omitted for brevity.

6. Proofs

Proof of Theorem 1. We first show the "if" part. Recall that

$$\begin{aligned} Q(f) &= \mathbb{E}[\phi(Yf(X))] \\ &= \mathbb{E}\left(\mathbb{E}\left[\phi(Yf(X))|X\right]\right) \\ &= \mathbb{E}\left[\eta(X)\phi(f(X)) + (1-\eta(X))\,\phi(-f(X))\right]. \end{aligned}$$

With slight abuse of notation, write

$$Q_{\eta(X)}(f(X)) = \eta(X)\phi(f(X)) + (1-\eta(X))\,\phi(-f(X)).$$

Then $f_\phi^*(X)$ minimizes $Q_{\eta(X)}(\cdot)$.

We now proceed by separately considering three different scenarios: (a) $\eta(X) < d$; (b) $\eta(X) > 1-d$; and (c) $d < \eta(X) < 1-d$. For brevity, we shall abbreviate the dependence of η and f_ϕ^* on X in the reminder of the proof when no confusion occurs.

First consider the case when $\eta < d$. Recall that $\phi'(-\delta) < \phi'(\delta) < 0$, and

$$\frac{\phi'(\delta)}{\phi'(-\delta)+\phi'(\delta)} = d.$$

Therefore,

$$\eta\phi'(-\delta) - (1-\eta)\phi'(\delta) > 0.$$

By the convexity of ϕ, for any $z > 0$,

$$\begin{aligned} \phi(z-\delta) - \phi(-\delta) &\geq \phi'(-\delta)z; \\ \phi(-z+\delta) - \phi(\delta) &\geq -\phi'(\delta)z. \end{aligned}$$

Hence

$$Q_\eta(z-\delta) - Q_\eta(-\delta) \geq \left[\eta\phi'(-\delta) - (1-\eta)\phi'(\delta)\right] z > 0,$$

which implies that $f_\phi^* \leq -\delta$.

It now suffices to show that $f_\phi^* \neq -\delta$. By the definition of $\phi'(-\delta)$ and $\phi'(\delta)$, for any $\varepsilon > 0$, there exists a $\zeta > 0$ such that for any $0 < z < \zeta$,

$$\begin{aligned} \frac{\phi(-z-\delta) - \phi(-\delta)}{-z} &\geq \phi'(-\delta) - \varepsilon; \\ \frac{\phi(z+\delta) - \phi(\delta)}{z} &\leq \phi'(\delta) + \varepsilon. \end{aligned}$$

Therefore for any $0 < z < \zeta$,

$$\begin{aligned} Q_\eta(-z-\delta) - Q_\eta(-\delta) &= \eta\left[\phi(-z-\delta) - \phi(-\delta)\right] + (1-\eta)\left[\phi(z+\delta) - \phi(\delta)\right] \\ &\leq -\eta\left[\phi'(-\delta) - \varepsilon\right] z + (1-\eta)\left[\phi'(\delta) + \varepsilon\right] z \\ &= \left(\left[(1-\eta)\phi'(\delta) - \eta\phi'(-\delta)\right] + \varepsilon\right) z. \end{aligned}$$

Recall that $(1-\eta)\phi'(\delta) - \eta\phi'(-\delta) < 0$. By setting ε small enough, we can ensure that $(1-\eta)\phi'(\delta) - \eta\phi'(-\delta) + \varepsilon$ remains negative. Hence

$$Q_\eta(-z-\delta) < Q_\eta(-\delta),$$

which implies that $f_\phi^* \neq -\delta$.

Now consider the case when $\eta > 1-d$. Observe that $Q_\eta(z) = Q_{1-\eta}(-z)$. From the previous discussion,

$$f_\phi^* = \arg\min_z Q_\eta(z) = -\arg\min_z Q_{1-\eta}(-z) > \delta.$$

At last, consider the case when $d < \eta < 1-d$. Observe that in this case,

$$\eta\phi'(\delta) - (1-\eta)\phi'(-\delta) > 0;$$
$$\eta\phi'(-\delta) - (1-\eta)\phi'(\delta) < 0.$$

Hence for any $z > 0$,

$$Q_\eta(z+\delta) - Q_\eta(\delta) \geq \left[\eta\phi'(\delta) - (1-\eta)\phi'(-\delta)\right] z > 0,$$

which implies that $f_\phi^* \leq \delta$. Similarly,

$$Q_\eta(-z-\delta) - Q_\eta(-\delta) \geq \left[-\eta\phi'(-\delta) + (1-\eta)\phi'(\delta)\right] z > 0,$$

which implies that $f_\phi^* \geq -\delta$. In summary, $f_\phi^* \in [-\delta, \delta]$.

We now consider the "only if" part. Let $[a_-, b_-]$ and $[a_+, b_+]$ be the subdifferential of ϕ at $-\delta$ and δ respectively. We need to show that $a_- = b_-$, $a_+ = b_+$ and $a_+/(a_+ + a_-) = d$. We begin by showing that $b_+ \leq 0$. Assume the contrary. The infinite sample consistency implies that for any $\eta > 1-d$, $f_\phi^* > \delta$. Because $b_+ > 0$, we have $\phi(f_\phi^*) > \phi(\delta)$. Together with the fact that $Q_\eta(f_\phi^*) < Q_\eta(\delta)$, this implies that $\phi(-f_\phi^*) < \phi(-\delta)$. Subsequently, we have $a_- > 0$. The convexity of ϕ also suggests that $a_- \leq a_+ \leq b_- \leq b_+$. Because

$$\begin{aligned} \phi(f_\phi^*) - \phi(\delta) &\geq b_+(f_\phi^* - \delta); \\ \phi(-\delta) - \phi(-f_\phi^*) &\leq a_-(f_\phi^* - \delta), \end{aligned}$$

we have

$$Q_\eta(f_\phi^*) - Q_\eta(\delta) \geq (\eta b_+ - (1-\eta)a_-)(f_\phi^* - \delta) > 0.$$

This contradiction suggests that $b_+ \leq 0$.

Given that $a_- \leq a_+ \leq b_- \leq b_+ \leq 0$, we have $|a_-| \geq |a_+| \geq |b_-| \geq |b_+|$, which implies that

$$\frac{b_+}{a_- + b_+} \leq \frac{a_+}{b_- + a_+}.$$

It suffices to show that

$$\frac{b_+}{a_- + b_+} \geq d \qquad \text{and} \qquad \frac{a_+}{b_- + a_+} \leq d.$$

Assume the contrary. First consider the case when $b_+/(a_- + b_+) < d$. Let η be such that $b_+/(a_- + b_+) < \eta < d$. By definition, for any $f < -\delta$,

$$\begin{aligned} \phi(f) - \phi(-\delta) &\geq a_-(f + \delta); \\ \phi(-f) - \phi(\delta) &\geq b_+(-f - \delta). \end{aligned}$$

Hence

$$Q_\eta(f) - Q_\eta(-\delta) \geq [\eta a_- - (1-\eta)b_+](f + \delta) > 0,$$

which implies that $\arg\min Q_\eta(z) \geq -\delta$. This contradicts with the infinite sample consistency. Therefore, $b_+/(a_-+b_+) \geq d$. Next we deal with the case of $a_+/(b_-+a_+) > d$. Let η be such that $a_+/(b_-+a_+) > \eta > d$. Following a similar argument as before, one can show that $Q_{1-\eta}(f) - Q_{1-\eta}(\delta) > 0$ for any $f > \delta$, which implies that $\arg\min Q_\eta(z) \leq \delta$. This again contradicts infinite sample consistency because $1-\eta < 1-d$. Therefore, $a_+/(b_-+a_+) \leq d$.

The proof is now concluded. ∎

Proof of Theorem 9. Recall that

$$Q_\eta(f) = \eta\phi(f) + (1-\eta)\phi(-f).$$

Similarly, write

$$R_\eta[C(f,\delta)] = \eta\ell(C(f,\delta),1) + (1-\eta)\ell(C(f,\delta),-1).$$

Also write $\Delta Q_\eta(f) = Q_\eta(f) - \inf Q_\eta(f)$ and $\Delta R_\eta(f) = R_\eta(f) - \inf R_\eta(f)$. It suffices to show that

$$\Delta R_\eta[C(f,\delta)] \leq 2C\left[\Delta Q_\eta(f)\right]^{1/s}. \tag{10}$$

The theorem can be deduced from (10) by Jensen's inequality:

$$\begin{aligned}
\Delta R[C(f,\delta)] &= \mathbb{E}\left[\Delta R_{\eta(X)}[C(f(X),\delta)]\right] \\
&\leq 2C\mathbb{E}\left[\Delta Q_{\eta(X)}(f(X))\right]^{1/s} \\
&\leq 2C\left(\mathbb{E}\left[\Delta Q_{\eta(X)}(f(X))\right]\right)^{1/s} \\
&= 2C\left[\Delta Q(f)\right]^{1/s}.
\end{aligned}$$

To show (10), we consider separately the different combinations of values of η and f. For brevity, we shall abbreviate their dependence on X in what follows.

Case 1. $\eta < d$ and $f < -\delta$. As shown before, in this case $f^*_\phi(X) < -\delta$. Thus,

$$\Delta R_\eta[C(f,\delta)] = 0 \leq C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 2. $\eta < d$ and $|f| < \delta$. Observe that

$$Q_\eta(f) - Q_\eta(-\delta) \geq \left[\eta\phi'(-\delta) - (1-\eta)\phi'(\delta)\right](f+\delta) = \frac{-\phi'(\delta)}{d}(d-\eta)(f+\delta) \geq 0.$$

Together with the fact that $C^s\Delta Q_\eta(-\delta) \geq |\eta-d|^s$, we have

$$\Delta Q_\eta(f) \geq \Delta Q_\eta(-\delta) \geq C^{-s}|\eta-d|^s.$$

Note that

$$\Delta R_\eta[C(f,\delta)] = d-\eta.$$

We have

$$\Delta R_\eta[C(f,\delta)] \leq C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 3. $\eta < d$ and $f > \delta$. Observe that

$$Q_\eta(f) - Q_\eta(\delta) \geq \left[\eta\phi'(\delta) - (1-\eta)\phi'(-\delta)\right](f-\delta) = \frac{-\phi'(\delta)}{d}(1-\eta-d)(f-\delta) \geq 0.$$

Together with the facts that $d < 1/2$ and $C^s \Delta Q_\eta(\delta) \geq |1-\eta-d|^s$, we have

$$\Delta Q_\eta(f) \geq \Delta Q_\eta(\delta) \geq C^{-s}|1-\eta-d|^s \geq (2C)^{-s}|1-2\eta|^s.$$

Note that

$$\Delta R_\eta[C(f,\delta)] = 1-2\eta.$$

Therefore,

$$\Delta R_\eta[C(f,\delta)] \leq 2C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 4. $d < \eta < 1-d$ and $f < -\delta$. Following a similar argument as before,

$$Q_\eta(f) - Q_\eta(-\delta) \geq \frac{-\phi'(\delta)}{d}(d-\eta)(f+\delta) \geq 0.$$

Therefore,

$$\Delta Q_\eta(f) \geq \Delta Q_\eta(-\delta) \geq C^{-s}|\eta-d|^s,$$

which, together with the fact that $\Delta R_\eta[C(f,\delta)] = \eta - d$, implies that

$$\Delta R_\eta[C(f,\delta)] \leq C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 5. $d < \eta < 1-d$ and $|f| < \delta$. In this case,

$$\Delta R_\eta[C(f,\delta)] = 0 \leq C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 6. $d < \eta < 1-d$ and $f > \delta$. Observe that

$$Q_\eta(f) - Q_\eta(\delta) \geq \frac{-\phi'(\delta)}{d}(1-\eta-d)(f-\delta) \geq 0.$$

Hence

$$\Delta Q_\eta(f) \geq \Delta Q_\eta(\delta) \geq C^{-s}|1-\eta-d|^s,$$

which, together with the fact that $\Delta R_\eta[C(f,\delta)] = 1-\eta-d$, implies that

$$\Delta R_\eta[C(f,\delta)] \leq C\left[\Delta Q_\eta(f)\right]^{1/s}.$$

Case 7. $\eta > 1-d$. Observe that $R_\eta[C(f,\delta)] = R_{1-\eta}[C(-f,\delta)]$, and $Q_\eta(f) = Q_{1-\eta}(-f)$. Because $1-\eta < d$, from Cases 1, 2 and 3, we have

$$\begin{aligned}
\Delta R_\eta[C(f,\delta)] &= \Delta R_{1-\eta}[C(-f,\delta)] \\
&\leq 2C\left[\Delta Q_{1-\eta}(-f)\right]^{1/s} \\
&= 2C\left[\Delta Q_\eta(f)\right]^{1/s}.
\end{aligned}$$

The proof is therefore completed. ∎

Proof of Theorem 10. The proof follows from the same argument as that of Theorem 9. The only difference takes place in Case 3 where under the current assumptions

$$\Delta R_\eta(f) = 1 - 2\eta \le C\left[\Delta Q_\eta(\delta)\right]^{1/s}. \blacksquare$$

Proof of Theorem 11. The last part of the proof is based on the proof of Theorem 3 in Bartlett, Jordan and McAuliffe (2006). Let $g = C(f,\delta)$ be the classification rule with reject option based on $f : \mathcal{X} \to \mathbb{R}$ and set $g^* = C(f^*_\phi,\delta)$. We have shown above that under the assumptions of Theorem 9,

$$\begin{aligned}
|d-\eta|1\{g \neq g^*\}\left(1\{g=-1\}+1\{g^*=-1\}\right) &\le C\left[\Delta Q_\eta(f)\right]^{1/s} \\
|1-d-\eta|1\{g \neq g^*\}\left(1\{g=1\}+1\{g^*=1\}\right) &\le C\left[\Delta Q_\eta(f)\right]^{1/s}.
\end{aligned}$$

Moreover, Lemma 1 in Herbei and Wegkamp (2006) states that

$$\begin{aligned}
\Delta R(g) &= \mathbb{E}\left[|d-\eta(X)|1\{g(X) \neq g^*(X)\}\left(1\{g(X)=-1\}+1\{g^*(X)=-1\}\right)\right] \\
&\quad +\mathbb{E}\left[|1-d-\eta(X)|1\{g(X) \neq g^*(X)\}\left(1\{g(X)=1\}+1\{g^*(X)=1\}\right)\right].
\end{aligned} \tag{11}$$

Hence, for any $\varepsilon > 0$,

$$\begin{aligned}
&\Delta R(g) \\
&= \mathbb{E}\left[|d-\eta(X)|1\{d-\eta(X)| \le \varepsilon\}1\{g(X) \neq g^*(X)\}\left(1\{g(X)=-1\}+1\{g^*(X)=-1\}\right)\right] \\
&+\mathbb{E}\left[|d-\eta(X)|1\{d-\eta(X)| > \varepsilon\}1\{g(X) \neq g^*(X)\}\left(1\{g(X)=-1\}+1\{g^*(X)=-1\}\right)\right] \\
&+\mathbb{E}\left[|1-d-\eta(X)|1\{|1-d-\eta(X)| \le \varepsilon\}1\{g(X) \neq g^*(X)\}\left(1\{g(X)=1\}+1\{g^*(X)=1\}\right)\right] \\
&+\mathbb{E}\left[|1-d-\eta(X)|1\{|1-d-\eta(X)| > \varepsilon\}1\{g(X) \neq g^*(X)\}\left(1\{g(X)=1\}+1\{g^*(X)=1\}\right)\right] \\
&\le 2\varepsilon\mathbb{P}\{g^*(X) \neq g(X)\} + 2\varepsilon^{1-s}\Delta Q(f)
\end{aligned}$$

where we used (11) and the inequality $|x|1\{|x| \ge \varepsilon\} \le |x|^r\varepsilon^{1-r}$ for $r \ge 1$. Using the bound

$$\mathbb{P}\{g(X) \neq g^*(X)\} \le \left[2(8A)^{1/\alpha}\Delta R(g)\right]^\beta$$

from the proof of Lemma 4 of Herbei and Wegkamp (2006), and choosing

$$\varepsilon = c\left[\Delta R(g)\right]^{1-\beta}$$

with $c = [2(8A)^{1/\alpha}]^\beta/4$ readily gives the desired claim with $K = 4Cc^{1-s}$. $\blacksquare$

Proof of Theorem 18. Recall that $\bar{f} \in \mathcal{F}$ minimizes $Q(f)$ over $f \in \mathcal{F}$. Let $h(yf(x)) = \phi(yf(x)) - \phi(y\bar{f}(x))$. Since

$$\begin{aligned}
\frac{Q(f)+Q(\bar{f})}{2} &\ge Q\left(\frac{f+\bar{f}}{2}\right) + c\mathbb{E}\left[(f-\bar{f})^2(X)\right] \\
&\ge Q(\bar{f}) + c\mathbb{E}\left[(f-\bar{f})^2(X)\right],
\end{aligned}$$

we have

$$\begin{aligned}
\mathbb{E}\left[h^2(Yf(X))\right] &\le L^2\mathbb{E}\left[(f-\bar{f})^2(X)\right] \\
&\le \frac{L^2}{2c}\{Q(f)-Q(\bar{f})\} \\
&= \frac{L^2}{2c}\mathbb{E}\left[h(Yf(X))\right],
\end{aligned}$$

see, for example, Bartlett, Jordan and McAuliffe (2006). Since $\widehat{f}_n$ minimizes $Q_n(f)$, we have

$$\begin{aligned} Q(\widehat{f}_n) - Q(\bar{f}) &= Ph(y\widehat{f}_n(x)) \\ &= 2\mathbb{P}_n h(Y\widehat{f}_n(X)) + (P - 2\mathbb{P}_n)h(Y\widehat{f}_n(X)) \\ &\leq 2\mathbb{P}_n h(Y\bar{f}(X)) + (P - 2\mathbb{P}_n)h(Y\widehat{f}_n(X)) \\ &\leq \sup_{f\in\mathcal{F}}(P - 2\mathbb{P}_n)h(Yf(X)) \end{aligned}$$

where $\mathbb{P}h(Yf(X)) = \mathbb{E}[h(Yf(X))]$ and $\mathbb{P}_n h(Yf(X)) = (1/n)\sum_{i=1}^n h(Y_i f(X_i))$ for any $f \in \mathcal{F}$. Next we observe that

$$\sup_{f\in\mathcal{F}}(\mathbb{P} - 2\mathbb{P}_n)h(Yf(X)) \leq \frac{3L}{n} + \max_{f\in\mathcal{F}_n}(\mathbb{P} - 2\mathbb{P}_n)h(Yf(X))$$

where $\mathcal{F}_n$ is the minimal $1/n$-net of $\mathcal{F}$. By Bernstein's inequality, we get

$$\begin{aligned} \mathbb{P}\left\{\sup_{f\in\mathcal{F}_n}(\mathbb{P} - 2\mathbb{P}_n)h(Yf(X)) \geq t\right\} &\leq N_n \exp\left[-\frac{n\{t + \mathbb{P}h(Yf(X))\}^2/8}{\mathbb{P}h^2(Yf(X)) + (2LB)\{t + \mathbb{P}h(Yf(X))\}/6}\right] \\ &\leq N_n \exp\left[-\frac{nt}{8}\left(\frac{L^2}{2c} + \frac{LB}{3}\right)^{-1}\right] \end{aligned}$$

and the conclusion follows easily. ∎

Acknowledgments

The authors gratefully acknowledge the fact that part of the research was done while the authors were visiting the Isaac Newton Institute for Mathematical Sciences (Statistical Theory and Methods for Complex, High-Dimensional Data Programme) at Cambridge University during Spring 2008. The research of Ming Yuan was supported in part by NSF grants DMS-MPSA-0624841 and DMS-0846234. The research of Marten Wegkamp was supported in part by NSF Grant DMS-0706829.

References

P.L. Bartlett, M.I. Jordan and J. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101:138-156, 2006.

P.L. Bartlett and M.H. Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9:1823-1840, 2008.

J. Friedman, T. Hastie and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337-407, 2000.

R. Herbei and M.H. Wegkamp. Classification with reject option. *Canadian Journal of Statistics*, 34(4):709-721, 2006.

Y. Lin. Support vector machines and the Bayes rule in classification. *Machine Learning and Knowledge Discovery*, 6:259-275, 2002.

E. Mammen and A.B. Tsybakov. Smooth discrimination analysis. *Annals of Statistics*, 27(6):1808-1829, 1999.

J.S. Marron, M. Todd and J. Ahn. Distance weighted discrimination. *Journal of the American Statistical Association*, 102:1267-1271, 2007.

B.D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.

A.B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32:135-166, 2004.

M.H. Wegkamp. Lasso type classifiers with a reject option. *Electronic Journal of Statistics*, 1:155-168, 2007.

T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32:56-134, 2004.

Journal of Machine Learning Research 11 (2010) 131-170 Submitted 4/09; Revised 12/09; Published 1/10

An Investigation of Missing Data Methods for Classification Trees Applied to Binary Response Data

Yufeng Ding YUFENG.DING@MOODYS.COM
Moody's Investors Service
250 Greenwich Street
New York, NY 10007

Jeffrey S. Simonoff JSIMONOF@STERN.NYU.EDU
New York University, Stern School of Business
44 West 4th Street
New York, NY 10012

Editor: Charles Elkan

Abstract

There are many different methods used by classification tree algorithms when missing data occur in the predictors, but few studies have been done comparing their appropriateness and performance. This paper provides both analytic and Monte Carlo evidence regarding the effectiveness of six popular missing data methods for classification trees applied to binary response data. We show that in the context of classification trees, the relationship between the missingness and the dependent variable, as well as the existence or non-existence of missing values in the testing data, are the most helpful criteria to distinguish different missing data methods. In particular, separate class is clearly the best method to use when the testing set has missing values and the missingness is related to the response variable. A real data set related to modeling bankruptcy of a firm is then analyzed. The paper concludes with discussion of adaptation of these results to logistic regression, and other potential generalizations.

Keywords: classification tree, missing data, separate class, RPART, C4.5, CART

1. Classification Trees and the Problem of Missing Data

Classification trees are a supervised learning method appropriate for data where the response variable is categorical. The simple methodology behind classification trees is to recursively split data based upon the predictors that best distinguish the response variable classes. There are, of course, many subtleties, such as the choice of criterion function used to pick the best split variable, stopping rules, pruning rules, and so on. In this study, we mostly rely on the built-in features of the tree algorithms C4.5 and RPART to implement tree methods. Details about classification trees can be found in various references, for example, Breiman, Friedman, Olshen, and Stone (1998) and Quinlan (1993). Classification trees are computationally efficient, can handle mixed variables (continuous and discrete) easily and the rules generated by them are relatively easy to interpret and understand. Classification trees are highly flexible, and naturally uncover interaction effects among the independent variables. Classification trees are also popular because they can easily be incorporated into learning ensembles or larger learning systems as base learners.

Like most statistics or machine learning methods, "base form" classification trees are designed assuming that data are complete. That is, all of the values in the data matrix, with the rows being the observations (instances) and the columns being the variables (attributes), are observed. However, missing data (meaning that some of the values in the data matrix are not observed) is a very common problem, and for this reason classification trees have to, and do, have ways of dealing with missing data in the predictors. (In supervised learning, an observation with missing response value has no information about the underlying relationship, and must be omitted. There is, however, research in the field of semi-supervised learning methods that tries to handle the situation where the response value is missing, for example, Wang and Shen 2007.)

Although there are many different ways of dealing with missing data in classification trees, there are relatively few studies in the literature about the appropriateness and performance of these missing data methods. Moreover, most of these studies limited their coverage to the simplest missing data scenario, namely, missing completely at random (MCAR), while our study shows that the missing data generating process is one of the two crucial criteria in determining the best missing data method. The other crucial criterion is whether or not the testing set is complete. The following two subsections describe in more detail these two criteria.

1.1 Different Types of Missing Data Generating Process

Data originate according to the data generating process (DGP) under which the data matrix is "generated" according to the probabilistic relationships between the variables. We can think of the missingness itself as a random variable, realized as the matrix of the missingness indicator I_m. I_m is generated according to the missingness generating process (MGP), which governs the relationship between I_m and the variables in the data matrix. I_m has the same dimension as the original data matrix, with each entry equal to 0 if the corresponding original data value is observed and 1 if the corresponding original data value is not observed (missing). Note that an I_m value not only can be related to its corresponding original data value, but can also be related to other variables of the same observation.

Depending on the relationship between I_m and the original data, Rubin (1976) and Little and Rubin (2002) categorize the missingness into three different types. If I_m is dependent upon the missing values (the unobserved original data values), then the missingness pattern is called "not missing at random" (NMAR). Otherwise, the missingness pattern is called "missing at random" (MAR). As a special case of MAR, when the missingness is also not dependent on the observed values (that is, is independent of all data values), the missingness pattern is called "missing completely at random" (MCAR). The definition of MCAR is rather restrictive, which makes MCAR unlikely in reality. For example, in the bankruptcy data discussed later in the paper, there is evidence that after the Enron scandal in 2001, when both government and the public became more wary about financial reporting misconduct, missingness of values in financial statement data was related to the well-being of the company, and thus other values in the data. This makes intuitive sense because when scrutinized, a company is more likely to have trouble reporting their financial data if there were problems. Thus, focusing on the MCAR case is a major limitation that will be avoided in this paper. In fact, this paper shows that the categorization of MCAR, MAR and NMAR itself is not appropriate for the missing data problem in classification trees, as well as in another supervised learning context (at least with respect to prediction), although it has been shown to be helpful with likelihood-based or Bayesian analysis.

	Missingness is related to				
	Missing values	Observed Predictors	Response Variable	LR	Three-Letter
1	No	No	No	MCAR	$---$
2	No	Yes	No	MAR	$-X-$
3	Yes	No	No	NMAR	$M--$
4	Yes	Yes	No	NMAR	$MX-$
5	No	No	Yes	MAR	$--Y$
6	No	Yes	Yes	MAR	$-XY$
7	Yes	No	Yes	NMAR	$M-Y$
8	Yes	Yes	Yes	NMAR	MXY

Table 1: Eight missingness patterns investigated in this study and their correspondence to the categorization MCAR, MAR and NMAR defined by Rubin (1976) and Little and Rubin (2002) (the LR column). The column Three-Letter shows the notation that is used in this paper.

In this paper, we investigate eight different missingness patterns, depending on the relationship between the missingness and three types of variables, the observed predictors, the unobserved predictors (the missing values) and the response variable. The relationship is conditional upon other factors, for example, missingness is not dependent upon the missing values means that the missingness is conditionally independent of the missing values given the observed predictors and/or the response variable. Table 1 shows their correspondence with the MCAR/MAR/NMAR categorization as well as the three-letter notation we use in this paper. The three letters indicate if the missingness is conditionally dependent on the missing values (M), on other predictors (X) and on the response variable (Y), respectively. As will be shown, the dependence of the missingness on the response variable (the letter Y) is the one that affects the choice of best missingness data method. Later in the paper, some derived notations are also used. For example, $*X*$ means the union of $-X-$, $-XY$, $MX-$ and MXY, that is, the missingness is dependent upon the observed predictors, and it may or may not be related to the missing values and/or the response variable.

1.2 Scenarios Where the Testing Data May or May Not Be Complete

There are essentially two stages of applying classification trees, the training phase where the historical data (training set) are used to construct the tree, and the testing phase where the tree is put into use and applied to testing data. Similar to most other studies, this study deals with the scenario where missing data occur in the training set, but the testing set may or may not have missing values. One basic assumption is, of course, that the DGP (as well as MGP if the testing set also contains missing values) is the same for both the training set and the testing set.

While it would probably typically be the case that the testing data would also have missing values (generated by the same process that generated them in the training set), it should be noted that in certain circumstances a testing set without missing values could be expected. For example, consider a problem involving prediction of bankruptcy from various financial ratios. If the training set comes from a publicly available database, there could be missing values corresponding to information that was not supplied by various companies. If the goal is to use these publicly available data to try

to predict bankruptcy from ratios from one's own company, it would be expected that all of the necessary information for prediction would be available, and thus the test set would be complete.

This study shows that when the missingness is dependent upon the response variable and the test set has missing values, separate class is the best missing data method to use. In other situations, the choice is not as clear, but some insights on effective choices are provided. The rest of paper provides detailed theoretical and empirical analysis and is organized as follows. Section 2 gives a brief introduction to the previous research on this topic. This is followed by discussion of the design of this study and findings in Section 3. The generality of the results are then tested on real data sets in Section 4. A brief extension of the results to logistic regression is presented in Section 5. We conclude with discussion of these results and future work in Section 6.

2. Previous Research

There have been several studies of missing data and classification trees in the literature. Liu, White, Thompson, and Bramer (1997) gave a general description of the problem, but did not discuss solutions. Saar-Tsechansky and Provost (2007) discussed various missing data methods in classification trees and proposed a cost-sensitive approach to the missing data problem for the scenario when missing data occur only at the testing phase, which is different from the problem studied here (where missing values occur in the training phase).

Kim and Yates (2003) conducted a simulation study of seven popular missing value methods but did not find any dominant method. Feelders (1999) compared the performance of surrogate split and imputation and found the imputation methods to work better. (These methods, and the methods described below, are described more fully in the next section.) Batista and Monard (2003) compared four different missing data methods, and found that 10 nearest neighbor imputation outperformed other methods in most cases. In the context of cost sensitive classification trees, Zhang, Qin, Ling, and Sheng (2005) studied four different missing data methods based on their performances on five data sets with artificially generated random missing values. They concluded that the internal node method (the decision rules for the observations with the next split variable missing will be made at the (internal) node) is better than the other three methods examined. Fujikawa and Ho (2002) compared several imputation methods based on preliminary clustering algorithms to probabilistic split on simulations based on several real data sets and found comparable performance. A weakness of all of the above studies is that they focused only on the restrictive MCAR situation.

Other studies examined both MAR and NMAR missingness. Kalousis and Hilario (2000) used simulations from real data sets to examine the properties of seven algorithms: two rule inducers, a nearest neighbor method, two decision tree inducers, a naive Bayes inducer, and linear discriminant analysis. They found that the naive Bayes method was by far most resilient to missing data, in the sense that its properties changed the least when the missing rate was increased (note that this resilience is related to, but not the same as, its overall predictive performance). They also found that the deleterious effects of missing data are more serious if a given amount of missing values are spread over several variables, rather than concentrated in a few.

Twala (2009) used computer simulations based on real data sets to compare the properties of different missing value methods, including using complete cases, single imputation of missing values, likelihood-based multiple imputation (where missing values are imputed several times, and the results of fitting trees to the different generated data sets are combined), probabilistic split, and surrogate split. He studied MAR, MCAR, and NMAR missingness generating processes, although

dependence of missingness on the response variable was not examined. Multiple imputation was found to be most effective, with probabilistic split also performing reasonably well, although little difference was found between methods when the proportion of missing values was low. As would be expected, MCAR missingness caused the least problems for methods, while NMAR missingness caused the most, and as was also found by Kalousis and Hilario (2000), missingness spread over several predictors is more serious than if it is concentrated in only one. Twala, Jones, and Hand (2008) proposed a method closely related to creating a separate class for missing values, and found that its performance was competitive with that of likelihood-based multiple imputation.

The study described in the next section extends these previous studies in several ways. First, theoretical analyses are provided for simple situations that help explain observed empirical performance. We then extend these analyses to more complex situations and data sets (including large ones) using Monte Carlo simulations based on generated and real data sets. The importance of whether missing is dependent on the response variable, which has been ignored in previous studies on classification trees yet turns out to be of crucial importance, is a fundamental aspect of these results. The generality of the conclusions is finally tested using real data sets and application to logistic regression.

3. The Effectiveness of Missing Data Methods

The recursive nature of classification trees makes them almost impossible to analyze analytically in the general case beyond 2×2 tables (where there is only one binary predictor and a binary response variable). On the other hand, trees built on 2×2 tables, which can be thought of as "stumps" with a binary split, can be considered as degenerate classification trees, with a classification tree being built (recursively) as a hierarchy of these degenerate trees. Therefore, analyzing 2×2 tables can result in important insights for more general cases. We then build on the 2×2 analyses using Monte Carlo simulation, where factors that might have impact on performance are incrementally added, in order to see the effect of each factor. The factors include variation in both the data generating process (DGP) and the missing data generating process (MGP), the number and type of predictors in the data, the number of predictors that contain missing values, and the number of observations with missing data.

This study examines six different missing data methods: probabilistic split, complete case method, grand mode/mean imputation, separate class, surrogate split, and complete variable method. Probabilistic split is the default method of C4.5 (Quinlan, 1993). In the training phase, observations with values observed on the split variable are split first. The ones with missing values are then put into each of the child nodes with a weight given as the proportion of non-missing instances in the child. In the testing phase, an observation with a missing value on a split variable will be associated with all of the children using probabilities, which are the weights recorded in the training phase. The complete case method deletes all observations that contain missing values in any of the predictors in the training phase. If the testing set also contains missing values, the complete case method is not applicable and thus some other method has to be used. In the simulations, we use C4.5 to realize the complete case method. In the training phase, we manually delete all of the observations with missing values and then run C4.5 on the pre-processed remaining complete data. In the testing phase, the default missing data method, probabilistic split, is used. Grand mode imputation imputes the missing value with the grand mode of that variable if it is categorical. Grand mean is used if the variable is continuous. The separate class method treats the missing values as a new class

(category) of the predictor. This is trivial to apply when the original variable is categorical, where we can create a new category called "missing". To apply the separate class method to a numerical variable, we give all of the missing values a single extremely large value that is obviously outside of the original data range. This creates the needed separation between the nonmissing values and the missing values, implying that any split that involves the variable with missing values will put all of the missing observations into the same branch of the tree. Surrogate split is the default method of CART (realized using RPART in this study; Breiman et al. 1998 and Therneau and Atkinson 1997). It finds and uses a surrogate variable (or several surrogates in order) within a node if the variable for the next split contains missing values. In the testing phase, if a split variable contains missing values, the surrogate variables in the training phase are used instead. The complete variable method simply deletes all variables that contain missing values.

Before we start presenting results, we define a performance measure that is appropriate for measuring the impact of missing data. Accuracy, calculated as the percentage of correctly classified observations, is often used to measure the performance of classification trees. Since it can be affected by both the data structure (some data are intrinsically easier to classify than others) and by the missing data, this is not necessarily a good summary of the impact of missing data. In this study, we define a measure called *relative accuracy* (*RelAcc*), calculated as

$$RelAcc = \frac{\text{Accuracy with missing data}}{\text{Accuracy with original full data}}.$$

This can be thought of as a standardized accuracy, as *RelAcc* measures the accuracy achievable with missing values relative to that achievable with the original full data.

3.1 Analytical Results

In the following consistency theorems, the data are assumed to reflect the DGP exactly, and therefore the training set and the testing set are exactly the same. Several of the theorems are for 2×2 tables, and in those cases stopping and pruning rules are not relevant, since the only question is whether or not the one possible split is made. The proofs are thus dependent on the underlying parameters of the DGP and MGP, rather than on data randomly generated from them. It is important to recognize that these results are only designed to be illustrative of the results found in the much more realistic simulation analyses to follow. Proofs of all of the results are given in the appendix.

Before presenting the theorems, we define some terms to avoid possible confusion. First, a partition of the data refers to the grouping of the observations defined by the classification tree's splitting rules. Note that it is possible for two different trees on the same data set to define the same partition. For example, suppose that there are only two binary explanatory variables, X_1 and X_2, and one tree splits on X_1 then X_2 while another tree splits on X_2 then X_1. In this case, these two trees have different structures, but they can lead to the same partition of the data. Secondly, the set of rules defined by a classification tree consists of the rules defined by the tree leaves on each of the groups (the partition) of the data.

3.1.1 WHEN THE TEST SET IS FULLY OBSERVED WITH NO MISSING VALUES

We start with Theorems 1 to 3 that apply to the complete case method. Theorems 4 and 5 apply to probabilistic split and mode imputation, respectively. Proofs of the theorems can be found in the appendix.

Theorem 1 *Complete Case Method: If the MGP is conditionally independent of Y given X, then the tree built on the data containing missing values using the complete case method gives the same set of rules as the tree built on the original full data set.*

Theorem 2 *Complete Case Method: If the partition of the data defined by the tree built on the incomplete data is not changed from the one defined by the tree built on the original full data, the loss in accuracy when the testing set is complete is bounded above by P_M, where P_M is the missing rate, defined as the percentage of observations that contain missing values.*

Theorem 3 *Complete Case Method: If the partition of the data defined by the tree built on the incomplete data is not changed from the one defined by the tree built on the original full data, the relative accuracy when the testing set is complete is bounded below by*

$$RelAcc_{min} = \frac{1-P_M}{1+P_M},$$

where P_M is the missing rate. Notice that the tree structure itself could change as long as it gives the same final partition of the data.

There are similar results in regression analyses as in Theorem 1. In regression analyses, when the missingness is independent of the response variable, by using only the complete observations, the parameter estimators are all unbiased (Allison, 2001). This implies that in theory, when the missingness is independent of the response variable, using complete cases only is not a bad approach on average. However, in practice, as will be seen later, deleting observations with missing values can cause severe loss in information, and thus has generally poor performance.

Theorem 4 *Probabilistic Split: In a 2×2 data table, if the MGP is independent of either Y or X, given the other variable, then the following results hold for probabilistic split.*

1. *If X is not informative in terms of classification, that is, the majority classes of Y for different X values are the same, then probabilistic split will give the same rule as the one that would be obtained from the original full data;*

2. *If probabilistic split shows that X is informative in terms of classification, that is, the majority classes of Y for different X values are different, then it finds the same rule as the one that would be obtained from the original full data;*

3. *The absolute accuracy when the testing set is complete is bounded below by 0.5. Since the original full data accuracy is at most 1, the relative accuracy is also bounded below by 0.5.*

Theorem 5 *Mode Imputation: If the MGP is independent of Y, given X, then the same results hold for mode imputation as for probabilistic split under the conditions of Theorem 4.*

Theorems 1, 2 and 3 (for the complete case method) are true for general data sets. Theorems 4 and 5 are for 2×2 tables only but they imply that probabilistic split and mode imputation have advantages over the complete case method, which can have very poor performance (as will be shown in Figure 1).

Moreover, with 2×2 tables, the complete variable method will always have a higher than 0.5 accuracy since by ignoring the only predictor, we will always classify all of the data to the overall majority class and achieve at least 0.5 accuracy, and thus at least 0.5 relative accuracy. Together with Theorems 4 and 5, as well as the evidence to be shown in Figure 1, this is an indication that classification trees tend not to be hurt much by missing values, since trees built on 2×2 tables can be considered as degenerate classification trees and more complex trees are composites of these degenerate trees. The performance of a classification tree is the average (weighted by the number of observations at each leaf) over the degenerate trees at the leaf level, and, as will be seen later in the simulations, can often be quite good.

Surrogate split is not applicable to 2×2 tables because there are no other predictors. For 2×2 table problems with a complete testing set, separate class is essentially the same as the complete case method, because as long as the data are split according to the predictor (and it is very likely that this will be so), the separate class method builds separate rules for the observations with missing values; when the testing set is complete, the rules that are used in the testing phase are exactly the ones built on the complete observations. When there is more than one predictor, however, the creation of the "separate class" will save the observations with missing values from being deleted and affect the tree building process. It will very likely lead to a change in the tree structure. This, as will be seen, tends to have a favorable impact on the performance accuracy.

Figure 1 illustrates the lower bound calculated in Theorem 3. The illustration is achieved by Monte Carlo simulation of 2×2 tables. A 2×2 table with missing values has only eight cells, that is, eight different value combinations of the binary variables X, Y and M, where M is the missingness indicator such that $M = 0$ if X is observed and $M = 1$ if X is missing. There is one constraint, that the sum of the eight cell probabilities must equal one. Therefore, this table is determined by seven parameters. In the simulation, for each 2×2 table, the following seven parameters (probabilities) are randomly and independently generated from a uniform distribution between $(0,1)$: (1)$P(X = 1)$, (2)$P(Y = 1|X = 0)$, (3)$P(Y = 1|X = 1)$, (4)$P(M = 1|X = 0,Y = 0)$, (5)$P(M = 1|X = 0,Y = 1)$, (6)$P(M = 1|X = 1,Y = 0)$ and (7)$P(M = 1|X = 1,Y = 1)$. Here we assume the data tables reflect the true underlying DGP and MGP without random variation, and thus the expected performance of the classification trees can be derived using the parameters. In this simulation, sets of the seven parameters are generated (but no data sets are generated using these parameters) repeatedly, and the relative accuracy of each missing data method on each parameter set is determined. One million sets of parameters are generated for each missingness pattern.

In Figure 1, the plot on the left is a scatter plot of relative accuracy versus missing rate for each Monte Carlo replication for the complete case method when the MGP depends on the response variable. The lower bound is clearly shown. We can see that when the missing rate is high, the lower bound can reduce to almost zero (implying that not only relative accuracy, but accuracy itself, can approach zero). This perhaps somewhat counterintuitive result can occur in the following way. Imagine the extreme case where almost all cases are positive and (virtually) all of the positive cases have missing predictor value at the training phase; in this situation the resultant rule will be to classify everything as negative. When this rule is applied to a complete testing set with almost all positive cases, the accuracy will be almost zero. The graph on the right is the quantile version of the scatter plot on the left. The lines shown in the quantile plot are the theoretical lower bound, the 10th, 20th, 30th, 40th and 50th percentile lines from the lowest to the highest. Higher percentile lines are the same as the 50th percentile (median) line, which is already the horizontal line at $RelAcc = 1$. The percentile lines are constructed by connecting the corresponding percentiles in a moving window

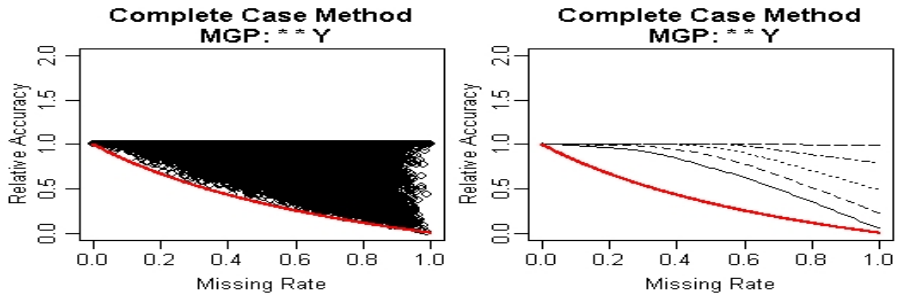


Figure 1: Scatter plot and the corresponding quantile plot of the complete testing set *RelAcc* vs. missing rate of the complete case method when the MGP is dependent on the response variable. Recall that "$* * Y$" means the MGP is conditionally dependent on the response variable but no restriction on the relationship between the MGP and other variables, missing or observed, is assumed. Each point in the scatter plot represents the result on one of the simulated data tables.

of data from the left to the right. Due to space limitations, we do not show quantile plots of other missing data methods and/or under different scenarios, but in all of the other plots, the quantile lines are all higher (that is, the quantile plot in Figure 1 shows the worst case scenario). The plots show that the missing data problem, when the missing rate is not too high, may not be as serious as we might have thought. For example, when 40% of the observations contain missing data, 80% of the time the expected relative accuracy is higher than 90%, and 90% of the time the expected relative accuracy is higher than 80%.

3.1.2 WHEN THE TEST SET HAS MISSING VALUES

Theorem 6 *Separate Class: In 2×2 data tables, if missing values occur in both the training set and the testing set, then the separate class method achieves the best possible performance.*

In the Monte Carlo simulation of the 2×2 tables, the head-to-head comparison between the separate class method and other missing data methods confirmed the uniform dominance of the separate class when the test set also contains missing values, regardless whether the MGP is dependent on the response variable or not. However, as shown in Figure 2, when the MGP is independent of the response variable, separate class never performances better than the performance on the original full data, indicated by relative accuracies less than one. This means that separate class is not gaining from the missingness. On the other hand, when the MGP is dependent on the response variable, a fairly large percentage of the time the relative accuracy of the separate class method is larger than one (the quantiles shown are from the 10th to the 90th percentile with increment 10 percent). This means that trees based on the separate class method can improve on predictive performance compared to the situation where there are no missing data. Our simulations show that other methods can also gain from the missingness when the MGP is dependent on the response variable, but not as frequently as the separate class method and the gains are in general not as large. We follow up on this behavior in more detail in the next section, but the simple explanation is that since missingness depends on the response variable, the tree algorithm can use the presence of missing data in an observation to improve prediction of the response for that observation. Duda, Hart, and Stork (2001) and Hand (1997) briefly mentioned this possibility in the classification context, but did not give any

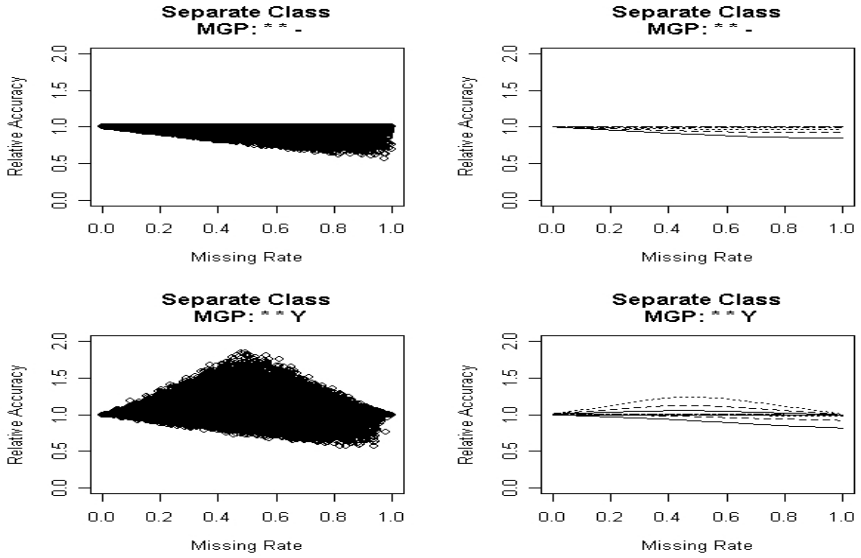


Figure 2: Scatter plot of the separate class method with incomplete testing set. Each point in the scatter plot represents the result on one of the simulated data tables.

supporting evidence. Theorem 6 makes a fairly strong statement in the simple situation, and it will be seen to be strongly indicative of the results in more general cases.

3.2 Monte Carlo Simulations of General Data Sets

In this section extensions of the simulations in the last section are summarized.

3.2.1 AN OVERVIEW OF THE SIMULATION

The following simulations are carried out.

1. 2×2 tables, missing values occur in the only predictor.
2. Up to seven binary predictors, missing values occur in only one predictor.
3. Eight binary predictors, missing values occur in two of them.
4. Twelve binary predictors, missing values occur in six of them.
5. Eight continuous predictors, missing values occur in two of them.
6. Twelve continuous predictors, missing values occur in six of them.

Two different scenarios of each of the last four simulations listed above were performed. In the first scenario, the six complete predictors are all independent of the missing ones, while in the second scenario three of the six complete predictors are related to the missing ones. Therefore, ten simulations were done in total.

In each of the simulations, 5000 sets of DGPs are simulated in order to cover a wide range of different-structured data sets so that a generalizable inference from the simulation is possible. For

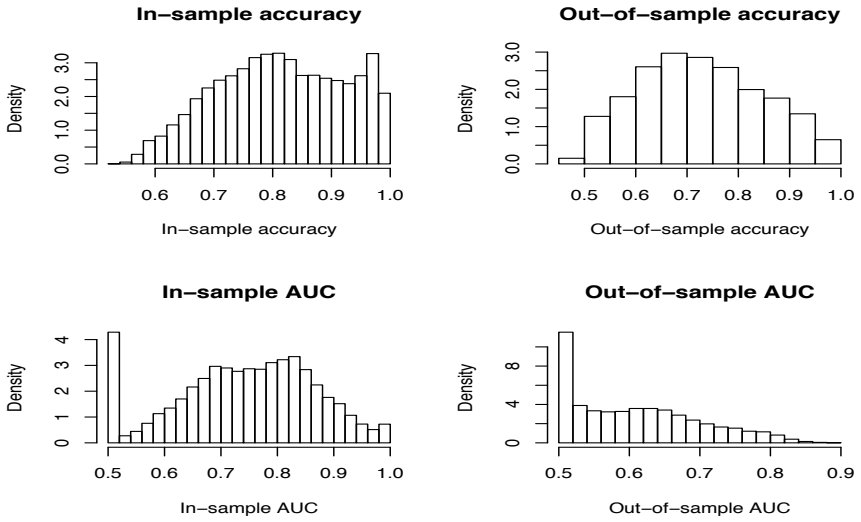


Figure 3: A summary of the tree performance on the simulated original full data.

each DGP, eight different MGPs are simulated to cover different types of missingness patterns. For each data set, the variables are generated sequentially in the order of the predictors, the response and the missingness. The probabilities associated with the binary response variable and the binary missingness variable are generated using conditional logit functions. The predictors may or may not be correlated with each other. Details about the simulations implementation can be found in Ding and Simonoff (2008). For each set of DGP/MGP, several different sample sizes are simulated to see any possible learning curve effect, since it was shown by Perlich, Provost, and Simonoff (2003) that sample size is an important factor in the effectiveness of classification trees. Figure 3 shows the distribution of the tree performance on the simulated original full data, as measured by accuracy and area under the ROC curve (AUC). As we can see, there is broad coverage of the entire range of strength of the underlying relationship. Also, as expected, the out-of-sample performance (on the test set) is generally worse than the in-sample performance (on the training set). When the in-sample AUC is close to 0.5, a tree is likely to not split and as a result, any missing data method will not actually be applied, resulting in equivalent performance over all of them. To make the comparisons more meaningful, we exclude the cases where the in-sample AUC is below 0.7. Lower thresholds for exclusion (0.55 and 0.6) yield very similar results.

Of the six missing data methods covered by this study, five of them, namely, complete case method, probabilistic split, separate class, imputation and complete variable method, are realized using C4.5. These methods are always comparable. However, surrogate split is carried out using RPART, which makes it less comparable to the other methods because of differences between RPART and C4.5 other than the missing data methods. To remedy this problem, we tuned the RPART parameters (primarily the parameter "cp") so that it gives balanced results compared to C4.5 when applied to the original full data (i.e., each has a similar probability of outperforming the other), and special attention is given when comparing RPART with other methods. The out-of-sample performances of each pair of missing data methods were compared based on both t-tests and nonparametric tests; each difference discussed in the following sections was strongly statistically significant.

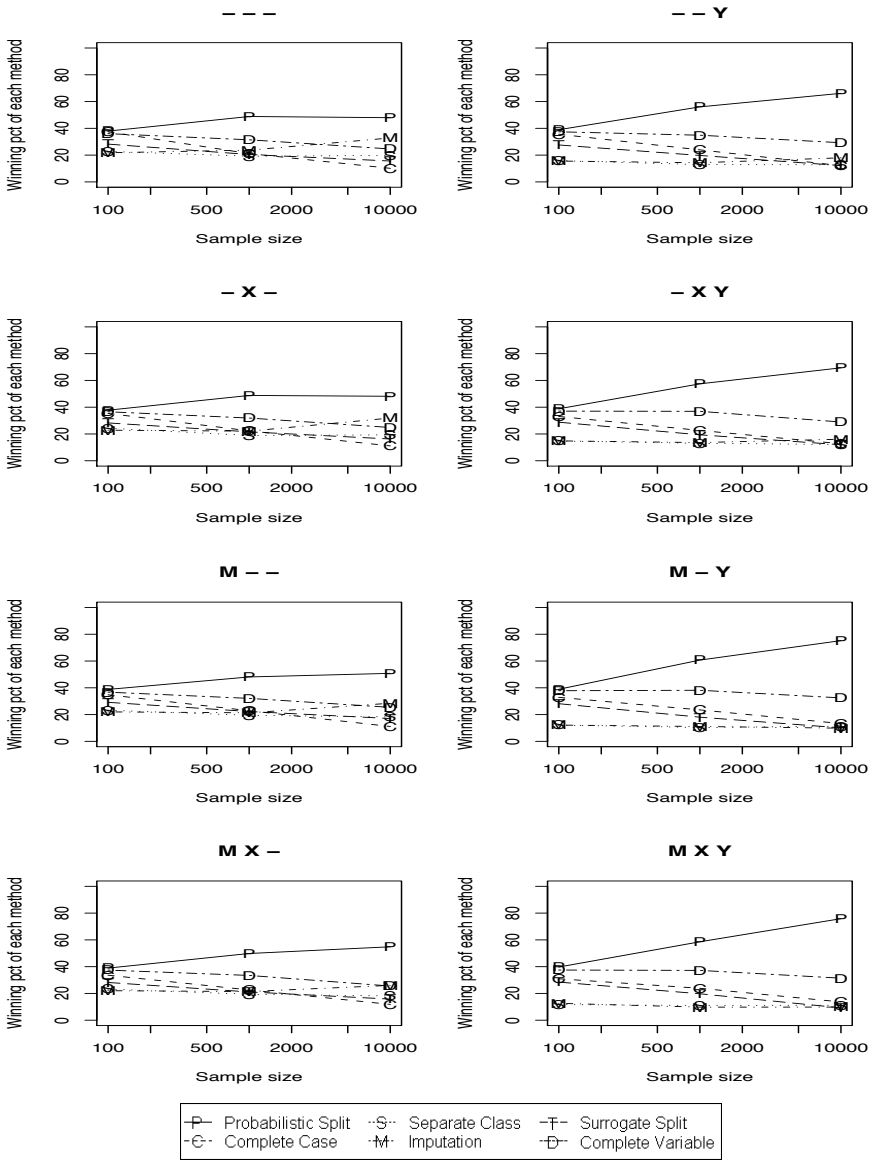


Figure 4: A summary of the order of six missing data methods when tested on a new complete testing set. The Y axis is the percentage of times each method is the best (including being tied with other methods; therefore the percentages do not sum up to one).

3.2.2 THE TWO FACTORS THAT DETERMINE THE PERFORMANCE OF DIFFERENT MISSING DATA METHODS

The simulations make clear that the dependence relationship between the missingness and the response variable is the most informative factor in differentiating different missing data methods, and thus is most helpful in determining the appropriateness of the methods. This can be clearly seen in Figures 4 and 5 (these figures refer to the case with twelve continuous predictors, six of which are subject to missing values, but results for other situations were broadly similar). The left column in

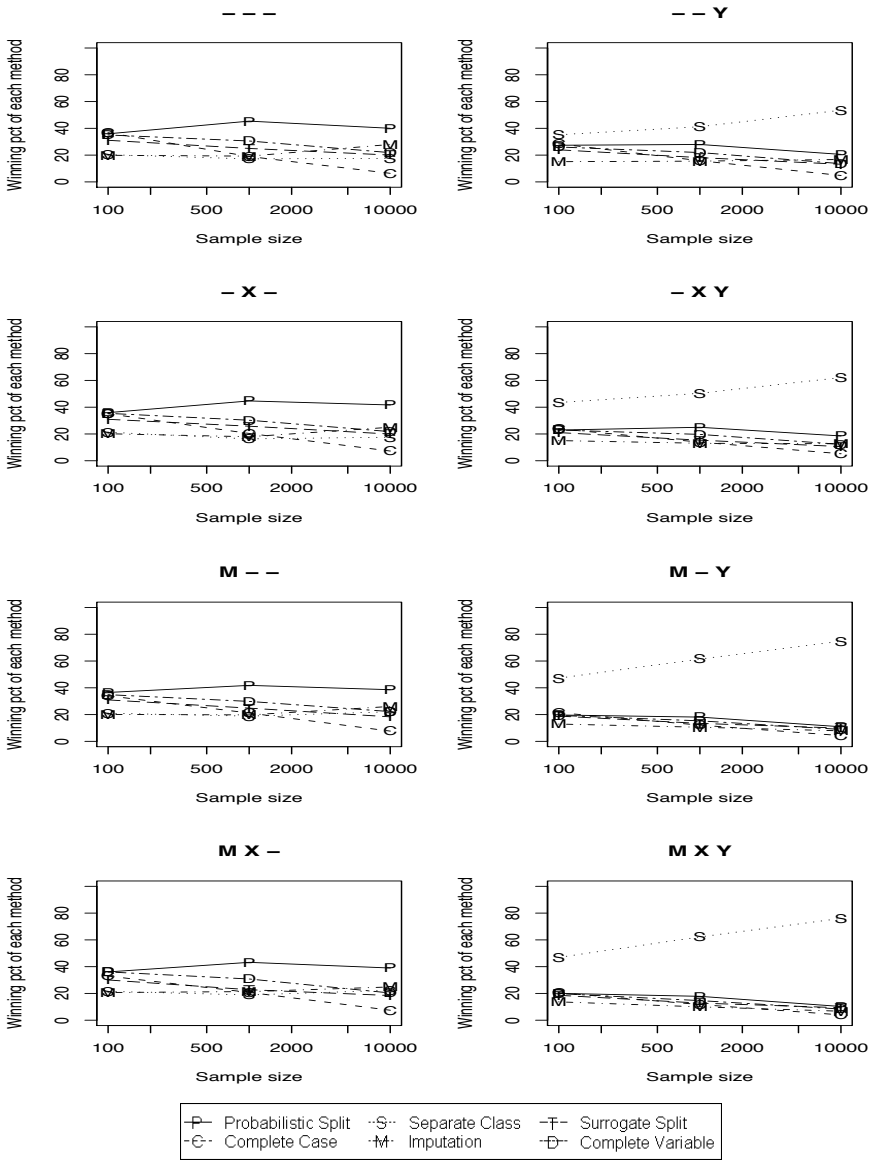


Figure 5: A summary of the order of six missing data methods when tested on a new incomplete testing set. The Y axis is the percentage of times each method is the best (including being tied with other methods).

the pictures shows the results when the missingness is independent of the response variable and the right column shows the results when the missingness is dependent on the response variable. We can see that there are clear differences between the two columns, but within each column there is essentially no difference. This also says the categorization of MCAR/MAR/NMAR (which is based upon the dependence relationship between the missingness and missing values, and does not distinguish the dependence of the missingness on other Xs and on Y) is not helpful in this context.

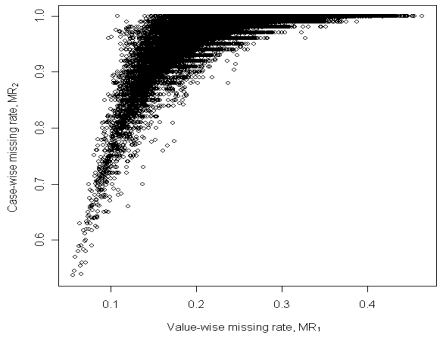


Figure 6: Plot of the case-wise missing rate MR_2 versus the value-wise missing rate MR_1 in the simulations using the 36 real data sets.

Comparison of the right columns of Figures 4 and 5 shows that whether or not there are missing values in the testing set is the second important criterion in differentiating between the methods. The separate class method is strongly dominant when the testing set contains missing values and the missingness is related to the response variable. The reason for this is that when missing data exist in both the training phase and the testing phase, they become part of the data and the MGP becomes an essential part of the DGP. This, of course, requires the assumption that the MGP (as well as the DGP) is the same in both the training phase and the testing phase. Under this scenario, if the missingness is related to the response variable, then there is information about the response variable in the missingness, which should be helpful when making predictions. Separate class, by taking the missingness directly as an "observed" variable, uses the information in the missingness about the response variable most effectively and thus is the best method to use. As a matter of fact, as can be seen in the bottom rows of Figures 7 and 8 (which give average relative accuracies separated by missing rate), the average relative accuracy of separate class under this situation is larger than one, indicating, on average, a better performance than with the original full data.

On the other hand, when the missing data only occur in the training phase and the testing set does not have missing values, or when the missingness is not related to and carries no information about the response variable, the existence of missing values is a nuisance. Its only effect is to obscure the underlying DGP and thus would most likely reduce a tree's performance. In this case, simulations show probabilistic split to be the dominantly best method. However, we don't see this dominance later in results based on real data sets. More discussion of this point will follow in Section 4.

3.2.3 MISSING RATE EFFECT

There are two ways of defining the missing rate: the percentage of predictor values that are missing from the data set (the value-wise missing rate, termed here MR_1), and the percentage of observations that contain missing values (the case-wise missing rate, termed here MR_2). If there is only one predictor, as is the case with 2×2 tables, then the two definitions are the same. We have seen earlier in the theoretical analyses that the missing rate has a clear impact on the performance of the missing data methods. In the simulations, there is also evidence of a relationship between relative performance and missing rate, whichever definition is used to define the missing rate.

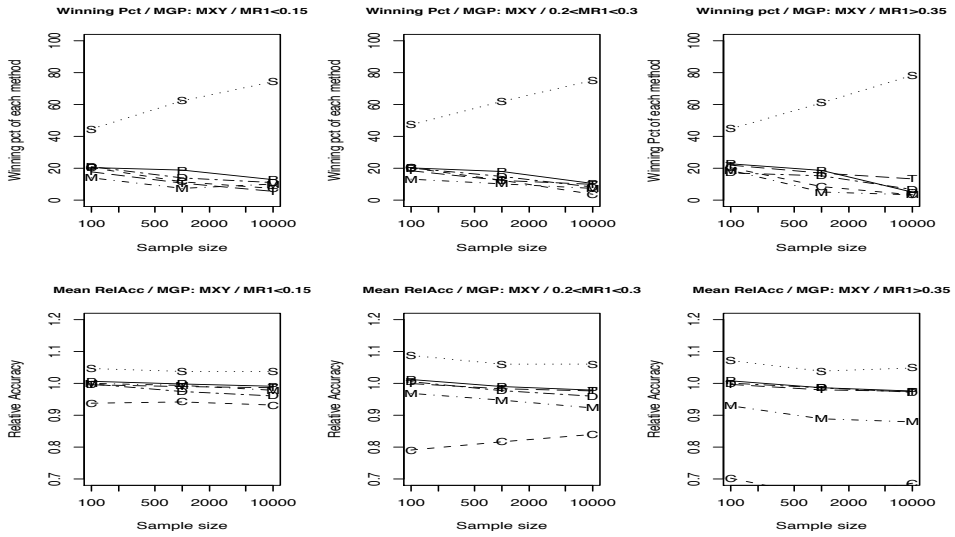


Figure 7: A comparison of the low, median and high missing rate situations. The top row shows the comparison in terms of winning percentage and the bottom row shows the comparison of the absolute performance of each missing data method.

Figure 6 shows the relationship between MR_1 and MR_2 in the simulations with 12 continuous predictors and 6 of them with missing values. Notice that in this setting, MR_1 is naturally between 0 and 0.5 (since half of the predictors can have missing values). MR_2 values are considerably larger than MR_1 values, as would be expected.

The simulations clearly show that the relative performance of different missing data methods is very consistent regardless of the missing rate (see the top row of Figure 7). However, the bottom row of Figure 7 shows that the absolute performance of the complete case method and the mean imputation method deteriorate as the missing rate gets higher. It also shows that separate class method performs best when the missing rate is neither too high or too low, although this effect is relatively small. Interestingly, the relative accuracy of the other missing data methods is very close to one regardless of the missing rate, indicating that they can almost achieve the same accuracy as if the data are complete without missing values.

A final effect connected to missing rate relates to results in earlier papers (Kalousis and Hilario, 2000; Twala, 2009) that suggested that missingness over several predictors is more problematic than missingness concentrated in a few predictors. This pattern was not evident here (e.g., in comparing the results for 8 predictors with 2 having missing values to those for 12 predictors with 6 having missing values), but it should be noted that the comparisons here are based on relative performance between methods, not absolute performance. That is, even if absolute performance deteriorates in the presence of missingness over multiple predictors, this is less important to the data analyst than is relative performance between methods (since a method must be chosen), and with respect to the latter criterion the observed patterns are reasonably stable.

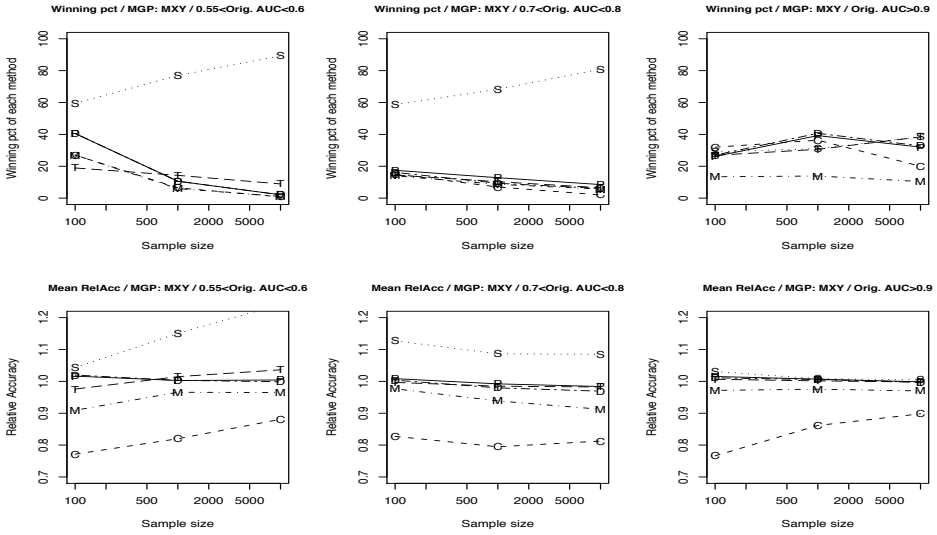


Figure 8: A comparison of the low, median and high original full data AUC situations. The top row shows the comparison in terms of winning percentage and the bottom row shows the comparison of the absolute performance of each missing data method.

3.2.4 THE IMPACT OF THE ORIGINAL FULL DATA AUC

Figure 8 shows that the original full data AUC primarily has an impact on the performance of separate class method. When the original full data AUC is higher, the loss in information due to missing values is less likely to be compensated by the information in the missingness, and thus separate class method deteriorates in performance (see the bottom row of Figure 8). When the original AUC is very high, although separate class still does a little better on average, it loses the dominance over the other methods.

Another observation is that the missing data methods other than separate class have fairly stable relative accuracy, with complete case and mean imputation consistently being the poorest performers (see the graphs in the bottom rows of both Figure 7 and Figure 8). This is true regardless of the AUC or the missing rate, even when the missingness does not depend on the response variable and there are no missing data in the testing set where, in theory, the complete case method can eventually recover the DGP.

4. Performance On Real Data Sets

In this section, we show that most of the previously described results hold when using real data sets. Moreover, we propose a method of determining the best missing data method to use when analyzing a real data set. Unlike in the previous sections, in these simulations based on real data, default settings of C4.5 are used and RPART is tuned (primarily using its parameter "cp") to get similar performance on the original full data as C4.5. Therefore, in particular, the effect of pruning is present. In Section 4.1, we show the results on 36 data sets that were originally complete. In Section 4.2, we propose a way to determine the best missing data method to use when facing real

Missingness is related to				
Missing values	Observed Predictors	Response Variable	LR	Three-Letter
No	No	No	MCAR	– – –
No	No	Yes	MAR	– – Y
Yes	No	No	NMAR	M – –

Table 2: Three missingness patterns used in simulations based on real data sets. The LR column shows the categorization according to Rubin (1976) and Little and Rubin (2002). The Three-Letter column shows the categorization used in this paper.

data sets that contain missing values (since in that case the true missingness generating process is not known by the data analyst).

4.1 Results on Real Data Sets with Simulated Missing Values

The same 36 data sets as in Perlich, Provost, and Simonoff (2003) are used here (except for Covertype and Patent, which are too big for RPART to handle; in those cases a random subset of 100,000 observations for each of them was used as the "true" underlying data set). They are either complete or were made complete by Perlich et al. (2003). Missing values with different missingness patterns were generated for the purpose of this study. According to the earlier results, the only important factor in the missingness generating process is the relationship between the missingness and the response variable. Therefore, two missingness patterns are included. In one of them, missingness is independent of all of the variables (including the response variable). In the other one, missingness is related to the response variable, but independent of all of the predictors. These two missingness patterns can be categorized as missing completely at random (MCAR) and missing at random (MAR), respectively. To account for this categorization of MGPs, the third type of missingness, not missing at random (NMAR), is also included. In the NMAR case, missingness is made dependent upon the missing values but not on the response variable (see Table 2). To maximize the possible effect of missing values, the first split variable of the original full data is chosen as the variable that contains missing values. It can be either numeric or categorical (binary or multi-categorical). Ten new data sets with missing values are generated for each combination of data set, training set size, and missingness pattern combination, with the missing rate chosen randomly for each. The performance of the missing data methods is measured out-of-sample, on a hold out test sample.

The same six missing data methods, namely, the complete case method, the complete variable method, probabilistic split, grand mode/mean imputation, surrogate split and the separate class method are applied. All of them are realized using C4.5 except for surrogate split, which is realized using RPART. C4.5 is run with its default settings. To make surrogate split comparable to the other missing data methods, the RPART parameters are tuned for each data set and each sample size so that RPART and C4.5 have comparable in sample performance on the original full data (by comparable performance we mean the average in sample original full data accuracies are similar to each other).

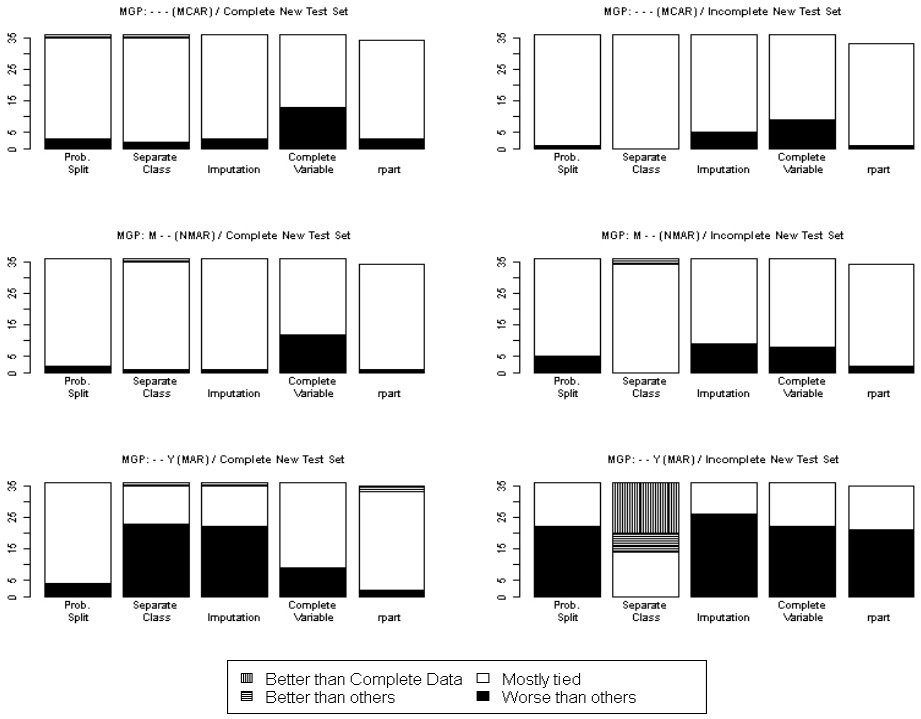


Figure 9: A tally of the relative out-of-sample performance measured in accuracy of all of the missing data methods on the 36 data sets.

4.1.1 THE TWO FACTORS AND THE BEST MISSING DATA METHOD

Consistent with the earlier results, the two factors that differentiate the performance of different missing data methods are whether the testing set is complete and whether the missingness is dependent upon the response variable. Figure 9 summarizes the relative out-of-sample performance in terms of accuracy of all of the missing data methods under different situations. In the graph, each bar represents one missing data method. Since the complete case method is consistently the worst method, it is omitted in the comparisons. Within each bar, the blank block shows the frequency that the missing data method has comparable performance with others. The shadowed block on the bottom shows the frequency that the missing data method has worse performance than others. The line-shadowed blocks on the top show the frequency that the missing data method has better performance than others, with the vertically line-shadowed block further indicating that the missing data method has better performance than with the original full data.

As was seen in the previous section, when missingness is related to the response variable and the test set contains missing values (the graph at the bottom right corner of Figure 9), the separate class method is dominant and in almost half of the cases, its performance is even better than the full data performance. Interestingly, the middle plot on the right shows that the separate class method still has an edge over the others (sometimes even over the original full data) when the test set contains missing values and the missingness is dependent upon the predictor but conditionally independent of the response variable. This is probably due to the indirect relationship between the missingness

and the response variable because both the missingness and the response variable are related to the predictor.

However, the dominance of probabilistic split is not observed in these real data sets. One possible reason could be the effect of pruning, which is used in these real data sets. The other two methods realized using C4.5 (imputation and separate class) both work with "filled-in" data sets, while probabilistic split takes the missing values as-is. Given this, we speculate that the branches with missing values are more likely to be pruned under probabilistic split, which causes it to lose predictive power. Another possible reason could be the competition from surrogate split, which is realized using RPART. Although we tried to tune RPART for each data set and each sample size, RPART and C4.5 are still two different algorithms. Different features of RPART and C4.5, other than the missing data methods, may cause RPART to outperform C4.5. Complete variable method performs a bit worse than the others, presumably because in these simulations the initial split variable on the full data was used as the variable with missing values.

In addition to accuracy, AUC was also tested as an alternative performance measure. We also examined the use of bagging (bootstrap aggregating) to reduce the variability of classification trees (discussion of bagging can be found in many sources, for example, Hastie et al. 2001). The learning curve effect (that is, the relationship between effectiveness and sample size) is also examined. We see patterns consistent with those in the simulated data sets. That is, the relative performance of the missing data methods is fairly consistent across different sample sizes.

4.1.2 THE EFFECT OF MISSING RATE

Figure 10 shows the distribution of the generated missing rates in these simulations. Recall that missing values occur in one variable, so this missing rate is the percentage of observations that have missing values, that is, MR_2 as defined earlier. Figure 11 shows a comparison between the case when the missing rate is low ($MR_2 < 0.2$) and the case when the missing rate is high ($MR_2 > 0.8$). For brevity, only the result when the MGP is dependent on the response variable is shown; differences between the low and high missing rate situations for other MGP's are similar. Since the missing rate is chosen at random, some of the original data sets do not have any generated data sets with simulated missing values with low missing rate, while for others we do not have any with high missing rate, which accounts for the "no data" category in the figures. Also, when the missing rate is high, the complete case method is obviously much worse than other missing data methods, and is therefore omitted from the comparison in that situation.

By comparing the graphs in Figures 11 with the corresponding ones in Figure 9, we can see some of the effects of missing rate. First, when the missing rate is lower than 0.2, the complete case method has comparable performance to other methods other than the complete variable method. This is unsurprising, as in this situation the complete case method does not lose much information from omitted observations. Secondly, the complete variable method has the worst performance when the missing rate is low, presumably (as noted earlier) because the complete variable method omits the most important explanatory variable in these simulations.

Moreover, in both the low and high missing rate cases, when the missingness depends on the response and the testing set is incomplete, the dominance of the separate class is not as strong as it is in Figure 9. This indicates that separate class works best when the missing rate is moderate. If the missing rate is too low, there might not be enough observations in the category of "missing" for the separate class method to be as effective. On the other hand, if the missing rate is very high, the

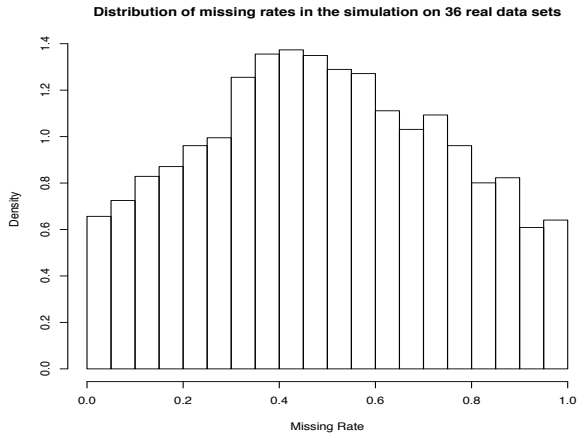


Figure 10: The distribution of missing rate in the simulation on 36 real data sets.

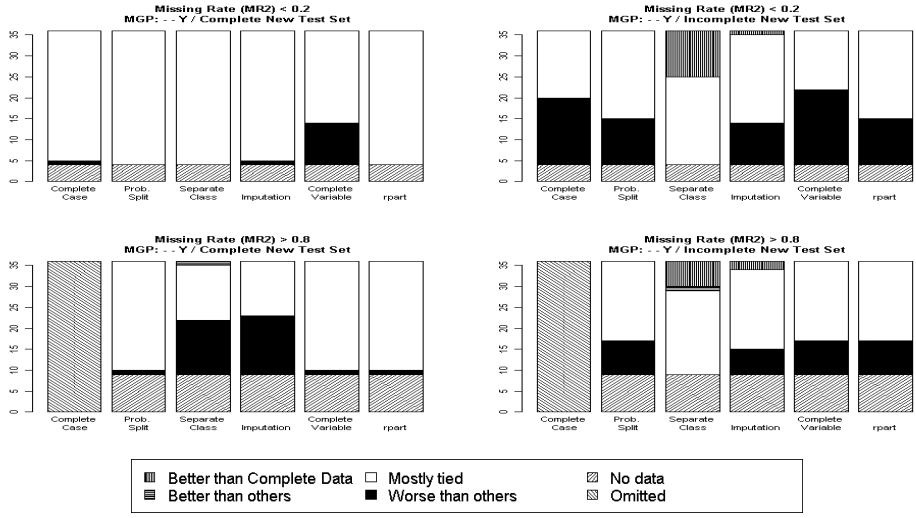


Figure 11: A comparison of the relative out-of-sample performance with low and high missing rates. Shown here, as an example, is the relative performance when the missingness is dependent upon the response variable. The left column is for the cases where the test set is fully observed and the right column for the cases where the test set has missing values. Top row shows the cases with low missing rate ($MR_2 < 0.2$) and bottom row shows the cases with high missing rate ($MR_2 > 0.8$)

.

information gained by separate class may not be enough to compensate for the lost information in the missing values, making all of the methods more comparable. This observation is consistent with Figure 2, where it is very clear that separate class gains the most when the missing rate is around 50%, as well as Figure 7, where the bottom row shows that separate class has better performance when the missing rate is neither too high or too low.

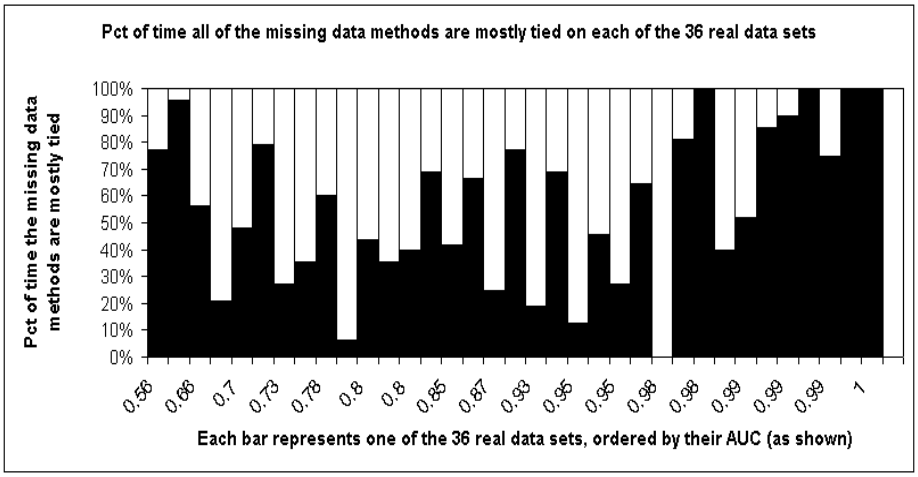


Figure 12: A tally of the missing data methods performance differentiation by data separability (measured by AUC).

4.1.3 IMPACT OF THE DATA SEPARABILITY, MEASURED BY ORIGINAL FULL DATA AUC

The experiment with these 36 data sets also shows that data separability (measured by AUC) is informative about the performance differentiation between different missing data methods (see Figure 12). In the graph, each vertical bar represents one of the 36 data sets, which are ordered from left to right according to their maximum full data AUC (as calculated by Perlich et al. 2003) from smallest to the greatest. The X-axis label shows the AUCs of the data sets. The height of each black bar shows the percentage of time when all of the missing data methods have mostly tied performance on the data set. The percentage is calculated as follows. There are three simulated missingness patterns (MCAR, NMAR and missingness depending on Y), four different testing sets (complete training set, complete new test set, incomplete training set and incomplete new test set) and four performance measures (accuracy, AUC and their bagged versions). This yields 48 measurement blocks for each data set. The performances of all of the missing data methods are compared within each block. If within a block, all of the missing data methods have very similar performance, the block is marked as mostly tied. Otherwise, the block is marked as having at least one method performing differently. The percentage is the proportion of the 48 blocks that are marked as mostly tied.

Figure 12 shows that when data separability is very high, as indicated by an AUC very close to 1 (the right end of the graph), the performances of different missing data methods are more likely to be tied. This is presumably due to the fact that strong signals in the data are less likely to be affected by missing data. The last data set (Nurse) is an exception because there is only one useful predictor. Since we picked the most significant predictor to create missing values in, when the complete variable method is used, the only useful predictor was always deleted and thus the complete variable method always had worse performance than others. As a result, on this data set, none of the measurement blocks is marked as mostly tied. This is consistent with the observations made in Figure 8.

4.2 A Real Data Set With Missing Values

We now present a real data example with naturally occurred missing values. In this example, we try to model a company's bankruptcy status given its key financial statement items. The data are annual financial statement data and the predictions are sequential. That is, we build the tree on one year's data and then test its performance on the following year's data. For example, we build a tree on 1987's data and test its performance on 1988's data, then build a tree on 1988's data and test it on 1989 data, and so on.

The data are retrieved from Compustat North America (a database of U.S. and Canadian fundamental and market information on more than 24,000 active and inactive publicly held companies). Following Altman and Sabato (2005), twelve variables from the data base are used as potential predictors: Current Assets, Current Liabilities, Assets, Sales, Operating Income Before Depreciation, Retained Earnings, Net Income, Operating Income After Depreciation, Working Capital, Liabilities, Stockholder's Equity and year. The response variable, bankruptcy status, is determined using two footnote variables, the footnote for Sales and the footnote for Assets. Companies with remarks corresponding to "Reflects the adoption of fresh-start accounting upon emerging from Chapter 11 bankruptcy" or "Company in bankruptcy or liquidation" are marked as bankruptcy. The data include all active companies, and span 19 years from 1987 to 2005. There are 177560 observations in the original retrieved data, but 76504 of the observations have no data except for the company identifications, and are removed from the data set, resulting in 99056 observations. There are 19238 (19.4%) observations containing missing values and there are 56820 (4.8%) missing data values.

According to the results in Sections 3 and 4.1, there are two criteria that differentiate the performance of different missing data methods, that is, whether or not there are missing values in the testing set and whether or not the missingness depends on the response variable. In the bankruptcy data, there are missing values in every year's data, and thus missing values in each testing data set. To assess the dependence of the missingness on the response variable, the following test is carried out. First, we define twelve new binary missingness indicators corresponding to the original twelve predictors. Each indicator takes on value 1 if the original value for the associated variable is missing and 0 if the original value is observed for that observation. We then build a tree for each year's data using the indicators as the predictors and the original response variable, the bankruptcy status, as the response variable. From 1987 to 2000, the tree makes no split, indicating the tree algorithm is not able to establish a relationship between the missingness and the response variable. From 2001 to 2005, the classification tree consistently splits on the missingness indicators of Sales and Retained Earnings. This indicates that the missingness of these predictors has information about the response variable in these years, and the MGP across the years is fairly consistent in missingness in sales and retained earnings being related to bankruptcy status. However, the AUC values calculated from the trees built with the missingness indicators are not very high, all being between 0.5 and 0.6. Therefore, the relationship is not a very strong one.

Given these observations and the fact that the sample sizes are fairly large, we would make the following propositions based on our earlier conclusions. First, from 1988 to 2001 (since the tree tested on 2001 data is built on 2000 data), different missing data methods should have similar performance, with no clear winners. However, from year 2002 to year 2005, the separate class method should have better performance than the others (but perhaps not much better since the relationship between missingness and the response is not very strong). The actual relative performance of different missing data methods is shown in Figure 13. Since surrogate split is realized using

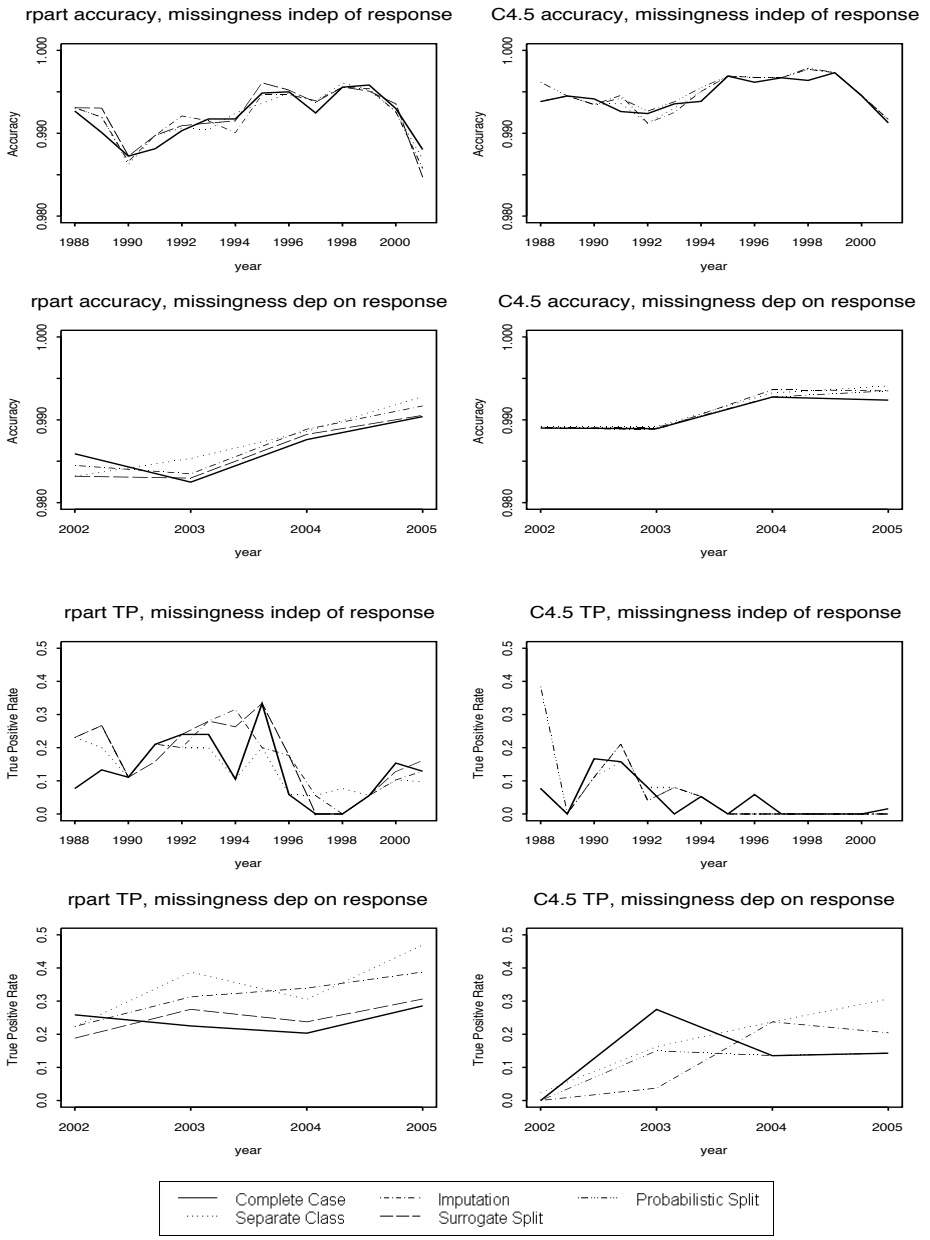


Figure 13: The relative performance of all of the missing data methods on the bankruptcy data. The left column gives methods using RPART (and includes all of the methods except for probabilistic split) and the right column gives methods using C4.5 (and includes all of the methods except for surrogate split). The top rows are performance in terms of accuracy while the bottom rows are in terms of true positive rate.

RPART while probabilistic split is realized using C4.5, we run all of the other methods using both RPART and C4.5 so that we can compare both surrogate split and probabilistic split with all of the other methods. In Figure 13, the plots on the left are the results from RPART, which include all of

the missing data methods except for probabilistic split. The plots on the right are the results from C4.5, which include all of the missing data methods except for surrogate split. The performances of methods common to both plots are slightly different because of differences between C4.5 and RPART in splitting and pruning rules. Both the accuracy and the true positive rates are shown. Since the number of actual bankruptcy cases in the data is small, the accuracy is always very high. The true positive rate is defined as

$$TP = \frac{\text{Number of correctly predicted bankruptcy cases}}{\text{Actual number of bankruptcy cases}}.$$

The graphs in the first and the second rows are for accuracies, with the first row for the first time period from 1988 to 2001 and the second row for the second time period from 2002 to 2005. The graphs in the third and the fourth rows are for true positive rates, with the third row for the first time period from 1988 to 2001 and the fourth row for the second time period from 2002 to 2005. It is apparent that in the first time period, there are no clear winners. However, in the second time period, separate class is a little better than the others, in line with expectations.

5. Extension To Logistic Regression

One obvious observation from this study is that when missing values occur in both the model building and model application stages, it should be considered as part of the data generating process rather than a separate mechanism. That is, taking the missingness into consideration can improve predictive performance, sometimes significantly. This should also apply to other supervised learning methodologies, non-parametric or parametric, when predictive performance is concerned. We present here the results from a real data analysis study involving logistic regression, similar to the one presented in Section 4.1. Missing values are generated the same way as in Section 4.1 and then logistic regression models (without variable selection) with different ways of handling missing data are applied to those data sets. Finally a tally is made on the relative performances of different missing data methods. Results measured in accuracy, bagged accuracy, AUC and bagged AUC are almost identical to each other; results in terms of accuracy are shown in Figure 14.

Included in the study are five ways of handling missing data: using only complete cases (complete case method), including a missingness dummy variable in the explanatory variable (dummy method, sometimes called the missing-indicator method),[1] building separate models for data with values missing and data without missing values (by-group method),[2] imputing missing values with grand mean/mode (imputation method), and only using predictors without missing values (complete variable method). Note that the methods using a dummy variable and building separate models for

1. If explanatory variable X_1 has missing values, then we create a missingness dummy variable M_1 that has value 1 if X_1 is observed and 0 otherwise. Then M_1 and $X_1 * M_1$ are both used as explanatory variables. The result of this set-up is that the effect of X_1 is fit on the observations with X_1 observed but a single mean value is fit to the observations with X_1 missing. All of the observations, with or without X_1 values, have the same coefficients for all of the other explanatory variables. Jones (1996) showed that this method can result in biased coefficient estimates in regression modeling, but did not address the question of predictive accuracy that is the focus here.
2. The biggest difference between the by-group method and the dummy method is whether the explanatory variables, other than the one containing missing values, have different coefficients or not. The by-group method fits two separate models to observations with and without missing values. Therefore, even if an explanatory variable is fully observed, its coefficient would most likely be different for fully observed observations and for observations with missing values. The dummy method, on the other hand, fits a single model to the entire data set so that variables that are fully observed will have the same coefficients whether an observation has missing values or not.

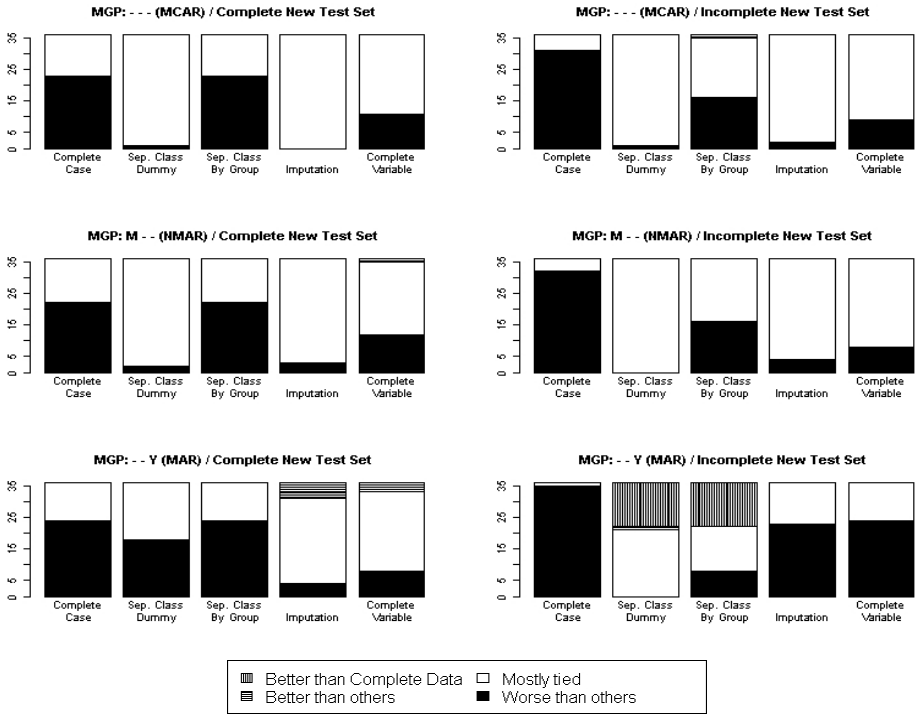


Figure 14: A tally of the relative out-of-sample performance with logistic regression measured in accuracy of all of the missing data methods on the 36 data sets.

observations with and without missing values each are analogous to the separate class method for trees. The most obvious observation is that when missingness is related to the response variable and missingness occurs in the test set, the dummy method and the by-group method dominate the other methods; in fact, more than a third of the time, they perform better than logistic regression on the original full data. Comparing Figure 14 with Figure 9, we see a clear similarity, in that the methods using a separate class model missingness directly, and thus use the information contained in the missingness about the response variable most efficiently. This suggests that the result that predictive performance of supervised learning methods is driven by the dependence (or lack of dependence) on the response variable is not limited to trees, but is rather a general phenomenon.

6. Conclusion And Future Study

The main conclusions from this study are as follows:

1. The two most important criteria that differentiate the performance of different missing data methods are whether or not the testing set is complete and whether or not the missingness depends on the response variable. There is strong evidence, both analytically and empirically, that separate class is the best missing data method to use when the testing data also contains missing values and the missingness is dependent upon the response variable.

   In practice, one way to detect the dependence of missingness on the response variable is to try building a model, with a classification tree being a natural choice, of the response variable on

the missingness indicators (which equals to 1 if the corresponding original value is missing and 0 otherwise). If such a model supports a relationship, then it is an indication that the missingness is related to the response variable.

2. The performance of classification trees is on average not too negatively affected by missing values, except for the complete case method and the mean imputation method, which are sensitive to different missing rates. Separate class tend to perform better when the missing rate is neither too high nor too low, trading off between information loss due to missing values and information gain from the informative MGP.

3. The original full data AUC has an impact on the performance of separate class method. The higher the original AUC, the more severe the information loss due to missing value, and thus relatively the worse the performance of the separate class method.

The consistency of these results across the theoretical analyses, simulations from the artificial data, and simulations based on real data provides strong support for their general validity.

The findings here also have implications beyond analysis of the data at hand. For example, since missingness that is dependent on the response variable can actually improve predictive performance, it is clear that expending time, effort, and money to recover the missing values is potentially a poor way to allocate resources. Another interesting implication of these results is related to data disclosure limitation. It is clear that any masking of values must be done in a way that is independent of the response variables of interest (or any predictors highly related to such variables), since otherwise data disclosure using regression-type methods (Palley and Simonoff, 1987) could actually increase.

Classification trees are designed for the situation where the response variable is categorical, not just binary; it would be interesting to see how these results carry over to that situation. Tree-based methodologies for the situation with a numerical response have also been developed (i.e., regression trees), and the problems of missing data occur in that context also. Investigating such trees would be a natural extension of this paper. In this paper, we focused on base form classification trees using C4.5 and RPART, although bagging was included in the study. It would be worthwhile to see how the performance of different missing data methods is affected by different tree features such as stopping and pruning or when techniques such as cross-validation, tree ensembles, etc. are used.

Moreover, as was shown in Section 5, the relationship between the missingness and the response variable can be helpful in prediction when missingness occurs in both the training data and the testing data in situations other than classification trees. This is very likely true for other supervised learning methods, and thus testing more learning methods would also be a natural extension to this study.

Acknowledgments

The authors would like to thank Foster Provost for his helpful comments, which have helped us to greatly improve the quality of this paper. The authors also thank several anonymous referees for helpful comments that greatly improved the presentation of results in the paper.

$P(Y=0\|X=0,$ with Missing Data)		$>T$	$>T$
$P(Y=0\|X=1,$ with Missing Data)		$>T$	$\leq T$
P(Y=0|X=0)	P(Y=0|X=1)		
$>T$	$>T$	1	$\frac{P(X=0,Y=0)+P(X=1,Y=1)}{P(Y=0)}$
$>T$	$\leq T$	$\frac{P(Y=0)}{P(X=0,Y=0)+P(X=1,Y=1)}$	1
$\leq T$	$>T$	$\frac{P(Y=0)}{P(X=0,Y=1)+P(X=1,Y=0)}$	$\frac{P(X=0,Y=0)+P(X=1,Y=1)}{P(X=0,Y=1)+P(X=1,Y=0)}$
$\leq T$	$\leq T$	$\frac{P(Y=0)}{P(Y=1)}$	$\frac{P(X=0,Y=0)+P(X=1,Y=1)}{P(Y=1)}$
$P(Y=0\|X=0,$ with Missing Data)		$\leq T$	$\leq T$
$P(Y=0\|X=1,$ with Missing Data)		$>T$	$\leq T$
P(Y=0|X=0)	P(Y=0|X=1)		
$>T$	$>T$	$\frac{P(X=0,Y=1)+P(X=1,Y=0)}{P(Y=0)}$	$\frac{P(Y=1)}{P(Y=0)}$
$>T$	$\leq T$	$\frac{P(X=0,Y=1)+P(X=1,Y=0)}{P(X=0,Y=0)+P(X=1,Y=1)}$	$\frac{P(Y=1)}{P(X=0,Y=0)+P(X=1,Y=1)}$
$\leq T$	$>T$	1	$\frac{P(Y=1)}{P(X=0,Y=1)+P(X=1,Y=0)}$
$\leq T$	$\leq T$	$\frac{P(X=0,Y=1)+P(X=1,Y=0)}{P(Y=1)}$	1

Table 3: *RelAcc* of tree built on data with missing values and tested on the original full data set when there is no variation from true DGP

Appendix A. Proofs of the Theorems

The relative accuracy (*RelAcc*) when there are missing values in the training set but not in the testing set can be summarized into Table 3, where T is the threshold value (an observation will be classified as class 0 if the predicted probability for it to be 0 is greater than T). The value of T reflects the misclassification cost. It is taken as 0.5 reflecting an equal misclassification cost. In Table 3, the columns show different rules given by the classification trees when there are missing values, and the rows show actual DGP's. The entries are the *RelAcc* values under different scenarios. For example, all of the entries on the diagonal are one's because the rules given by the classification trees when there are missing values are the same as the true DGP's and thus the accuracy achieved by the trees are the same with or without the missing values and thus $RelAcc = 1$. Cell $(1,2)$, for example, shows that if the true DGP is $P(Y=0|X=0) > T$ and $P(Y=0|X=1) > T$ but the classification tree gives rule $P(Y=0|X=0) > T$ and $P(Y=0|X=1) \leq T$ when there are missing values, that is, $P(Y=0|X=0,\text{with missing value}) > T$ and $P(Y=0|X=1\text{with missing value}) \leq T$, then the relative accuracy is determined to be

$$\frac{P(X=0,Y=0)+P(X=1,Y=1)}{P(Y=0)}.$$

Proof of Theorem 1 : The expected performance of the complete case method when the missingness does not depend on the response variable and the testing set is complete.

Proof

First, we define A as the case-wise missingness indicator which equals 1 if the observation contains missing values in one or more of the predictors or 0 if the observation does not

contain missing values in any of the predictors. Y is the response variable and $\underline{X}$ is the vector of the predictors.

If only the complete cases are used, if $P(Y|A=0,\underline{X}) = P(Y|\underline{X})$, then only the diagonal in Table 3 can be achieved, and thus there is no loss in accuracy.

This condition will be satisfied if and only if the MGP is conditionally independent of Y given $\underline{X}$, that is, $P(A=0|\underline{X},Y) = P(A=0|\underline{X})$.

1. "$P(Y|A=0,\underline{X}) = P(Y|\underline{X}) \Rightarrow P(A=0|\underline{X},Y) = P(A=0|\underline{X})$"

   $$\begin{aligned} \text{P(A=0}|\underline{\text{X}},\text{Y)} &= \frac{P(A=0,\underline{X},Y)}{P(\underline{X},Y)} \\ &= \frac{P(Y|A=0,\underline{X})P(A=0,\underline{X})}{P(\underline{X},Y)} \\ &= \frac{P(Y|\underline{X})P(A=0,\underline{X})}{P(Y|\underline{X})P(\underline{X})} \\ &= P(A=0|\underline{X}) \end{aligned}$$

2. "$P(Y|A=0,\underline{X}) = P(Y|\underline{X}) \Leftarrow P(A=0|\underline{X},Y) = P(A=0|\underline{X})$"

   $$\begin{aligned} \text{P(Y}|\text{A=0,}\underline{\text{X}}\text{)} &= \frac{P(A=0,\underline{X},Y)}{P(A=0,\underline{X})} \\ &= \frac{P(A=0|\underline{X},Y)P(\underline{X},Y)}{P(A=0,\underline{X})} \\ &= \frac{P(A=0|\underline{X})P(\underline{X},Y)}{P(A=0|\underline{X})P(\underline{X})} \\ &= \frac{P(\underline{X},Y)}{P(\underline{X})} \\ &= P(Y|\underline{X}) \end{aligned}$$

■

Proof of Theorems 2 and 3 : The expected performance of the complete case method when the missingness depends on the response variable and the testing set is complete.

We first observe the following lemmas.

Lemma 7 *For the partition defined by the tree built on the original full data (and not changed by missing values), let the k^{th} section contain P^k proportion of data and within the partition, the majority class have proportion P^k_{mj}. Note that $\sum_{k=1}^{K} P^k = 1$, while the full data set accuracy, that is, the accuracy achievable with the full data set, is $\sum_k P^k P^k_{mj}$.)*

The rule for the k^{th} section will be classifying it as the majority class of the section. The impact of missing data on its rule is to either leave it unchanged or make it classify the data as the minority class instead of the majority class.

The smallest missing rate needed in k^{th} section to change the rule is $P(A=1|k) = 2P^k_{mj} - 1$, where A is defined as in Theorem 1, that is, it is the case-wise indicator, which takes value 1 if the observation contains missing value or 0 otherwise. If the rule is changed the loss in accuracy within that section is $2P^k_{mj} - 1$.

Proof

We assume the partition of the data is not changed by the missing values. The structure of the trees need not to be the same because different trees may lead to the same partition of data.

For any k, to make the rule of the k^{th} section change, we need to observe more minority class cases than the majority ones within that section. To achieve this in the most efficient way, we only make the majority ones missing. Originally, there are P_{mj}^k majorities and $1-P_{mj}^k$ minorities. Only when there are $P_{mj}^k-(1-P_{mj}^k)=2P_{mj}^k-1$ majorities missing will it become less than the minorities, so this is the smallest missing rate we need to make the rule change.

After the rule is changed, only $1-P_{mj}^k$ of the data, that is, the minorities, will be correctly classified. Therefore, the loss in accuracy is $P_{mj}^k-(1-P_{mj}^k)=2P_{mj}^k-1$.

■

Lemma 8 *For a given data set and the partition defined by the tree built on the full data set (which is not changed by the missing values), the largest loss in accuracy is* $\sum_k 2P_{mj}^k-1$. *The smallest missing rate needed to achieve this is also* $\sum_k 2P_{mj}^k-1$.

Proof

The largest loss is achieved if and only if the rules are changed in every section of data in the partition. The result then follows from Lemma 7.

■

Lemma 9 *For a certain missing rate, say* P_m, *the largest effect it can have on the classification accuracy of any data that won't be split is* P_m *itself.*

In this case, the data set has its majority proportion $P_{mj}=\frac{1}{2}(1+P_m)$.

Proof

Similar to the proof of Lemma 7, for missing values to have an impact on the classification rule, it has to switch the order status of the majority and minority. To achieve this, it has to be that $P_{mj}-(1-P_{mj})\leq P_m$. We know that once the rule is changed, the loss in accuracy is $P_{mj}-(1-P_{mj})$. Therefore, the largest loss is P_m when the equality holds. In this case, we have $P_{mj}=\frac{1}{2}(1+P_m)$.

■

We now prove Theorem 2.

Proof

For any data set, once it is partitioned and the partition is not changed by missing values, the rules in different sections of data are independent of each other, so we can look at them separately.

Suppose the data are partitioned into K segments, in which some contain missing data and the others do not. Let K_0 be the set of sections whose rules are changed by missing data and K_1 be the set of all other sections. Also let the k^{th} segment ($k = 1 \ldots K$) contain proportion P^k of the data. We have $\sum_{k=1}^{K} P^k = 1$.

Assume that the k^{th} segment ($k \in K_0$) contains proportion P_m^k of missing data. Then we have

$$\sum_{k \in K_0} P_m^k P^k \leq P_m.$$

For the k^{th} segment ($k \in K_0$), by Lemma 9, the largest possible loss in accuracy is P_m^k and it occurs if and only if $P_{mj}^k = \frac{1}{2}(1 + P_m^k)$. Therefore, the possible loss for the entire data set is

$$\sum_{k \in K_0} P_m^k P^k \leq P_m,$$

the largest loss being achieved when the equality holds. In that case, the rules in all of the categories that contain missing values are changed and the maximum loss is P_m.

■

We now prove Theorem 3.

Proof

Assuming the partitions of data are not changed by the missing values, we have

$$\begin{aligned} RelAcc &= \frac{\sum_{k=1}^{K} P_{mj}^k P^k - \sum_{k \in K_0} (\text{loss in accuracy in } k^{th} \text{ segment})}{\sum_{k=1}^{K} P_{mj}^k P^k} \\ &= 1 - \frac{\sum_{k \in K_0} (\text{loss in accuracy in } k^{th} \text{ segment})}{\sum_{k=1}^{K} P_{mj}^k P^k} \\ &= 1 - \frac{\sum_{k \in K_0} (\text{loss in accuracy in } k^{th} \text{ segment})}{\sum_{k \in K_0} P_{mj}^k P^k + \sum_{k \in K_1} P_{mj}^k P^k} \end{aligned}$$

This is an increasing function of $\sum_{k \in K_1} P_{mj}^k P^k$ in the denominator, which is independent of other factors; setting it to zero minimize the *relative accuracy*, so

$$RelAcc \leq 1 - \frac{\sum_{k \in K_0} (\text{loss in accuracy in } k^{th} \text{ segment})}{\sum_{k \in K_0} P_{mj}^k P^k}$$

Denote the numerator $\sum_{k \in K_0} (\text{loss in accuracy in } k^{th} \text{ segment})$ as a. Now, from the proof of Theorem 2, the numerator $a \leq P_m$ and the denominator $\sum_{k \in K_0} P_{mj}^k P^k = \frac{1}{2}(1 + a)$. So,

$$RelAcc \leq 1 - \frac{a}{\frac{1}{2}(1 + a)}$$

This is a decreasing function of a and subject to $a \leq P_m$. Therefore, the minimum *RelAcc* is achieved when $a = P_m$. This gives

$$\begin{aligned} RelAcc &\leq 1 - \frac{P_m}{\frac{1}{2}(1+P_m)} \\ &= \frac{1-P_m}{1+P_m} \end{aligned}$$

■

Proof of Theorem 4 : Some properties of probabilistic split when the missingness does not depend on both the predictor and the response variable.

Proof

1. Part 1
   - If the MGP is independent of Y given X, that is, $P(M|X,Y) = P(M|X)$ then $P(Y|M,X) = P(Y|X)$ by the proof of Theorem 1.
     The rules given by probabilistic split when there are missing values are as follows:

$$\begin{aligned} & P(Y=0|X=0, Prob\_split) \\ = \; & P(Y=0|M=0,X=0)P(M=0) + P(Y=0|M=1)P(M=1) \\ = \; & P(Y=0|X=0)P(M=0) + P(Y=0|M=1)P(M=1) \\ = \; & P(Y=0|X=0)P(M=0) \\ & +[P(Y=0,X=0|M=1) + P(Y=0,X=1|M=1)]P(M=1) \\ = \; & P(Y=0|X=0)P(M=0) \\ & +[P(Y=0|M=1,X=0)P(X=0|M=1) \\ & +P(Y=0|M=1,X=1)P(X=1|M=1)]P(M=1) \\ = \; & P(Y=0|X=0)P(M=0) + [P(Y=0|X=0)P(X=0|M=1) \\ & +P(Y=0|X=1)P(X=1|M=1)]P(M=1) \\ = \; & P(Y=0|X=0)P(M=0) + P(Y=0|X=0)P(M=1,X=0) \\ & +P(Y=0|X=1)P(M=1,X=1) \\ = \; & P(Y=0|X=0)[P(M=0) + P(M=1,X=0)] \\ & +P(Y=0|X=1)P(M=1,X=1) \end{aligned}$$

     and following the similar route, we can get

$$\begin{aligned} & P(Y=0|X=1, Prob\_split) \\ = \; & P(Y=0|X=1)[P(M=0) + P(M=1,X=1)] \\ & +P(Y=0|X=0)P(M=1,X=0). \end{aligned}$$

     Note that

$$P(M=0) + P(M=1,X=1) + P(M=1,X=0) = 1.$$

Therefore, both $P(Y=0|X=0,Prob\_split)$ and$P(Y=0|X=1,Prob_split)$ are weighted averages of $P(Y=0|X=0)$ and $P(Y=0|X=1)$.
It follows that if both $P(Y=0|X=0)$ and $P(Y=0|X=1)$ are greater (less) than 0.5, then both $P(Y=0|X=0,Prob\_split)$ and $P(Y=0|X=1,Prob\_split)$ are also greater (less) than 0.5.

- If the MGP is independent of X given Y, without loss of generality, we prove the case when $P(Y=0|X=0) > T = 0.5$ and $P(Y=0|X=1) > T = 0.5$.

$$
\begin{aligned}
& P(Y=0|X=0,Prob\_split) \\
= \; & \frac{P(M=0,X=0,Y=0)}{P(M=0,X=0)}P(M=0)+P(Y=0|M=1)P(M=1) \\
= \; & \frac{P(M=0|X=0,Y=0)P(X=0,Y=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1,Y=0) \\
= \; & \frac{P(M=0|Y=0)P(X=0,Y=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1,X=0,Y=0)+P(M=1,X=1,Y=0) \\
= \; & \frac{P(M=0|Y=0)P(Y=0|X=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1,X=0|Y=0)P(Y=0) \\
& +P(M=1,X=1|Y=0)P(Y=0) \\
= \; & \frac{P(M=0|Y=0)P(Y=0|X=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1|Y=0)P(X=0|Y=0)P(Y=0) \\
& +P(M=1|Y=0)P(X=1|Y=0)P(Y=0) \\
= \; & \frac{P(M=0|Y=0)P(Y=0|X=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1|Y=0)P(Y=0|X=0)P(X=0) \\
& +P(M=1|Y=0)P(Y=0|X=1)P(X=1) \\
> \; & T[\frac{P(M=0|Y=0)P(M=0)}{P(M=0,X=0)} \\
& +P(M=1|Y=0)P(X=0)+P(M=1|Y=0)P(X=1)] \\
= \; & T[\frac{P(M=0|Y=0)P(M=0)}{P(M=0,X=0)}+P(M=1|Y=0)] \\
> \; & T\,(P(M=0|Y=0)+P(M=1|Y=0)) \\
= \; & T
\end{aligned}
$$

A similar argument gives $P(Y=0|X=1,Prob\_split) > T$.

2. Part 2

- If the MGP is independent of Y given X, then from the proof of part 1,

$$P(Y=0|X=0,Prob\_split)$$

$$\begin{aligned} &= P(Y=0|X=0)(P(M=0)+P(M=1,X=0)) \\ &\quad +P(Y=0|X=1)P(M=1,X=1) \end{aligned}$$

and

$$\begin{aligned} &P(Y=0|X=1,Prob\_split) \\ &= P(Y=0|X=1)(P(M=0)+P(M=1,X=1)) \\ &\quad +P(Y=0|X=0)P(M=1,X=0). \end{aligned}$$

Taking the difference, we get

$$\begin{aligned} &P(Y=0|X=0,Prob\_split)-P(Y=0|X=1,Prob\_split) \\ &= P(Y=0|X=0)(P(M=0)+P(M=1,X=0)) \\ &\quad +P(Y=0|X=1)P(M=1,X=1) \\ &\quad -[P(Y=0|X=1)(P(M=0)+P(M=1,X=1)) \\ &\quad +P(Y=0|X=0)P(M=1,X=0)] \\ &= P(Y=0|X=0)P(M=0)-P(Y=0|X=1)P(M=0) \\ &= (P(Y=0|X=0)-P(Y=0|X=1))P(M=0). \end{aligned}$$

Without loss of generality, assume $P(Y=0|X=0,Prob\_split) > T$ and $P(Y=0|X=1,Prob\_split) < T$. It then follows that $P(Y=0|X=0) > P(Y=0|X=1)$. There are three possibilities:

(a) $P(Y=0|X=0) > T > P(Y=0|X=1)$

(b) $T > P(Y=0|X=0) > P(Y=0|X=1)$

(c) $P(Y=0|X=0) > P(Y=0|X=1) > T$

Conditions (b) and (c) are not possible because in these two cases, X is actually not informative and by Part 1, probabilistic split will show they are not informative. Therefore, it holds that $P(Y=0|X=0) > T > P(Y=0|X=1)$.

- If the MGP is independent of X given Y, that is, $P(M|X,Y) = P(M|Y)$, we have

$$\begin{aligned} &P(Y=0|X=0,Prob\_split) \\ &= \frac{P(M=0,X=0,Y=0)}{P(M=0,X=0)}P(M=0)+P(Y=0|M=1)P(M=1) \\ &= \frac{P(M=0|X=0,Y=0)P(X=0,Y=0)P(M=0)}{P(M=0|X=0,Y=0)P(X=0,Y=0)+P(M=0|X=0,Y=1)P(X=0,Y=1)} \\ &\quad +P(Y=0|M=1)P(M=1) \\ &= \frac{P(M=0|Y=0)P(X=0,Y=0)P(M=0)}{P(M=0|Y=0)P(X=0,Y=0)+P(M=0|Y=1)P(X=0,Y=1)} \\ &\quad +P(Y=0|M=1)P(M=1) \\ &= \frac{P(M=0|Y=0)P(Y=0|X=0)P(M=0)}{P(M=0|Y=0)P(Y=0|X=0)+P(M=0|Y=1)P(Y=1|X=0)} \\ &\quad +P(Y=0|M=1)P(M=1), \end{aligned}$$

and following the same route, we have

$$\begin{aligned} &P(Y=0|X=1,Prob\_split) \\ &= \frac{P(M=0|Y=0)P(Y=0|X=1)P(M=0)}{P(M=0|Y=0)P(Y=0|X=1)+P(M=0|Y=1)P(Y=1|X=1)} \\ &\quad +P(Y=0|M=1)P(M=1). \end{aligned}$$

Therefore,

$$\begin{aligned}
& P(Y=0|X=0,Prob_split) - P(Y=0|X=1,Prob_split) \\
= & \frac{P(M=0|Y=0)P(Y=0|X=0)P(M=0)}{P(M=0|Y=0)P(Y=0|X=0)+P(M=0|Y=1)P(Y=1|X=0)} \\
& - \frac{P(M=0|Y=0)P(Y=0|X=1)P(M=0)}{P(M=0|Y=0)P(Y=0|X=1)+P(M=0|Y=1)P(Y=1|X=1)} \\
= & [P(Y=0|X=0)P(M=0|Y=0)P(Y=0|X=1) \\
& +P(Y=0|X=0)P(M=0|Y=1)P(Y=1|X=1) \\
& -P(Y=0|X=1)P(M=0|Y=0)P(Y=0|X=0) \\
& -P(Y=0|X=1)P(M=0|Y=1)P(Y=1|X=0)]\frac{P(M=0|Y=0)P(M=0)}{D_1D_2} \\
= & [P(Y=0|X=0)-P(Y=0|X=1)]\frac{P(M=0|Y=1)P(M=0|Y=0)P(M=0)}{D_1D_2} \\
= & [P(Y=0|X=0)-P(Y=0|X=1)]K
\end{aligned}$$

where

$$D_1 = P(M=0|Y=0)P(Y=0|X=0)+P(M=0|Y=1)P(Y=1|X=0),$$
$$D_2 = P(M=0|Y=0)P(Y=0|X=1)+P(M=0|Y=1)P(Y=1|X=1)$$

and

$$K = \frac{P(M=0|Y=1)P(M=0|Y=0)P(M=0)}{D_1D_2}.$$

Since K is always positive as long as there are different Y values observed, we can see that the probabilistic split preserves the order of the conditional probability of Y given X.

Now, without loss of generality, assume $P(Y=0|X=0,Prob\_split) > T$ and $P(Y=0|X=1,Prob\_split) < T$. It follows that $P(Y=0|X=0) > P(Y=0|X=1)$ because probabilistic split preserves the correct order. There are three possibilities:

(a) $P(Y=0|X=0) > T > P(Y=0|X=1)$

(b) $T > P(Y=0|X=0) > P(Y=0|X=1)$

(c) $P(Y=0|X=0) > P(Y=0|X=1) > T$

Conditions (b) and (c) are not possible because in these two cases, X is actually not informative and by the earlier result in Part 1, probabilistic split will show they are not informative. Therefore, it holds that $P(Y=0|X=0) > T > P(Y=0|X=1)$.

3. Part 3

The results of Part 1 and Part 2 lead to the simplification of Table 3 into Table 4.

Without loss of generality, we provide the proof only for the case when $P(Y=0|X=0) > T$ and $P(Y=0|X=1) \leq T$ but $P(Y=0|X=0,prob\_split) > T$ and $P(Y=0|X=1,prob\_split) > T$, where *RelAcc* is

$$RelAcc = \frac{P(Y=0)}{P(X=0,Y=0)+P(X=1,Y=1)}.$$

It suffices to show that $P(Y=0) > 0.5$

- If M is independent of Y given X,

$$\begin{aligned}
& P(Y=0) \\
= & P(X=0,Y=0)+P(X=1,Y=0)
\end{aligned}$$

Simplified possibilities	$>T$ $>T$	$>T$ $\leq T$	$\leq T$ $>T$	$\leq T$ $\leq T$
Full data				
$>T$ $>T$	1	—	—	—
$>T$ $\leq T$	$\frac{P(Y=0)}{P(X=0,Y=0)+P(X=1,Y=1)}$	1	—	$\frac{P(Y=1)}{P(X=0,Y=0)+P(X=1,Y=1)}$
$\leq T$ $>T$	$\frac{P(Y=0)}{P(X=0,Y=1)+P(X=1,Y=0)}$	—	1	$\frac{P(Y=1)}{P(X=0,Y=1)+P(X=1,Y=0)}$
$\leq T$ $\leq T$	—	—	—	1

Table 4: *RelAcc* with a 2×2 table of probabilistic split when the missingness is independent of either X or Y or both

$$
\begin{aligned}
= \;& P(M=0,X=0,Y=0)+P(M=1,X=0,Y=0) \\
& +P(Y=0|X=1)P(X=1) \\
= \;& P(Y=0|M=0,X=0)P(M=0,X=0) \\
& +P(Y=0|M=1,X=0)P(M=1,X=0)+P(Y=0|X=1)P(X=1) \\
=^{1} \;& P(Y=0|X=0)P(M=0,X=0)+P(Y=0|X=0)P(M=1,X=0) \\
& +P(Y=0|X=1)P(X=1) \\
>^{2} \;& P(Y=0|X=0)P(M=1,X=0)+P(Y=0|X=1)P(M=0,X=0) \\
& +P(Y=0|X=1)P(X=1) \\
= \;& P(Y=0|X=1)(P(M=0)+P(M=1,X=1)) \\
& +P(Y=0|X=0)P(M=1,X=0) \\
= \;& P(Y=0|X=1,prob\_split) \\
> \;& 0.5
\end{aligned}
$$

where 1 follows because $P(Y|M,X)=P(Y|X)$ and 2 follows because $P(Y=0|X=0)>T\geq P(Y=0|X=1)$. Therefore,

$$\frac{P(Y=0)}{P(X=0,Y=0)+P(X=1,Y=1)}>P(Y=0)>0.5$$

- If M is independent of X given Y,

$$P(Y=0)=P(M=0,Y=0)+P(M=1,Y=0)$$

and by assumption,

$$
\begin{aligned}
& P(Y=0|X=1,prob\_split) \\
= \;& P(Y=0|M=0,X=1)P(M=0)+P(M=1,Y=0) \\
> \;& 0.5
\end{aligned}
$$

If $P(M=0,Y=0)>P(Y=0|M=0,X=1)P(M=0)$, then $P(Y=0)>P(Y=0|X=1,prob\_split)>0.5$, it suffices to show

$$P(M=0,Y=0)>P(Y=0|M=0,X=1)P(M=0).$$

By the earlier results in Part 2, probabilistic split preserves the order of conditional probabilities of Y given X when the missingness is conditionally independent of X given Y, that is, in this case, since

$$P(Y=0|X=0) > T \geq P(Y=0|X=1),$$

we have

$$\begin{aligned}
& P(Y=0|X=0,Prob_split) - P(Y=0|X=1,Prob_split) \\
= \; & P(Y=0|M=0,X=0)P(M=0) + P(Y=0|M=1)P(M=1) \\
& -(P(Y=0|M=0,X=1)P(M=0) + P(Y=0|M=1)P(M=1)) \\
= \; & (P(Y=0|M=0,X=0) - P(Y=0|M=0,X=1))P(M=0) \\
> \; & 0.
\end{aligned}$$

That is, $P(Y=0|M=0,X=0) > P(Y=0|M=0,X=1)$. We then have

$$\begin{aligned}
& P(Y=0|M=0,X=0) > P(Y=0|M=0,X=1) \\
\Rightarrow \; & P(Y=0|M=0,X=0)P(M=0,X=0) \\
& > P(Y=0|M=0,X=1)P(M=0,X=0) \\
\Rightarrow \; & P(M=0,X=0,Y=0) > P(Y=0|M=0,X=1)P(M=0,X=0) \\
\Rightarrow \; & P(M=0,X=0,Y=0) + P(M=0,X=1,Y=0) \\
& > P(Y=0|M=0,X=1)P(M=0,X=0) + P(M=0,X=1,Y=0) \\
\Rightarrow \; & P(M=0,Y=0) \\
& > P(Y=0|M=0,X=1)P(M=0,X=0) + P(Y=0|M=0,X=1)P(M=0,X=1) \\
\Rightarrow \; & P(M=0,Y=0) > P(Y=0|M=0,X=1)(P(M=0,X=0) + P(M=0,X=1) \\
\Rightarrow \; & P(M=0,Y=0) > P(Y=0|M=0,X=1)P(M=0)
\end{aligned}$$

■

Proof of Theorem 5 : Some properties of the mode imputation when the missingness does not depend on the response variable.

Proof

Without loss of generality, we assume that $P(X=0|M=0) > P(X=1|M=0)$, that is, there are more X=0 cases observed than X=1 ones. As a result, all of the missing X values will be labeled as X=0, the observed mode. Then the decision rules when the mode imputation is used can be written as

$$\begin{aligned}
& P(Y=0|X=0,Imp) \\
= \; & \frac{P(M=0,X=0,Y=0) + P(M=1,Y=0)}{P(M=0,X=0) + P(M=1)} \\
= \; & \frac{P(M=0,X=0,Y=0) + P(M=1,X=0,Y=0) + P(M=1,X=1,Y=0)}{P(M=0,X=0) + P(M=1)} \\
= \; & \frac{P(X=0,Y=0) + P(M=1,X=1,Y=0)}{P(X=0) + P(M=1,X=1)} \\
= \; & \frac{P(X=0,Y=0) + P(Y=0|M=1,X=1)P(M=1,X=1)}{P(X=0) + P(M=1,X=1)}
\end{aligned}$$

$$
\begin{aligned}
= & \frac{P(Y=0|X=0)P(X=0)+P(Y=0|X=1)P(M=1,X=1)}{P(X=0)+P(M=1,X=1)} \\
& P(Y=0|X=1,Imp) \\
= & P(Y=0|M=0,X=1) \\
= & P(Y=0|X=1)
\end{aligned}
$$

1. Note that $P(Y=0|X=0,Imp)$ is a weighted average of $P(Y=0|X=0)$ and $P(Y=0|X=1)$. Therefore, if they are both larger (or smaller) than 0.5, $P(Y=0|X=0,Imp)$ will also be, and thus it gives the same rule as $P(Y=0|X=0)$. Moreover, $P(Y=0|X=1,Imp)=P(Y=0|X=1)$, so it also gives the correct rule.
2. Suppose

$$
\begin{aligned}
P(Y=0|X=0,Imp) &> 0.5 \\
P(Y=0|X=1,Imp) &< 0.5,
\end{aligned}
$$

   then $P(Y=0|X=1)=P(Y=0|X=1,Imp)<0.5$, which is always correct. Moreover, note that $P(Y=0|X=0,Imp)$ is a weighted average of $P(Y=0|X=0)$ and $P(Y=0|X=1)$. Since $P(Y=0|X=0,Imp)>0.5$ and $P(Y=0|X=1)<0.5$, we must have $P(Y=0|X=0)>0.5$. Therefore, $P(Y=0|X=0,Imp)$ gives the correct rule.
3. Again the possibilities simplify to Table 4. Without loss of generality, we prove the situation when both $P(Y=0|X=0,Imp)$ and $P(Y=0|X=1,Imp)$ are greater than 0.5, that is

$$
\begin{aligned}
& P(Y=0|X=0,Imp) \\
= & \frac{P(Y=0|X=0)P(X=0)+P(Y=0|X=1)P(M=1,X=1)}{P(X=0)+P(M=1,X=1)} \\
> & 0.5 \\
& P(Y=0|X=1,Imp) \\
= & P(Y=0|X=1) \\
> & 0.5
\end{aligned}
$$

   Under the assumption that $P(X=0|M=0)>P(X=1|M=0)$, the missing values have an effect only if $P(Y=0|X=0)<0.5$ and $P(Y=0|X=1)>0.5$. In this case, the relative accuracy is $\frac{P(Y=0)}{P(X=0,Y=1)+P(X=1,Y=0)}$. This is the cell of the 3^{rd} row and the 1^{st} column in Table 4.
   But,

$$
\begin{aligned}
& \frac{P(Y=0)}{P(X=0,Y=1)+P(X=1,Y=0)} \\
> & P(Y=0) \\
= & P(X=0,Y=0)+P(X=1,Y=0) \\
>^{1} & 0.5(P(X=0)+P(M=1,X=1))-P(Y=0|X=1)P(M=1,X=1) \\
& +P(X=1,Y=0) \\
= & 0.5(P(X=0)+P(M=1,X=1))+P(Y=0|X=1)(P(X=1)-P(M=1,X=1)) \\
= & 0.5(1-P(M=0,X=1))+P(Y=0|X=1)P(M=0,X=1) \\
= & 0.5-0.5P(M=0,X=1))+P(Y=0|X=1)P(M=0,X=1) \\
>^{2} & 0.5-0.5P(M=0,X=1))+0.5P(M=0,X=1) \\
> & 0.5,
\end{aligned}
$$

where 1 follows because

$$\begin{aligned} & P(Y=0|X=0,Imp) \\ = & \frac{P(Y=0|X=0)P(X=0)+P(Y=0|X=1)P(M=1,X=1)}{P(X=0)+P(M=1,X=1)} \\ > & 0.5. \end{aligned}$$

By rearranging terms,

$$\begin{aligned} & P(Y=0|X=0)P(X=0) \\ = & P(X=0,Y=0) \\ > & 0.5(P(X=0)+P(M=1,X=1))-P(Y=0|X=1)P(M=1,X=1), \end{aligned}$$

where 2 follows because P(Y=0|X=1)=P(Y=0|X=1,Imp)>0.5.

■

Proof of Theorem 6 : The dominance of the separate class method when there are missing values in both the training set and the testing set and the missingness depends on the response variable.

Proof

When there are missing data in X in both the training set and the testing set, the finest partition of the data will be $X=0$, $X=1$ and X is missing. The best rule we can derive is to classify the majority class in each of these three partitions. This is achieved by using the separate class method.

■

References

P.D. Allison. Missing data. In *Sage University Papers Series on Quantitative Applications in the Social Sciences*, 07-136. Sage, Thousand Oaks, CA, 2001.

E.I. Altman and G. Sabato. Effects of the new basel capital accord on bank capital requirements for smes. *Journal of Financial Services Research*, 28(1):15–42, 2005. URL `http://econpapers.repec.org/RePEc:kap:jfsres:v:28:y:2005:i:1:p:15-42`.

G.E.A.P.A. Batista and M.C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17:519–533, 2003.

L. Breiman, J. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, Boca Raton, Fla, 1998.

Y. Ding and J.S. Simonoff. An investigation of missing data methods for classification trees applied to binary response data. Working paper 2008-SOR-1, Stern School of Business, New York University, 2008.

R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-Interscience, 2001.

A. Feelders. Handling missing data in trees: Surrogate splits or statistical imputation? *Principles of Data Mining and Knowledge Discovery*, 1704:329–334, 1999.

Y. Fujikawa and T.B. Ho. Cluster-based algorithms for filling missing values. *Lecture Notes in Artificial Intelligence*, 2336:549–554, 2002. In *6th Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, Taiwan, 6-9 May.

D.J. Hand. *Construction and Assessment of Classification Rules*. John Wiley and Sons, Chichester, 1997.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, pages 246–249. Springer Series in Statistics, 2001.

M.P. Jones. Indicator and stratification methods for missing explanatory variables in multiple linear regression. *Journal of the American Statistical Association*, 91:222–230, 1996.

A. Kalousis and M. Hilario. Supervised knowledge discovery from incomplete data. Cambridge, UK, 2000. Proceedings of the 2nd International Conference on Data Mining 2000, WIT Press.

H. Kim and S. Yates. Missing value algorithms in decision trees. In H. Bozdogan, editor, *Statistical Data Mining and Knowledge Discovery*, pages 155–172. Chapman & Hall/CRC, Boca Raton, Fla, 2003.

R.J.A. Little and D.B. Rubin. *Statistical Analysis with Missing Data*. Wiley, New York, second edition, 2002.

W.Z. Liu, A.P. White, S.G. Thompson, and M.A. Bramer. Techniques for dealing with missing values in classification. *Lecture Notes in Computer Science*, 1280:527–536, 1997.

M.A. Palley and J.S. Simonoff. The use of regression methodology for the compromise of confidential information in statistical databases. *ACM Transactions on Database Systems*, 12:593–608, 1987.

C. Perlich, F. Provost, and J.S. Simonoff. Tree induction vs. logistic regression: A learning curve analysis. *Journal of Machine Learning Research*, 4:211–255, 2003.

J.R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publishers, San Francisco, CA, 1993.

D.B. Rubin. Inference and missing data. *Biometrika*, 63:581–592, 1976.

M. Saar-Tsechansky and F. Provost. Handling missing values when applying classification models. *Journal of Machine Learning Research*, 8:1625–1657, 2007.

T.M. Therneau and E.J. Atkinson. An introduction to recursive partitioning using the rpart routines. Technical report, Mayo Foundation, 1997.

B. Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Applied Artificial Intelligence*, 23:373–405, 2009.

B. Twala, M.C. Jones, and D.J. Hand. Good methods for coping with missing data in decision trees. *Pattern Recognition Letters*, 29:950–956, 2008.

J. Wang and X. Shen. Large margin semi-supervised learning. *Journal of Machine Learning Research*, 8:1867–1891, 2007.

S. Zhang, Z. Qin, C.X. Ling, and S. Sheng. Missing is useful: Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 17:1689–1693, 2005.

Journal of Machine Learning Research 11 (2010) 171-234 Submitted 7/07; Revised 2/08; Published 1/10

Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation

Constantin F. Aliferis CONSTANTIN.ALIFERIS@NYUMC.ORG
Center of Health Informatics and Bioinformatics
Department of Pathology
New York University
New York, NY 10016, USA

Alexander Statnikov ALEXANDER.STATNIKOV@MED.NYU.EDU
Center of Health Informatics and Bioinformatics
Department of Medicine
New York University
New York, NY 10016, USA

Ioannis Tsamardinos TSAMARD@ICS.FORTH.GR
Computer Science Department, University of Crete
Institute of Computer Science, Foundation for Research and Technology, Hellas
Heraklion, Crete, GR-714 09, Greece

Subramani Mani SUBRAMANI.MANI@VANDERBILT.EDU
Discovery Systems Laboratory
Department of Biomedical Informatics
Vanderbilt University
Nashville, TN 37232, USA

Xenofon D. Koutsoukos XENOFON.KOUTSOUKOS@VANDERBILT.EDU
Department of Electrical Engineering and Computer Science
Vanderbilt University
Nashville, TN 37212, USA

Editor: Marina Meila

Abstract

We present an algorithmic framework for learning local causal structure around target variables of interest in the form of direct causes/effects and Markov blankets applicable to very large data sets with relatively small samples. The selected feature sets can be used for causal discovery and classification. The framework (*Generalized Local Learning*, or GLL) can be instantiated in numerous ways, giving rise to both existing state-of-the-art as well as novel algorithms. The resulting algorithms are sound under well-defined sufficient conditions. In a first set of experiments we evaluate several algorithms derived from this framework in terms of predictivity and feature set parsimony and compare to other local causal discovery methods and to state-of-the-art non-causal feature selection methods using real data. A second set of experimental evaluations compares the algorithms in terms of ability to induce local causal neighborhoods using simulated and resimulated data and examines the relation of predictivity with causal induction performance.

Our experiments demonstrate, consistently with causal feature selection theory, that local causal feature selection methods (under broad assumptions encompassing appropriate family of distribu-

tions, types of classifiers, and loss functions) exhibit strong feature set parsimony, high predictivity and local causal interpretability. Although non-causal feature selection methods are often used in practice to shed light on causal relationships, we find that they cannot be interpreted causally even when they achieve excellent predictivity. Therefore we conclude that only local causal techniques should be used when insight into causal structure is sought.

In a companion paper we examine in depth the behavior of GLL algorithms, provide extensions, and show how local techniques can be used for scalable and accurate global causal graph learning.

Keywords: local causal discovery, Markov blanket induction, feature selection, classification, causal structure learning, learning of Bayesian networks

1. Introduction

This paper addresses the problem of how to learn local causal structure around a target variable of interest using observational data. We focus on two specific types of local discovery: (a) identification of variables that are direct causes or direct effects of the target, and (b) discovery of Markov blankets. A Markov Blanket of a variable T is a minimal variable subset conditioned on which all other variables are probabilistically independent of T.

Discovery of local causal relationships is significant because it plays a central role in causal discovery and classification, because of its scalability benefits, and because by naturally bridging causation with predictivity, it provides significant benefits in feature selection for classification. More specifically, solving the local causal induction problem helps understanding how natural and artificial systems work; it helps identify what interventions to pursue in order for these systems to exhibit desired behaviors; under certain assumptions, it provides minimal feature sets required for classification of a chosen response variable with maximum predictivity; and finally local causal discovery can form the basis of efficient algorithms for learning the global causal structure of all variables in the data.

The paper is organized as follows: Section 2 provides necessary background material. The section summarizes related prior work in feature selection and causal discovery; reviews recent results that connect causality with predictivity; explains the central role of local causal discovery for achieving scalable global causal induction; reviews prior methods for local causal and Markov blanket discovery and published applications; finally it introduces the open problems that are the focus of the present report. Section 3 provides formal concepts and definitions used in the paper. Section 4 provides a general algorithmic framework, *Generalized Local Learning (GLL)*, which can be instantiated in many different ways yielding sound algorithms for local causal discovery and feature selection. Section 5 evaluates a multitude of algorithmic instantiations and parameterizations from GLL and compares them to state-of-the-art local causal discovery and feature selection methods in terms of classification performance, feature set parsimony, and execution time in many real data sets. Section 6 evaluates and compares new and state-of-the-art algorithms in terms of ability to induce correct local neighborhoods using simulated data from known networks and resimulated data from real-life data sets. Section 7 discusses the experimental findings and their significance.

The experiments presented here support the conclusion that local structural learning in the form of Markov blanket and local neighborhood induction is a theoretically well-motivated and empirically robust learning framework that can serve as a powerful tool for data analysis geared toward classification and causal discovery. At the same time several existing open problems offer possibilities for non-trivial theoretical and practical discoveries making it an exciting field of research. A companion paper (part II of the present work) studies the GLL algorithm properties

empirically and theoretically, introduces algorithmic extensions, and connects local to global causal graph learning (Aliferis et al., 2010). An online supplement to the present work is available at `http://www.nyuinformatics.org/downloads/supplements/JMLR2009/index.html`. In addition to supplementary tables and figures, the supplement provides all software and data needed to reproduce the analyses of the present paper.

2. Background

In the present section we provide a brief review of feature selection and causal discovery research, summarize theoretical results motivating this work, present methods to speed-up scalability of discovery, give desiderata for local algorithms, review prior methods for Markov blanket and local neighborhood induction, and finally discuss open problems and focus of this paper.

2.1 Brief Review of Feature Selection and Causal Discovery Research

Variable selection for predictive modeling (also called feature selection) has received considerable attention during the last three decades both in statistics and in machine learning (Guyon and Elisseeff, 2003; Kohavi and John, 1997). Intuitively, variable selection for prediction aims to select only a subset of variables for constructing a diagnostic or predictive model for a given classification or regression task. The reasons to perform variable selection include (a) improving the model predictivity and addressing the curse-of-dimensionality, (b) reducing the cost of observing, storing, and using the predictive variables, and finally, (c) gaining an understanding of the underlying process that generates the data. The problem of variable selection is more pressing than ever, due to the recent emergence of extremely large data sets, sometimes involving tens to hundreds of thousands of variables and exhibiting a very small sample-to-variable ratio. Such data sets are common in gene expression array studies, proteomics, computational biology, text categorization, information retrieval, image classification, business data analytics, consumer profile analysis, temporal modeling, and other domains and data-mining applications.

There are many different ways to define the variable selection problem depending on the needs of the analysis. Often however, the feature selection problem for classification/prediction is defined as identifying the minimum-size subset of variables that exhibit the maximal predictive performance (Guyon and Elisseeff, 2003). Variable selection methods can be broadly categorized into *wrappers* (i.e., heuristic search in the space of all possible variable subsets using a classifier of choice to assess each subset's predictive information), or *filters* (i.e., not using the classifier per se to select features, but instead applying statistical criteria to first select features and then build the classifier with the best features). In addition, there exist learners that perform *embedded variable selection*, that is, that attempt to simultaneously maximize classification performance while minimizing the number of variables used. For example, shrinkage regression methods introduce a bias into the parameter estimation regression procedure that imposes a penalty on the size of the parameters. The parameters that are close to zero are essentially filtered-out from the predictive model.

A variety of embedded variable selection methods have been recently introduced. These methods are linked to a statement of the classification or regression problem as an optimization problem with specified loss and penalty functions. These techniques usually fall into a few broad classes: One class of methods uses the $\mathcal{L}^2$-norm penalty (also known as ridge penalty), for example, the recursive feature elimination (RFE) method is based on the $\mathcal{L}^2$-norm formulation of SVM classification problem (Rakotomamonjy, 2003; Guyon et al., 2002). Other methods are based on the $\mathcal{L}^1$-norm

penalty (also known as lasso penalty), for example, feature selection via solution of the $\mathcal{L}^1$-norm formulation of SVM classification problem (Zhu et al., 2004; Fung and Mangasarian, 2004) and penalized least squares with lasso penalty on the regression coefficients (Tibshirani, 1996). A third set of methods is based on convex combinations of the $\mathcal{L}^1$- and $\mathcal{L}^2$-norm penalties, for example, feature selection using the doubly SVM formulation (Wang et al., 2006) and penalized least squares with elastic net penalty (Zou and Hastie, 2005). A fourth set uses the $\mathcal{L}^0$-norm penalty, for example, feature selection via approximate solution of the $\mathcal{L}^0$-norm formulation of SVM classification problem (Weston et al., 2003). Finally other methods use other penalties, for example, smoothly clipped absolute deviation penalty (Fan and Li, 2001).

Despite the recent emphasis on mathematically sophisticated methods such as the ones mentioned, the majority of feature selection methods in the literature and in practice are heuristic in nature in the sense that in most cases it is unknown what consists an optimal feature selection solution *independently of the class of models fitted*, and under which conditions an algorithm will output such an optimal solution.

Typical variable selection approaches also include forward, backward, forward-backward, local and stochastic search wrappers (Guyon and Elisseeff, 2003; Kohavi and John, 1997; Caruana and Freitag, 1994). The most common family of filter algorithms ranks the variables according to a score and then selects for inclusion the top k variables (Guyon and Elisseeff, 2003). The score of each variable is often the univariate (pairwise) association with the outcome variable T for different measures of associations such as the signal-to-noise ratio, the G^2 statistic and others. Information-theoretic (estimated mutual information) scores and multivariate scores, such as the weights received by a Support Vector Machine, have also been suggested (Guyon and Elisseeff, 2003; Guyon et al., 2002). Excellent recent reviews of feature selection can be found in Guyon et al. (2006a), Guyon and Elisseeff (2003) and Liu and Motoda (1998).

An emerging successful but also principled filtering approach in variable selection, and the one largely followed in this paper, is based on identifying the Markov blanket of the response ("target") variable T. The Markov blanket of T (denoted as $MB(T)$) is defined as a minimal set conditioned on which all other *measured* variables become independent of T (more details in Section 3).

While classification is often useful for *recognizing or predicting the behavior* of a system, in many problem-solving activities one needs to *change the behavior* of the system (i.e., to "manipulate it"). In such cases, knowledge of the causal relations among the various parts of the system is necessary. Indeed, in order to design new drugs and therapies, institutional policies, or economic strategies, one needs to know how the diseased organism, the institution, or the economy work. Often, heuristic methods based on multivariate or univariate associations and prediction accuracy are used to induce causation, for example, consider as causally "related" the features that have a strong association with T. Such heuristics may lead to several pitfalls and erroneous inductions, as we will show in the present paper. For principled causal discovery with known theoretical properties a causal theory is needed and classification is not, in general, sufficient (Spirtes et al., 2000; Pearl, 2000; Glymour and Cooper, 1999). Consider the classical epidemiologic example of the tar-stained finger of the heavy smoker: it does predict important outcomes (e.g., increased likelihood for heart attack and lung cancer). However, eliminating the yellow stain by washing the finger does not alter these outcomes. While experiments can help discover causal structure, quite often experimentation is impossible, impractical, or unethical. For example, it is unethical to force people to smoke and it is currently impossible to manipulate most genes in humans in order to discover which genes cause disease and how they interact in doing so. Moreover, the discoveries anticipated due to the explosive

growth of biomedical and other data cannot be made in any reasonable amount of time using solely the classical experimental approach where a single gene, protein, treatment, or intervention is attempted each time, since the space of needed experiments is immense. It is clear that computational methods are needed to catalyze the discovery process.

Fortunately, relatively recently (1980's), it was shown that it is possible to soundly infer causal relations from *observational* data in many practical cases (Spirtes et al., 2000; Pearl, 2000; Glymour and Cooper, 1999; Pearl, 1988). Since then, algorithms that infer such causal relations have been developed that can greatly reduce the number of experiments required to discover the causal structure. Several empirical studies have verified their applicability (Tsamardinos et al., 2003b; Spirtes et al., 2000; Glymour and Cooper, 1999; Aliferis and Cooper, 1994).

One of the most common methods to model and induce causal relations is by learning causal Bayesian networks (Neapolitan, 2004; Spirtes et al., 2000; Pearl, 2000). A special, important and quite broad class of such networks is the family of *faithful networks* intuitively defined as those whose probabilistic properties, and specifically the dependencies and independencies, are a direct function of their structure (Spirtes et al., 2000). Cooper and Herskovits were the first to devise a score measuring the fit of a network structure to the data based on Bayesian statistics, and used it to learn the highest score network structure (Cooper and Herskovits, 1992). Heckerman and his colleagues studied theoretically the properties of the various scoring metrics as they pertain to causal discovery (Glymour and Cooper, 1999; Heckerman, 1995; Heckerman et al., 1995). Heckerman also recently showed that Bayesian-scoring methods also assume (implicitly) faithfulness, see Chapter 4 of Glymour and Cooper (1999). Another prototypical method for learning causal relationships by inducing causal Bayesian networks is the constraint-based approach as exemplified in the PC algorithm by Spirtes et al. (2000). The PC induces causal relations by assuming faithfulness and by performing tests of independence. A network with a structure consistent with the results of the tests of independence is returned. Several other methods for learning networks have been devised subsequently (Chickering, 2003; Moore and Wong, 2003; Cheng et al., 2002a; Friedman et al., 1999b).

There may be many different networks that fit the data equally well, even in the sample limit, and that exhibit the same dependencies and independencies and are thus statistically equivalent. These networks belong to the same Markov equivalence class of causal graphs and contain the same causal edges but may disagree on the direction of some of them, that is, whether A causes B or vice-versa (Chickering, 2002; Spirtes et al., 2000). An *essential graph* is a graph where the directed edges represent the causal relations on which all equivalent networks agree upon their directionality and all the remaining edges are undirected. Causal discovery by employing causal Bayesian networks is based on the following principles. The PC (Spirtes et al., 2000), Greedy Equivalence Search (Chickering, 2003) and other prototypical or state-of-the-art Bayesian network-learning algorithms provide theoretical guarantees, that under certain conditions such as faithfulness they will converge to a network that is statistically indistinguishable from the true, causal, data-generating network, if there is such. Thus, if the conditions hold the existence of all and the direction of some of the causal relations can be induced by these methods and graphically identified in the essential graph of the learnt network.

A typical condition of the aforementioned methods is causal sufficiency (Spirtes et al., 2000). This condition requires that for every pair of measured variables all their common direct causes are also measured. In other words, there are no hidden, unmeasured confounders for any pair of variables. Algorithms, such as the FCI, that in some cases can discover causal relationships in the

presence of hidden confounding variables and selection bias, have also been designed (see Spirtes et al. 2000 and Chapter 6 of Glymour and Cooper 1999).

As it was mentioned above, using observational data alone (even a sample of an infinite size), one can infer only a Markov equivalence class of causal graphs, which may be inadequate for causal discovery. For example, it is not possible to distinguish with observational data any of these two graphs that belong to the same Markov equivalence class: $X \rightarrow Y$ and $X \leftarrow Y$. However, experimental data can distinguish between these graphs. For example, if we manipulate X and see no change in the distribution of Y, we can conclude that the data-generative graph is not $X \rightarrow Y$. This principle is exploited by active learning algorithms. Generally speaking, causal discovery with active learning can be described as follows: learn an approximation of a causal network structure from available data (which is initially only observational data), select and perform an experiment that maximizes some utility function, augment data and possibly current best causal network with the result of experiment, and repeat the above steps until some termination criterion is met.

Cooper and Yoo (1999) proposed a Bayesian scoring metric that can incorporate both observational and experimental data. Using a similar metric (Tong and Koller, 2001) designed an algorithm to select experiments that reduce the entropy of probability of alternative edge orientations. A similar but more general algorithm has been proposed in Murphy (2001) where the expected information gain of a new experiment is calculated and the experiment with the largest information gain is selected. Both above methods were designed for discrete data distributions. Pournara and Wernisch (2004) proposed another active learning algorithm that uses a loss function defined in terms of the size of transition sequence equivalence class of networks (Tian and Pearl, 2001) and can handle continuous data. Meganck et al. (2006) have introduced an active learning algorithm that is based on a general decision theoretic framework that allows to assign costs to each experiment and each measurement. It is also worthwhile to mention the GEEVE system of Yoo and Cooper (2004) that recommends which experiments to perform to discover gene-regulation pathway. This instance of causal active learning allows to incorporate preferences of the experimenter. Recent work has also provided theoretical bounds and related algorithms to minimize the number of experiments needed to infer causal structure (Eberhardt et al., 2006, 2005).

2.2 Synopsis of Theoretical Results Motivating Present Research

A key question that has been investigated in the feature selection literature is which family of methods is more advantageous: filters or wrappers. A second one is what are the "relevant" features? The latter question presumably is important because "relevant" features should be important for discovery and so several definitions appeared defining relevancy (Guyon and Elisseeff, 2003; Kohavi and John, 1997). Finally, how can we design optimal and efficient feature selection algorithms? Fundamental theoretical results connecting Markov blanket induction for feature selection and local causal discovery to standard notions of relevance were given in Tsamardinos and Aliferis (2003). The latter paper provides a technical account and together with Spirtes et al. (2000), Pearl (2000), Kohavi and John (1997) and Pearl (1988) they constitute the core theoretical framework underpinning the present work. Here we provide a very concise description of the results in Tsamardinos and Aliferis (2003) since they partially answer these questions and pave the way to principled feature selection:

1. Relevance cannot be defined independently of the learner and the model-performance metric (e.g., the loss function used) in a way that the relevant features are the solution to the feature

selection problem. The quest for a universally applicable notion of relevancy for prediction is futile.

2. Wrappers are subject to the No-Free Lunch Theorem for optimization: averaged out on all possible problems any wrapper algorithm will do as well as a random search in the space of feature subsets. Therefore, there cannot be a wrapper that is a priori more efficient than any other (i.e., without taking into account the learner and model-performance metric). The quest for a universally efficient wrapper is futile as well.

3. Any filter algorithm can be viewed as the implementation of a definition of relevancy. Because of #1, there is no filter algorithm that is universally optimal, independently of the learner and model-performance metric.

4. Because of #2, wrappers cannot guarantee universal efficiency and because of #3, filters cannot guarantee universal optimality and in that respect, neither approach is superior to the other.

5. Under the conditions that (i) the learner that constructs the classification model can actually learn the distribution $P(T|MB(T))$ and (ii) that the loss function is such that perfect estimation of the probability distribution of T is required with the smallest number of variables, the Markov blanket of T is the optimal solution to the feature selection problem.

6. Sound Markov blanket induction algorithms exist for faithful distributions.

7. In faithful distributions and under the conditions of #5, the strongly/weakly/irrelevant taxonomy of variables (Kohavi and John, 1997) can be mapped naturally to causal graph properties. Informally stated, strongly relevant features were defined by Kohavi and John (1997) to be features that contain information about the target not found in other variables; weakly relevant features are informative but redundant; irrelevant features are not informative (for formal definitions see Section 3). Under the causal interpretation of this taxonomy of relevancy, strongly relevant features are the members of the Markov blanket of the target variable, weakly relevant features are all variables with an undirected path to T which are not themselves members of $MB(T)$, and irrelevant features are variables with no undirected path to the target.

8. Since in faithful distributions the $MB(T)$ contains the direct causes and direct effects of T, and since state-of-the-art $MB(T)$ algorithms output the spouses separately from the direct causes and direct effects, inducing the $MB(T)$ not only solves the feature selection problem but also a form of local causal discovery problem.

Figure 1 provides a summary of the connection between causal structure and predictivity.

We will refer to algorithms that perform feature selection by formal causal induction as *causal feature selection* and algorithms that do not as *non-causal*. As highly complementary to the above results we would add the arguments in favor of causal feature selection presented in Guyon et al. (2007) and recent theoretical (Hardin et al., 2004) and empirical (Statnikov et al., 2006) results that show that under the same sufficient conditions that make Markov blanket the optimal solution to the feature selection and local causal discovery problem, state-of-the-art methods such as ranking features by SVM weights (RFE being a prototypical algorithm Guyon et al. 2002) do not return the

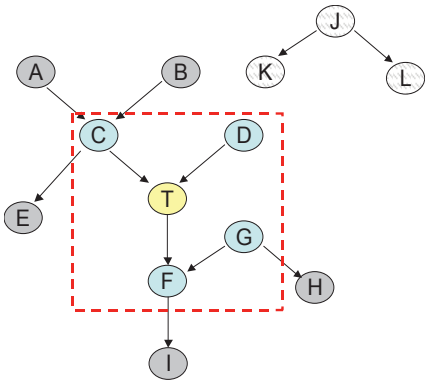


Figure 1: Relationship between causal structure and predictivity in faithful distributions. Cyan variables are members of Markov blanket of T. They are depicted inside the red dotted square (i.e., variables that have undirected path to target T and that are predictive of T given the remaining variables which makes them strongly relevant). Markov blanket variables include direct causes of T (C,D), direct effects (F), and "spouses" of T (i.e., direct causes of the direct effects of T) (G). Grey variables are non-members of Markov blanket of T that have undirected path to T. They are not predictive of T given the remaining variables but they are predictive given a subset of the remaining variables (which makes them weakly relevant). Light-gray variables are variables that do not have an undirected path to T. They are not predictive of T given any subset of the remaining variables, thus they are irrelevant.

correct causal neighborhood and are not minimal, that is, do not solve the feature selection problem) even in the large sample limit.

The above theoretical results also suggest that one should not attempt to define and identify the relevant features for prediction, when discovery is the goal of the analysis. Instead, we argue that a set of features with well-defined *causal* semantics should be identified instead: for example, the $MB(T)$, the set of direct causes and direct effects of T, the set of all (direct and indirect) causes of T, and so on.

We will investigate limitations of prominent non-causal feature selection algorithms in the companion paper (Aliferis et al., 2010).

2.3 Methods to Speed-up Discovery: Local Discovery as a Critical Tool for Scalability

As appealing as causal discovery may be for understanding a domain, predicting effects of intervention, and pursuing principled feature selection for classification, a major problem up until recent years has been scalability. The PC algorithm is worst-case exponential (Spirtes et al., 2000) and in practical settings it cannot typically handle more than a hundred variables. The FCI algorithm is similarly worst-case intractable (Spirtes et al., 2000) and does not handle more than a couple of dozen of variables practically. Learning Bayesian networks with Bayesian scoring techniques is NP-Hard (Chickering et al., 1994). Heuristic hill-climbing techniques such as the Sparse Candidate Algorithm (Friedman et al., 1999b) do not provide guaranteed correct solutions, neither they are very efficient (they can cope with a few hundred variables at the most in practical applications).

With the advent of massive data sets in biology, medicine, information retrieval, the WWW, finance, economics, and so on, scalability has become a critical requirement for practical algorithms. In early 2000's predictions about the feasibility of causal discovery in high-dimensional data were bleak (Silverstein et al., 2000). A variety of methods to scale up causal discovery have been devised to address the problem:

1. Learn the full graph but focus on special types of distributions;

2. Exploit domain knowledge to speed-up learning;

3. Abandon the effort to learn the full causal graph and instead develop methods that find a portion of the true arcs (not specific to some target variable);

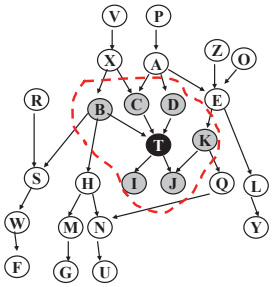
4. Abandon the effort to learn the full causal graph and instead develop methods that learn the local neighborhood of a specific target variable directly;

5. Abandon the effort to learn the fully oriented causal graph and instead develop methods that learn the unoriented graph;

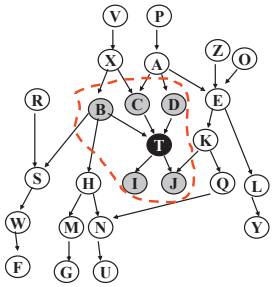
6. Induce constrains of the possible relationships among variables and then learn the full causal graph.

Techniques #1 and #2 were introduced in Chow and Liu (1968) for learning tree-like graphs and Naïve-Bayes graphs (Duda and Hart, 1973), while modern versions are exemplified in (i) TAN/BAN classifiers that relax the Naïve-Bayes structure (Cheng and Greiner, 2001, 1999; Friedman et al., 1997), (ii) efficient complete model averaging of Naïve-Bayes classifiers (Dash and Cooper, 2002), and (iii) algorithm TPDA which restricts the class of distributions so that learning becomes from worst-case intractable to solvable in 4^{th} degree polynomial time to the number of variables (and quadratic if prior knowledge about the ordering of variables is known) (Cheng et al., 2002a). Technique #3 was introduced by Cooper (1997) and replaced learning the complete graph by learning only a small portion of the edges (not pre-specified by the user but determined by the discovery method). Techniques $\#4-6$ pertain to local learning: Technique #4 seeks to learn the complete causal neighbourhood around a target variable provided by the user (Aliferis et al., 2003a; Tsamardinos et al., 2003b). We emphasize that local learning (technique #4) is not the same as technique #3 (incomplete learning) although inventors of incomplete methods often call them 'local'. Technique #5 abandons directionality and learns only a fully connected but undirected graph by using local learning methods (Tsamardinos et al., 2006; Brown et al., 2005). Often post-processing with additional algorithms can provide directionality. The latter can also be obtained by domain-specific criteria or experimentation. Finally, technique #6 uses local learning to restrict the search space for full-graph induction algorithms (Tsamardinos et al., 2006; Aliferis and Tsamardinos, 2002b).

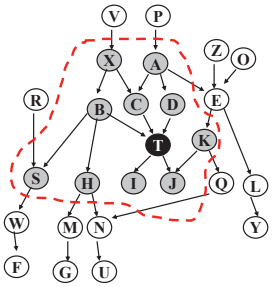
In the present paper we explore methods to learn local causal neighborhoods and test them in high-dimensional data sets. In the companion paper (Aliferis et al., 2010) we provide a framework for building global graphs using the local methods. Incomplete learning (technique #3) is not pursued because it is redundant in light of the other (complete) local and global learning approaches. Figure 2 provides a visual reference guide to the kinds of causal discovery problems the methods in the present work are able to address by starting from local causal discovery.



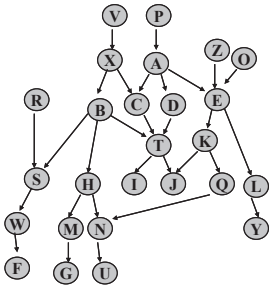
Problem #1: Consider a target variable T and discover Markov Blanket of T.



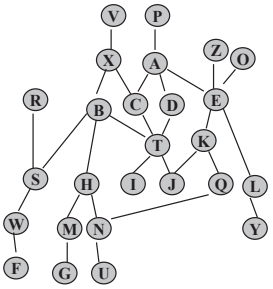
Problem #2: Consider a target variable T and discover Parents and Children of T.



Problem #3: Consider a target variable T and discover regions (e.g., of depth 2 edges) around T.



Problem #4: Discover directed graph.



Problem #5: Discover undirected graph.

Figure 2: Five types of causal discovery from local (types 1, 2), to global (4, 5) and intermediate (3). Specialized algorithms that solve type 2 (local causes and effects) can become building blocks for relatively efficiently solving all other types of causal discovery as well (see text for details).

2.4 Desiderata for Local Algorithms, Brief Review of Prior Methods for Markov Blanket and Local Neighborhood Induction

An ideal local learning algorithm should have three characteristics: (a) well-defined properties, especially broadly applicable conditions that guarantee correctness, (b) good performance in practical distributions and corresponding data sets, including ones with small sample and many features, and finally (c) scalability in terms of running time. We briefly review progress made in the field toward these goals.

Firm theoretical foundations of Bayesian networks were laid down by Pearl and his co-authors (Pearl, 1988). Furthermore, all local learning methods exploit either the constraint-based framework for causal discovery developed by Spirtes, Glymour, Schienes, Pearl, and Verma and their co-authors (Spirtes et al., 2000; Pearl, 2000; Pearl and Verma, 1991) or the Bayesian search-and-score Bayesian network learning framework introduced by Cooper and Herskovits (1992). The relevant key contributions were covered in Section 2.1 and will not be repeated here.

While the above foundations were introduced and developed in the span of at least the last 30 years, local learning is no more than 10 years old. Specialized Markov blanket learning methods were first introduced in 1996 (Koller and Sahami, 1996), incomplete causal methods in 1997 (Cooper, 1997), and local causal discovery methods (for targeted complete induction of direct causes and effects) were first introduced in 2002 and 2003 (Tsamardinos et al., 2003b; Aliferis and Tsamardinos, 2002a). In 1996, Koller et al. introduced a heuristic algorithm for inducing the

Markov blanket from data and tested the algorithm in simulated, real text, and other types of data from the UCI repository (Koller and Sahami, 1996). In 1997 Cooper and colleagues introduced and applied the heuristic method K2MB for finding the Markov blanket of a target variable in the task of predicting pneumonia mortality (Cooper, 1997). In 1997 Cooper introduced an incomplete method for causal discovery (Cooper et al., 1997). The algorithm was able to circumvent lack of scalability of global methods by returning a subset of arcs from the full network. To avoid notational confusion we point out that the algorithm was termed LCD (local causal discovery) despite being an *incomplete rather than local* algorithm as local algorithms are defined in the present paper (i.e., focused on some user-specified target variable or localized region of the network). A revision of the algorithm termed LCD2 was presented in Mani and Cooper (1999).

In 1999 Margaritis and Thrun introduced the GS algorithm with the intent to induce the Markov blanket for the purpose of speeding up global network learning (i.e., not for feature selection) (Margaritis and Thrun, 1999). GS was the first published sound Markov blanket induction algorithm. The weak heuristic used by GS combined with the need to condition on at least as many variables simultaneously as the Markov blanket size makes it impractical for many typical data sets since the required sample grows exponentially to the size of the Markov blanket. This in turn forces the algorithm to stop its execution prematurely (before it identifies the complete Markov blanket) because it cannot grow the conditioning set while performing reliable tests of independence. Evaluations of GS by its inventors were performed in data sets with a few dozen variables leaving the potential of scalability largely unexplored.

In 2001 Cheng et al. applied the TPDA algorithm (a global BN learner) (Cheng et al., 2002a) to learn the Markov blanket of the target variable in the Thrombin data set in order to solve a prediction problem of drug effectiveness on the basis of molecular characteristics (Cheng et al., 2002b). Because TPDA could not be run with more than a few hundred variables efficiently, they pre-selected 200 variables (out of 139,351 total) using univariate filtering. Although this procedure in general will not find the true Markov blanket (because otherwise-unconnected with the target spouses can be missed, many true parents and children may not be in the first 200 variables, and many non-Markov blanket members cannot be eliminated), the resulting classifier performed very well winning the 2001 KDD Cup competition.

Friedman et al. proposed a simple Bootstrap procedure for determining membership in the Markov blanket for small sample situations (Friedman et al., 1999a). The Markov blanket in this method is to be extracted from the full Bayesian network learned by the SCA (Sparse Candidate Algorithm) learner (Friedman et al., 1999b).

In 2002 and 2003 Tsamardinos, Aliferis, et al. presented a modified version of GS, termed IAMB and several variants of the latter that through use of a better inclusion heuristic than GS and optional post-processing of the tentative and final output of the local algorithm with global learners would achieve true scalability to data sets with many thousands of variables and applicability in modest (but not very small) samples (Tsamardinos et al., 2003a; Aliferis et al., 2002). IAMB and several variants were tested both in the high-dimensional Thrombin data set (Aliferis et al., 2002) and in data sets simulated from both existing and random Bayesian networks (Tsamardinos et al., 2003a). The former study found that IAMB scales to high-dimensional data sets. The latter study compared IAMB and its variants to GS, Koller-Sahami, and PC and concluded that IAMB variants on average perform best in the data sets tested.

In 2003 Tsamardinos and Aliferis presented a full theoretical analysis explaining relevance as defined by Kohavi and John (1997) in terms of Markov blanket and causal connectivity (Tsamardi-

nos and Aliferis, 2003). They also provided theoretical results about the strengths and weaknesses of filter versus wrapper algorithms, the impossibility of a universal definition of relevance, and the optimality of Markov blanket as a solution to the feature selection problem in formal terms. These results were summarized in Section 2.2.

The extension of Sparse Candidate Algorithm to create a local-to-global learning strategy was first introduced in Aliferis and Tsamardinos (2002b) and led to the MMHC algorithm introduced and evaluated in Tsamardinos et al. (2006). MMHC was shown in Tsamardinos et al. (2006) to achieve best-of-class performance in quality and scalability compared to most state-of-the-art global network learning algorithms.

In 2002 Aliferis et al. also introduced parallel and distributed versions of the IAMB family of algorithms (Aliferis et al., 2002). These serve as the precursor of the parallel and distributed local neighborhood learning method presented in the companion paper (Aliferis et al., 2010). The precursor of the GLL framework was also introduced by Aliferis and Tsamardinos in 2002 for the explicit purpose of reducing the sample size requirements of IAMB-style algorithms (Aliferis and Tsamardinos, 2002a).

In 2003 Aliferis et al. introduced algorithm HITON[1] Aliferis et al., and Tsamardinos et al. introduced algorithms MMPC and MMMB (Aliferis et al., 2003a; Tsamardinos et al., 2003b). These are the first concrete algorithms that would find sets of direct causes or direct effects and Markov blankets in a scalable and efficient manner. HITON was tested in 5 biomedical data sets spanning clinical, text, genomic, structural and proteomic data and compared against several feature selection methods with excellent results in parsimony and classification accuracy (Aliferis et al., 2003a). MMPC was tested in data simulated from human-derived Bayesian networks with excellent results in quality and scalability. MMMB was tested in the same data sets and compared to prior algorithms such as Koller-Sahami algorithm and IAMB variants with superior results in the quality of Markov blankets. These benchmarking and comparative evaluation experiments provided evidence that the local learning approach held not only theoretical but also practical potential.

HITON-PC, HITON-MB, MMPC, and MMMB algorithms lacked so-called "symmetry correction" (Tsamardinos et al., 2006), however HITON used a wrapping post-processing that at least in principle removed this type of false positives. The symmetry correction was introduced in 2005 and 2006 by Tsamardinos et al. in the context of the introduction of MMHC (Tsamardinos et al., 2006, 2005). Peña et al. also published work pointing to the need for a symmetry correction in MMPC (Peña et al., 2005b).

HITON was applied in 2005 to understand physician decisions and guideline compliance in the diagnosis of melanomas (Sboner and Aliferis, 2005). HITON has been applied for the discovery of biomarkers in human cancer data using microarrays and mass spectrometry and is also implemented in the GEMS and FAST-AIMS systems for the automated analysis of microarray and mass spectrometry data respectively (Statnikov et al., 2005b; Fananapazir et al., 2005). In a recent extensive comparison of biomarker selection algorithms (Aliferis et al., 2006a,b) it was found that HITON outperforms 16 state-of-the-art representatives from all major biomarker algorithmic families in terms of combined classification performance and feature set parsimony. This evaluation used 9 human cancer data sets (gene expression microarray and mass spectrometry) in 10 diagnostic and outcome (i.e., survival) prediction classification tasks. In addition to the above real data, resimulation was also used to create two gold standard network structures, one re-engineered from

1. From the Greek word "Χιτών" meaning "cloak", and pronounced <hee tó n>.

human lung cancer data and one from yeast data. Several applications of HITON in text categorization have been published where the algorithm was used to understand complex "black box" SVM models and convert complex models to Boolean queries usable by Boolean interfaces of Medline (Aphinyanaphongs and Aliferis, 2004), to examine the consistency of editorial policies in published journals (Aphinyanaphongs et al., 2006), and to predict drug-drug interactions (Duda et al., 2005). HITON was also compared with excellent results to manual and machine feature selection in the domain of early graft failure in patients with liver transplantations (Hoot et al., 2005).

In 2003 Frey et al. explored the idea of using decision tree induction to indirectly approximate the Markov blanket (Frey et al., 2003). They produced promising results, however a main problem with the method was that it requires a threshold parameter that cannot be optimized easily. Furthermore, as we show in the companion paper (Aliferis et al., 2010) decision tree induction is subject to synthesis and does not select only the Markov blanket members.

In 2004 Mani et al. introduced BLCD-MB, which resembles IAMB but using a Bayesian scoring metric rather than conditional independence testing (Mani and Cooper, 2004). The algorithm was applied with promising results in infant mortality data (Mani and Cooper, 2004).

A method for learning regions around target variables by recursive application of MMPC or other local learning methods was introduced in Tsamardinos et al. (2003c). Peña et al. applied interleaved MMPC for learning regions in the domain of bioinformatics (Peña et al., 2005a).

In 2006 Gevaert et al. applied K2MB for the purpose of learning classifiers that could be used for prognosis of breast cancer from microarray and clinical data (Gevaert et al., 2006) . Univariate filtering was used to select 232 genes before applying K2MB.

Other recent efforts in learning Markov blankets include the following algorithms: PCX, which post-processes the output of PC (Bai et al., 2004); KIAMB, which addresses some violations of faithfulness using a stochastic extension to IAMB (Peña et al., 2007); FAST-IAMB, which speeds up IAMB (Yaramakala and Margaritis, 2005); and MBFS, which is a PC-style algorithm that returns a graph over Markov blanket members (Ramsey, 2006).

2.5 Open Problems and Focus of Paper

The focus of the present paper is to describe state-of-the-art algorithms for inducing direct causes and effects of a response variable or its Markov blanket using a novel cohesive framework that can help in the analysis, understanding, improvement, application (including configuration / parameterization) and dissemination of the algorithms. We furthermore study comparative performance in terms of predictivity and parsimony of state-of-the-art local causal algorithms; we compare them to non-causal algorithms in real and simulated data sets using the same criteria; and show how novel algorithms can be obtained. A second major hypothesis (and set of experiments in the present paper) is that non-causal feature selection methods may yield predictively optimal feature sets while from a causal perspective their output is unreliable. Testing this hypothesis has tremendous implications in many areas (e.g., analysis of biomedical molecular data) where highly predictive variables (biomarkers) of phenotype (e.g., disease or clinical outcome) are often interpreted as being causally implicated for the phenotype and great resources are invested in pursuing these markers for new drug development and other research.

In the second part of our work (Aliferis et al., 2010) we address gaps in the theoretical understanding of local causal discovery algorithms and provide empirical and theoretical analyses of their

behavior as well as several extensions including algorithms for learning the full causal graph using a divide-and-conquer local learning approach.

3. Notation and Definitions

In the present paper we use Bayesian networks as the language in which to represent data generating processes and causal relationships. We thus first formally define causal Bayesian networks. Recall that in a directed acyclic graph (DAG), a node A is the parent of B (B is the child of A) if there is a direct edge from A to B, A is the ancestor of B (B is the descendant of A) if there is a direct path from A to B. "Nodes", "features", and "variables" will be used interchangeably.

3.1 Notation

We will denote variables with uppercase letters X, Y, Z, values with lowercase letters, x, y, z, and sets of variables or values with boldface uppercase or lowercase respectively. A "target" (i.e., response) variable is denoted as T unless stated otherwise.

Definition 1 ***Conditional Independence***. *Two variables X and Y are conditionally independent given $\boldsymbol{Z}$, denoted as $I(X,Y|\boldsymbol{Z})$, iff $P(X=x,Y=y|\boldsymbol{Z}=\boldsymbol{z}) = P(X=x|\boldsymbol{Z}=\boldsymbol{z})P(Y=y|\boldsymbol{Z}=\boldsymbol{z})$, for all values $x, y, \boldsymbol{z}$ of $X, Y, \boldsymbol{Z}$ respectively, such that $P(\boldsymbol{Z}=\boldsymbol{z}) > 0$.*

Definition 2 ***Bayesian network*** $\langle \boldsymbol{V}, G, J \rangle$. *Let $\boldsymbol{V}$ be a set of variables and J be a joint probability distribution over all possible instantiations of $\boldsymbol{V}$. Let G be a directed acyclic graph (DAG) such that all nodes of G correspond one-to-one to members of $\boldsymbol{V}$. We require that for every node $A \in \boldsymbol{V}$, A is probabilistically independent of all non-descendants of A, given the parents of A (i.e.,* Markov Condition *holds). Then we call the triplet $\langle \boldsymbol{V}, G, J \rangle$ a Bayesian network (abbreviated as "BN"), or equivalently a belief network or probabilistic network (Neapolitan, 1990).*

Definition 3 ***Operational criterion for causation***. *Assume that a variable A can be forced by a hypothetical experimenter to take values a_i. If the experimenter assigns values to A according to a uniformly random distribution over values of A, and then observes $P(B|A=a_i) \neq P(B|A=a_j)$ for some i and j, (and within a time window dt), then variable A is a cause of variable B (within dt).*

We note that randomization of values of A serves to eliminate any combined causative influences on both A and B. We also note that universally acceptable definitions of causation have eluded scientists and philosophers for centuries. Indeed the provided criterion is not a proper definition, because it examines one cause at a time (thus multiple causation can be missed), it assumes that a hypothetical experiment is feasible even when in practice this is not attainable, and the notion of "forcing" variables to take values presupposes a special kind of causative primitive that is formally undefined. Despite these limitations, the above criterion closely matches the notion of a Randomized Controlled Experiment which is a de facto standard for causation in many fields of science, and following common practice in the field (Glymour and Cooper, 1999) will serve operationally the purposes of the present paper.

Definition 4 ***Direct and indirect causation***. *Assume that a variable A is a cause of variable B according to the operational criterion for causation in definition 3. A is an indirect cause for B with respect to a set of variables $\boldsymbol{V}$, iff A is not a cause of B for some instantiation of values of $\boldsymbol{V} \setminus \{A, B\}$, otherwise A is a direct cause of B.*

Definition 5 ***Causal probabilistic network (a.k.a. causal Bayesian network)***. *A causal probabilistic network (abbreviated as "CPN") $\langle \boldsymbol{V}, G, J \rangle$ is the Bayesian network $\langle \boldsymbol{V}, G, J \rangle$ with the additional semantics that if there is an edge $A \rightarrow B$ in G then A directly causes B (for all $A, B \in \boldsymbol{V}$) (Spirtes et al., 2000).*

Definition 6 ***Faithfulness***. *A directed acyclic graph G is faithful to a joint probability distribution J over variable set $\boldsymbol{V}$ iff every independence present in J is entailed by G and the Markov Condition. A distribution J is faithful iff there exists a directed acyclic graph G such that G is faithful to J (Spirtes et al., 2000; Glymour and Cooper, 1999).*

It follows from the Markov Condition that in a CPN $C = \langle \boldsymbol{V}, G, J \rangle$ every conditional independence entailed by the graph G is also present in the probability distribution J encoded by C. *Thus, together faithfulness and the causal Markov Condition establish a close relationship between a causal graph G and some empirical or theoretical probability distribution J. Hence we can associate statistical properties of the sample data with causal properties of the graph of the CPN.* The d-separation criterion determines all independencies entailed by the Markov Condition and a graph G.

Definition 7 ***d-separation, d-connection***. *A* collider *on a path p is a node with two incoming edges that belong to p. A path between X and Y given a conditioning set $\boldsymbol{Z}$ is open, if (i) every collider of p is in $\boldsymbol{Z}$ or has a descendant in $\boldsymbol{Z}$, and (ii) no other nodes on p are in $\boldsymbol{Z}$. If a path is not open, then it is* blocked. *Two variables X and Y are d-separated given a conditioning set $\boldsymbol{Z}$ in a BN or CPN C iff every path between X, Y is blocked (Pearl, 1988).*

Property 1 *Two variables X and Y are d-separated given a conditioning set $\boldsymbol{Z}$ in a faithful BN or CPN iff $I(X, Y | \boldsymbol{Z})$ (Spirtes et al., 2000). It follows, that if they are d-connected, they are conditionally dependent.*

Thus, in a faithful CPN, d-separation captures *all* conditional dependence and independence relations that are encoded in the graph.

Definition 8 ***Markov blanket of T, denoted as $MB(T)$***. *A set $MB(T)$ is a minimal set of features with the following property: for every variable subset $\boldsymbol{S}$ with no variables in $MB(T)$, $I(\boldsymbol{S}, T | MB(T))$. In Pearl's terminology this is called the Markov Boundary (Pearl, 1988).*

Property 2 *The $MB(T)$ of any variable T in a faithful BN or a CPN is unique (Tsamardinos et al., 2003b) (also directly derived from Pearl and Verma 1991 and Pearl and Verma 1990).*

Property 3 *The $MB(T)$ in a faithful CPN is the set of parents, children, and parents of children (i.e., "spouses") of T (Pearl, 2000, 1988).*

Definition 9 ***Causal sufficiency***. *For every pair of measured variables, all their common causes are also measured.*

Definition 10 ***Feature selection problem***. *Given a sample S of instantiations of variable set $\boldsymbol{V}$ drawn from distribution D, a classifier induction algorithm C and a loss function L, find: smallest subset of variables $\boldsymbol{F} \subseteq \boldsymbol{V}$ such that $\boldsymbol{F}$ minimizes expected loss $L(M, D)$ in distribution D where M is the classifier model (induced by C from sample S projected on $\boldsymbol{F}$).*

In the above definition, we mean "exact" minimization of $L(M,D)$. In other words, out of all possible subsets of variable set $\boldsymbol{V}$, we are interested in subsets $\boldsymbol{F} \subseteq \boldsymbol{V}$ that satisfy the following two criteria: (i) $\boldsymbol{F}$ minimizes $L(M,D)$ and (ii) there is no subset $\boldsymbol{F}^* \subseteq \boldsymbol{V}$ such that $|\boldsymbol{F}^*| < |\boldsymbol{F}|$ and $\boldsymbol{F}^*$ also minimizes $L(M,D)$.

Definition 11 ***Wrapper feature selection algorithm***. *An algorithm that tries to solve the Feature Selection problem by searching in the space of feature subsets and evaluating each one with a user-specified classifier and loss function estimator.*

Definition 12 ***Filter feature selection algorithm***. *An algorithm designed to solve the Feature Selection problem by looking at properties of the data and not by applying a classifier to estimate expected loss for different feature subsets.*

Definition 13 ***Causal feature selection algorithm***. *An algorithm designed to solve the Feature Selection problem by (directly or indirectly) inducing causal structure and by exploiting formal connections between causation and predictivity.*

Definition 14 ***Non-causal feature selection algorithm***. *An algorithm that tries to solve the Feature Selection problem without reference to the causal structure that underlies the data.*

Definition 15 ***Irrelevant, strongly relevant, weakly relevant, relevant feature (with respect to target variable*** T***)***. *A variable set $\boldsymbol{I}$ that conditioned on every subset of the remaining variables does not carry predictive information about T is irrelevant to T. Variables that are not irrelevant are called relevant. Relevant variables are strongly relevant if they are predictive for T given the remaining variables, while a variable is weakly relevant if it is non-predictive for T given the remaining variables (i.e., it is not strongly relevant) but it is predictive given some subset of the remaining variables.*

4. A General Framework for Local Learning

In this section we present a formal general framework for learning local causal structure. Such a framework enables a systematic exploration of a family of related but not identical algorithms which can be seen as instantiations of the same broad algorithmic principles encapsulated in the framework. Also, the framework allows us to think about formal conditions for correctness not only at the algorithm level but also at the level of algorithm family. We are thus able to identify two distinct sets of assumptions for correctness: the more general set of assumptions (*admissibility rules*) applies to the generative algorithms and provides a set of flexible rules for constructing numerous algorithmic instantiations each one of which is guaranteed to be correct provided that in addition a more specific and fixed set of assumptions hold (i.e., specific sufficient conditions for correctness of the algorithms that are instantiations of the generative framework).

We consider the following two problems of local learning:

Problem 1 *Given a set of variables $\boldsymbol{V}$ following distribution P, a sample D drawn from P, and a target variable of interest $T \in \boldsymbol{V}$: determine the direct causes and direct effects of T.*

Problem 2 *Given a set of variables $\boldsymbol{V}$ following distribution P, a sample D drawn from P, and a target variable of interest $T \in \boldsymbol{V}$: determine the direct causes, direct effects, and the direct causes of the direct effects of T.*

From the work of Spirtes et al. (2000) and Pearl (2000, 1988) we know that when the data are observational, causal sufficiency holds for the variables $\boldsymbol{V}$, and the distribution P is faithful to a causal Bayesian network, then the direct causes, direct effects, and direct causes of the direct effects of T, correspond to the parents, children, and spouses of T respectively in that network.

Thus, in the context of the above assumptions, Problem 1 seeks to identify the parents and children set of T in a Bayesian network G faithful to P; we will denote this subset as $PC_G(T)$. There may be several networks that faithfully capture distribution P, however, as we have shown in Tsamardinos et al. (2003b) (also directly derived from Pearl and Verma 1991, 1990) $PC_G(T) = PC_{G'}(T)$, for any two networks G and G' faithful to the same distribution. So, the set of parents and children of T is unique among all Bayesian networks faithful to the same distribution and so we will drop the superscript and denote it simply as $PC(T)$. Notice that, a node may be a parent of T in one network and a child of T in another, for example, the graphs $X \leftarrow T$ and $X \rightarrow T$ may both be faithful to the same distribution. However, the set of parents and children of T, that is, $\{X\}$, remains the same in both networks. Finally, by Theorem 4 in Tsamardinos et al. (2003b) we know that the Markov blanket $MB(T)$ is unique in all networks faithful to the same distribution. Therefore, under the assumptions of the existence of a causal Bayesian network that faithfully captures P and causal sufficiency of $\boldsymbol{V}$, the problems above can be recast as follows:

Problem 3 *Given a set of variables $\boldsymbol{V}$ following distribution P, a sample D drawn from P, and a target variable of interest $T \in \boldsymbol{V}$: determine the $PC(T)$.*

Problem 4 *Given a set of variables $\boldsymbol{V}$ following distribution P, a sample D drawn from P, and a target variable of interest $T \in \boldsymbol{V}$: determine the $MB(T)$.*

Problem 1 is geared toward local causal discovery, while Problem 2 is oriented toward causal feature selection for classification. The solutions to these problems can form the basis for solving several other related local discovery problems, such as learning the unoriented set of causal relations (skeleton of a Bayesian network), a region of interest of a given depth of d edges around T, or further analyze the data to discover the orientation of the causal relations.

The *Generalized Local Learning* (GLL) framework consists of two main types of algorithms: GLL-PC (GLL Parent and Children) for Problem 1 and GLL-MB for Problem 2.

4.1 Discovery of the $PC(T)$ Set

Identification of the $PC(T)$ set is based on the following theorem in Spirtes et al. (2000):

Theorem 1 *In a faithful BN $\langle \boldsymbol{V}, G, P \rangle$ there is an edge between the pair of nodes $X \in \boldsymbol{V}$ and $Y \in \boldsymbol{V}$ iff $\neg I(X, Y | \boldsymbol{Z})$, for all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X, Y\}$.*

Any variable X that does have an edge with T belongs to the $PC(T)$. Thus, the theorem gives rise to an immediate algorithm for identifying $PC(T)$: for any variable $X \in \boldsymbol{V} \setminus \{T\}$, and all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X, T\}$, test whether $\mathrm{I}(X, T | \boldsymbol{Z})$. If such a $\boldsymbol{Z}$ exists for which $\mathrm{I}(X, T | \boldsymbol{Z})$, then $X \notin PC(T)$, otherwise $X \in PC(T)$. This algorithm is equivalent to a "localized version" of SGS (Spirtes et al., 2000). The problem of course is that the algorithm is very inefficient because it tests all subsets of the variables and thus does not scale beyond problems of trivial size. The order of complexity is $\mathrm{O}(|\boldsymbol{V}| 2^{|\boldsymbol{V}|-2})$. The general framework presented below attempts to characterize not only the above algorithm but also efficient implementations of the theorem that maintain soundness.

There are several observations that lead to more efficient but still sound algorithms. First notice that, once a subset $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X,T\}$ has been found s.t. $\mathrm{I}(X,T|\boldsymbol{Z})$ there is no need to perform any other test of the form $\mathrm{I}(X,T|\boldsymbol{Z}')$: we know that $X \notin PC(T)$. Thus, the sooner we identify good candidate subsets $\boldsymbol{Z}$ that can render the variables conditionally independent from T, the fewer tests will be necessary.

Second, to determine whether $X \in PC(T)$ there is no need to test whether $\neg\mathrm{I}(X,T|\boldsymbol{Z})$ for all subsets $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X,T\}$ but only for all subsets $\boldsymbol{Z}' \subseteq Parents_G(T) \setminus \{X\}$ and all $\boldsymbol{Z}' \subseteq Parents_G(X) \setminus \{T\}$ where G is any network faithful to the distribution. To see this, let us first assume that there is no edge between X and T. Notice that either X is a non-descendant of T or T is a non-descendant of X since the network is acyclic and they cannot be both descendants of each other. If X is a non-descendant of T in G, then by the Markov Condition we know that there is a subset $\boldsymbol{Z}$ of $Parents_G(T) = Parents_G(T) \setminus \{X\}$ (the equality because we assume no edge between T and X) such that $\mathrm{I}(X,T|\boldsymbol{Z})$. Similarly, if T is a non-descendant of X in G then there is $\boldsymbol{Z} \subseteq Parents_G(X) \setminus \{T\}$ such that $\mathrm{I}(X,T|\boldsymbol{Z})$. Conversely, if there is an edge $X \rightarrow T$ or $T \rightarrow X$, then the dependence $\neg\mathrm{I}(X,T|\boldsymbol{Z})$ holds for all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X,T\}$ (by the theorem), thus also holds for all $\boldsymbol{Z} \subseteq Parents_G(T) \setminus \{X\}$ or $\boldsymbol{Z} \subseteq Parents_G(X) \setminus \{T\}$. We just proved that:

Proposition 1 *In a faithful BN $\langle \boldsymbol{V}, G, P \rangle$ there is an edge between the pair of nodes $X \in \boldsymbol{V}$ and $T \in \boldsymbol{V}$ iff $\neg I(X,T|\boldsymbol{Z})$, for all $\boldsymbol{Z} \subseteq Parents_G(X) \setminus \{T\}$ and $\boldsymbol{Z} \subseteq Parents_G(T) \setminus \{X\}$.*

Since the networks in most practical problems are relatively sparse, if we knew the sets $Parents_G(T)$ and $Parents_G(X)$ then the number of subsets that would need to be checked for conditional independence for each $X \in PC(T)$ is significantly smaller: $|2^{|\boldsymbol{V}\setminus\{T,X\}|}| \gg |2^{|Parents_G(X)|}| + |2^{|Parents_G(T)|}|$. Of course, we do not know the sets $Parents_G(T)$ and $Parents_G(X)$ but one could work with any superset of them as shown by the following proposition:

Proposition 2 *In a faithful BN $\langle \boldsymbol{V}, G, P \rangle$ there is an edge between the pair of nodes $X \in \boldsymbol{V}$ and $T \in \boldsymbol{V}$ iff $\neg I(X,T|\boldsymbol{Z})$, for all $\boldsymbol{Z} \subseteq \boldsymbol{S}$ and $\boldsymbol{Z} \subseteq \boldsymbol{S}'$, where $Parents_G(X) \setminus \{T\} \subseteq \boldsymbol{S} \subseteq \boldsymbol{V} \setminus \{X,T\}$ and $Parents_G(X) \setminus \{T\} \subseteq \boldsymbol{S}' \subseteq \boldsymbol{V} \setminus \{X,T\}$.*

Proof If there is an edge between the pair of nodes X and T then $\neg\mathrm{I}(X,T|\boldsymbol{Z})$, for all subsets $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X,T\}$ (by Theorem 1) and so $\neg\mathrm{I}(X,T|\boldsymbol{Z})$ for all $\boldsymbol{Z} \subseteq \boldsymbol{S}$ and $\boldsymbol{Z} \subseteq \boldsymbol{S}'$ too. Conversely, if there is no edge between the pair of nodes X and T, then $\mathrm{I}(X,T|\boldsymbol{Z})$, for some $\boldsymbol{Z} \subseteq Parents_G(X) = Parents_G(X) \setminus \{T\} \subseteq \boldsymbol{S}$ or $\boldsymbol{Z} \subseteq Parents_G(T) = Parents_G(T) \setminus \{X\} \subseteq \boldsymbol{S}'$ (by Proposition 1). ■

Now, the sets $Parents_G(X)$ and $Parents_G(T)$ depend on the specific network G that we are trying to learn. As we mentioned however, there may be several such statistically equivalent networks among which we cannot differentiate from the data, forming an equivalence class. Thus, it is preferable to work with supersets of $Parents_G(T)$ and $Parents_G(X)$ that do not depend on a specific network member of the class: these supersets are the sets $PC(T)$ and $PC(X)$.

Let us suppose that we have available a superset of $PC(T)$ called $TPC(T)$ (tentative PC). For any node $X \in TPC(T)$ if $\mathrm{I}(X,T|\boldsymbol{Z})$ for some $\boldsymbol{Z} \subseteq TPC(T) \setminus \{X,T\}$, then by Proposition 2, we know that X has no edge with T, that is, $X \notin PC(T)$. So, X should also be removed from $TPC(T)$ to obtain a better approximation of $PC(T)$. If however, $\neg\mathrm{I}(X,T|\boldsymbol{Z})$ for all $\boldsymbol{Z} \subseteq TPC(T) \setminus \{X,T\}$, then it is still possible that $X \notin PC(T)$ because there may be a set $\boldsymbol{Z} \subseteq PC(X)$ where $\boldsymbol{Z} \nsubseteq PC(T)$ for which $\mathrm{I}(X,T|\boldsymbol{Z})$.

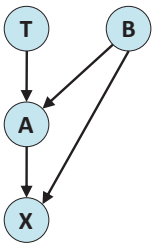


Figure 3: $PC(T) = \{A\}, PC(X) = \{A,B\}, X \notin PC(T)$. Notice that, there is no subset of $PC(T)$ that makes T conditionally independent of $X : \neg I(X,T|\emptyset), \neg I(X,T|A)$. However, there is a subset of $PC(X)$ for which X and T become conditionally independent: $I(X,T|\{A,B\})$. The Extended $PC(T)$ (see Definition 16 in this section) is $EPC(T) = \{A,X\}$.

Is there actually a case, where X cannot be made independent of T by conditioning on some subset of $PC(T)$? We know that all non-descendants of T can be made independent of T conditioned on a subset of its parents, thus, if there is such an X it has to be a descendant of T. Figure 3 shows such a case. These situations are rare in practice as indicated by our empirical results in Sections 5 and 6, which implies that by conditioning on all subsets of $TPC(T)$ one will approximate $PC(T)$ quite closely.

Definition 16 *We call the Extended $PC(T)$, denoted as $EPC(T)$, the set $PC(T)$ union the set of variables X for which $\neg I(X,T|\boldsymbol{Z})$, for all $\boldsymbol{Z} \subseteq PC(T) \setminus \{X\}$.*

The previous results allow us to start building algorithms that operate locally around T in order to find $PC(T)$ efficiently and soundly. Consider first the sketch of the algorithm below:

Algorithm 1

Find a superset $TPC(T)$ of $PC(T)$
for** each variable $X \in TPC(T)$ **do
  if** $\exists \boldsymbol{Z} \subseteq TPC(T) \setminus \{X\}$, s.t. $I(X,T|\boldsymbol{Z})$ **then
    remove X from $TPC(T)$
  end if
end for
Return $TPC(T)$

This algorithm will output $TPC(T) \subseteq EPC(T)$. To ensure we end up with the exact $PC(T)$ we can use the following pruning algorithm:

Algorithm 2

***for all** $X \in TPC(T)$ **do** {returned from Algorithm 1}*
  if** $T \notin TPC(X)$ **then
    remove X from $TPC(T)$ {$TPC(X)$ is obtained by running Algorithm 1}
  end if
end for

GLL-PC: High-level pseudocode and main components of Generalized Local Learning - Parents and Children. Returns PC(*T*)

1. $\boldsymbol{U} \leftarrow$ GLL-PC-nonsym(T) // first approximate *PC(T)* without symmetry check
2. For all $X \in \boldsymbol{U}$
3. If $T \notin$ GLL-PC-nonsym(X) then $\boldsymbol{U} \leftarrow \boldsymbol{U} \setminus \{X\}$ // check for symmetry
4. Return $\boldsymbol{U}$ // true set of parents and children

GLL-PC-nonsym(*T*) // returns a set which is a subset of $EPC(T)$ and a superset of $PC(T)$

1. Initialization
   a. Initialize a set of candidates for the true $PC(T)$ set: *TPC(T)* $\leftarrow S$, s.t. $S \subseteq \boldsymbol{V} \setminus \{T\}$
   b. Initialize a priority queue of variables to be examined for inclusion in *TPC(T)*: OPEN $\leftarrow \boldsymbol{V} \setminus \{T \cup TPC(T)\}$
2. Apply inclusion heuristic function
   a. Prioritize variables in OPEN for inclusion in *TPC(T)*;
   b. Throw away non-eligible variables from OPEN;
   c. Insert in *TPC(T)* the highest-priority variable(s) in OPEN and remove them from OPEN
3. Apply elimination strategy to remove variables from *TPC(T)*
4. Apply interleaving strategy by repeating steps #2 and #3 until a termination criterion is met
5. Return *TPC(T)*

Figure 4: High-level outline and main components (underlined) of GLL-PC algorithm.

In essence, the second algorithm checks for every $X \in TPC(T)$ whether the *symmetrical relation* holds: $T \in TPC(X)$. If the symmetry is broken, we know that $X \notin PC(T)$ since the parents-and-children relation is symmetrical.

What is the complexity of the above algorithms? In Algorithm 1 if step 1 is performed by an Oracle with constant cost, and with $TPC(T)$ equal to $PC(T)$, then the first algorithm requires an order of $O(|\boldsymbol{V}|2^{|PC(T)|})$ tests. The second algorithm will require an order of $O(|\boldsymbol{V}|2^{|PC(X)|})$ tests for each X in $TPC(T)$. Two observations to notice are: (i) the complexity order of the first algorithm depends linearly on the size of the problem $|\boldsymbol{V}|$, exponentially on $|PC(T)|$, which is a structural property of the problem, and how close $TPC(T)$ is to $PC(T)$ and (ii) the second algorithm requires multiple times the time of the first algorithm for minimal returns in quality of learning, that is, just to take care of the scenario in Figure 3 and remove the variables $EPC(T) \setminus PC(T)$ (i.e., X in Figure 3).

Since an Oracle is not available the complexity of both algorithms strongly depends on how close approximation of the $PC(T)$ is and how efficiently this approximation is found. The simplest strategy for example is to set $TPC(T) = \boldsymbol{V}$, essentially getting the local version of the algorithm SGS described above. In general any heuristic method that returns a superset of $PC(T)$ is admissible, that is, it could lead to sound algorithms.

Also notice that in the first algorithm the identification of the members of the $TPC(T)$ (step 1) and the removal of variables from it (step 3) can be interleaved. $TPC(T)$ can grow gradually by one, many variables, or all members of it at a time before it satisfies the requirement that is a superset of $PC(T)$. The requirement for the algorithm to be sound is that, in the end, all tests $I(X, T|\boldsymbol{Z})$ for all subsets $\boldsymbol{Z}$ of $PC(T) \setminus \{X\}$ have been performed.

Given the above, the components of Generalized Local Learning GLL-PC, that is, an algorithm for $PC(T)$ identification based on the above principles are the following: an *inclusion heuristic function* to prioritize variables for consideration as members of $TPC(T)$ and include them in $TPC(T)$ according to established priority. The second component of the framework is an *elimination strategy*, which eliminates variables from the $TPC(T)$ set. An *interleaving strategy* is the third component and it iterates between inclusion and elimination until a stopping criterion is satisfied. Finally

the fourth component is the check that the *symmetry requirement* mentioned above is satisfied. See Figure 4 for details. The main algorithm calls an internally defined subroutine that induces parents and children of T without symmetry correction (i.e., returns a set which is a subset of $EPC(T)$ and a superset of $PC(T)$). Note that in all references to $TPC(T)$ hereafter, due to generality of the stated algorithms and the process of convergence of $TPC(T)$ to $PC(T)$, $TPC(T)$ stands for just an approximation to $PC(T)$.

Also notice that the term "priority queue" in the schema of Figure 4 indicates an abstract data structure that satisfies the requirement that its elements are ranked by some priority function so that the highest-priority element is extracted first. $TPC(T)$ in step 1a of the GLL-PC-nonsym subroutine will typically be instantiated with the empty set when no prior knowledge about membership in $PC(T)$ exists. When the user does have prior knowledge indicating that X is a member of $PC(T)$, $TPC(T)$ can be instantiated to contain X. This prior knowledge may come from domain knowledge, experiments, or may be the result of running GLL-PC on variable X and finding that T is in $PC(X)$ when conducting local-to-global learning (Aliferis et al., 2009; Tsamardinos et al., 2006).

Steps #2, 3, 4 in GLL-PC-nonsym can be instantiated in various ways. Obeying a set of specific rules generates what we call "admissible" instantiations. These admissibility rules are given in Figure 5.

Theorem 2 *When the following sufficient conditions hold:*

a. There is a causal Bayesian network faithful to the data distribution P;

b. The determination of variable independence from the sample data D is correct;

c. Causal sufficiency in $\mathbf{V}$

any algorithmic instantiation of GLL-PC in compliance with the admissibility rules $\#1 - \#3$ *above will return the direct causes and direct effects of* T.

The proof is provided in the Appendix.

We note that the algorithm schema does not address various optimizations and does not address the issue of statistical decisions in finite sample. These will be discussed later. We also note that initialization of $TPC(T)$ in step 1a of the GLL-PC-nonsym function is arbitrary because correctness (unlike efficiency) of the algorithm is not affected by the initial contents of $TPC(T)$.

We next instantiate the GLL-PC schema to derive two pre-existing algorithms, interleaved HITON-PC with symmetry correction and MMPC with symmetry correction (Tsamardinos et al., 2006; Aliferis et al., 2003a; Tsamardinos et al., 2003b). Figure 6 depicts the instantiations needed to obtain interleaved HITON-PC.

The interleaved HITON-PC with symmetry correction algorithm starts with an empty set of candidates, then ranks variables for priority for inclusion in the candidate set by univariate association. It discards variables with zero univariate association. It then accepts each variable into $TPC(T)$. If any variable inside the candidate set becomes independent of the response variable T given some subset of the candidate set, then the algorithm removes that variable from the candidate set and never considers it again. In other words, the algorithm attempts to eliminate weakly relevant features from the $TPC(T)$ every time the $TPC(T)$ receives a new member. Iterations of insertion and elimination stop when there are no more variables to examine for inclusion. Once iterating has stopped, the candidate set is filtered using symmetry criterion. Finally, the candidate set is output. Because the

GLL-PC: Admissibility rules

1. The inclusion heuristic function should respect the following requirement:

   // *Admissibility rule #1*
   All variables $X \in PC(T)$ are eligible for inclusion in the candidate set $TPC(T)$ and each one is assigned a non-zero value by the ranking function. Variables with zero values are discarded and never considered again.

   Note that variables may be re-ranked after each update of the candidate set, or the original ranking may be used throughout the algorithm's operation.

2. The elimination strategy should satisfy the following requirement:

   // *Admissibility rule #2*
   All and only variables that become independent of the target variable T given any subset of the candidate set $TPC(T)$ are discarded and never considered again (whether they are inside or outside $TPC(T)$).

3. The interleaving strategy iterates inclusion and elimination any number of times provided that iterating stops when the following criterion is satisfied:

   //Admissibility rule #3
   At termination no variable outside the set $TPC(T)$ is eligible for inclusion and no variable in the candidate set can be removed at termination.

Figure 5: GLL-PC admissibility rules.

Interleaved HITON-PC with symmetry correction
Derived from GLL-PC with following instantiation specifics:

Initialization
$TPC(T) \leftarrow \varnothing$

Inclusion heuristic function

a. Sort in descending order the variables X in OPEN according to their pairwise association with T, i.e., $Assoc(X, T|\varnothing)$.
b. Remove from OPEN variables with zero association with T, *i.e.,* when $I(X, T|\varnothing)$
c. Insert at end of $TPC(T)$ the first variable in OPEN and remove it from OPEN

Elimination strategy

For each $X \in TPC(T)$
If $\exists \mathbf{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\mathbf{Z})$ remove X from $TPC(T)$

Interleaving strategy

Repeat
steps #2 and #3 of GLL-PC-nonsym
Until OPEN=$\varnothing$

Figure 6: Interleaved HITON-PC with symmetry correction as an instance of GLL-PC.

admissibility criteria are obeyed, the algorithm is guaranteed to be correct when the assumptions of Theorem 2 hold.

Below we prove that that admissibility rules are obeyed in interleaved HITON-PC with symmetry under the assumptions of Theorem 2:

1. Rule #1 (inclusion) is obeyed because all $PC(T)$ members have non-zero univariate association with T in faithful distributions.

2. Rule #2 (elimination) is directly implemented so it holds.

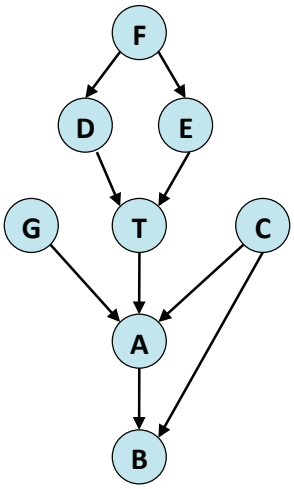


Figure 7: Bayesian network used to trace the algorithms.

Step of GLL-PC-nonsym	*Comments*	OPEN	*TPC(T)*
1	Initialize *TPC(T)* and OPEN	*{A, B, C, D, E, F, G}*	∅
2a (I)	Prioritize variables in OPEN for inclusion in *TPC(T)*	*{F, D, E, A, B, G, C}*	∅
2b (I)	Throw away non-eligible members of OPEN (*G* and *C*)	*{F, D, E, A, B}*	∅
2c (I)	Insert in *TPC(T)* the highest-priority variable in OPEN (*F*) and remove it from OPEN	*{D, E, A, B}*	*{F}*
3 (I)	Apply elimination strategy to *TPC(T)*: no effect	*{D, E, A, B}*	*{F}*
2 (II)	Insert the highest-priority variable (*D*) in *TPC(T)* and remove it from OPEN	*{E, A, B}*	*{F, D}*
3 (II)	Apply elimination strategy to *TPC(T)*: no effect	*{E, A, B}*	*{F, D}*
2 (III)	Insert the highest-priority variable (*E*) in *TPC(T)* and remove it from OPEN	*{A, B}*	*{F, D, E}*
3 (III)	Apply elimination strategy to *TPC(T)*: remove *F* since I(*T, F*|*{D,E}*)	*{A, B}*	*{D, E}*
2 (IV)	Insert the highest-priority variable (*A*) in *TPC(T)* and remove it from OPEN	*{B}*	*{D, E, A}*
3 (IV)	Apply elimination strategy to *TPC(T)*: no effect	*{B}*	*{D, E, A}*
2 (V)	Insert the highest-priority variable (*B*) in *TPC(T)* and remove it from OPEN	∅	*{D, E, A, B}*
3 (V)	Apply elimination strategy to *TPC(T)*: no effect	∅	*{D, E, A, B}*
4	Stop interleaving since OPEN = ∅	∅	*{D, E, A, B}*

Table 1: Trace of GLL-PC-nonsym(T) during execution of interleaved HITON-PC algorithm.

3. Rule #3 (termination) is obeyed because termination requires empty OPEN and thus eligible variables (i.e., members of $PC(T)$) outside $TPC(T)$ could only be previously discarded from OPEN or $TPC(T)$. Neither case can happen because of admissibility rules #1, #2 respectively. Similarly all variables in $TPC(T)$ that can be removed are removed because of admissibility rule #2.

A trace of the algorithm is provided below for data coming out of the example BN of the Figure 7. We assume that the network is faithful and so the conditional dependencies and indepen-

dencies can be read off the graph directly using the d-separation criterion. Consider that we want to find parents and children of the target variable T using interleaved HITON-PC with symmetry. Table 1 gives a complete trace of step 1 of the instantiated GLL-PC algorithm, that is, execution of GLL-PC-nonsym subroutine for variable T. The Roman numbers in the table refer to iterations of steps 2 and 3 in GLL-PC-nonsym.

Thus we have $TPC(T) = \{D,E,A,B\}$ by the end of GLL-PC-nonsym subroutine, so $\boldsymbol{U} = \{D,E,A,B\}$ in step 1 of GLL-PC. Next, in steps 2 and 3 we first run GLL-PC-nonsym for all $X \in \boldsymbol{U}$:

- GLL-PC-nonsym$(D) \rightarrow \{T,F\}$
- GLL-PC-nonsym$(E) \rightarrow \{T,F\}$
- GLL-PC-nonsym$(A) \rightarrow \{T,G,C,B\}$
- GLL-PC-nonsym$(B) \rightarrow \{A,C\}$

and then check symmetry requirement. Since $T \notin$ GLL-PC-nonsym(B), the variable B is removed from $\boldsymbol{U}$. Finally, the GLL-PC algorithm returns $\boldsymbol{U} = \{D,E,A\}$ in step 4.

Figure 8 shows how algorithm MMPC is obtained from GLL-PC. MMPC is also guaranteed to be sound when the conditions of Theorem 2 hold. Interleaving consists of iterations of just the inclusion heuristic function until OPEN is empty. The heuristic inserts into $TPC(T)$ the next variable F that maximizes the minimum association of variables in OPEN with T given all subsets of $TPC(T)$. In the algorithm, this minimum association of X with T conditioned over all subsets of $\boldsymbol{Z}$ is denoted by $\text{Min}_{\boldsymbol{Z}}\text{Assoc}(X,T|\boldsymbol{Z})$. The intuition is that we accept next the variable that despite our best efforts to be made conditionally independent of T (i.e., conditioned on all subsets of our current estimate $TPC(T)$) is still highly associated with T. The two main differences of the MMPC algorithm from interleaved HITON-PC are the more complicated inclusion heuristic function and the absence of interleaving of the inclusion-exclusion phases before all variables have been processed by the inclusion heuristic function. A set of optimizations and caching operations render the algorithm efficient; for complete details see Tsamardinos et al. (2006, 2003b).

Below we prove that admissibility rules are obeyed in MMPC with symmetry under the assumptions of Theorem 2:

1. Rule #1 (inclusion) is obeyed because all $PC(T)$ members have non-zero conditional association with T in faithful distributions.
2. Rule #2 (elimination) is directly implemented so it holds.
3. Rule #3 (termination) is obeyed because termination requires empty OPEN and thus eligible variables (i.e., members of $PC(T)$) outside $TPC(T)$ could only be previously discarded from OPEN or $TPC(T)$. Neither case can happen because of admissibility rules #1, #2 respectively. Similarly all variables in $TPC(T)$ that can be removed are removed because of admissibility rule #2.

We now introduce a new algorithm, semi-interleaved HITON-PC with symmetry correction, see Figure 9. Semi-interleaved HITON-PC operates like interleaved HITON-PC with one major difference: it does not perform a full variable elimination in $TPC(T)$ with each $TPC(T)$ expansion.

MMPC with symmetry correction

Derived from GLL-PC with following instantiation specifics:

Initialization

TPC(T) ← ∅

Inclusion heuristic function

a. Sort in descending order the variables X in OPEN according to $\text{Min}_{\boldsymbol{Z}}\text{Assoc}(X, T|\boldsymbol{Z})$ for $\boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$
b. Remove from OPEN variables X with zero association with T, given some $\boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$
c. Insert at end of *TPC(T)* the first variable in OPEN and remove it from OPEN

Elimination strategy

If OPEN=∅
  For each $X \in TPC(T)$
    If $\exists\, \boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\boldsymbol{Z})$ remove X from *TPC(T)*

Interleaving strategy

Repeat
  steps #2 and #3 of GLL-PC-nonsym
Until OPEN=∅

Figure 8: MMPC with symmetry correction as an instance of GLL-PC.

Semi-Interleaved HITON-PC with symmetry correction

Derived from GLL-PC with following instantiation specifics:

Initialization

TPC(T) ← ∅

Inclusion heuristic function

a. Sort in descending order the variables X in OPEN according to their pairwise association with T, i.e., $Assoc(X, T|\emptyset)$.
b. Remove from OPEN variables with zero association with T, *i.e.,* when $I(X, T|\emptyset)$
c. Insert at end of *TPC(T)* the first variable in OPEN and remove it from OPEN

Elimination strategy

If OPEN=∅
  For each $X \in TPC(T)$
    If $\exists\, \boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\boldsymbol{Z})$ remove X from *TPC(T)*
Else
  X ← last variable added to *TPC(T)* // in step 2 of GLL-PC-nonsym
  If $\exists\, \boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\boldsymbol{Z})$ remove X from *TPC(T)*

Interleaving strategy

Repeat
  steps #2 and #3 of GLL-PC-nonsym
Until OPEN=∅

Figure 9: Semi-interleaved HITON-PC with symmetry correction as an instance of GLL-PC.

On the contrary, once a new variable is selected for inclusion, it attempts to eliminate it and if successful it discards it without further attempted eliminations. If it is not eliminated, it is added to the end of the $TPC(T)$ and new candidates for inclusion are sought. Because the admissibility criteria are obeyed the algorithm is guaranteed to be correct under the assumptions of Theorem 2.

Below we prove that admissibility rules are obeyed in semi-interleaved HITON-PC with symmetry under the assumptions of Theorem 2:

1. Rule #1 (inclusion) is obeyed because all $PC(T)$ members have non-zero univariate association with T in faithful distributions.

Step of GLL-PC-nonsym	*Comments*	OPEN	*TPC(T)*
1	Initialize *TPC(T)* and OPEN	*{A, B, C, D, E, F, G}*	∅
2a (I)	Prioritize variables in OPEN for inclusion in *TPC(T)*	*{F, D, E, A, B, G, C}*	∅
2b (I)	Throw away non-eligible members of OPEN (*G* and *C*)	*{F, D, E, A, B}*	∅
2c (I)	Insert in *TPC(T)* the highest-priority variable in OPEN (*F*) and remove it from OPEN	*{D, E, A, B}*	*{F}*
3 (I)	Apply elimination strategy to *TPC(T)*: no effect	*{D, E, A, B}*	*{F}*
2 (II)	Insert the highest-priority variable (*D*) in *TPC(T)* and remove it from OPEN	*{E, A, B}*	*{F, D}*
3 (II)	Apply elimination strategy to *TPC(T)*: no effect	*{E, A, B}*	*{F, D}*
2 (III)	Insert the highest-priority variable (*E*) in *TPC(T)* and remove it from OPEN	*{A, B}*	*{F, D, E}*
3 (III)	Apply elimination strategy to *TPC(T)*: No effect	*{A, B}*	*{F, D, E}*
2 (IV)	Insert the highest-priority variable (*A*) in *TPC(T)* and remove it from OPEN	*{B}*	*{F, D, E, A}*
3 (IV)	Apply elimination strategy to *TPC(T)*: no effect	*{B}*	*{F, D, E, A}*
2 (V)	Insert the highest-priority variable (*B*) in *TPC(T)* and remove it from OPEN	∅	*{F, D, E, A, B}*
3 (V)	Apply elimination strategy to *TPC(T)*: remove *F* since I(*T, F*|*{D,E}*)	∅	*{D, E, A, B}*
4	Stop interleaving since OPEN = ∅	∅	*{D, E, A, B}*

Table 2: Trace of GLL-PC-nonsym(T) during execution of semi-interleaved HITON-PC algorithm.

2. Rule #2 (elimination) is directly implemented so it holds.

3. Rule #3 (termination) is obeyed because termination requires empty OPEN and thus eligible variables (i.e., members of $PC(T)$) outside $TPC(T)$ could only be previously discarded from OPEN or $TPC(T)$. Neither case can happen because of admissibility rules #1, #2 respectively. Similarly all variables in $TPC(T)$ that can be removed are removed because of admissibility rule #2.

A trace of the algorithm is provided below for data coming out of the example faithful BN of the Figure 7. Consider that we want to find parents and children of the target variable T using semi-interleaved HITON-PC with symmetry. Table 2 gives a complete trace of step 1 of the instantiated GLL-PC algorithm, that is, execution of GLL-PC-nonsym subroutine for variable T. The Roman numbers in the table refer to iterations of steps 2 and 3 in GLL-PC-nonsym.

Thus we have $TPC(T) = \{D, E, A, B\}$ by the end of GLL-PC-nonsym subroutine, so $\boldsymbol{U} = \{D, E, A, B\}$ in step 1 of GLL-PC. Next, in steps 2 and 3 we first run GLL-PC-nonsym for all $X \in \boldsymbol{U}$:

- GLL-PC-nonsym$(D) \rightarrow \{T, F\}$
- GLL-PC-nonsym$(E) \rightarrow \{T, F\}$
- GLL-PC-nonsym$(A) \rightarrow \{T, G, C, B\}$
- GLL-PC-nonsym$(B) \rightarrow \{A, C\}$

and then check symmetry requirement. Since $T \in$ GLL-PC-nonsym(B), the variable B is removed from $\boldsymbol{U}$. Finally, the GLL-PC algorithm returns $\boldsymbol{U} = \{D, E, A\}$ in step 4.

4.2 Discovery of the $MB(T)$ Set

As mentioned in Section 3 the $MB(T)$ contains all information sufficient for the determination of the conditional distribution of T : $P(T|MB(T)) = P(T|\boldsymbol{V} \setminus \{T\})$ and further, it coincides with the parents, children and spouses of T in any network faithful to the distribution (if any) under causal sufficiency. The previous subsection described a general family of algorithms to obtain the $PC(T)$ set, and so in order to find the $MB(T)$ one needs in addition to $PC(T)$, to also identify the spouses of T.

First notice that, approximating $MB(T)$ with $PC(T)$ and missing the spouse nodes may in theory discard very informative nodes. For example, suppose that X and T are two uniformly randomly chosen numbers in $[0,1]$ and that $Y = \min(1, X+T)$. Then, the only faithful network representing the joint distribution is $X \rightarrow Y \leftarrow T$, where X is the spouse of T. In predicting T, the spouse node X may reduce the uncertainty completely: conditioned on Y, T may become completely determined (when both X and T are less than 0.5). Thus, it theoretically makes sense to develop algorithms that identify the spouses in addition to the $PC(T)$, even though later in Section 5 we empirically determine that within the scope of distributions and problems tried, the $PC(T)$ resulted in feature subsets almost as predictive as the full $MB(T)$. In the companion paper (Aliferis et al., 2010) we also provide possible reasons explaining the good performance of $PC(T)$ versus $MB(T)$ for classification in practical tasks.

The theorem on which the algorithms in this family are based to discover the $MB(T)$ is the following:

Theorem 3 *In a faithful BN $\langle \boldsymbol{V}, G, P \rangle$, if for a triple of nodes X, T, Y in G, $X \in PC(Y)$, $Y \in PC(T)$, and $X \notin PC(T)$, then $X \rightarrow Y \leftarrow T$ is a subgraph of G iff $\neg I(X, T | \boldsymbol{Z} \cup \{Y\})$, for all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X, T\}$ (Spirtes et al., 2000).*

We distinguish two cases: (i) X is a spouse of T but it is also a parent or child, for example, $X \rightarrow T \rightarrow Y$ and also $X \rightarrow Y$. In this case, we cannot use the theorem above to identify Y as a collider and X as a spouse. But at the same time we do not have to: $X \in PC(T)$ and so it will be identified by GLL-PC. (ii) $X \in MB(T) \setminus PC(T)$ in which case we can use the theorem to locally discover the subgraph $X \rightarrow Y \leftarrow T$ and determine that X should be included in $MB(T)$.

We now introduce the GLL-MB in Figure 10. The admissibility requirement is simply to use an admissible GLL-PC instantiation.

For the identification of $PC(T)$ any method of GLL-PC can be used. Also, in step 5a we know such a $\boldsymbol{Z}$ exist since $X \notin PC(T)$ (by Theorem 1); this $\boldsymbol{Z}$ has been previously determined and is cached during the call to GLL-PC.

Theorem 4 *When the following sufficient conditions hold*

a. There is a causal Bayesian network faithful to the data distribution P;

b. The determination of variable independence from the sample data D is correct;

GLL-MB: Generalized Local Learning - Markov Blanket

1. *PC(T)* ← GLL-PC*(T)* // obtain *PC*(*T*) by running *GLL-PC* for variable *T*
2. For every variable $Y \in PC(T)$
   $PC(Y)$ ← GLL-PC(Y) // obtain $PC(Y)$ for every member Y of $PC(T)$
3. *TMB(T)* ← *PC(T)* // initialize *TMB(T)* with *PC*(*T*) members
4. $\boldsymbol{S} \leftarrow \{\cup_{Y \in PC(T)} PC(Y)\} \setminus \{PC(T) \cup \{T\}\}$ // these are the potential spouses
5. For every variable $X \in \boldsymbol{S}$
   a. Retrieve a subset $\boldsymbol{Z}$ s.t. I(X, T | $\boldsymbol{Z}$) // subset was identified and stored in steps 1 and 2
   b. For every variable $Y \in PC(T)$ *s.t.* $X \in PC(Y)$ // Y is a potential common child of T and X
   c. If $\neg$I(X, T| $\boldsymbol{Z} \cup \{Y\}$) // X is a spouse
   d. Insert X into *TMB(T)*
6. Optionally: Eliminate from *TMB(T)* predictively redundant members using a backward wrapper approach.
7. Return *TMB*(*T*)

Figure 10: GLL-MB: Generalized Local Learning - Markov Blanket algorithm.

c. Causal sufficiency in $\boldsymbol{V}$

any algorithmic instantiation of GLL-MB in compliance with the admissibility rule will return $MB(T)$ *(with no need for step 6).*

The proof is provided in the Appendix.

A new Markov blanket algorithm, semi-interleaved HITON-MB, can be obtained by instantiating GLL-MB (Figure 10) with the semi-interleaved HITON-PC algorithm with symmetry correction for GLL-PC.

Semi-interleaved HITON-MB is guaranteed to be correct under the assumptions of Theorem 4, hence the only proof of correctness needed is the proof of correctness for semi-interleaved HITON-PC with symmetry (which was provided earlier).

A trace of the semi-interleaved HITON-MB algorithm for data coming out of the example faithful BN of the Figure 7 follows below. Please refer to Figure 10 for step numbers. Consider that we want to find Markov blanket of T. In step 1, we find $PC(T) = \{D, E, A\}$. Then in step 2 we find $PC(X)$ for all $X \in PC(T)$:

- $PC(D) = \{T, F\}$,
- $PC(E) = \{T, F\}$,
- $PC(A) = \{T, G, C, B\}$,

In step 3 we initialize $TMB(T) \leftarrow \{D, E, A\}$. The set $\boldsymbol{S}$ in step 4 contains the following variables: $\{F, G, C, B\}$. In step 5 we loop over all members of $\boldsymbol{S}$ to find spouses of T. Let us consider each variable separately:

- Loop for $X = F$: In step 5a we retrieve a subset $\boldsymbol{Z} = \{D, E\}$ that renders $X = F$ independent of T. In step 5b we loop over all potential common children of F and T, that is, $Y = D$ and $Y = E$. When we consider $Y = D$, we find that $X = F$ is independent of T given $\boldsymbol{Z} \cup \{Y\} = \{D, E\}$ and thus do not include F in $TMB(T)$ in step 5d. When we consider $Y = E$, we also do not include F in $TMB(T)$ in step 5d.

- Loop for $X = G$: In step 5a we retrieve a subset $\boldsymbol{Z} = \emptyset$ that renders $X = G$ independent of T. In step 5b we loop over all potential common children of G and T, that is, variable $Y = A$. We find that $X = G$ is dependent on T given $\boldsymbol{Z} \cup \{Y\} = \{A\}$ and thus include G in $TMB(T)$ in step 5d.

- Loop for $X = C$: In step 5a we retrieve a subset $\boldsymbol{Z} = \emptyset$ that renders $X = C$ independent of T. In step 5b we loop over all potential common children of C and T, that is, variable $Y = A$. We find that $X = C$ is dependent on T given $\boldsymbol{Z} \cup \{Y\} = \{A\}$ and thus include C in $TMB(T)$ in step 5d.

- Loop for $X = B$: In step 5a we retrieve a subset $\boldsymbol{Z} = \{A, C\}$ that renders $X = B$ independent of T. In step 5b we loop over all potential common children of B and T, that is, variable $Y = A$. We find that $X = B$ is independent of T given $\boldsymbol{Z} \cup \{Y\} = \{A, C\}$ and thus do not include G in $TMB(T)$ in step 5d.

By the end of step 5, we have $TMB(T) = \{D, E, A, G, C\}$. Notice that it is the true $MB(T)$. In step 6 we perform wrapping to remove members of $TMB(T)$ that are redundant for classification. Let us assume that we used a backward wrapping procedure that led to removal of variable G, for example because omitting this variable does not increase classification loss. Thus, we have $TMB(T) = \{D, E, A, C\}$ in step 7 when the algorithm terminates.

The above algorithm specifications and proofs demonstrate that it is relatively straightforward to derive correct algorithms and prove their correctness using the GLL framework. It is also straightforward to derive relaxed versions (for example non-symmetry corrected versions of interleaved and semi-interleaved HITON and MMPC) which trade-off correctness for improved tractability.

4.3 Computational Complexity

The complexity of all algorithms presented depends on the time for the tests of independence and measures of associations. For the G^2 test of independence for discrete variables, for example, we use in reported experiments an implementation linear to the sample size and exponential to the number of variables in the conditional set. However, because the latter number is small in practice, tests are relatively efficient. Faster implementations exist that only take time $n \log(n)$ to the number n of training instances, independent of the size of the conditioning set. Also, advanced data structures (Moore and Wong, 2003) can be employed to improve the time complexity (see Tsamardinos et al. 2006 for details on the implementation of the tests). In reported experiments we also implement the measure of association $\text{Assoc}(X, T|\boldsymbol{Z})$ to be the negative *p-value* returned by the test $I(X, T|\boldsymbol{Z})$ and so it takes exactly the same time to compute as a test of independence. In the following discussion, we consider the complexity of the algorithms in terms of the number of tests and measures of association they perform.

The number of tests of the GLL-PC algorithm in Figure 4 depends on several factors. These are the inclusion heuristic efficiency in approximating the $PC(T)$, the time required by the inclusion heuristic, and the size of the $PC(T)$ which is a structural property of the problem to solve. Interleaved-HITON-PC (algorithm in Figure 6) for example, will sort the variables using $|\boldsymbol{V}|$ measures of associations. Subsequently, it will perform a test $I(X, T|\boldsymbol{Z})$ for all subsets of the largest $TPC(T)$ in any iteration of interleaving of the inclusion-exclusion steps. With appropriate caching a test will never have to be repeated. Thus, assuming the largest size of the $TPC(T)$ is in the order of the $PC(T)$, the complexity of the GLL-PC-nonsym subroutine is $O(|\boldsymbol{V}|2^{|PC(T)|})$. In step 3, it will

execute the GLL-PC-nonsym subroutine again for all $X \in TPC(T)$. Assuming each neighborhood of X is about the same as the $PC(T)$, when checking the symmetry condition, the algorithm will perform another $O(|\boldsymbol{V}||PC(T)|2^{|PC(T)|})$tests.

To identify $MB(T)$ by the GLL-MB algorithm in Figure 10 we first need to initialize subset $\boldsymbol{S}$. Assuming all neighborhoods are about the same size (i.e., equal to $|PC(T)|$), the total complexity to find the set $\boldsymbol{S}$ is $O(|\boldsymbol{V}||PC(T)|^2 2^{|PC(T)|})$ since we call GLL-PC for each member of the $PC(T)$. In fact, several optimizations can reduce this order to $O(|\boldsymbol{V}||PC(T)|2^{|PC(T)|})$ but we will not elaborate further in this paper. In step 5, in the worst case we perform a single test for each node in $\boldsymbol{S}$ and each node in $PC(T)$ for a total of at most $O(|PC(T)|^2)$ tests (the subset $\boldsymbol{Z}$ in step 5a is cached and retrieved). So the order of the algorithm is $O(|\boldsymbol{V}||PC(T)|^2 2^{|PC(T)|})$ tests given the structural assumptions above.

All other algorithmic instantiations of the template in this section have similar complexity.

At this point it is worth noting a number of polynomial approximation algorithms in the literature that increase efficiency without sacrificing quality to a large degree. The identification of a subset $\boldsymbol{Z}$ in step 3 of the GLL-PC-nonsym subroutine as described in algorithm instantiations of GLL-PC is a step exponential to the size of the $TPC(T)$; however, one could attempt to discover it in a greedy fashion, for example by starting with the empty set and adding to $\boldsymbol{Z}$ the variable decreasing the association with T the most. These ideas started with the TPDA algorithm (Cheng et al., 2002a) and were further explored in Brown et al. (2005). Similar improvements can be applicable to inclusion strategy.

For the above analysis we assumed that all tests $I(X,T|\boldsymbol{Z})$ can or should be performed and return the correct results. However, in the next sub-section we discuss how the statistical decisions of independence or dependence are made; these decisions severely affect the complexity of the algorithms as well.

4.4 Dealing with Statistical Decisions

The quality of the algorithms in practice highly depends on their ability to statistically determine whether $I(X,T|\boldsymbol{Z})$ or $\neg I(X,T|\boldsymbol{Z})$ (equivalently whether $\text{Assoc}(X,T|\boldsymbol{Z})$ is zero or non-zero) for a pair of variables X and T and a set of variables Z. The test $I(X,T|\boldsymbol{Z})$ is implemented as a statistical hypothesis test with null hypothesis H_0: X and T are independent given $\boldsymbol{Z}$. A *p-value* corresponding to this test statistic's distribution expresses the probability of seeing the same or more extreme (i.e., indicative of dependence) test statistic values when sampling from distributions where H_0 is true. If the *p-value* is lower than a given threshold (i.e., significance level "alpha") α, then we consider the independence hypothesis to be improbable and reject it. Thus, for a sufficiently low *p-value* we accept *dependence*. If however, the *p-value* is not low enough to provide confidence in rejecting H_0 then there are two possibilities:

a) H_0 actually holds, that is, the variables are indeed conditionally independent.

b) H_0 does not hold, the variables are conditionally dependent but we cannot confidently reject H_0.

The reasons for b) are that either the dependence is weak relatively to the available sample to be detected (in order words, we have low probability to reject the null hypothesis H_0 when it does not hold, that is, low statistical power), or we are using the wrong statistical test for this type of dependency. In essence, we would like to distinguish between the following cases:

a) $I(X,T|\boldsymbol{Z})$ holds with high-probability

b) $\neg I(X,T|\boldsymbol{Z})$ holds with high-probability

c) Undetermined case given the available sample

To deal with case c) in our implementations we take the following approach, introduced by Spirtes et al. (2000): we consider that we are facing case c) if there is no sufficient power according to a *reliability criterion*. In our implementations this criterion depends on parameter *h-ps*. The criterion dictates that if and only if we have at least *h-ps* sample instances per number of cells (i.e., number of parameters to be estimated) in the contingency tables for the discrete statistical tests then the test is reliable.

Once a test is deemed unreliable an algorithm needs to decide how to handle the corresponding statistical decision. For example, the PC algorithm for global causal discovery (Spirtes et al., 2000) considers that given no other evidence, all variables are dependent with each other. That is, a pair of variables is always connected by an edge in the graph unless a subset $\boldsymbol{Z}$ is discovered that renders them conditionally independent.

The implementations of GLL instantiations in the present paper do not perform an unreliable test either. However, ignoring unreliable tests with 0-order conditioning test (i.e., univariate tests) is equivalent to assuming $I(X,T|\boldsymbol{Z})$ whereas ignoring unreliable tests with higher-order conditioning test (i.e., conditioning sets with 1 or more conditioning variables) is equivalent to assuming $\neg I(X,T|\boldsymbol{Z})$ as far as this unreliable test is concerned (because the final judgment on independence, is deferred to reliable, typically lower-order tests). Thus, given no evidence of dependence, we assume the unreliable tests to return $I(X,T|\boldsymbol{Z})$. The different treatment of the PC implementation leads to problems as discussed in Tsamardinos et al. (2006) pointing to the importance of this implementation aspect of the algorithms.

Another practical implementation issue arises when prior knowledge, experiments, or domain substantive knowledge ensures that a variable X is in $PC(T)$ or that X is not in $PC(T)$. In such cases the algorithm can be modified to "lock" X inside or outside $TPC(T)$ respectively in order to avoid the possibility that errors in statistical decisions will counter previously validated knowledge and possibly propagate more statistical decision errors.

In addition to *h-ps*, a second restriction on the conditioning set size is provided by parameter *max-k*. This parameter places an absolute limit on the number of elements in a conditioning set size, *without reference to available sample size*. As such *max-k* participates in the reliability judgment but also restricts the computational complexity of the algorithms by trading off computational complexity for fit to data.

Specifically first consider that more variables than the actual $PC(T)$ could be output by the algorithm. A variable X that becomes independent of T only when we condition on $\boldsymbol{Z}$, with $|\boldsymbol{Z}| >$ *max-k* could enter the $TPC(T)$ and will not be removed afterwards. For example, if *max-k* $= 1$, then variable F in Figure 7 cannot be d-separated from T given any $\boldsymbol{Z}$ with $|\boldsymbol{Z}| \leq 1$. Thus, the reliability criterion may increase the number of tests performed, since these depend on the size of the $TPC(T)$. On the other hand, the criterion forces certain tests not to be performed, specifically those whose conditioning set $\boldsymbol{Z}$ size is larger than *max-k*. Thus, since only $\binom{TPC(T)}{max\text{-}k}$ subsets are tested out of all possible $2^{|TPC(T)|}$ ones, the complexity of the algorithm GLL-PC-nonsym now becomes $O(|\boldsymbol{V}||TPC(T)|^{max\text{-}k})$, that is, polynomial of order *max-k*.

The parameters *h-ps* and *max-k* are user-specified or, alternatively, optimized automatically by cross-validation, or optimized for a whole domain. The role and importance of these two parameters, especially with respect to quality of statistical decisions, is explored in detail in the companion paper (Aliferis et al., 2010). Finally, because the quality of statistical decisions is not addressed in the proofs of correctness provided earlier, it was implicitly assumed that whenever sufficient sample size is provided to the algorithms statistical decisions are reliable.

A recent treatment that specifically addresses the role of statistical decisions in finite sample is presented in Tsamardinos and Brown (2008a). In this work, a bound of the p-value of the existence of an edge is provided; the bound can be used to control the False Discovery Rate of the identification of the $PC(T)$ or all the edges in a network.

5. Comparative Evaluation of Local Causal and Non-Causal Feature Selection Algorithms in Terms of Feature Selection Parsimony and Classification Accuracy

In the present section we examine the ability of GLL algorithms to discover compact sets of features with as high classification performance as possible for each data set and compare them with other local causal structure discovery methods as well as non-causal feature selection methods.

In order to avoid bias in error estimation we apply nested N-fold cross-validation. The inner loop is used to try different parameters for the feature selection and classifier methods while the outer loop tests the best configuration on an independent test set. Details are given in Statnikov et al. (2005b), Dudoit and van der Laan (2003) and Scheffer (1999).

All experiments discussed in this section and elsewhere in this paper were conducted on ACCRE (Advanced Computing Center for Research and Education) High Performance Computing system at Vanderbilt University. The ACCRE system consists of 924 x86 processors (the majority of which 2 GHz) and 668 PowerPC processors (2.2 GHz) running 32 and 64-bit Linux OS. The overall computational capacity of the cluster is approximately 6 TFLOPS. For preliminary and exploratory experiments we used a smaller cluster of eight 3.2 GHz x86 processors.

The evaluated algorithms are listed in the Appendix Tables 5-7. They were chosen on the basis of prior independently published results showing their state-of-the-art performance and applicability to the range of domains represented in the evaluation data sets. We compare several versions of GLL, including parents and children (PC) and Markov blanket (MB) inducers. Whenever we refer to HITON-PC algorithm in this paper, we mean semi-interleaved HITON-PC without symmetry correction, unless mentioned otherwise. Also, other GLL algorithms evaluated do not have symmetry correction unless mentioned otherwise. Finally, unless otherwise noted, GLL-MB does not implement a wrapping step.

Tables 8-9 in the Appendix present the evaluation data sets. The data sets were chosen on the basis of being representative of a wide range of problem domains (biology, medicine, economics, ecology, digit recognition, text categorization, and computational biology) in which feature selection is essential. These data sets are challenging since they have a large number of features with small-to-large sample sizes. Several data sets used in prior feature selection and classification challenges were included. All data sets have a single binary target variable.

To perform imputation in data sets with missing values, we applied a non-parametric nearest neighbor method (Batista and Monard, 2003). Specifically, this method imputes each missing value of a variable with the present value of the same variable in the most similar instance according to Euclidian distance metric. Discretization in non-sparse continuous data sets was performed by a

univariate method (Liu et al., 2002) implemented in *Causal Explorer* (Aliferis et al., 2003b). For a given continuous variable, the method considers many binary and ternary discretization thresholds (by means of a sliding window) and chooses the one that maximizes statistical association with the target variable. In sparse continuous data sets, discretization was performed by assigning value 1 to all non-zero values. All variables in each data set were also normalized to be in [0, 1] range to facilitate classification by SVM and KNN. All computations of statistics for the preprocessing steps were performed based on training data only to ensure unbiased classification error estimation. Statistical comparison between algorithms was done using two-sided permutation test (with 10,000 permutations) at 5% alpha level (Good, 2000). The null hypothesis of this test is that algorithms perform the same.

Both polynomial SVMs and KNN were used for building classifiers from each selected feature set. In complementary experiments, the *native* classifier for each one of several feature selection methods (LARS-EN, L0, and RFVS) was used and its performance was compared against classifiers induced by SVMs and KNN. For SVMs, the misclassification cost C and kernel degree d were optimized over values [1, 10, 100] and [1, 2, 3, 4], respectively. For KNN, the number of nearest neighbors k was optimized over values [1,...,min(1000, number of instances in the training set)]. All optimization was conducted in nested cross-validation using training data only, while the testing data was used only once to obtain an error estimate for the final classifier. We used the libSVM implementation of SVM classifiers (Fan et al., 2005) and our own implementation of KNN.

We note that use of SVMs and KNN does not imply that GLL methods are designed to be filters for these two algorithms only, or that the algorithm comparison results narrowly apply to these two classifiers. Rather as explained in Section 2.2, GLL algorithms provide performance guarantees as long as the classifier used has universal approximator properties. SVMs and KNN are two exemplars of practical and scalable such methods in wide use. We also emphasize that selecting features with a wrapper or embedded feature selection method that *is not* SVM or KNN specific is not affected by the inductive bias mismatch because such mismatch is affecting performance only when the classifier used is "handicapped" relative to the native classifier (Tsamardinos and Aliferis, 2003; Kohavi and John, 1997). We provide experimental data substantiating this point in the Appendix Table 10 (and Table S1 in the online supplement) where we compare classification performance of RFVS, LARS-EN, and L0 with features selected by each corresponding method to the classification performance of SVMs and KNN using the same features. It is shown that SVM predictivity matches, whereas KNN predictivity compares favorably, with the classifiers that are native to each feature selector. On the other hand, the choice of SVMs and KNN provides several advantages to the research design of the evaluation: (a) the same classifiers can be used with all data sets removing a possible confounder in the evaluation; (b) they can be used without feature selection (i.e., full variable set) to give a reference point of predictivity under no feature selection (that in practice is as good as empirically optimal predictivity especially when using SVMs); (c) they can be used when sample size is smaller than number of variables; (d) prior evidence suggests that they are suitable classifiers for the domains; (e) they can be executed in tractable time using nested cross-validation as required by our protocol.

In all cases when an algorithm had not terminated within 2 days of single-CPU time per run on a single training set (including optimization of the feature selector parameters) and in order to make the experimental comparison feasible with all methods and data sets in the study, we deemed it to be impractical and terminated it. While the practicality of spending more than two days of single-CPU time on a single training set can be debated, we believe that use of slower algorithms in

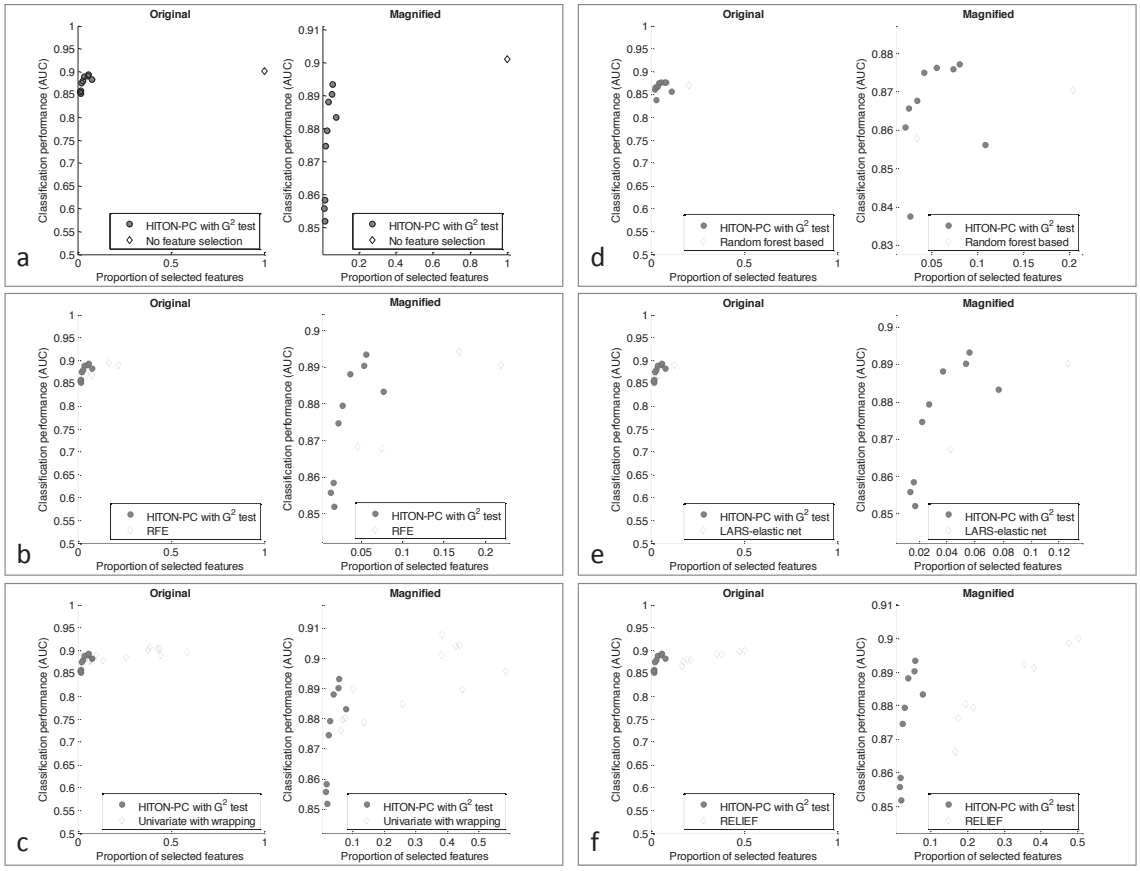


Figure 11: Causal Feature Selection Returns More Compact Feature Sets Than Non-Causal Feature Selection—Comparison of each algorithmic family with semi-interleaved HITON-PC with G^2 test. HITON-PC is executed with 9 different configurations: $\{max\text{-}k = 1, \alpha = 0.05\}$, $\{max\text{-}k = 2, \alpha = 0.05\}$, $\{max\text{-}k = 3, \alpha = 0.05\}$, $\{max\text{-}k = 4, \alpha = 0.05\}$, $\{max\text{-}k = 1, \alpha = 0.01\}$, $\{max\text{-}k = 2, \alpha = 0.01\}$, $\{max\text{-}k = 3, \alpha = 0.01\}$, $\{max\text{-}k = 4, \alpha = 0.01\}$, and a configuration that selects one of the above parameterizations by nested cross-validation. Results shown are averaged across all real data sets where both HITON-PC with G^2 test and an algorithmic family under consideration are applicable and terminate within 2 days of single-CPU time per run on a single training set. Multiple points for each algorithm correspond to different parameterizations/configurations. See Appendix Tables 5- 7 for detailed list of algorithms. The left graph has x-axis (proportion of selected features) ranging from 0 to 1 and y-axis (classification performance AUC) ranging from 0.5 to 1. The right graph has the same data, but the axes are magnified to see the details better. This figure is continued in Figures 12 and 13.

practice is problematic due to the following reasons: (i) in the context of N-fold cross-validation the total running time is at least N times longer (i.e., >20 days single-CPU time); (ii) the analyst does not know whether the algorithm will terminate within a reasonable amount of time, and (iii) when quantification of uncertainty about various parameters (e.g., estimating variance in error estimates via bootstrapping) is needed the analysis becomes prohibitive regardless of analyst flexibility and

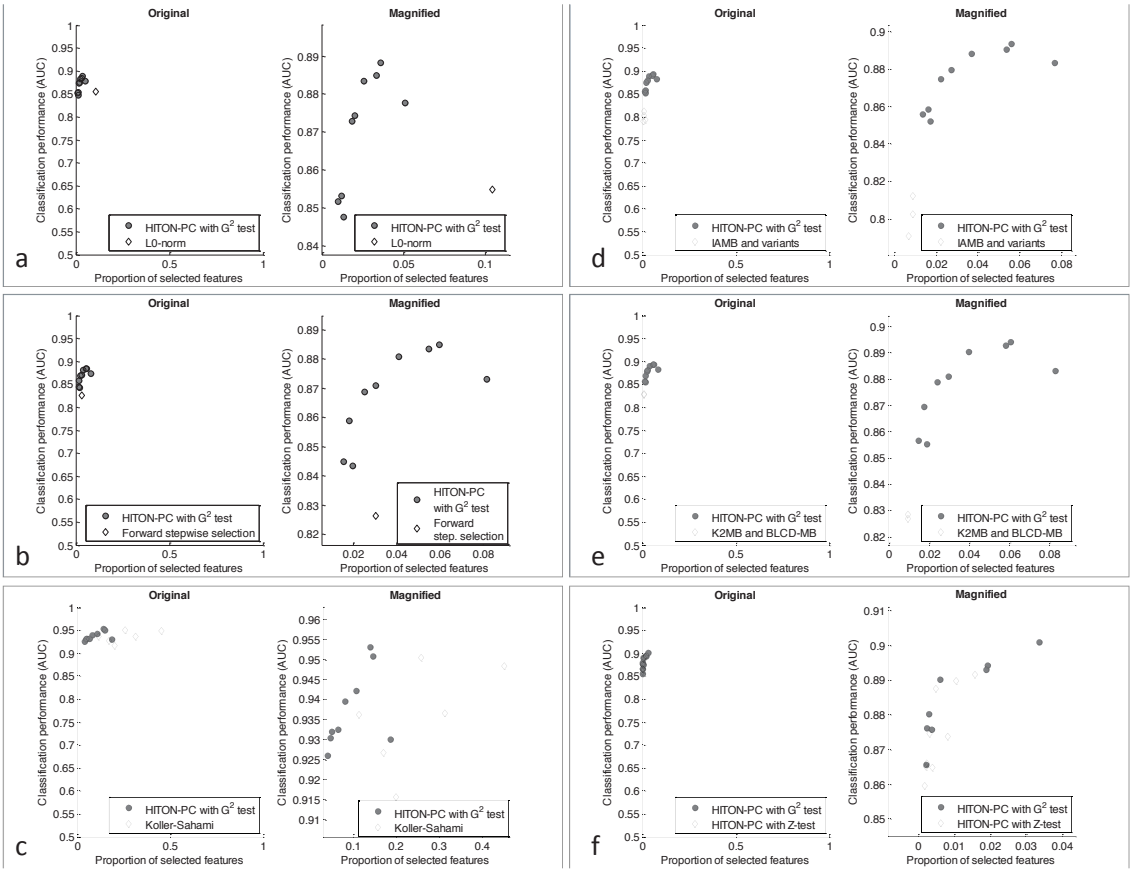


Figure 12: Continued from Figure 11.

computational resources. When comparing a pair of algorithms we consider only the data sets where both algorithms terminate within the allotted time.

We evaluate the algorithms using the following metrics:

1. Number of features selected;
2. Proportion of features selected relative to the original number of features (i.e., prior to feature selection);
3. Classification performance measured as area under ROC curve (AUC) (Fawcett, 2003);
4. Feature selection time in minutes.[2]

Figure 11 compares each evaluated algorithm to semi-interleaved HITON-PC with G^2 test as a reference performance for GLL, in the two-dimensional space defined by proportion of selected features and classification performance by SVM (results for KNN are similar and are available in

2. In all cases we used the implementations provided by the authors of methods, or state-of-the-art implementations, and thus reported time should be considered representative of what practitioners can expect in real-life with equipment and data similar to the ones used in the present study. However, we note that running times should be interpreted as indicative only since numerous implementation details and possible optimizations as well as computer platform discrepancies can affect results.

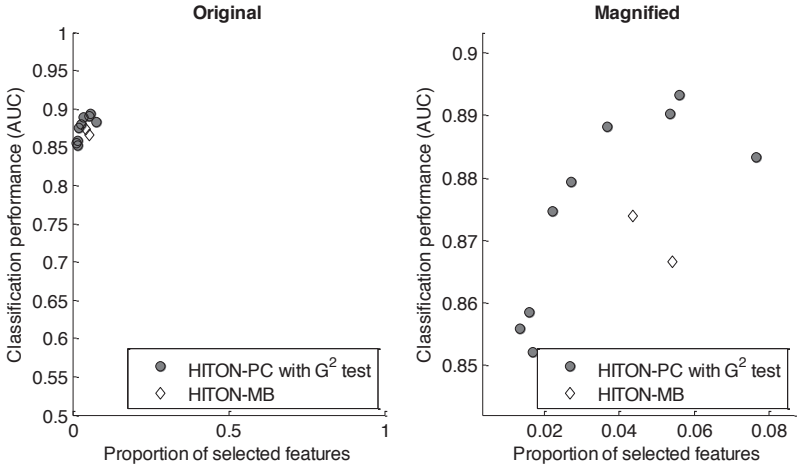


Figure 13: Continued from Figure 12.

Table S5 in the online supplement). As can be seen in the figure (and also in Figure S1 of the online supplement), GLL algorithms typically return much more compact sets than other methods. More compact results are provided by versions that induce the PC set rather than the MB for obvious reasons. Out of GLL methods, the most compact sets are returned when the Z-test is applicable (continuous data) compared to G^2 test (discrete or discretized data). As seen in Tables S2-S3 in the online supplement, depending on the parameterization of GLL, compactness varies. However, regardless of configuration, both GLL and other local causal methods (i.e., IAMB, BLCD-MB, FAST-IAMB, K2MB) with the exception of Koller-Sahami are typically more compact than non-causal feature selection methods (i.e., univariate methods with backward wrapping, RFE, RELIEF, Random Forest-based Variable Selection, L0, and LARS-EN). Forward stepwise selection and some configurations of LARS-EN, Random Forest-based Variable Selection, and RFE are often very parsimonious, however their parsimony varies greatly across data sets. Notice that whenever an algorithm variant employed statistical comparison among feature sets (in particular non-causal ones), it improved compactness (Figure S1 and Tables S2-S3 in the online supplement). Table 3 gives statistical comparisons of compactness between one reference GLL algorithm (semi-interleaved HITON-PC with G^2 test and cross-validation-based optimization of the algorithm parameters) and 43 non-GLL algorithms and variants (including no feature selection). In 21 cases the GLL reference method gives statistically significantly more compact sets compared to all other methods, in 16 cases parsimony is not statistically distinguishable, and in 6 cases HITON-PC gives less compact feature sets. These 6 cases correspond strictly to non-GLL causal feature selection algorithms and at the expense of severe predictive suboptimality (0.06 to 0.10 AUC) relative to the reference GLL method (see Tables S4-S5 in the online supplement).

5.1 Compactness Versus Classification Performance

Compactness is only one of the two requirements for solving the feature selection problem. A maximally compact algorithm that does not achieve optimal predictivity does not solve the feature selection problem. Figure 11 examines the trade-off of compactness and SVM predictivity (results for KNN are similar and available in Table S5 in the online supplement). The best possible point for each graph is at the upper left corner. For ease of visualization the results are plotted for each

	Predicitivity		*Reduction*	
Feature selection method	P-value	Nominal winner	P-value	Nominal winner
No feature selection	0.1890	Other	**<0.0001**	HITON-PC
RFE: 4 variants	0.9754	Other	**0.0046**	HITON-PC
	0.8030	Other	**0.0042**	HITON-PC
	0.1312	HITON-PC	0.3634	HITON-PC
	0.1008	HITON-PC	0.6816	Other
UAF-KruskalWallis-SVM: 4 variants	0.2248	Other	**0.0028**	HITON-PC
	0.0098	Other	**0.0004**	HITON-PC
	1.0000	HITON-PC	0.1414	HITON-PC
	0.3232	HITON-PC	0.3998	HITON-PC
UAF-Signal2Noise-SVM: 4 variants	0.0710	Other	**0.0018**	HITON-PC
	0.0752	Other	**0.0030**	HITON-PC
	0.4420	HITON-PC	0.7850	HITON-PC
	0.2820	HITON-PC	0.6604	HITON-PC
UAF-Neal-SVM: 4 variants	0.5046	Other	**<0.0001**	HITON-PC
	0.9782	HITON-PC	**<0.0001**	HITON-PC
	0.6980	HITON-PC	**0.0044**	HITON-PC
	0.3806	HITON-PC	**0.0186**	HITON-PC
Random Forest Variable Selection: 2 variants	0.6064	HITON-PC	0.3252	HITON-PC
	0.5050	HITON-PC	0.1338	Other
LARS-Elastic Net: 2 variants	1.0000	Other	0.1112	HITON-PC
	0.0832	HITON-PC	0.5216	Other
RELIEF: 8 variants	0.2032	Other	**<0.0001**	HITON-PC
	0.9362	Other	**<0.0001**	HITON-PC
	0.4388	Other	**0.0014**	HITON-PC
	0.8432	Other	**0.0010**	HITON-PC
	0.4290	HITON-PC	**0.0108**	HITON-PC
	0.3114	HITON-PC	0.0518	HITON-PC
	0.4424	HITON-PC	0.0706	HITON-PC
	0.2748	HITON-PC	**0.0404**	HITON-PC
L0-norm	**0.0258**	HITON-PC	0.1942	HITON-PC
Forward Stepwise Selection	**0.0028**	HITON-PC	0.2758	Other
Koller-Sahami: 6 variants	0.7506	HITON-PC	**<0.0001**	HITON-PC
	0.6234	HITON-PC	**<0.0001**	HITON-PC
	0.6278	HITON-PC	**<0.0001**	HITON-PC
	<0.0001	HITON-PC	**<0.0001**	Other
	0.1278	HITON-PC	0.3856	HITON-PC
	0.1236	HITON-PC	**<0.0001**	HITON-PC
IAMB: 3 variants	**<0.0001**	HITON-PC	**<0.0001**	Other
	<0.0001	HITON-PC	**<0.0001**	Other
	<0.0001	HITON-PC	0.1202	Other
K2MB	**<0.0001**	HITON-PC	**<0.0001**	Other
BLCD-MB	**<0.0001**	HITON-PC	**<0.0001**	Other
FAST-IAMB	**<0.0001**	HITON-PC	**<0.0001**	Other

Table 3: Statistical comparison via permutation test (Good, 2000) of 43 non-GLL algorithms (including no feature selection) to the reference GLL algorithm (semi-interleaved HITON-PC with G^2 test and cross-validation-based optimization of the algorithm parameters by SVM classifier) in terms of SVM predictivity and parsimony. Each non-GLL algorithm compared to HITON-PC in each row is denoted by "Other". Bolded p-values are statistically significant at 5% alpha.

algorithmic family separately. To avoid overfitting and to examine robustness of various methods to parameterization we did not select the best performing configuration, but plotted all of them. Notice that some algorithms did not run on all 13 real data sets (i.e., algorithms with Fisher's Z-test are applicable only to continuous data, while some algorithms did not terminate within 2 days of single-CPU time per run on a single training set). For such cases, we plotted results only for data sets where the algorithms were applicable and the results for HITON-PC correspond to the same data sets. As can be seen, GLL algorithms that induce PC sets dominate both other causal and non-causal feature

selection algorithms. This is also substantiated in Table 3 (and Table S7 in the online supplement that provides results for KNN classifier) that gives statistical comparisons of predictivity between the reference GLL algorithm and all 43 non-GLL algorithms and variants (including no feature selection). In 9 cases the GLL reference method gives statistically significantly more predictive sets compared to all other methods, in 33 cases predictivity is not statistically distinguishable, and in 1 case GLL gives less predictive feature sets (however the magnitude of the GLL suboptimal predictivity is only 0.018 AUC on average, whereas the difference in compactness is more than 33% features selected on average).

The overall performance patterns of combined predictivity and parsimony are highly consistent with Markov blanket induction theory (Section 2.2) which predicts maximum compactness and optimal classification performance when using the MB. Different instantiations of the GLL method give different trade-offs between predictivity and parsimony (details and statistical comparisons to the reference method are provided in online supplement Tables S2-S6 and S8).

In the companion paper (Aliferis et al., 2010), we examine in detail conditions under which PC induction can give optimal classification performance (the empirical illustration is provided in Figure 13). The comparison of HITON-PC with G^2 test and HITON-PC with Z-test reveals that both statistics perform similarly, while the latter (where it is applicable) does not require discretization of continuous data that can simplify data analysis significantly (see Figure 12 and statistical comparisons in Table S9 in the online supplement). In Table S10 of the online supplement we provide statistical comparisons of non-GLL causal feature selection methods in terms of predictivity and parsimony. K2MB, BLCD-MB, IAMB, and FAST-IAMB rather unexpectedly perform statistically indistinguishably in terms of predictivity and parsimony. Since BLCD-MB differs from K2MB by an additional backward elimination step, this implies that this step rarely results in elimination of features in the real data sets tested.

5.2 Analysis of Running Times

Table S6 in the online supplement gives detailed running times for all feature selection experiments. Major observations include that: (i) univariate methods, RELIEF, RFE, LARS-EN are in general the fastest ones, (ii) Koller-Sahami is probably the slowest method since it does not terminate on several data sets within the allotted time limit, (iii) FAST-IAMB is two orders of magnitude faster than IAMB on the average, and (iv) GLL algorithms are practical for very high-dimensional data (e.g., in the Thrombin data set with $>$ 100,000 features GLL-PC requires 10 to 52 minutes single-CPU time depending on fixed-parameter configuration, and less than 3 hours when GLL-PC is automatically optimized by cross-validation).

In conclusion, the GLL reference algorithm dominates most feature selection methods in predictivity and compactness. Some non-GLL causal methods are more parsimonious than the reference GLL method at the expense of severe classification suboptimality. One univariate method exhibits slightly higher predictivity but with severe disadvantage in parsimony. No feature selection method achieves equal or better compactness with equal or better classification performance than GLL.

6. Comparative Evaluation of Markov Blanket Induction, Local Causal Neighborhood and Other Non-Causal Algorithms for Local Structure Discovery

In the present section we study the ability of GLL algorithms to discover local causal structure (in the form of parent and children sets and Markov blankets) and compare them with other local

structure discovery methods as well as non-causal feature selection. While many researchers apply feature selection techniques strictly to improve the cost and effectiveness of classification, in many fields researchers routinely apply feature selection in order to gain insights *about the causal structure of the domain*. A frequently encountered example is in bioinformatics where a plethora of feature selection methods are applied in high-throughput genomic and proteomic data to discover biomarkers suitable for new drug development, personalizing medical treatments, and orienting subsequent experimentation (Zhou et al., 2002; Li et al., 2001; Holmes et al., 2000; Eisen et al., 1998). It is thus necessary to test the appropriateness of various feature selection techniques for causal discovery, not just classification.

In order to compare the performance of the tested techniques for causal discovery, we simulate data from known Bayesian networks and also use resimulation, whereby real data is used to elicit a causal network and then data is simulated from the obtained network (see Table 11 in the Appendix). For each network, we randomly select 10 different targets and generate 5 samples (except for sample size 5,000 where one sample is generated) to reduce variability due to sampling.[3] An independent sample of 5,000 instances is used for evaluation of classification performance.

In order to avoid overfitting of the results to the method used to induce the causal network, an algorithm with different inductive bias is used than the algorithms tested. In our case we use SCA (Friedman et al., 1999b). We note that SCA has greatly different inductive bias from the GLL variants and thus the comparison (provided that the causal generative model is a Bayesian network) is not unduly biased toward them, while still allowing induction of a credible causal graphical model. Specifically, the inductive biases of the two methods can be described as follows: SCA performs global, heuristically constrained, Bayesian search-and-score, greedy TABU iterative search for a Bayesian network that has maximum-a-posteriori probability given the data under uninformative prior on all possible network structures. GLL algorithms induce a local causal neighborhood, under the distributional assumption of faithfulness and causal sufficiency, employing statistical tests of conditional independence, and preferring to assume a variable is in the local neighborhood whenever a conditional test is not applicable due to small sample (provided that a univariate association exists, otherwise independence is the default) in order to minimize false negative risk of losing a true member and overall risk of false positives and false negatives if true network is not dense. More about the inductive bias of GLL can be found in Aliferis et al. (2010).

We obtained two resimulated networks as follows: (a) *Lung_Cancer* network: We randomly selected 799 genes and a phenotype target (cancer versus normal tissue indicator) from human gene expression data of Bhattacharjee et al. (2001). Then we discretized continuous gene expression data and applied SCA to elicit network structure. (b) *Gene* network: It was obtained from a subset of variables of yeast gene expression data of Spellman et al. (1998) that contained 800 randomly selected genes and a target variable denoting cell cycle state. Continuous gene expression data was also discretized and SCA was applied to learn network. This research design follows Friedman et al. (2000).

Furthermore, we note that additional factors not captured in the simulation or resimulation process make real-life discovery potentially harder than in our experiments. Such factors include for example, deviations of faithfulness, existence of temporal and cellular aggregation effects, unmea-

3. For networks *Lung_Cancer* and *Gene*, we also add an eleventh target that corresponds to the natural response variable: lung cancer diagnosis and cell cycle state, respectively. For network *Munin* we use only 6 targets because of extreme probability distributions of the majority of variables that do not allow variability in the finite sample of size 500 and even 5000. Because of the same reason, we did not experiment with sample size 200 in the *Munin* network.

sured variables, and various measurement, normalization, and noise artifacts. However evaluations with simulated and resimulated data yield comparative performances that are still highly informative since if a method cannot induce the correct structure from relatively easier settings, it is unlikely that in harder real-life situations it will perform any better. In other words successful causal structure discovery performance in simulated and resimulated networks represents at a minimum "gate-keeper" level performance that will filter the more promising from the less promising methods (Spirtes et al., 2000). Finally, as Spirtes et al. (2000) note the behavior of constraint-based algorithms is particularly complex and theoretical analyses are very difficult to perform. The same is true for several other modern feature selection methods. Hence, simulation experiments are necessary in order to gain a deeper understanding of the strengths and limitations of many state-of-the-art algorithms. The evaluated algorithms are provided in Appendix Table 12.

We evaluate the algorithms using the following metrics:

1. *Graph distance*. This metric calculates the average shortest unoriented graph distance of each variable returned by an algorithm to the local neighborhood of target, normalized by the average such distance of all variables in the graph. The rationale is to normalize the score to allow for comparisons across data sets and to correct the score for randomly choosing variables. The score is a non-negative number and has the following interpretation: value 0 means that current feature set is a subset of the true local neighborhood of the target, values less than 1 are better than random selection in the specific network, values equal to 1 are as good as random selection in the specific network and values higher than 1 are worse than random selection. The metric is computed using Dijkstra's shortest path algorithm.

2. *Euclidean distance from the perfect sensitivity and specificity (in the ROC space)* for discovery of local neighborhood of the target variable. This is computed as in Tsamardinos et al. (2003b) and provides a loss function-neutral combination of sensitivity and specificity.

3. *Proportion of false positives and proportion of false negatives*.

4. *Classification performance using polynomial SVM and KNN classifiers* with parameters optimized by nested cross-validation (misclassification cost C and kernel degree d for SVMs and number of nearest neighbors k for KNN) on an independently sampled test data set with large sample (n=5000). The performance is measured by AUC (Fawcett, 2003) on binary tasks and proportion of correct classifications on multiclass tasks.

5. *Feature selection time in minutes*. All caveats regarding interpretation of running times stated in Section 5 apply here as well.

We note that the causal discovery evaluations emphasize *local* discovery of direct causes and direct effects and this choice is supported by several reasons. First, in many domains searching for direct causes and effects is natural (e.g., biological pathway discovery). Second, for non-causal feature selection methods, a natural causal interpretation of their output is being among the direct causes and direct effects (or the Markov blanket) of the target. Consider for example clustering or differential gene expression in bioinformatics where if *Gene1* clusters with *Gene2*, or if *Gene3* is more strongly differentially expressed with respect to some phenotype than *Gene4* then *Gene1* and *Gene2* are interpreted to be members of the same pathway (i.e., in close proximity in the gene regulatory/causal network), and *Gene 3* is interpreted to be more likely to determine the phenotype

than *Gene4*. Similar interpretations abound for other non-causal feature selection methods. We notice that if a method is locally causally inconsistent then it is very unlikely that it will be globally causally consistent either. The logic of this argument is that algorithms either return global or local causal knowledge. If an algorithm outputs a global causal graph and this is incorrect, then this implies that locally it will be wrong for at least some variables. Conversely, if the global graph is correct then locally it is correct as well. If algorithm B outputs a correct local causal set (e.g., direct causes and direct effects) then we can "piece together" these sets and obtain a correct global graph. Finally, if an algorithm outputs an incorrect non-empty local causal set, this implies that B returns non-causes as direct causes or remote causes as direct causes (and the same for effects). Thus, it is not possible to construct the full causal graph strictly from knowledge provided by the algorithm. As a result, local causal consistency is necessary for global consistency as well.

A second reason for focusing on local causal discovery is that it is much harder in practice than indirect causal discovery in highly interconnected causal networks. In our bioinformatics example, because cancer affects many pathways, it is trivial to find genes affected by cancer, since a large proportion (e.g., half) of the measured genes are expected to be affected. However, it is vastly harder to find the chain of events that leads from occurrence of cancer to *Gene1* becoming under- or over-expressed. In such settings, discovery of remote causation is not particularly hard, neither it is particularly interesting. Conversely, when one has a locally correct causal discovery algorithm as elucidated in Section 2, global causal learners can be relatively easily constructed.

Finally, in our evaluations we do not examine quality of causal orientation of the algorithms output for several reasons: First, while GLL algorithms' output can be oriented by constraint-based or other post-processing, non-causal feature selection methods do not readily admit orientation. Second, orientation is not needed when target T is a terminal variable as is often the case in the real data. Third, oriented local causal discovery is harder than unoriented one (Ramsey et al., 2006), and it makes sense to examine the ability of the feature selection algorithms for causal discovery in tasks of incremental difficulty, especially since as we will see most of the non-causal algorithms do not perform well even when seeking unoriented causality. Fourth, orientation information can be obtained subsequently by experiments or knowledge-based post-processing and in many practical settings it is not the primary obstacle to causal discovery.

6.1 Superiority of Causal Over Non-Causal Feature Selection Methods for Causal Discovery

Causal methods achieve, consistently under a variety of conditions and across all metrics employed, superior causal discovery performance than non-causal feature selection methods in our experiments. Figures 14(a) and 15 compare semi-interleaved HITON-PC to HITON-MB, RFE, UAF, L0, and LARS-EN in terms of graph distance and for different sample sizes. Other GLL instantiations such as Interleaved-HITON-PC, MMPC, and Interleaved-MMPC perform similarly to HITON-PC (data in Table S12 in the online supplement). We apply HITON-PC as is and also with a variable pre-filtering step such that only variables that pass a test of univariate association with the target at 5% False Discovery Rate (FDR) threshold are input into the algorithm (Benjamini and Yekutieli, 2001; Benjamini and Hochberg, 1995). Motivation and analysis of incorporating FDR in GLL is provided in Aliferis et al. (2010).

As can be seen, in all samples HITON-PC variants return features closely localized near the target while HITON-MB requires relatively larger sample size to localize well. The distance is smaller as sample size grows. Methods such as univariate filtering localize features well in some

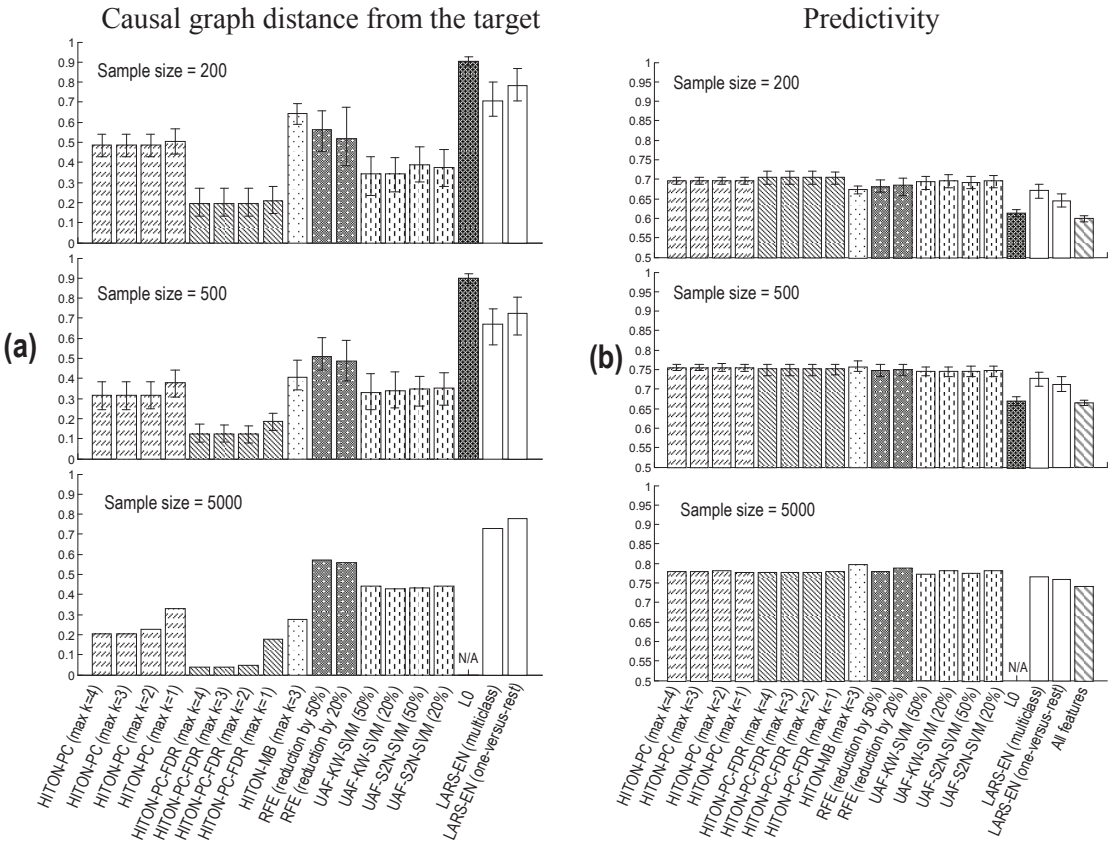


Figure 14: Performance of feature selection algorithms in 9 simulated and resimulated data sets: (a) *graph distance*, (b) *classification performance of polynomial SVM classifiers*. The smaller is causal graph distance and the larger is classification performance, the better is the algorithm. The results are given for training sample sizes = 200, 500, and 5000. The bars denote maximum and minimum performance over multiple training samples of each size (data is available only for sample sizes 200 and 500). The metrics reported in the figure are averaged over all data sets, selected targets, and multiple samples of each size. L0 did not terminate within 2 days (per target) for sample size 5000. Please see text for more details.

data sets and badly in others. As sample size grows, localization of univariate filtering deteriorates. Methods L0, and LARS-EN exhibit a *reverse-localization* bias (i.e., preferentially select features *away* from the target). Performance of RFE varies greatly across data sets in its ability to localize features and this is independent of sample size. A "bull's eye" plot for *Insurance10* data set is provided in Figure 16. A localization example for *Insurance10* data set is shown in Figure 17. The presented visualization examples are representative of the relative performance of causal versus non-causal algorithms. Table 4 provides p-values (via a permutation test at 5% alpha) for the differences of localization among algorithms.

Tables S13-S16 and Figure S2(a)-(d) in the online supplement compare the same algorithms in terms of (a) Euclidian distance from the point of perfect sensitivity and specificity, (b) proportion of false negatives, (c) proportion of false positives, and (d) running time in minutes. Consistent with the results presented in the main text, local causal discovery algorithms strongly outperform

Sample size 200

	Child10	Insurance10	Alarm10	Hailfinder10	Pigs	Link	Lung_Cancer	Gene	*Average*
HITON-PC (max k=4)	0.43	0.41	0.42	0.83	0.41	0.44	0.44	0.50	0.48
HITON-PC (max k=3)	0.43	0.41	0.42	0.83	0.41	0.44	0.44	0.50	0.48
HITON-PC (max k=2)	0.43	0.41	0.42	0.83	0.41	0.44	0.44	0.50	0.48
HITON-PC (max k=1)	0.45	0.42	0.42	0.83	0.41	0.46	0.53	0.50	0.50
HITON-PC-FDR (max k=4)	0.29	0.15	0.24	0.18	0.10	0.17	0.24	0.18	0.19
HITON-PC-FDR (max k=3)	0.29	0.15	0.24	0.18	0.10	0.17	0.24	0.18	0.19
HITON-PC-FDR (max k=2)	0.29	0.15	0.24	0.18	0.10	0.17	0.24	0.18	0.19
HITON-PC-FDR (max k=1)	0.29	0.15	0.24	0.18	0.10	0.17	0.34	0.18	0.21
HITON-MB (max k=3)	0.70	0.68	0.50	0.99	0.49	0.66	0.50	0.64	0.64
RFE (reduction of features by 50%)	0.58	0.38	0.50	0.71	0.52	0.45	0.75	0.59	0.56
RFE (reduction of features by 20%)	0.57	0.46	0.54	0.65	0.46	0.30	0.63	0.54	0.52
UAF-KruskalWallis-SVM (50%)	0.45	0.27	0.32	0.50	0.26	0.34	0.34	0.26	0.34
UAF-KruskalWallis-SVM (20%)	0.43	0.32	0.38	0.55	0.27	0.29	0.29	0.22	0.34
UAF-Signal2Noise-SVM (50%)	0.47	0.31	0.44	0.47	0.33	0.35	0.46	0.27	0.39
UAF-Signal2Noise-SVM (20%)	0.44	0.35	0.40	0.56	0.28	0.29	0.44	0.25	0.38
L0	0.95	0.93	0.83	0.97	0.99	0.83	0.82	0.92	0.90
LARS-EN (for multiclass response)	0.67	0.70	0.64	0.79	0.78	0.66	0.64	0.78	0.71
LARS-EN (one-versus-rest)	0.83	0.68	0.67	0.92	0.89	0.70	0.67	0.89	0.78

Sample size 500

	Child10	Insurance10	Alarm10	Hailfinder10	Pigs	Link	Munin	Lung_Cancer	Gene	*Average*
HITON-PC (max k=4)	0.23	0.26	0.32	0.57	0.27	0.33	0.24	0.28	0.32	0.31
HITON-PC (max k=3)	0.23	0.26	0.32	0.57	0.27	0.33	0.24	0.28	0.32	0.31
HITON-PC (max k=2)	0.23	0.26	0.32	0.57	0.27	0.33	0.24	0.29	0.32	0.32
HITON-PC (max k=1)	0.24	0.28	0.37	0.57	0.34	0.39	0.24	0.52	0.45	0.38
HITON-PC-FDR (max k=4)	0.09	0.08	0.20	0.13	0.02	0.11	0.29	0.14	0.07	0.12
HITON-PC-FDR (max k=3)	0.09	0.08	0.20	0.13	0.02	0.11	0.29	0.13	0.07	0.12
HITON-PC-FDR (max k=2)	0.09	0.08	0.20	0.13	0.02	0.11	0.29	0.11	0.07	0.12
HITON-PC-FDR (max k=1)	0.09	0.11	0.23	0.13	0.08	0.12	0.29	0.40	0.22	0.19
HITON-MB (max k=3)	0.28	0.34	0.37	0.85	0.30	0.43	0.35	0.34	0.38	0.41
RFE (reduction of features by 50%)	0.63	0.51	0.61	0.53	0.37	0.40	0.26	0.70	0.56	0.51
RFE (reduction of features by 20%)	0.54	0.48	0.69	0.53	0.41	0.39	0.26	0.58	0.49	0.49
UAF-KruskalWallis-SVM (50%)	0.37	0.27	0.42	0.49	0.21	0.39	0.34	0.27	0.24	0.33
UAF-KruskalWallis-SVM (20%)	0.40	0.27	0.41	0.48	0.26	0.40	0.30	0.26	0.25	0.34
UAF-Signal2Noise-SVM (50%)	0.40	0.27	0.42	0.51	0.22	0.45	0.29	0.33	0.22	0.35
UAF-Signal2Noise-SVM (20%)	0.42	0.30	0.43	0.51	0.23	0.43	0.30	0.32	0.24	0.35
L0	0.98	0.97	0.93	0.98	0.99	0.87	0.53	0.87	0.97	0.90
LARS-EN (for multiclass response)	0.67	0.71	0.70	0.75	0.78	0.68	0.33	0.60	0.79	0.67
LARS-EN (one-versus-rest)	0.70	0.74	0.74	0.91	0.90	0.77	0.30	0.62	0.82	0.72

Sample size 5000

	Child10	Insurance10	Alarm10	Hailfinder10	Pigs	Link	Munin	Lung_Cancer	Gene	*Average*
HITON-PC (max k=4)	0.13	0.16	0.25	0.35	0.20	0.19	0.04	0.23	0.30	0.20
HITON-PC (max k=3)	0.13	0.16	0.25	0.35	0.20	0.19	0.04	0.23	0.30	0.20
HITON-PC (max k=2)	0.13	0.17	0.25	0.33	0.22	0.19	0.04	0.36	0.33	0.23
HITON-PC (max k=1)	0.18	0.27	0.29	0.33	0.30	0.42	0.04	0.63	0.50	0.33
HITON-PC-FDR (max k=4)	0.00	0.03	0.10	0.10	0.00	0.08	0.04	0.00	0.00	0.04
HITON-PC-FDR (max k=3)	0.00	0.03	0.10	0.10	0.00	0.08	0.04	0.00	0.00	0.04
HITON-PC-FDR (max k=2)	0.00	0.05	0.10	0.10	0.00	0.08	0.04	0.08	0.00	0.05
HITON-PC-FDR (max k=1)	0.01	0.17	0.14	0.11	0.16	0.16	0.04	0.55	0.23	0.18
HITON-MB (max k=3)	0.17	0.20	0.28	0.38	0.27	0.30	0.20	0.33	0.35	0.28
RFE (reduction of features by 50%)	0.63	0.64	0.58	0.59	0.40	0.90	0.28	0.66	0.48	0.57
RFE (reduction of features by 20%)	0.58	0.58	0.69	0.54	0.54	0.92	0.22	0.50	0.43	0.56
UAF-KruskalWallis-SVM (50%)	0.37	0.37	0.62	0.55	0.42	0.69	0.38	0.39	0.20	0.44
UAF-KruskalWallis-SVM (20%)	0.37	0.40	0.60	0.54	0.27	0.59	0.41	0.42	0.24	0.43
UAF-Signal2Noise-SVM (50%)	0.46	0.35	0.65	0.54	0.43	0.67	0.24	0.31	0.25	0.43
UAF-Signal2Noise-SVM (20%)	0.39	0.42	0.58	0.51	0.31	0.60	0.39	0.50	0.25	0.44
LARS-EN (for multiclass response)	0.67	0.85	0.65	0.87	0.74	0.75	0.52	0.71	0.79	0.73
LARS-EN (one-versus-rest)	0.71	0.86	0.74	0.84	0.95	0.80	0.48	0.74	0.88	0.78

Figure 15: Causal graph distance results for training sample sizes = 200, 500 and 5000. The results reported in the figure are averaged over all selected targets. Lighter cells correspond to smaller (better) values of graph distance; darker cells correspond to larger (worse) values of graph distance. L0 did not terminate within 2 days (per target) for sample size 5000.

non-causal feature selection methods in ability to find the direct causes and effects of the target variable.

6.2 Classification Performance is Misleading for Causal Discovery

Despite causally wrong outputs (i.e., failing to return the Markov blanket or parents and children set), several non-causal feature selection methods achieve comparable classification performance with causal algorithms in the simulated data. Figure 14(b) (and Tables S17-S18 and Figure S2(e)

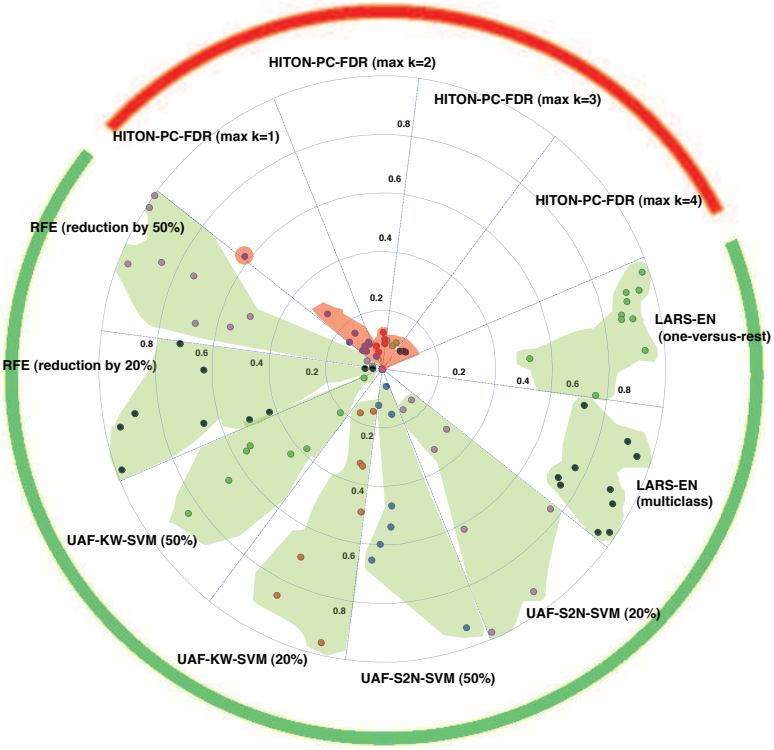


Figure 16: Visualization of graph distances for *Insurance10* network and sample size 5000 by "bull's eye" plot. For each method, results for 10 randomly selected targets are shown. The closer are points to the origin, the better is ability for local causal discovery. Results for GLL method HITON-PC-FDR are highlighted with red; results for baseline methods are highlighted with green.

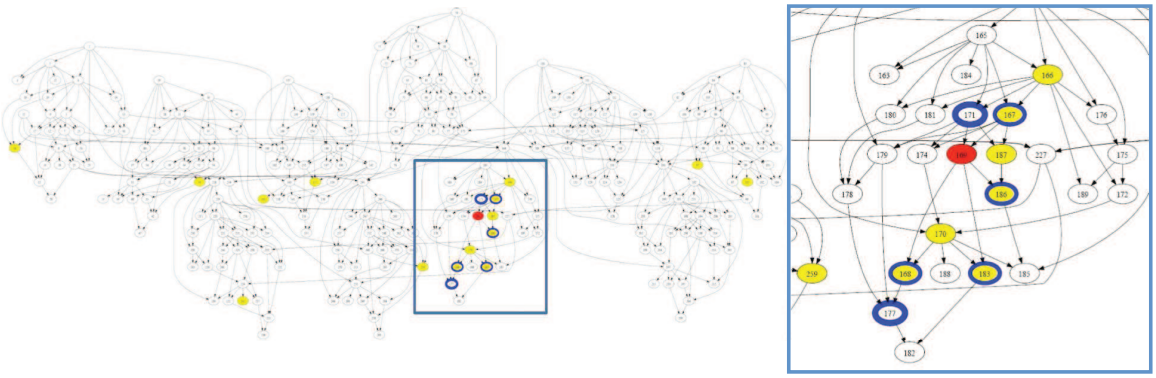


Figure 17: An example of poor localization by a baseline method and good localization by a GLL method. *Left*: Graph of the adjacency matrix of *Insurance10* network. Target variable is shown with red. HITON-PC discovers all 5 members of the parents and children set and a false positive variable #177 that is located close to the true neighborhood (discovered variables are shown with blue bolded circles). RFE discovers 4 out of 5 members of the PC set and introduces many false positives scattered throughout the network (discovered variables are shown with yellow circles). *Right*: A magnified area of the *Insurance10* network close to the target variable.

	Sample size = 200		Sample size = 500		Sample size = 5000	
Comparison	P-value	Nominal winner	P-value	Nominal winner	P-value	Nominal winner
average semi-interleaved HITON-PC with G2 test vs. HITON-MB	**<0.0001**	HITON-PC	**0.0042**	HITON-PC	**0.0472**	HITON-PC
average semi-interleaved HITON-PC with G2 test vs. average RFE	0.2594	HITON-PC	**0.0076**	HITON-PC	**<0.0001**	HITON-PC
average semi-interleaved HITON-PC with G2 test vs. average UAF	**0.0078**	UAF	0.6788	HITON-PC	**0.0086**	HITON-PC
average semi-interleaved HITON-PC with G2 test vs. L0	**<0.0001**	HITON-PC	**<0.0001**	HITON-PC		N/A
average semi-interleaved HITON-PC with G2 test vs. average LARS-EN	**<0.0001**	HITON-PC	**<0.0001**	HITON-PC	**<0.0001**	HITON-PC
average semi-interleaved HITON-PC-FDR with G2 test vs. HITON-MB	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR
average semi-interleaved HITON-PC-FDR with G2 test vs. average RFE	**<0.0001**	HITON-PC-FDR	**0.0028**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR
average semi-interleaved HITON-PC-FDR with G2 test vs. average UAF	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR
average semi-interleaved HITON-PC-FDR with G2 test vs. L0	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR		N/A
average semi-interleaved HITON-PC-FDR with G2 test vs. average LARS-EN	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR	**<0.0001**	HITON-PC-FDR

Table 4: Statistical comparison between semi-interleaved HITON-PC with G^2 test (with and w/o FDR correction) and other methods in terms of graph distance. Bolded p-values are statistically significant at 5% alpha.

in the online supplement) shows the average AUC and proportion of correct classifications. This phenomenon is related to information redundancy of features in relation to the target in non-sparse causal processes. In addition, it is facilitated by the relative insensitivity of state-of-the-art classifiers to irrelevant and redundant features. *Good classification performance is thus greatly misleading as a criterion for quality of causal hypotheses* generated by non-causal feature selection algorithms.

In conclusion, the results in the present section strongly undermine the hope that non-causal feature selection methods can be used as good heuristics for causal discovery. The idea that non-causal feature selection can be used for causal discovery should be viewed with caution (Guyon et al., 2007). Whole research programs are, in many domains, built on experiments motivated by causal hypotheses that were generated by non-causal feature selection results (Zhou et al., 2002; Li et al., 2001; Holmes et al., 2000; Eisen et al., 1998) and this seems an unfortunate and inadvisable practice, in light of existence of principled causal algorithms. On the other hand, generalized local learning algorithms in simulated and resimulated experiments show great potential for local causal discovery.

7. Discussion

In the present section we discuss main findings of this research, state limitations and outline open problems, and give an overview of problems addressed in the companion paper.

7.1 Main Findings

Our experimental evaluation shows that GLL algorithms typically attain the theoretically expected benefits of strong feature set parsimony without loss of performance relative to the best classification attained by any method used in the experiments. The wide range of data sets and algorithms used shows that the sufficient conditions stated in the proofs for correctness for GLL are likely to hold and/or that violations may be small or well tolerated.

The second major result from our experiments is that we showed that use of non-causal feature selection methods for learning causality although very widespread, is generally inadvisable. We used resimulated and simulated data and showed that causally-motivated feature selection methods connect local causal discovery with feature selection for classification consistent with recent theoretical work. Feature selection algorithms that are not causal have a tendency to return highly predictive feature sets that are scattered all over the network, or that are in the periphery of the network, and cannot be otherwise interpreted in a way that makes useful and consistent causal sense. We strongly caution practitioners to use principled causal discovery algorithms whenever available and to not substitute causal discovery methods with predictive/non-causal feature selection ones for reasons of convenience or due to non familiarity with such methods. Practical software widely exists that can be used to apply state-of-the-art causal methods including the methods studied in the present paper that is available for download from the online supplement.

Finally, the theoretical framework that is based in large part on faithfulness and other assumptions summarized in Sections 2 and 3 is a valuable frame of reference both conceptually and algorithmically. However, we do not consider it to be an absolute and immutable measure by which to judge all new and existing algorithms. Our data shows that algorithms that are not deemed correct under the more general assumptions of the framework (e.g., algorithms that do not employ symmetry correction, or algorithms that use $PC(T)$ instead of $MB(T)$ for feature selection for classification) offer in many real data sets same predictive quality and better computational tractability than the sound algorithms. This is a reflection of several factors. One of them is the existence of distributions that are special classes of faithful ones and are easier to analyze (e.g., where symmetry correction is not required, or in other words where $EPC(T) = PC(T)$). A second factor is mitigating circumstances for violations of assumptions (Aliferis et al., 2010). A third factor is that practical implementations of sound algorithms are statistically imperfect (in other words, a theoretical assumption that conveniently leads to a proof of correctness, for example that a conditional test of independence is correct, does not entail immediate or flawless practical feasibility since all such tests admit errors in practice). An alternative set of assumptions for correctness may require vaguely 'sufficient sample size' disregarding the practical difficulty of determining whether in any given analysis this requirement is met. As a result, practical implementations may claim soundness without being demonstrably sound in applied settings. We address the small-sample behavior of GLL algorithms with empirical analysis in the companion paper (Aliferis et al., 2010).

7.2 Limitations and Open Problems

A possible critique of the present work is that Markov blanket features may not work well with a plethora of classifiers, distributions and loss functions. Indeed, a feature selector that is uniformly optimal is not attainable as shown by the results in Tsamardinos and Aliferis (2003), and several (possibly infinite) conceivable classifiers will fail to capture the information in the selected features. Our focus was to examine if the GLL framework has merit in the sense of whether GLL instantia-

tions when applied and compared to reasonable state-of-the-art baseline feature selectors in many complex data sets from typical analysis domains and with practical classifiers, loss function and sample sizes, yield good performance consistent with the theoretical claims of GLL.

Another possibility we would like to address is that best predictivity achieved in our experiments for each data set may not be optimal since some classifier other than SVMs and KNN may yield better predictivity. We believe that this possibility is remote for the following reason: Evidence from earlier published work where we have applied instances of GLL with classifiers such as ANNs, Decision Trees, Simple Bayes, as well as SVMs and KNN supports that the choice of classifier matters very little in practice and similar predictivity/parsimony patterns as the ones reported here were found (Aliferis et al., 2003a). On the other hand, the use of SVMs and KNN as classifiers uniformly across our experiments confers many benefits explained in Section 5. To further support the use of these classifiers we provide additional experimental results in Appendix Table 10 where we use features extracted from embedded or wrapper-based feature selectors (L0, RFVS, LARS-EN) and compare SVMs and KNN to classifiers native to the above embedded and wrapper-based methods. We found that SVMs and KNN achieve predictivity comparable to the classifiers from the aforementioned feature selectors.

Additional strong evidence in favor of our conclusions that GLL algorithms yield highly predictive and parsimonious feature sets is given by the simulated and resimulated data experiments where both the data-generative model and optimal feature sets are known. In those experiments the true Markov blanket is directly given by the model and does constitute the gold standard for the smallest and optimally informative feature set for common loss functions in the sense that *it contains all information available for predicting the target*. The experiments showed that the GLL algorithms identify this Markov blanket very well and better than the baseline comparison algorithms.

Although the GLL framework and the studied instantiations and implementations are theoretically well motivated and empirically robust in many practical data analysis domains, as demonstrated in our experiments, as with all machine learning methods they should be expected to not perform well in quality or efficiency in certain distributions. Such distributions may include cases where the Markov blanket is very large and thus the combinatorics of the elimination phase makes it too slow. Another case can be when extreme non-linearities render the $PC(T)$ members "invisible" to the algorithm (because univariate association with the target is zero). Another possibility for hurting efficiency arises when excessive synthesis of information exists such that the true members of $PC(T)$ are not considered before other weakly relevant variables enter the $TPC(T)$. Also when certain types of deterministic relationships exist or more broadly target information equivalence (i.e., special types of violations of faithfulness), many Markov blankets may exist and the algorithms will return a predictively optimal feature set but both causal localization and optimal parsimony may be lost (Statnikov, 2008). The practical importance of these possibilities needs to be assessed domain-by-domain.

Some of the adverse situations described in the limitations sub-section can be addressed by relaxing the algorithm operation (e.g., for very large Markov blankets the analyst can set *max-k* to a very small number and achieve faster execution but incur some false positives). In some domains, violation of assumptions are mitigated by other factors (e.g., Aliferis et al. 2010 describes how connectivity can make extremely epistatic parents visible to the algorithms). These and other situations constitute open research areas and very recent research efforts attempt to address these issues. For example, Statnikov (2008) provides algorithms that address multiplicity of Markov blankets and Tsamardinos and Brown (2008b) introduce a method for kernel mapping of extremely

non-linear functions to a faithful feature space that can be used to do feature selection via GLL in the transformed feature space.

Although the emphasis of the present work was in classification, Markov blanket theory applies equally well to regression and thus the GLL framework can be used for regression problems as well. An empirical analysis of performance of regression-oriented GLL instantiations and comparisons to state-of-the-art methods were not pursued here however.

7.3 Further Problems Addressed in the Companion Paper

While the theory motivating local learning and especially Markov blanket induction for feature selection has wide implications, it is far from complete. To begin with, all theoretical arguments to-date apply to the large sample case. While the theory implies that the large-sample Markov blanket and the corresponding classifiers fitted from large sample, are predictively optimal, it is not known to what extend learning from small samples affects the optimality of Markov blanket based feature selection. More specifically, it is not clear how often in small samples and real-life distributions the true Markov blanket (i.e., obtained from the data-generative process) gives an optimal classifier when the latter is fitted from small samples with state-of-the-art classifiers. Similarly, we do not know whether the estimated Markov blanket gives an optimal classifier when the latter is fitted from small samples or even when it is fitted from the large sample. Related to the above for practical applications, we do not know how fast is convergence of the estimated Markov blanket/classifier to true Markov blanket/optimal classification as a function of sample size, for the available state-of-the-art Markov blanket inducing algorithms. In the second part of our work (Aliferis et al., 2010) we examine these issues. We also provide explanations why counter-intuitively relaxed versions of some algorithms that trade-off computational efficiency for theoretical soundness tend to outperform sound versions in some domains. Moreover, we systematically study the factors that influence the quality and number of statistical decisions, explain the inductive bias of the algorithms, show how non-causal feature selection methods can be understood in light of Markov blanket induction theory, and address divide-and-conquer local to global causal graph learning strategies.

Appendix A.

This Appendix provides proofs of theorems and additional tables referenced in the paper.

A.1 Proof of Theorem 2

Consider the algorithm in Figure 4. First notice, that as we mentioned above, when conditions (a) and (c) hold the direct causes and direct effects of T will coincide with the parents and children of T in the causal Bayesian network G that faithfully captures the distribution (Spirtes et al., 2000). As we have shown in Section 4 and in Tsamardinos et al. (2003b), the $PC_G(T) = PC(T)$ is unique in all networks faithfully capturing the distribution.

First we show that the algorithm will terminate, that is that the termination criterion of admissibility rule #3 will be met. The criterion requires that no variable eligible for inclusion will fail to enter $TPC(T)$ and that no variable that can be eliminated from $TPC(T)$ is left inside. Indeed because (a) due to admissibility rule #1 all eligible variables in OPEN are identified, (b) $\boldsymbol{V}$ is finite and OPEN instantiated to $\boldsymbol{V} \setminus \{T\}$, and (c) termination will not happen before all eligible members of OPEN are moved from OPEN to $TPC(T)$, the first part of the termination criterion will be satisfied.

The second part of the termination criterion will also be satisfied because of admissibility rule #2 which examines for removal all variables and discards the ones that can be removed.

Lemma 1 *The output of GLL-PC-nonsym $TPC(T)$ is such that: $PC(T) \subseteq TPC(T) \subseteq EPC(T)$.*

Proof Let us assume that $X \in PC(T)$ and show that $X \in TPC(T)$ by the end of GLL-PC-nonsym. By admissibility rule #3, X will never fail to enter $TPC(T)$ by the end of GLL-PC-nonsym. By Theorem 1, for all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X\}$, $\neg I(X,T|\boldsymbol{Z})$ and so the elimination strategy because of admissibility rule #2 will never remove X from $TPC(T)$ by the end of GLL-PC-nonsym.

Now, let us assume that $X \in TPC(T)$ by the end of GLL-PC-nonsym and show that $X \in EPC(T)$. Let us assume the opposite, that is, that $X \notin EPC(T)$ and so by definition $I(X,T|\boldsymbol{Z})$, for some $\boldsymbol{Z} \subseteq PC(T) \setminus \{X\}$. By the same argument as in the previous paragraph, we know that at some point before termination of the algorithm, in step 4, $TPC(T)$ will contain the $PC(T)$. Since $X \notin EPC(T)$, the elimination strategy will find that $I(X,T|\boldsymbol{Z})$, for some $\boldsymbol{Z} \subseteq PC(T) \setminus \{X\}$ and remove X from $TPC(T)$ contrary to what we assumed. Thus, $X \in EPC(T)$ by the end of GLL-PC-nonsym. ■

Lemma 2 *If $X \in EPC(T) \setminus PC(T)$, then $T \notin EPC(X) \setminus PC(X)$*

Proof Let us assume that $X \in EPC(T) \setminus PC(T)$. For every network G faithful to the distribution P $Parents_G(T) \subseteq PC_G(T) = PC(T)$. X has to be a descendant of T in every network G faithful to the distribution because if it is not a descendant, then there is a subset $\boldsymbol{Z}$ of T's parents s.t., $I(X,T|\boldsymbol{Z})$ (by the Markov Condition). Since $X \in EPC(T) \setminus PC(T)$, we know that by definition $\neg I(X,T|\boldsymbol{Z})$, for all $\boldsymbol{Z} \subseteq PC(T) \setminus \{X\}$. By the same argument, if also $T \in EPC(X) \setminus PC(X)$, T would have to be a descendant of X in the every network G which is impossible since the networks are acyclic. So, $T \notin EPC(X) \setminus PC(X)$. ■

Let us assume that $X \in PC(T)$. By Lemma 1, $X \in TPC(T)$ by the end of GLL-PC-nonsym. Since also $T \in PC(X)$, substituting X for T, we also have that by the end of GLL-PC-nonsym, $T \in TPC(X)$. So, X will not be removed from $\boldsymbol{U}$ by the symmetry requirement of GLL-PC either, and will be in the final output of the algorithm.

Conversely, let us assume that $X \notin PC(T)$ and show $X \notin \boldsymbol{U}$ at termination of algorithm GLL-PC. If X never enters $TPC(T)$ by the inclusion heuristic, the proof is done. Similarly, if X enters but is later removed from $TPC(T)$ by the exclusion strategy, the proof is done too. So, let us assume that X enters $TPC(T)$ at some point and by the end of GLL-PC-nonsym(T) is not removed by the exclusion strategy. By Lemma 1, we get that by the end of GLL-PC-nonsym, $X \in EPC(T)$ and since we assumed $X \notin PC(T)$, we get that $X \in EPC(T) \setminus PC(T)$. By Lemma 2, we get that $T \notin EPC(X) \setminus PC(X)$. Since also $T \notin PC(X)$, we get that $T \notin EPC(X)$. Step 3 of GLL-PC will thus eliminate X from $\boldsymbol{U}$.

A.2 Proof of Theorem 4

Since we assume faithful Bayesian networks, d-separation in the graph of such a network is equivalent to independence and can be used interchangeably (Spirtes et al., 2000).

Method	Additional Information	Reference
No feature selection		
RFE (recursive feature elimination SVM-based method)	• reduction by 50% at each iteration, best performing feature subset is returned • reduction by 20% at each iteration, best performing feature subset is returned • reduction by 50% at each iteration, statistically same as best performing feature subset is returned • reduction by 20% at each iteration, statistically same as best performing feature subset is returned	(Guyon et al., 2002)
UAF-KruskalWallis-SVM (univariate ranking by Kruskal-Wallis statistic and feature selection with SVM backward wrapper)	• reduction by 50% at each iteration, best performing feature subset is returned • reduction by 20% at each iteration, best performing feature subset is returned • reduction by 50% at each iteration, statistically same as best performing feature subset is returned • reduction by 20% at each iteration, statistically same as best performing feature subset is returned	(Statnikov et al., 2005a; Hollander and Wolfe, 1999)
UAF-Signal2Noise-SVM (univariate ranking by signal-to-noise statistic and feature selection with SVM backward wrapper)	• reduction by 50% at each iteration, best performing feature subset is returned • reduction by 20% at each iteration, best performing feature subset is returned • reduction by 50% at each iteration, statistically same as best performing feature subset is returned • reduction by 20% at each iteration, statistically same as best performing feature subset is returned	(Guyon et al., 2006b; Statnikov et al., 2005a; Furey et al., 2000)
UAF-Neal-SVM (univariate ranking by Radford Neal's statistic and feature selection with SVM backward wrapper)	• reduction by 50% at each iteration, best performing feature subset is returned • reduction by 20% at each iteration, best performing feature subset is returned • reduction by 50% at each iteration, statistically same as best performing feature subset is returned • reduction by 20% at each iteration, statistically same as best performing feature subset is returned	Chapter 10 in Guyon et al. (2006a)
Random Forest Variable Selection (RFVS)	• best performing feature subset is returned • statistically same as best performing feature subset is returned	(Diaz-Uriarte and Alvarez de Andres, 2006; Breiman, 2001)

Table 5: Algorithms used in evaluation on real data sets. When statistical comparison was performed inside a wrapper, we used a non-parametric method by DeLong et al. (1988). The only exception is Random Forest-based Variable Selection (RFVS), where we used a method recommended by its authors (Diaz-Uriarte and Alvarez de Andres, 2006). For GLL algorithms (i.e., variants of HITON-PC, HITON-MB, MMPC, MMMB) we experimented with both G^2 and Fisher's Z-test whenever the latter was applicable. This table is continued in Tables 6 and 7.

If $X \in MB(T)$, we show $X \in TMB(T)$ in the end. If $X \in MB(T)$ and $X \in PC(T)$, it will be included in the $TMB(T)$ in step 3, will not be removed afterwards and will be included in the final output.

If $X \in MB(T) \setminus PC(T)$ then X will be included in $\boldsymbol{S}$ since if X is a spouse of T, there exists Y (by definition of spouse) s.t., $X \in PC(Y), Y \in PC(T)$ and $X \notin PC(T)$. For that Y, by Theorem 3 we know that $\neg I(X,T|\boldsymbol{Z} \cup \{Y\})$, for all $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus \{X,T\}$ and so the test in step 5c will succeed and X will be included in $TMB(T)$ in the end.

Method	Additional Information	Reference
LARS-Elastic Net (LARS-EN)	• best performing feature subset is returned • statistically same as best performing feature subset is returned	(Zou and Hastie, 2005)
RELIEF (with backward wrapping by SVM)	• Number of neighbors = 1, reduction by 50% at each iteration, best performing feature subset is returned • Number of neighbors = 1, reduction by 20% at each iteration, best performing feature subset is returned • Number of neighbors = 5, reduction by 50% at each iteration, best performing feature subset is returned • Number of neighbors = 5, reduction by 20% at each iteration, best performing feature subset is returned • Number of neighbors = 1, reduction by 50% at each iteration, statistically same as best performing feature subset is returned • Number of neighbors = 1, reduction by 20% at each iteration, statistically same as best performing feature subset is returned • Number of neighbors = 5, reduction by 50% at each iteration, statistically same as best performing feature subset is returned • Number of neighbors = 5, reduction by 20% at each iteration, statistically same as best performing feature subset is returned	(Kononenko, 1994; Kira and Rendell, 1992)
L0-norm		(Weston et al., 2003)
Forward Stepwise Selection	using SVM classifier for wrapping	(Caruana and Freitag, 1994)
Koller-Sahami (with backward wrapping by SVM)	• $k = 0$, best performing feature subset is returned • $k = 1$, best performing feature subset is returned • $k = 2$, best performing feature subset is returned • $k = 0$, statistically same as best performing feature subset is returned • $k = 1$, statistically same as best performing feature subset is returned • $k = 2$, statistically same as best performing feature subset is returned	(Koller and Sahami, 1996)
IAMB	• G^2 test and $a = 0.05$ • G^2 test and $a = 0.01$ • mutual information criterion with threshold=0.01	(Tsamardinos and Aliferis, 2003; Tsamardinos et al., 2003a)
K2MB		(Cooper et al., 1997; Cooper and Herskovits, 1992)

Table 6: Continued from Table 5.

Conversely, if $X \notin MB(T)$ we show that $X \notin TMB(T)$ by the end of the algorithm. Let $\boldsymbol{Z}$ be the subset in step 5a, s.t., $I(X, T|\boldsymbol{Z})$ (i.e., $\boldsymbol{Z}$ d-separates X and T). Then, $\boldsymbol{Z}$ blocks all paths from X to T. For the test in step 5c to succeed a node Y must exist that opens a new path, previously closed by $\boldsymbol{Z}$, from X to T. Since by conditioning on an additional node a path opens, Y has to be a collider (by the d-separation definition) or a descendant of a collider on a path from X to T. In addition, this path must have length two edges since all nodes in $\boldsymbol{S}$ are the parents and children of the $PC(T)$ but without belonging in $PC(T)$. Thus, for the test in step 5c to succeed there has to be a path of length two from X to T with a collider in-between, that is, X has to be a spouse of T. Since $X \notin MB(T)$ the test will fail for all Y and $X \notin TMB(T)$ by the end of the algorithm.

Method	Additional Information	Reference
BLCD-MB		(Mani and Cooper, 2004)
FAST-IAMB	G^2 test and $a = 0.05$	(Yaramakala and Margaritis, 2005)
HITON-PC (semi-interleaved)	• *max-k* = 4 and $a = 0.05$ • *max-k* = 3 and $a = 0.05$ • *max-k* = 2 and $a = 0.05$ • *max-k* = 1 and $a = 0.05$ • *max-k* = 4 and $a = 0.01$ • *max-k* = 3 and $a = 0.01$ • *max-k* = 2 and $a = 0.01$ • *max-k* = 1 and $a = 0.01$ • *max-k* and a selected by cross-validation	Novel algorithm
Interleaved HITON-PC	• *max-k* = 4 and $a = 0.05$ • *max-k* = 3 and $a = 0.05$ • *max-k* = 2 and $a = 0.05$ • *max-k* = 1 and $a = 0.05$ • *max-k* = 4 and $a = 0.01$ • *max-k* = 3 and $a = 0.01$ • *max-k* = 2 and $a = 0.01$ • *max-k* = 1 and $a = 0.01$ • *max-k* and a selected by cross-validation	(Aliferis et al., 2003a)
MMPC	• *max-k* = 4 and $a = 0.05$ • *max-k* = 3 and $a = 0.05$ • *max-k* = 2 and $a = 0.05$ • *max-k* = 1 and $a = 0.05$ • *max-k* = 4 and $a = 0.01$ • *max-k* = 3 and $a = 0.01$ • *max-k* = 2 and $a = 0.01$ • *max-k* = 1 and $a = 0.01$ • *max-k* and a selected by cross-validation	(Tsamardinos et al., 2006, 2003b)
Interleaved MMPC	• *max-k* = 4 and $a = 0.05$ • *max-k* = 3 and $a = 0.05$ • *max-k* = 2 and $a = 0.05$ • *max-k* = 1 and $a = 0.05$ • *max-k* = 4 and $a = 0.01$ • *max-k* = 3 and $a = 0.01$ • *max-k* = 2 and $a = 0.01$ • *max-k* = 1 and $a = 0.01$ • *max-k* and a selected by cross-validation	Novel algorithm
HITON-MB (semi-interleaved)	• *max-k* = 3 and $a = 0.05$ • *max-k* = 3 and $a = 0.01$	Novel algorithm
MMMB	• *max-k* = 3 and $a = 0.05$ • *max-k* = 3 and $a = 0.01$	(Tsamardinos et al., 2003b)

Table 7: Continued from Table 6.

Data set name	Domain	Num. variables	Num. samples	Target	Data type	Cross-val. design	Discretization applied	Notes	Reference
Infant Mortality	Clinical	86	5,337	Died within the first year	Discrete	1-fold cross-val.	Already discrete	Imputed by nearest neighbor method	(Mani and Cooper, 1999)
Ohsumed	Text	14,373	5,000	Relevant to neonatal diseases	Continuous	1-fold cross-val.	Word absent / present		(Joachims, 2002)
ACPJ Etiology	Text	28,228	15,779	Relevant to etiology	Continuous	1-fold cross-val.	Word absent / present		(Aphinyanaphongs et al., 2006)
Lymphoma	Gene expression	7,399	227	3-year survival: dead vs. alive	Continuous	10-fold cross-val.	Binary/ternary univariate; used window sizes 10,15, 20, 25, 30 for ternary		(Rosenwald et al., 2002)
Gisette	Digit recognition	5,000	7,000	Separate 4 from 9	Continuous	1-fold cross-val.	Pixel present / absent	Used original training & validation sets only	NIPS 2003 Feature Selection Challenge (Guyon et al., 2006a)
Dexter	Text	19,999	600	Relevant to corporate acquisitions	Continuous	10-fold cross-val.	Word absent / present	Used original training & validation sets only	NIPS 2003 Feature Selection Challenge (Guyon et al., 2006a)
Sylva	Ecology	216	14,394	Ponderosa pine vs. everything else	Continuous & discrete	1-fold cross-val.	Binary/ternary univariate; used window sizes 1000, 1500, 2000, 2500, 3000 for ternary	Used original training & validation sets only	WCCI 2006 Performance Prediction Challenge

Table 8: Real data sets used in evaluation of predictivity and compactness. This table is continued in Table 9.

Data set name	**Domain**	**Num. variables**	**Num. samples**	**Target**	**Data type**	**Cross-val. design**	**Discretization applied**	**Notes**	**Reference**
Ovarian Cancer	Proteomics	2,190	216	Cancer vs. normals	Continuous	10-fold cross-val.	Binary/ternary univariate; used window sizes 10, 15, 20, 25, 30 for ternary		(Conrads et al., 2004)
Thrombin	Drug discovery	139,351	2,543	Binding to thrombin	Discrete (binary)	1-fold cross-val.	Already discrete		KDD Cup 2001
Breast Cancer	Gene expression	17,816	286	Estrogen-receptor positive (ER+) vs. ER-	Continuous	10-fold cross-val.	Binary/ternary univariate, used window sizes 10, 15, 20, 25, 30 for ternary		(Wang et al., 2005)
Hiva	Drug discovery	1,617	4,229	Activity to AIDS HIV infection	Discrete (binary)	1-fold cross-val.	Already discrete	Used original training & validation sets only	WCCI 2006 Performance Prediction Challenge
Nova	Text	16,969	1,929	Separate politics from religion topics	Discrete (binary)	1-fold cross-val.	Already discrete	Used original training & validation sets only	WCCI 2006 Performance Prediction Challenge
Bankruptcy	Financial	147	7,063	Personal bankruptcy	Continuous & discrete	1-fold cross-val.	Binary/ternary univariate, used window sizes 1000, 1500, 2000, 2500, 3000 for ternary	Imputed by nearest neighbor method	(Foster and Stine, 2004)

Table 9: Continued from Table 8.

References

C. F. Aliferis and G. F. Cooper. An evaluation of an algorithm for inductive learning of Bayesian belief networks using simulated data sets. *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 1994.

C. F. Aliferis and I. Tsamardinos. Algorithms for large-scale local causal discovery and feature selection in the presence of small sample or large causal neighborhoods. *Technical Report DSL 02-08*, 2002a.

C. F. Aliferis and I. Tsamardinos. Using local causal induction to improve global causal discovery: Enhancing the sparse candidate set. *Technical Report DSL 02-04*, 2002b.

C. F. Aliferis, I. Tsamardinos, and A. Statnikov. Large-scale feature selection using Markov blanket induction for the prediction of protein-drug binding. *Technical Report DSL 02-06*, 2002.

C. F. Aliferis, I. Tsamardinos, and A. Statnikov. HITON: a novel Markov blanket algorithm for optimal variable selection. *AMIA 2003 Annual Symposium Proceedings*, pages 21–25, 2003a.

C. F. Aliferis, I. Tsamardinos, A. Statnikov, and L. E. Brown. Causal explorer: a causal probabilistic network learning toolkit for biomedical discovery. *Proceedings of the 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS)*, 2003b.

C. F. Aliferis, A. Statnikov, E. Kokkotou, P. P. Massion, and I. Tsamardinos. Local regulatory-network inducing algorithms for biomarker discovery from mass-throughput datasets. *Technical Report DSL 06-05*, 2006a.

C. F. Aliferis, A. Statnikov, and P. P. Massion. Pathway induction and high-fidelity simulation for molecular signature and biomarker discovery in lung cancer using microarray gene expression data. *Proceedings of the 2006 American Physiological Society Conference "Physiological Genomics and Proteomics of Lung Disease"*, 2006b.

C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification. Part II: Analysis and extensions. *Journal of Machine Learning Research*, 11:235–284, 2010.

Y. Aphinyanaphongs and C. F. Aliferis. Learning boolean queries for article quality filtering. *Medinfo 2004.*, 11(Pt 1):263–267, 2004.

Y. Aphinyanaphongs, A. Statnikov, and C. F. Aliferis. A comparison of citation metrics to machine learning filters for the identification of high quality medline documents. *J.Am.Med.Inform.Assoc.*, 13(4):446–455, Jul 2006.

X. Bai, C. Glymour, R. Padman, J. Ramsey, P. Spirtes, and F. Wimberly. PCX: Markov blanket classification for large data sets with few cases. *Technical Report, Center for Automated Learning and Discovery*, 2004.

G. E. A. P. A. Batista and M. C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5-6):519–533, 2003.

Feature subset	*Classifier*	Infant_Mortality	Ohsumed	ACPJ_Etiology	Lymphoma	Gisette	Dexter	Sylva	Ovarian_Cancer	Thrombin	Breast_Cancer	Hiva	Nova	Bankruptcy
LARS-EN (w/o stat. comp.)	SVM	0.88	0.80	0.89	0.60	0.99	0.98	1.00	0.98	0.89	0.92	0.73	0.96	0.95
	LARS-EN	0.88	0.81	0.88	0.60	1.00	0.98	1.00	0.99	0.89	0.92	0.77	0.94	0.94
LARS-EN (with stat. comp.)	SVM	0.86	0.77	0.82	0.57	0.99	0.98	1.00	0.96	0.85	0.94	0.62	0.96	0.95
	LARS-EN	0.87	0.78	0.82	0.57	1.00	0.97	0.99	0.96	0.90	0.94	0.69	0.93	0.94
L0	SVM	0.82	0.72	0.84	0.60	0.99	0.97	1.00	0.97	0.81	0.91	0.68	0.96	T
	L0	0.81	0.72	0.87	0.58	0.99	0.97	1.00	0.96	0.81	0.91	0.69	0.95	T
RFVS (w/o stat. comp.)	SVM	0.82	T	T	0.61	T	0.98	1.00	0.97	T	0.93	0.74*	T	0.96
	RF	0.84	T	T	0.63	T	0.98	1.00	0.97	T	0.91	0.78	T	0.97
RFVS (with stat. comp.)	SVM	0.86	T	T	0.61	T	0.98	1.00	0.96	T	0.93	0.68*	T	0.97
	RF	0.78	T	T	0.63	T	0.98	1.00	0.97	T	0.92	0.75	T	0.97

Table 10: Classification performance (AUC) for polynomial SVMs and classifiers native to LARS-EN, L0, and RFVS feature selection algorithms induced with features selected by the latter three methods. In cells marked with "T", the corresponding feature selection method did not terminate within the allotted time.

Bayesian network	**Number of variables**	**Training samples**	**Number of selected targets**
Child10	200	5 x 200, 5 x 500, 1 x 5000	10
Insurance10	270	5 x 200, 5 x 500, 1 x 5000	10
Alarm10	370	5 x 200, 5 x 500, 1 x 5000	10
Hailfinder10	560	5 x 200, 5 x 500, 1 x 5000	10
Munin	189	5 x 500, 1 x 5000	6
Pigs	441	5 x 200, 5 x 500, 1 x 5000	10
Link	724	5 x 200, 5 x 500, 1 x 5000	10
Lung_Cancer	800	5 x 200, 5 x 500, 1 x 5000	11
Gene	801	5 x 200, 5 x 500, 1 x 5000	11

Table 11: Simulated and resimulated data sets used for experiments. *Lung_Cancer* network is resimulated from human lung cancer gene expression data (Bhattacharjee et al., 2001) using SCA algorithm (Friedman et al., 1999b). *Gene* network is resimulated from yeast cell cycle gene expression data (Spellman et al., 1998) using SCA algorithm. More details about data sets are provided in Tsamardinos et al. (2006).

HITON-PC (max k=4)	**HITON-PC-FDR** (max k=4)
HITON-PC (max k=3)	**HITON-PC-FDR** (max k=3)
HITON-PC (max k=2)	**HITON-PC-FDR** (max k=2)
HITON-PC (max k=1)	**HITON-PC-FDR** (max k=1)
Interleaved HITON-PC (max k=4)	**HITON-MB** (max k=3)
Interleaved HITON-PC (max k=3)	**MMMB** (max k=3)
Interleaved HITON-PC (max k=2)	**RFE** (reduction of features by 50%)
Interleaved HITON-PC (max k=1)	**RFE** (reduction of features by 20%)
MMPC (max k=4)	**UAF-KruskalWallis-SVM** (50%)
MMPC (max k=3)	**UAF-KruskalWallis-SVM** (20%)
MMPC (max k=2)	**UAF-Signal2Noise-SVM** (50%)
MMPC (max k=1)	**UAF-Signal2Noise-SVM** (20%)
Interleaved MMPC (max k=4)	**L0**
Interleaved MMPC (max k=3)	**LARS-EN** (for multiclass response)
Interleaved MMPC (max k=2)	**LARS-EN** (one-versus-rest)
Interleaved MMPC (max k=1)	

Table 12: Algorithms used in local causal discovery experiments with simulated and resimulated data.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society.Series B (Methodological)*, 57(1):289–300, 1995.

Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Ann.Statist*, 29(4):1165–1188, 2001.

A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. J. Mark, E. S. Lander, W. Wong, B. E. Johnson, T. R. Golub, D. J. Sugarbaker, and M. Meyerson. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc.Natl.Acad.Sci.U.S.A*, 98(24):13790–13795, Nov 2001.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. E. Brown, I. Tsamardinos, and C. F. Aliferis. A comparison of novel and state-of-the-art polynomial Bayesian network learning algorithms. *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, 2005.

R. Caruana and D. Freitag. Greedy attribute selection. *Proceedings of the Eleventh International Conference on Machine Learning*, pages 28–36, 1994.

J. Cheng and R. Greiner. Comparing Bayesian network classifiers. *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 101–107, 1999.

J. Cheng and R. Greiner. Learning Bayesian belief network classifiers: Algorithms and system. *Proceedings of 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, 2001.

J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. Learning Bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137(1):43–90, 2002a.

J. Cheng, C. Hatzis, H. Hayashi, M. A. Krogel, S. Morishita, D. Page, and J. Sese. Kdd cup 2001 report. *ACM SIGKDD Explorations Newsletter*, 3(2):47–64, 2002b.

D. M. Chickering. Learning equivalence classes of bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, 2002.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3(3):507–554, 2003.

D. M. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks is NP-hard. *Technical Report MSR-TR-94-17*, 1994.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

T. P. Conrads, V. A. Fusaro, S. Ross, D. Johann, V. Rajapakse, B. A. Hitt, S. M. Steinberg, E. C. Kohn, D. A. Fishman, G. Whitely, J. C. Barrett, L. A. Liotta, E. F. I. I. I. Petricoin, and T. D. Veenstra. High-resolution serum proteomic features for ovarian cancer detection. *Endocr.Relat Cancer*, 11(2):163–178, Jun 2004.

G. F. Cooper. A simple constraint-based algorithm for efficiently mining observational databases for causal relationships. *Data Mining and Knowledge Discovery*, 1(2):203–224, 1997.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

G. F. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. *Proceedings of Uncertainty in Artificial Intelligence*, pages 116–125, 1999.

G. F. Cooper, C. F. Aliferis, R. Ambrosino, J. Aronis, B. G. Buchanan, R. Caruana, M. J. Fine, C. Glymour, G. Gordon, and B. H. Hanusa. An evaluation of machine-learning methods for predicting pneumonia mortality. *Artificial Intelligence in Medicine*, 9(2):107–138, 1997.

D. Dash and G. F. Cooper. Exact model averaging with naive Bayesian classifiers. *Proc.19th Int.Conf.Machine Learning (ICML 2002)*, pages 91–98, 2002.

E. R. DeLong, D. M. DeLong, and D. L. Clarke-Pearson. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics*, 44(3): 837–845, Sep 1988.

R. Diaz-Uriarte and S. Alvarez de Andres. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7:3, 2006.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

S. Duda, C. F. Aliferis, R. Miller, A. Statnikov, and K. Johnson. Extracting drug-drug interaction articles from medline to improve the content of drug databases. *AMIA 2005 Annual Symposium Proceedings*, pages 216–220, 2005.

S. Dudoit and M. J. van der Laan. Asymptotics of cross-validated risk estimation in model selection and performance assessment. *UC Berkeley Division of Biostatistics Working Paper Series*, 126, 2003.

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. *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 178–183, 2005.

F. Eberhardt, C. Glymour, and R. Scheines. N-1 experiments suffice to determine the causal relations among n variables. *Innovations in Machine Learning: Theory And Applications*, 2006.

M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc.Natl.Acad.Sci.U.S.A*, 95(25):14863–14868, Dec 1998.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1361, 2001.

R. E. Fan, P. H. Chen, and C. J. Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6(1889):1918, 2005.

N. Fananapazir, M. Li, D. Spentzos, and C. F. Aliferis. Formative evaluation of a prototype system for automated analysis of mass spectrometry data. *AMIA 2005 Annual Symposium Proceedings*, pages 241–245, 2005.

T. Fawcett. Roc graphs: Notes and practical considerations for researchers. *Technical Report, HPL-2003-4, HP Laboratories*, 2003.

D. P. Foster and R. A. Stine. Variable selecion in data mining: Building a predictive model for bankruptcy. *Journal of the American Statistical Association*, 99(466):303–314, 2004.

L. Frey, D. Fisher, I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Identifying Markov blankets with decision tree induction. *Proceedings of the Third IEEE International Conference on Data Mining (ICDM)*, 2003.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29 (2):131–163, 1997.

N. Friedman, M. Goldszmidt, and A. Wyner. Data analysis with Bayesian networks: A bootstrap approach. *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, pages 206–215, 1999a.

N. Friedman, I. Nachman, and D. Pe'er. Learning Bayesian network structure from massive datasets: the "sparse candidate" algorithm. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 1999b.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *J Comput.Biol.*, 7(3-4):601–620, 2000.

G. M. Fung and O. L. Mangasarian. A feature selection newton method for support vector machine classification. *Computational Optimization and Applications*, 28(2):185–202, 2004.

T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, Oct 2000.

O. Gevaert, Smet F. De, D. Timmerman, Y. Moreau, and Moor B. De. Predicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks. *Bioinformatics*, 22(14):e184–e190, Jul 2006.

C. N. Glymour and G. F. Cooper. *Computation, Causation, and Discovery*. AAAI Press, Menlo Park, Calif, 1999.

P. I. Good. *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses*, volume 2nd. Springer, New York, 2000.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(1):1157–1182, 2003.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, 2002.

I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction: Foundations and Applications*. Springer-Verlag, Berlin, 2006a.

I. Guyon, J. Li, T. Mader, P. A. Pletscher, G. Schneider, and M. Uhr. Feature selection with the clop package. Technical report, http://clopinet.com/isabelle/Projects/ETH/TM-fextract-class.pdf, 2006b.

I. Guyon, C. F. Aliferis, and A. Elisseeff. *Computational Methods of Feature Selection*, chapter Causal Feature Selection. Chapman and Hall, 2007.

D. Hardin, I. Tsamardinos, and C. F. Aliferis. A theoretical characterization of linear SVM-based feature selection. *Proceedings of the Twenty First International Conference on Machine Learning (ICML)*, 2004.

D. Heckerman. A tutorial on learning with Bayesian networks. *Technical Report MSR-TR-95-06*, 1995.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

M. Hollander and D. Wolfe. *Nonparametric Statistical Methods*, volume 2nd. Wiley, New York, NY, USA, 1999.

J. H. Holmes, D. R. Durbin, and F. K. Winston. The learning classifier system: an evolutionary computation approach to knowledge discovery in epidemiologic surveillance. *Artif.Intell.Med.*, 19(1):53–74, May 2000.

N. Hoot, I. Feurer, C. W. Pinson, and C. F. Aliferis. Modelling liver transplant survival: comparing techniques of deriving predictor sets. *Journal of Gastrointestinal Surgery*, 9(4):563, Apr 2005.

T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer Academic Publishers, Boston, 2002.

K. Kira and L. A. Rendell. A practical approach to feature selection. *Proceedings of the Ninth International Workshop on Machine Learning*, pages 249–256, 1992.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2): 273–324, 1997.

D. Koller and M. Sahami. Toward optimal feature selection. *Proceedings of the International Conference on Machine Learning*, 1996, 1996.

I. Kononenko. Estimating attributes: Analysis and extensions of relief. *Proceedings of the European Conference on Machine Learning*, pages 171–182, 1994.

L. Li, C. R. Weinberg, T. A. Darden, and L. G. Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the ga/knn method. *Bioinformatics*, 17(12):1131–1142, Dec 2001.

H. Liu and H. Motoda. *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Kluwer Academic, Boston, 1998.

H. Liu, F. Hussain, C. L. Tan, and M. Dash. Discretization: an enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423, 2002.

S. Mani and G. F. Cooper. A study in causal discovery from population-based infant birth and death records. *Proceedings of the AMIA Annual Fall Symposium*, 319, 1999.

S. Mani and G. F. Cooper. Causal discovery using a Bayesian local causal discovery algorithm. *Medinfo 2004*., 11(Pt 1):731–735, 2004.

D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. *Advances in Neural Information Processing Systems*, 12:505–511, 1999.

S. Meganck, P. Leray, and B. Manderick. Learning causal Bayesian networks from observations and experiments: A decision theoretic approach. *Modeling Decisions in Artificial Intelligence, LNCS*, pages 58–69, 2006.

A. Moore and W. K. Wong. Optimal reinsertion: a new search operator for accelerated and more accurate Bayesian network structure learning. *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 552–559, 2003.

K. P. Murphy. Active learning of causal Bayes net structure. *Technical Report, University of California, Berkeley*, 2001.

R. E. Neapolitan. *Probabilistic Reasoning in Expert Systems: Theory and Algorithms*. Wiley, New York, 1990.

R. E. Neapolitan. *Learning Bayesian networks*. Pearson Prentice Hall, Upper Saddle River, NJ, 2004.

J. Peña, J. Bjorkegren, and J. Tegner. Growing Bayesian network models of gene networks from seed genes. *Bioinformatics*, 21(2):224–229, 2005a.

J. Peña, J. Bjorkegren, and J. Tegner. Scalable, efficient and correct learning of Markov boundaries under the faithfulness assumption. *Proceedings of the Eighth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, 2005b.

J. Peña, R. Nilsson, J. Bjorkegren, and J. Tegnér. Towards scalable and data efficient learning of Markov boundaries. *International Journal of Approximate Reasoning*, 45(2):211–232, 2007.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Mateo, California, 1988.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, U.K, 2000.

J. Pearl and T. Verma. A theory of inferred causation. *Principles of Knowledge Representation and Reasoning: Proceedings of Second International Conference*, pages 441–452, 1991.

J. Pearl and T. S. Verma. Equivalence and synthesis of causal models. *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1990.

I. Pournara and L. Wernisch. Reconstruction of gene networks using Bayesian learning and manipulation experiments. *Bioinformatics*, 20(17):2934–2942, Nov 2004.

A. Rakotomamonjy. Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, 3(7-8):1357–1370, 2003.

J. Ramsey. A pc-style Markov blanket search for high-dimensional datasets. *Technical Report, CMU-PHIL-177, Carnegie Mellon University, Department of Philosophy*, 2006.

J. Ramsey, J. Zhang, and P. Spirtes. Adjacency-faithfulness and conservative causal inference. *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, 2006.

A. Rosenwald, G. Wright, W. C. Chan, J. M. Connors, E. Campo, R. I. Fisher, R. D. Gascoyne, H. K. Muller-Hermelink, E. B. Smeland, J. M. Giltnane, E. M. Hurt, H. Zhao, L. Averett, L. Yang, W. H. Wilson, E. S. Jaffe, R. Simon, R. D. Klausner, J. Powell, P. L. Duffey, D. L. Longo, T. C. Greiner, D. D. Weisenburger, W. G. Sanger, B. J. Dave, J. C. Lynch, J. Vose, J. O. Armitage, E. Montserrat, A. Lopez-Guillermo, T. M. Grogan, T. P. Miller, M. LeBlanc, G. Ott, S. Kvaloy, J. Delabie, H. Holte, P. Krajci, T. Stokke, and L. M. Staudt. The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma. *N.Engl.J Med.*, 346(25): 1937–1947, Jun 2002.

A. Sboner and C. F. Aliferis. Modeling clinical judgment and implicit guideline compliance in the diagnosis of melanomas using machine learning. *AMIA 2005 Annual Symposium Proceedings*, pages 664–668, 2005.

T. Scheffer. *Error Estimation and Model Selection*. PhD thesis, Ph.D.Thesis, Technischen Universitet Berlin, School of Computer Science, 1999.

C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. *Data Mining and Knowledge Discovery*, 4(2):163–192, 2000.

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol.Biol Cell*, 9(12):3273–3297, Dec 1998.

P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, Prediction, and Search*, volume 2nd. MIT Press, Cambridge, Mass, 2000.

A. Statnikov. Algorithms for discovery of multiple Markov boundaries: Application to the molecular signature multiplicity problem. *Ph.D.Thesis, Department of Biomedical Informatics, Vanderbilt University*, 2008.

A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5):631–643, Mar 2005a.

A. Statnikov, I. Tsamardinos, Y. Dosbayev, and C. F. Aliferis. Gems: a system for automated cancer diagnosis and biomarker discovery from microarray gene expression data. *Int.J.Med.Inform.*, 74 (7-8):491–503, Aug 2005b.

A. Statnikov, D. Hardin, and C. F. Aliferis. Using SVM weight-based methods to identify causally relevant and non-causally relevant variables. *Proceedings of the NIPS 2006 Workshop on Causality and Feature Selection*, 2006.

J. Tian and J. Pearl. Causal discovery from changes: A bayesian approach. *UCLA Cognitive Systems Laboratory, Technical Report (R-285)*, 2001.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society.Series B (Methodological)*, 58(1):267–288, 1996.

S. Tong and D. Koller. Active learning for structure in bayesian networks. *Proceedings of the International Joint Conference on Artificial Intelligence*, 17:863–869, 2001.

I. Tsamardinos and C. F. Aliferis. Towards principled feature selection: relevancy, filters and wrappers. *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (AI & Stats)*, 2003.

I. Tsamardinos and L. E. Brown. Bounding the false discovery rate in local Bayesian network learning. *Proceedings of the Twenty Third National Conference on Artificial Intelligence (AAAI)*, 2008a.

I. Tsamardinos and L. E. Brown. Markov blanket-based variable selection in feature space. *Technical report DSL-08-01*, 2008b.

I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Algorithms for large scale Markov blanket discovery. *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pages 376–381, 2003a.

I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Time and sample efficient discovery of Markov blankets and direct causal relations. *Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 673–678, 2003b.

I. Tsamardinos, C. F. Aliferis, A. Statnikov, and L. E. Brown. Scaling-up Bayesian network learning to thousands of variables using local learning technique. *Technical Report DSL 03-02*, 12, 2003c.

I. Tsamardinos, Brown L.E., and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Technical report DSL-05-01*, 2005.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

L. Wang, J. Zhu, and H. Zou. The doubly regularized support vector machine. *Statistica Sinica*, 16: 589–615, 2006.

Y. Wang, J. G. Klijn, Y. Zhang, A. M. Sieuwerts, M. P. Look, F. Yang, D. Talantov, M. Timmermans, M. E. Meijer-van Gelder, J. Yu, T. Jatkoe, E. M. Berns, D. Atkins, and J. A. Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365(9460):671–679, Feb 2005.

J. Weston, A. Elisseeff, B. Scholkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3(7):1439–1461, 2003.

S. Yaramakala and D. Margaritis. Speculative Markov blanket discovery for optimal feature selection. *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 809–812, 2005.

C. Yoo and G. F. Cooper. An evaluation of a system that recommends microarray experiments to perform to discover gene-regulation pathways. *Artif.Intell.Med.*, 31(2):169–182, Jun 2004.

X. Zhou, M. C. J. Kao, and W. H. Wong. Transitive functional annotation by shortest-path analysis of gene expression data. *Proceedings of the National Academy of Sciences*, 99(20):12783–12788, 2002.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. *Advances in Neural Information Processing Systems (NIPS)*, 16, 2004.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B(Statistical Methodology)*, 67(2):301–320, 2005.

Journal of Machine Learning Research 11 (2010) 235-284 Submitted 7/07; Revised 2/08; Published 1/10

Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part II: Analysis and Extensions

Constantin F. Aliferis CONSTANTIN.ALIFERIS@NYUMC.ORG
Center of Health Informatics and Bioinformatics
Department of Pathology
New York University
New York, NY 10016, USA

Alexander Statnikov ALEXANDER.STATNIKOV@MED.NYU.EDU
Center of Health Informatics and Bioinformatics
Department of Medicine
New York University
New York, NY 10016, USA

Ioannis Tsamardinos TSAMARD@ICS.FORTH.GR
Computer Science Department, University of Crete
Institute of Computer Science, Foundation for Research and Technology, Hellas
Heraklion, Crete, GR-714 09, Greece

Subramani Mani SUBRAMANI.MANI@VANDERBILT.EDU
Discovery Systems Laboratory
Department of Biomedical Informatics
Vanderbilt University
Nashville, TN 37232, USA

Xenofon D. Koutsoukos XENOFON.KOUTSOUKOS@VANDERBILT.EDU
Department of Electrical Engineering and Computer Science
Vanderbilt University
Nashville, TN 37212, USA

Editor: Marina Meila

Abstract

In part I of this work we introduced and evaluated the *Generalized Local Learning* (GLL) framework for producing local causal and Markov blanket induction algorithms. In the present second part we analyze the behavior of GLL algorithms and provide extensions to the core methods. Specifically, we investigate the empirical convergence of GLL to the true local neighborhood as a function of sample size. Moreover, we study how predictivity improves with increasing sample size. Then we investigate how sensitive are the algorithms to multiple statistical testing, especially in the presence of many irrelevant features. Next we discuss the role of the algorithm parameters and also show that Markov blanket and causal graph concepts can be used to understand deviations from optimality of state-of-the-art non-causal algorithms. The present paper also introduces the following extensions to the core GLL framework: parallel and distributed versions of GLL algorithms, versions with false discovery rate control, strategies for constructing novel heuristics for specific domains, and divide-and-conquer *local-to-global learning* (LGL) strategies. We test the generality of the LGL approach by deriving a novel LGL-based algorithm that compares favorably

to the state-of-the-art global learning algorithms. In addition, we investigate the use of non-causal feature selection methods to facilitate global learning. Open problems and future research paths related to local and local-to-global causal learning are discussed.

Keywords: local causal discovery, Markov blanket induction, feature selection, classification, causal structure learning, learning of Bayesian networks

1. Introduction

The present paper constitutes the second part of the study of *Generalized Local Learning* (GLL) which provides a unified framework for discovering local causal structure around a target variable of interest using observational data under broad assumptions. GLL supports local discovery of variables that are direct causes or direct effects of the target and of the Markov blanket of the target. In the first part of the work (Aliferis et al., 2010) we introduced GLL and explained the importance of local causal discovery both for identification of highly predictive and parsimonious feature sets (feature selection problem), and for scaling up causal discovery. We then evaluated GLL instantiations against a plethora of state-of-the-art alternatives in many real, simulated and resimulated data sets. The main conclusions were that GLL algorithms achieved excellent predictivity, compactness and ability to learn local neighborhoods. Moreover, state-of-the-art non-causal feature selection methods often achieve excellent predictivity but are misleading in terms of causal discovery.

In the present paper we provide several extensions to GLL, study its properties, and extend to global graph learning using GLL as the core method. Because of the close relationship with Aliferis et al. (2010) we do not repeat here background material, technical definitions, or algorithm specifications. These are found in Aliferis et al. (2010), Sections 2-4.

The paper is organized as follows: Section 2 studies the empirical convergence of GLL instantiations to the true local neighborhood and to optimal predictivity as a function of sample size. Section 3 studies the effects of multiple statistical testing and the sensitivity of GLL algorithms to large numbers of irrelevant features. Section 4 provides a theoretical analysis of GLL algorithms with respect to determinants of statistical decisions, heuristic efficiency and construction of inclusion heuristic functions, reasons for good performance of direct causes and effects instead of induced Markov blanket, and reduced sensitivity to error estimation problems that affect wrappers and traditional filters. Section 5 covers two algorithmic extensions, parallel processing and False Discovery Rate pre-filtering. Section 6 investigates the use of local learners like GLL for global learning and provides a general local-to-global learning framework. In that section we also derive a new algorithm HHC and compare it to the previously described MMHC, and show the potential of local induction variable ordering for tractability and quality improvements. Section 7 uses causal feature selection theory to shed light on limitations of established and newer feature selection methods and the inappropriateness of causally interpreting their output. Section 8 concludes with a discussion of the findings of the present paper and several open problems. An appendix and an online supplement (`http://www.nyuinformatics.org/downloads/supplements/JMLR2009/index.html`) provide additional results, as well as code and data sets that can be used to replicate the experiments.

2. Empirical Convergence and Comparison of Theoretical to Estimated Markov Blanket

As explained in Aliferis et al. (2010), arguments about the suitability of Markov blanket induction for feature selection for classification are based on large sample results, with convergence of small sample performance to the theoretical optimum being unknown. In the present section we use simulated data sets from published Bayesian networks to produce an empirical evaluation of classification performance convergence with respect to training sample size of two types of classifiers: one that uses the estimated Markov blanket ($MB(T)$) or parents and children set ($PC(T)$) and one that uses the true $MB(T)$ or $PC(T)$ set (obtained from the known generative network). We use polynomial SVMs and KNN to fit each classifier type from three training sample sizes: 200, 500 and 5,000 samples. We note that GLL algorithms provide predictive and optimality guarantees for universal approximator classifiers and SVMs and KNN are used here as exemplars of this class of algorithms. In Aliferis et al. (2010) we also discuss more generally suitable classifiers, distributions and loss functions for GLL instantiations. An independent sample of 5,000 instances is used as evaluation test for classification performance (measured by AUC for binary and proportion of correct classifications for multiclass classification tasks). We use data sets sampled from 9 different Bayesian networks (See Table 15 in the Appendix). For each Bayesian network, we randomly select 10 different targets and generate 5 samples (except for sample size 5,000 where one sample is generated) to reduce variability due to sampling.[1] An independent sample of 5,000 instances is used as evaluation test for classification performance. Several local causal induction algorithms are used (including algorithms that induce direct causes/direct effects, and Markov blankets), and are compared to several non-causal algorithms to obtain reference points for baseline performance: RFE, UAF (univariate association filtering), L0, and LARS-EN (see Table 16 in the Appendix for the list of all algorithms). Classifier parameters (misclassification cost C and degree d for polynomial SVMs and number of neighbors K for KNN) are optimized by nested cross-validation following the same methodology as in Aliferis et al. (2010).

Results are presented in Figure 1 (and more details are given in Tables S19 and S20 of the online supplement). The main conclusions follow. Note that similar patterns are present when KNN is used instead of SVMs (with the only difference that convergence is slightly slower for KNN than for SVMs). For brevity we discuss here the SVM results only.

(a) Classification performance of the true parents and children and Markov blanket feature sets are not statistically significantly different at the 0.05 alpha level in sample 200 (p-value = 0.1440) and are statistically significantly different for larger samples (p-values = 0.0098 and $<$0.0001 for sample sizes 500 and 5,000, respectively). The difference in SVM classification performance between using the $PC(T)$ and $MB(T)$ sets however does not exceed 0.02 AUC in favor of the $MB(T)$ set. This means that even when the true $PC(T)$ and $MB(T)$ sets are known in the tested data, fitting classifiers from small data using the $PC(T)$ set is as good as using the $MB(T)$ set. In large sample, $MB(T)$ features have a small predictive advantage over $PC(T)$ features.

1. For networks *Lung_Cancer* and *Gene*, we also add an eleventh target that corresponds to the natural response variable: lung cancer diagnosis and cell cycle state, respectively. For network *Munin* we use only 6 targets because of extreme probability distributions of the majority of variables that do not allow variability in the finite sample of size 500 and even 5000. Because of the same reason, we did not experiment with sample size 200 in the *Munin* network.

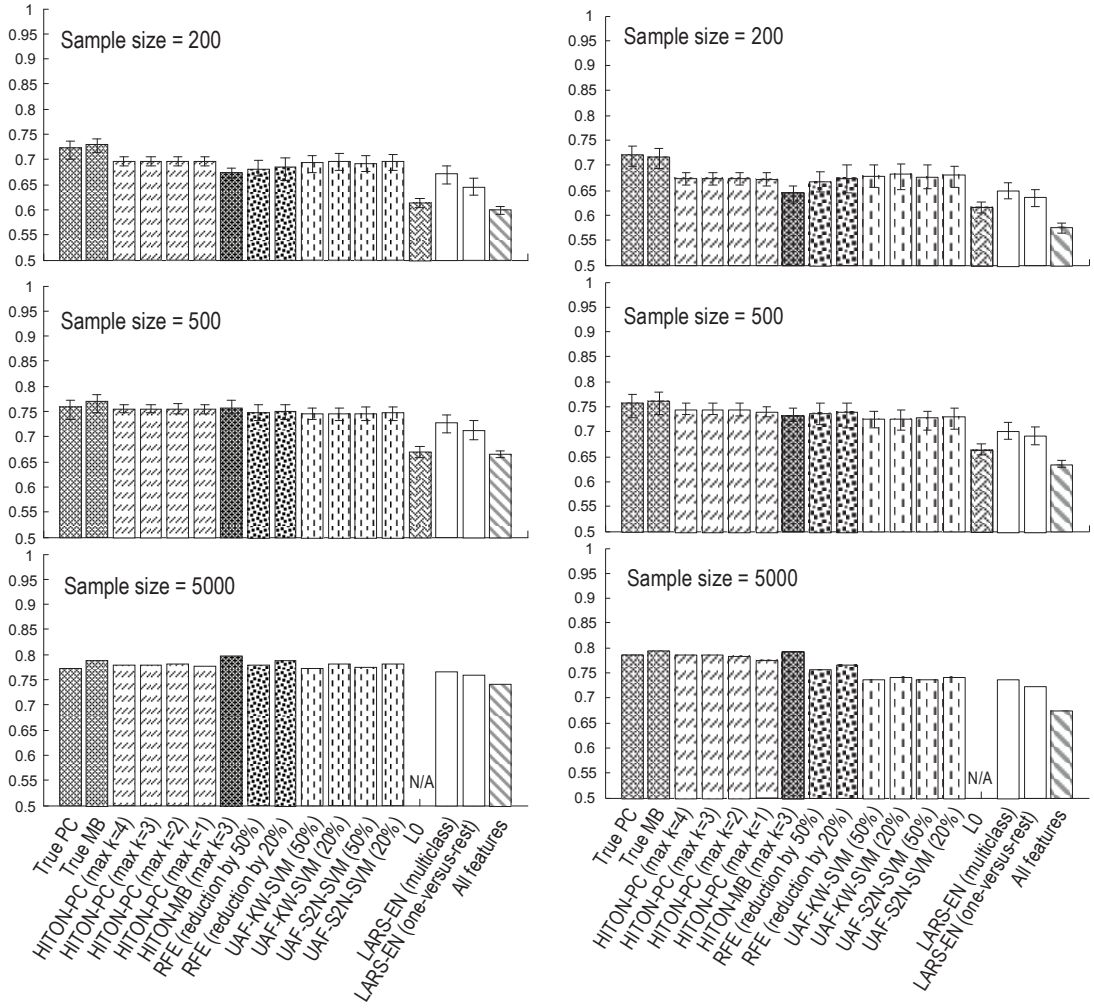


Figure 1: Classification performance of polynomial SVM (left) and KNN (right) classifiers in 9 simulated and resimulated data sets. Results are given for training sample sizes = 200, 500, and 5000. "True-PC" and "True-MB" correspond to the true $PC(T)$ and $MB(T)$ feature sets obtained from the known generative network. The bars denote maximum and minimum performance over multiple training samples of each size (data is available only for sample sizes 200 and 500). The performances reported in the figure are averaged over all data sets, selected targets, and multiple samples of each size. L0 did not terminate within the allotted time limit for sample size 5000.

(b) In small samples, feature selection increases classification performance for all tested classifier types (i.e., both when we know the $PC(T)$ or $MB(T)$ sets and when we estimate them from data) over using all features. This advantage becomes smaller but does not vanish in large sample. The difference in SVM classification performance between an average feature selection method and using all features is statistically significant at the 0.05 alpha level (p-values = $<0.0001, 0.0028, <0.0001$ for sample sizes 200, 500, and 5,000, respectively).

(c) The true $PC(T)$ or true $MB(T)$ features set when fitted from sample size of 200 has a small (0.02-0.03 AUC/proportion of correct classifications for SVM) advantage over the estimated $PC(T)$ or $MB(T)$ features fitted from small sample. This difference is statistically significant at the 0.05 alpha level with p-values 0.0144 and $<$0.0001 for the $PC(T)$ and $MB(T)$ classifiers, respectively. Very quickly (as sample size becomes 500), this advantage becomes insignificant (0.01 point of AUC/proportion of correct classifications for SVM) with corresponding p-values 0.4708 and 0.0506 for the $PC(T)$ and $MB(T)$ classifiers, respectively. This implies that predictivity of estimated $MB(T)$ and $PC(T)$ sets converge to the optimal one very quickly with respect to sample size.

(d) Classifiers for estimated $MB(T)$/$PC(T)$ sets fitted from small sample and classifiers for the true $MB(T)$/$PC(T)$ sets fitted from small sample have indistinguishable performance in sample size 500 (as shown in (c) above); then performance increases in sample size 5,000 for both types of classifiers (p-values ranging from $<$0.0001 to 0.0174 with AUC increases between 0.01 and 0.04). We thus conclude that fitting the right classifier parameters to the identified features is less sample efficient than identifying the right feature set.

(e) Some of the non-causal feature selection methods (e.g., L0, LARS-EN) tend to compare less favorably in small sample to their large sample performance compared to GLL algorithms.

3. Multiple Statistical Tests and Insensitivity to Irrelevant Variables

In this section we focus our attention to a subtle but an important problem facing many feature and causal discovery algorithms operating in very high dimensional spaces, namely the problem of multiple statistical comparisons, which is exacerbated when many irrelevant features are present. We will show that GLL algorithms have inherent control to false positives due to multiple comparisons while the same is not true for other non-causal feature selection methods tested.

Briefly stated, when conducting n statistical tests with an error type I level α (i.e., statistical significance level, that is probability that a truly null hypothesis is rejected, thus falsely concluding that a statistical difference or association or dependence exists when in reality it does not) it is expected that $\alpha \cdot n$ false positives will occur on average. Consider a common analysis situation in bioinformatics research where a researcher conducts one test per variable (i.e., single nucleotide polymorphism (SNP)) in an assay with 10,000 SNP probes in total. 10,000 such tests need be conducted to see whether univariately each SNP probe is differentially present in two or more phenotype categories. If the researcher uses α equal to 5%, then under the null hypothesis (i.e., all 10,000 SNPs are not truly differentially expressed) the analysis will yield 500 false positive SNP probes. Standard statistical practice involves addressing the problem via one of two basic approaches. The first approach, the classic Bonferroni correction (Casella and Berger, 2002), adjusts the α by replacing it by α/n so that in our example the 5% false positive rate is preserved for each feature selected by the multiple tests. This approach preserves the desired α, but reduces the power to detect statistically significant features (namely the features that are truly differentially expressed and detectable at α but non-detectable at α/n), hence creates false negatives that were not present before the correction. The second approach, False Discovery Rate (FDR) control (Benjamini and Yekutieli, 2001; Benjamini and Hochberg, 1995), trades off false positives and false negatives by ensuring not that each feature passing the chosen p-value threshold preserves the original α, but that from the all features found to be significant (i.e., for which the null hypothesis is rejected) a desired proportion will be false

Lung_Cancer

Sample size	*Version 1* (original network) max-k parameter					*Version 2* (original network + irrelevant variables) max-k parameter					*Version 3* (weakened signal + irrelevant variables) max-k parameter					*Version 4* (only irrelevant variables) max-k parameter				
	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	1.00	0.99	0.99	0.99	0.99	0.97	0.99	0.98	0.98	0.98	0.63	0.63	0.62	0.62	0.62	0.50	0.50	0.50	0.50	0.50
200	1.00	1.00	0.99	0.98	0.98	0.99	1.00	0.99	0.99	0.99	0.67	0.69	0.67	0.66	0.66	0.51	0.50	0.49	0.50	0.50
500	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.67	0.72	0.73	0.72	0.71	0.50	0.50	0.51	0.49	0.49
1000	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.68	0.74	0.73	0.74	0.72	0.50	0.52	0.51	0.50	0.49
2000	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.69	0.74	0.74	0.74	0.74	0.49	0.50	0.49	0.50	0.49
5000	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.72	0.74	0.74	0.74	0.74	0.51	0.51	0.49	0.49	0.49

Alarm10

Sample size	*Version 1* (original network) max-k parameter					*Version 2* (original network + irrelevant variables) max-k parameter					*Version 3* (weakened signal + irrelevant variables) max-k parameter					*Version 4* (only irrelevant variables) max-k parameter				
	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	0.95	0.95	0.95	0.95	0.95	0.83	0.92	0.92	0.92	0.92	0.66	0.69	0.69	0.69	0.69	0.50	0.50	0.50	0.50	0.50
200	0.96	0.95	0.95	0.95	0.95	0.89	0.95	0.95	0.95	0.95	0.68	0.77	0.78	0.78	0.78	0.50	0.50	0.50	0.50	0.50
500	0.96	0.96	0.96	0.96	0.96	0.93	0.95	0.95	0.95	0.95	0.71	0.80	0.80	0.80	0.81	0.50	0.51	0.50	0.50	0.50
1000	0.97	0.97	0.97	0.97	0.97	0.94	0.97	0.96	0.96	0.96	0.73	0.82	0.81	0.82	0.82	0.50	0.50	0.50	0.50	0.50
2000	0.97	0.97	0.97	0.97	0.97	0.96	0.97	0.97	0.97	0.97	0.76	0.82	0.82	0.82	0.82	0.50	0.50	0.50	0.50	0.50
5000	0.97	0.98	0.97	0.97	0.97	0.97	0.98	0.97	0.97	0.97	0.81	0.83	0.83	0.83	0.83	0.50	0.50	0.50	0.50	0.50

Low classification performance — *High classification performance*

Table 1: Classification performance (AUC) of polynomial SVM estimated on 5,000 sample independent testing set for features selected by HITON-PC with parameter *max-k*={0,1,2,3,4} on different training sample sizes {100,200,500,1000,2000,5000}. The color of each table cell denotes strength of predictivity with yellow (light) corresponding to low classification performance and red (dark) to high classification performance.

positives on average. In our example, FDR methods may, for example, allow the researcher to ensure that on average no more than 10 out of 100 SNPs selected are false positives. This is highly useful in exploratory analysis of high-dimensional data where subsequent experimentation can sort out false positives easily but where false negatives have high cost.

Constraint-based causal methods employ, in large data sets and depending on connectivity and inclusion heuristic efficiency, many thousands of statistical tests of independence and are thus expected a priori to be particularly sensitive to the multiple testing problem. We note that, rather not obviously at first, testing under the null hypothesis does not only occur when irrelevant features exist but also whenever we test weakly relevant features conditioned on a set of variables that blocks all paths connecting it with the target. Other feature selection methods do not explicitly conduct statistical tests of independence but may also be sensitive to many irrelevant features as we will show. In the present section we first systematically explore empirically and then examine theoretically the degree of sensitivity of GLL algorithms to irrelevant features, how they address the multiple testing problem, and how other feature selection and causal discovery algorithms compare along these dimensions.

In the first set of experiments we run only semi-interleaved HITON-PC without symmetry correction on two networks and variants. The networks, described in Aliferis et al. (2010), are the *Lung_Cancer* resimulated network and the *Alarm10* network. The former is chosen for its higher

Lung_Cancer

	Version 1 (original network)					*Version 2* (original network + irrelevant variables)					*Version 3* (weakened signal + irrelevant variables)				
	max-k parameter														
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	3.30	15.30	18.20	18.20	18.20	3.30	15.40	18.40	18.40	18.40	9.40	21.90	23.40	23.40	23.40
200	1.20	7.70	17.70	19.60	19.60	1.20	7.70	17.70	19.60	19.60	4.40	17.50	23.20	23.40	23.40
500	0.80	1.30	5.70	15.10	18.00	0.80	1.30	5.70	15.10	18.00	1.00	4.60	17.50	21.70	21.90
1000	0.30	1.00	1.50	5.40	11.70	0.30	1.00	1.50	5.40	11.70	0.80	1.70	6.60	17.50	19.90
2000	0.30	0.90	1.00	1.80	4.10	0.30	0.90	1.00	1.80	4.10	0.70	1.00	1.80	8.70	15.80
5000	0.00	0.40	1.00	1.10	1.10	0.00	0.40	1.00	1.10	1.10	0.30	0.80	1.00	1.40	4.80

Alarm10

	Version 1 (original network)					*Version 2* (original network + irrelevant variables)					*Version 3* (weakened signal + irrelevant variables)				
	max-k parameter														
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	1.70	4.10	4.10	4.10	4.10	1.70	4.10	4.20	4.20	4.20	2.20	5.00	5.00	5.00	5.00
200	1.40	3.90	4.00	4.00	4.00	1.40	3.90	4.00	4.00	4.00	1.80	4.50	4.70	4.70	4.70
500	0.40	2.60	2.70	2.70	2.70	0.40	2.60	2.90	3.00	3.00	0.60	3.90	4.40	4.40	4.40
1000	0.10	2.00	2.10	2.10	2.10	0.10	2.00	2.20	2.20	2.20	0.80	3.60	3.90	4.00	4.00
2000	0.00	1.40	1.50	1.50	1.50	0.00	1.40	1.50	1.50	1.50	0.10	3.10	3.60	3.50	3.50
5000	0.00	0.50	1.10	1.20	1.20	0.00	0.50	1.10	1.20	1.20	0.00	1.40	1.70	1.80	1.80

Small number of false negatives — *Large number of false negatives*

Table 2: Number of false negatives in the parents and children set for features selected by HITON-PC with parameter *max-k*=$\{0,1,2,3,4\}$ on different training sample sizes $\{100,200,500,1000,2000,5000\}$. For Version 4 of the network the parents and children set is empty since there are no relevant variables. The color of each table cell denotes number of false negatives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

connectivity whereas the latter is designed to have lower connectivity. In the *Lung_Cancer* network we focused our attention on the natural target variable; this target has 26 members of the parents and children set and 18 spouses, 14 irrelevant variables, and 741 weakly relevant ones. We created four versions of this network: *Version 1* contains the original network (total number of variables 800). In *Version 2* we augment the original network with 7990 irrelevant variables (total number of variables 8790). *Version 3* is the same as Version 2, except for 10% of values of the target are randomly flipped to weaken the signal (total number of variables 8790). Finally, *Version 4* is same as Version 2, except that there are only irrelevant variables and the target (total number of variables is $8790-741-18-26=8005$). The tiled *Alarm10* has also four corresponding versions but its target was chosen randomly and it has only 6 members of the parents and children set and no spouses. In both networks (and their variants) we create irrelevant variables by randomly permuting values of weakly and strongly variables so that the distribution of each variable values is realistic. With these 8 data set versions we can systematically examine the effects of presence of irrelevant variables, strength of predictive signal of features for the target, network connectivity and of the values of the GLL *max-k* parameter (Aliferis et al., 2010).

We run HITON-PC and build SVM classifiers for all networks and variants, varying sample size and the *max-k* parameter, and measure AUC, false negatives, false positives that are weakly relevant,

Lung_Cancer

Sample size	*Version 1* (original network) max-k parameter					*Version 2* (original network + irrelevant variables) max-k parameter					*Version 3* (weakened signal + irrelevant variables) max-k parameter				
	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	65.00	0.80	0.30	0.30	0.30	65.00	0.70	0.40	0.40	0.40	62.40	0.90	0.50	0.50	0.50
200	120.50	3.00	0.10	0.00	0.00	120.50	3.00	0.10	0.00	0.00	85.60	2.90	0.60	0.60	0.60
500	149.00	5.80	0.00	0.10	0.00	149.00	5.80	0.00	0.10	0.00	110.70	4.20	0.40	0.30	0.30
1000	202.90	11.60	0.10	0.00	0.00	202.90	11.60	0.10	0.00	0.00	123.70	5.70	0.00	0.00	0.00
2000	236.10	16.40	0.50	0.10	0.00	236.10	16.40	0.50	0.10	0.00	171.10	12.00	0.40	0.00	0.00
5000	410.40	30.80	2.60	0.10	0.00	410.40	30.80	2.60	0.10	0.00	272.60	20.30	1.10	0.00	0.00

Alarm10

Sample size	*Version 1* (original network) max-k parameter					*Version 2* (original network + irrelevant variables) max-k parameter					*Version 3* (weakened signal + irrelevant variables) max-k parameter				
	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	22.10	3.70	3.70	3.70	3.70	22.10	2.40	2.40	2.40	2.40	22.50	1.80	1.80	1.80	1.80
200	26.50	0.80	0.80	0.80	0.80	26.50	0.60	0.50	0.50	0.50	25.20	1.30	0.90	0.90	0.90
500	32.20	0.90	0.10	0.10	0.10	32.20	0.80	0.10	0.10	0.10	32.00	1.00	0.20	0.20	0.20
1000	30.20	1.40	0.00	0.00	0.00	30.20	1.30	0.00	0.00	0.00	27.10	0.70	0.10	0.30	0.30
2000	33.50	2.90	0.30	0.30	0.30	33.50	2.80	0.30	0.30	0.30	32.40	1.80	0.60	0.20	0.20
5000	38.00	5.40	0.30	0.20	0.10	38.00	5.30	0.30	0.20	0.10	37.30	3.10	0.20	0.20	0.20

Small number of false positives — *Large number of false positives*

Table 3: Number of false positives (within weakly relevant variables) in the parents and children set for features selected by HITON-PC with parameter *max-k*=$\{0,1,2,3,4\}$ on different training sample sizes $\{100,200,500,1000,2000,5000\}$. For Version 4 of the network there are no weakly relevant variables. The color of each table cell denotes number of false positives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

false positives that are irrelevant and total false positives. To ensure that our results are not affected by variability in small samples, we generate 10 random samples of each size and average results.

Tables 1– 5 provide evidence for the following conclusions:

(a) Classification performance is mildly or not affected by false positives and false negatives (Table 1). When many false negatives are present, predictivity is compensated by the few remaining strong relevant features plus strongly predictive weakly relevant ones. This implies that classification performance cannot be used to inform us about the presence of false positives/negatives.

(b) As expected, false negatives are reduced as sample size grows (because power increases), however they also increase as *max-k* grows, because the number of tests increases as *max-k* grows and thus overall power decreases (Table 2).

(c) When no irrelevant features are present, as sample size grows the number of false positives that are weakly relevant increases if *max-k* is not sufficient to block paths from/to each weakly relevant to/from the target. As *max-k* increases the false positives decrease to the point that they vanish (Table 3). Overall, both false negatives and false positives vanish given enough

Lung_Cancer

	Version 1 (original network)					*Version 2* (original network + irrelevant variables)					*Version 3* (weakened signal + irrelevant variables)					*Version 4* (only irrelevant variables)				
	max-k parameter																			
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	65.20	0.80	0.30	0.30	0.30	476.60	2.30	1.90	1.90	1.90	551.20	12.60	9.10	9.10	9.10	411.60	12.70	9.80	9.80	9.80
200	122.00	3.00	0.10	0.00	0.00	609.10	4.20	0.10	0.00	0.00	557.20	17.80	3.50	3.60	3.60	488.60	17.30	5.80	5.50	5.50
500	149.20	5.80	0.00	0.10	0.00	595.00	7.90	0.00	0.10	0.00	535.60	17.50	1.30	1.50	1.70	446.00	28.10	6.40	5.00	4.90
1000	203.40	11.60	0.10	0.00	0.00	625.60	13.20	0.10	0.00	0.00	536.90	18.40	0.20	0.30	0.30	422.70	31.20	6.90	5.30	5.10
2000	236.90	16.40	0.50	0.10	0.00	645.10	18.00	0.50	0.10	0.00	579.00	23.10	0.80	0.00	0.00	409.00	31.80	6.10	4.00	4.00
5000	411.10	30.80	2.60	0.10	0.00	813.50	32.50	2.60	0.10	0.00	670.40	32.10	1.10	0.00	0.00	403.10	30.90	6.20	4.70	4.10

Alarm10

	Version 1 (original network)					*Version 2* (original network + irrelevant variables)					*Version 3* (weakened signal + irrelevant variables)					*Version 4* (only irrelevant variables)				
	max-k parameter																			
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	22.10	3.70	3.70	3.70	3.70	414.20	25.40	25.20	25.20	25.20	431.20	28.00	28.20	28.20	28.20	392.10	23.30	23.40	23.40	23.40
200	26.50	0.80	0.80	0.80	0.80	439.40	6.30	4.30	4.30	4.30	453.00	11.60	7.40	7.40	7.40	412.90	19.30	9.70	9.70	9.70
500	32.20	0.90	0.10	0.10	0.10	443.80	4.70	0.90	0.90	0.90	449.90	15.80	4.60	4.10	4.00	411.60	24.40	6.80	6.60	6.60
1000	30.20	1.40	0.00	0.00	0.00	444.30	3.70	0.90	0.60	0.60	427.00	13.30	3.40	3.10	3.00	414.10	22.70	7.20	6.40	6.30
2000	33.50	2.90	0.30	0.30	0.30	415.50	4.40	0.30	0.30	0.30	412.40	11.90	2.40	1.80	1.70	382.00	25.00	8.80	6.50	5.90
5000	38.00	5.40	0.30	0.20	0.10	419.00	6.70	0.40	0.20	0.10	404.40	10.80	1.20	0.50	0.50	381.00	22.90	6.10	5.00	4.90

Small number of false positives — *Large number of false positives*

Table 4: Number of false positives in the parents and children set for features selected by HITON-PC with parameter *max-k*=$\{0,1,2,3,4\}$ on different training sample sizes $\{100,200,500,1000,2000,5000\}$. The color of each table cell denotes number of false positives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

sample size and sufficient (but not excessive) *max-k*, (i.e., sample size $\geq$ 2,000, *max-k*=2) (Tables 2 and 4).

(d) When irrelevant features are present, as sample size grows the number of false positives that are weakly relevant increases if *max-k* is not sufficient to block paths from/to each weakly relevant to/from the target. As *max-k* increases, the false positives decrease to the point that they vanish (Table 3). False positives due to irrelevant features (Table 5) quickly vanish as *max-k* becomes 2 or higher and this holds as long as sample size is larger than 200. False negatives are not affected by presence of irrelevant features (Table 2). Thus, overall, with enough sample size and right value of *max-k*, both false negatives and false positives vanish (Tables 2 and 4).

(e) When the predictive signal is weaker, both false negatives are increased and false positives within weakly relevant variables are decreased for a given sample size (because power is smaller) (Tables 2 and 3). However false positive irrelevant variables (Table 5) are increased. This is due to the fact that fewer features enter the $TPC(T)$ set thus leading to fewer tests that can be performed hence smaller capacity to remove irrelevant false positives. As previously with enough sample and right *max-k*, false positives and negatives are fully eliminated (Tables 2 and 4).

(f) When the data consists only of irrelevant features, false positives (irrelevant) are reduced as *max-k* increases for all sample sizes (Table 5). There is a very small persistent residual number of false positives regardless of how small the sample is or how big the *max-k*. These phenom-

Lung_Cancer

	Version 1 *(original network)*					***Version 2*** *(original network + irrelevant variables)*					***Version 3*** *(weakened signal + irrelevant variables)*					***Version 4*** *(only irrelevant variables)*				
	max-k parameter																			
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	0.20	0.00	0.00	0.00	0.00	411.60	1.60	1.50	1.50	1.50	488.80	11.70	8.60	8.60	8.60	411.60	12.70	9.80	9.80	9.80
200	1.50	0.00	0.00	0.00	0.00	488.60	1.20	0.00	0.00	0.00	471.60	14.90	2.90	3.00	3.00	488.60	17.30	5.80	5.50	5.50
500	0.20	0.00	0.00	0.00	0.00	446.00	2.10	0.00	0.00	0.00	424.90	13.30	0.90	1.20	1.40	446.00	28.10	6.40	5.00	4.90
1000	0.50	0.00	0.00	0.00	0.00	422.70	1.60	0.00	0.00	0.00	413.20	12.70	0.20	0.30	0.30	422.70	31.20	6.90	5.30	5.10
2000	0.80	0.00	0.00	0.00	0.00	409.00	1.60	0.00	0.00	0.00	407.90	11.10	0.40	0.00	0.00	409.00	31.80	6.10	4.00	4.00
5000	0.70	0.00	0.00	0.00	0.00	403.10	1.70	0.00	0.00	0.00	397.80	11.80	0.00	0.00	0.00	403.10	30.90	6.20	4.70	4.10

Alarm10

	Version 1 *(original network)*					***Version 2*** *(original network + irrelevant variables)*					***Version 3*** *(weakened signal + irrelevant variables)*					***Version 4*** *(only irrelevant variables)*				
	max-k parameter																			
Sample size	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
100	0.00	0.00	0.00	0.00	0.00	392.10	23.00	22.80	22.80	22.80	408.70	26.20	26.40	26.40	26.40	392.10	23.30	23.40	23.40	23.40
200	0.00	0.00	0.00	0.00	0.00	412.90	5.70	3.80	3.80	3.80	427.80	10.30	6.50	6.50	6.50	412.90	19.30	9.70	9.70	9.70
500	0.00	0.00	0.00	0.00	0.00	411.60	3.90	0.80	0.80	0.80	417.90	14.80	4.40	3.90	3.80	411.60	24.40	6.80	6.60	6.60
1000	0.00	0.00	0.00	0.00	0.00	414.10	2.40	0.90	0.60	0.60	399.90	12.60	3.30	2.80	2.70	414.10	22.70	7.20	6.40	6.30
2000	0.00	0.00	0.00	0.00	0.00	382.00	1.60	0.00	0.00	0.00	380.00	10.10	1.80	1.60	1.50	382.00	25.00	8.80	6.50	5.90
5000	0.00	0.00	0.00	0.00	0.00	381.00	1.40	0.10	0.00	0.00	367.10	7.70	1.00	0.30	0.30	381.00	22.90	6.10	5.00	4.90

Small number of false positives — *Large number of false positives*

Table 5: Number of false positives (within irrelevant variables) in the parents and children set for features selected by HITON-PC with parameter *max-k*=$\{0,1,2,3,4\}$ on different training sample sizes $\{100,200,500,1000,2000,5000\}$. The color of each table cell denotes number of false positives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

ena happen because the algorithm needs a sufficient number of elements in the $TPC(T)$ set (i.e., tentative parents and children of T) in order to execute conditional independence tests and remove the false positive irrelevant features.

(g) The above trends are remarkably consistent in both networks suggesting that different redundancy and connectivity do not affect the above algorithm behavior.

In the second set of experiments we compare empirically in the above two networks (four variants for each as previously) and 6 sample sizes the following algorithms: semi-interleaved HITON-PC, MMPC, a version of HITON-PC where we pre-filter features by Benjamini FDR control (at FDR rate threshold of 5%) (Benjamini and Yekutieli, 2001), the true $PC(T)$ set extracted from the data generating network (denoted as "True-PC" in Table 6), UAF (univariate association filtering) with Bonferroni correction, UAF with Benjamini FDR control, uncorrected UAF, "wrapped" UAF, RFE, and LARS-EN. Tables 6–9 provide support for the following conclusions:

(h) Due to strength of signal and redundancy of predictors, AUC reaches the theoretical maximum (provided by the generative network) very quickly and for all methods (Table 6).

(i) When no irrelevant features are present and in the stronger signal setting, simple and FDR-corrected UAF (but not wrapped UAF) has the least false negatives in very small samples (Table 7). As sample size grows all methods reduce their false negatives (Table 7). GLL methods pick up the strongly relevant features without false positives and reach near perfect

Lung_Cancer

FS method	*Version 1* (original network) sample size 100	200	500	1000	2000	5000	*Version 2* (original network + irrelevant variables) 100	200	500	1000	2000	5000	*Version 3* (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000	*Version 4* (only irrelevant variables) 100	200	500	1000	2000	5000
True-PC	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.71	0.72	0.73	0.72	0.73	0.75	0.50	0.50	0.50	0.50	0.50	0.50
UAF	1.00	1.00	1.00	1.00	1.00	1.00	0.97	0.99	1.00	1.00	1.00	1.00	0.63	0.67	0.67	0.68	0.69	0.72	0.50	0.51	0.50	0.50	0.49	0.51
UAF+Bonferroni	0.97	1.00	1.00	1.00	1.00	1.00	0.94	1.00	1.00	1.00	1.00	1.00	0.60	0.70	0.74	0.73	0.74	0.74	0.50	0.50	0.50	0.50	0.50	0.50
UAF+FDR	0.98	1.00	1.00	1.00	1.00	1.00	0.95	1.00	1.00	1.00	1.00	1.00	0.61	0.72	0.74	0.74	0.74	0.74	0.50	0.50	0.50	0.50	0.50	0.50
HITON-PC	0.99	0.99	1.00	1.00	1.00	1.00	0.98	0.99	1.00	1.00	1.00	1.00	0.62	0.67	0.73	0.73	0.74	0.74	0.50	0.49	0.51	0.51	0.49	0.49
HITON-PC-FDR	0.97	0.99	1.00	1.00	1.00	1.00	0.94	0.99	1.00	1.00	1.00	1.00	0.61	0.68	0.73	0.73	0.74	0.74	0.49	0.49	0.49	0.49	0.49	0.49
MMPC	0.99	0.99	1.00	1.00	1.00	1.00	0.95	0.99	1.00	1.00	1.00	1.00	0.62	0.67	0.73	0.73	0.74	0.74	0.50	0.49	0.50	0.50	0.50	0.50
LARS-EN	0.91	1.00	1.00	1.00	1.00	1.00	0.97	0.98	1.00	1.00	1.00	1.00	0.64	0.70	0.72	0.71	0.73	0.74	0.49	0.50	0.50	0.50	0.50	0.51
RFE (reduction 50%)	0.97	1.00	1.00	1.00	1.00	1.00	0.97	0.99	1.00	1.00	1.00	1.00	0.61	0.65	0.71	0.70	0.73	0.74	0.50	0.50	0.50	0.50	0.49	0.50
RFE (reduction 20%)	0.95	0.99	1.00	1.00	1.00	1.00	0.96	0.98	1.00	1.00	1.00	1.00	0.60	0.68	0.71	0.71	0.74	0.74	0.50	0.50	0.50	0.49	0.50	0.50
UAF-KW-SVM (50%)	0.98	0.99	1.00	1.00	1.00	1.00	0.97	0.99	1.00	1.00	1.00	1.00	0.65	0.69	0.73	0.71	0.74	0.74	0.50	0.50	0.50	0.50	0.49	0.50
UAF-KW-SVM (20%)	0.95	0.99	1.00	1.00	1.00	1.00	0.96	0.98	1.00	1.00	1.00	1.00	0.66	0.71	0.73	0.72	0.74	0.74	0.50	0.50	0.50	0.49	0.49	0.51
UAF-S2N-SVM (50%)	0.94	0.99	1.00	1.00	1.00	1.00	0.94	0.99	1.00	1.00	1.00	1.00	0.58	0.64	0.74	0.73	0.74	0.74	0.50	0.50	0.50	0.50	0.50	0.50
UAF-S2N-SVM (20%)	0.93	0.98	1.00	1.00	1.00	1.00	0.93	0.98	1.00	1.00	1.00	1.00	0.58	0.67	0.74	0.73	0.74	0.74	0.50	0.50	0.50	0.49	0.50	0.50

Alarm10

FS method	*Version 1* (original network) sample size 100	200	500	1000	2000	5000	*Version 2* (original network + irrelevant variables) 100	200	500	1000	2000	5000	*Version 3* (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000	*Version 4* (only irrelevant variables) 100	200	500	1000	2000	5000
True-PC	0.97	0.96	0.97	0.97	0.97	0.97	0.97	0.96	0.96	0.97	0.97	0.97	0.83	0.82	0.83	0.82	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50
UAF	0.95	0.96	0.96	0.97	0.97	0.97	0.83	0.89	0.93	0.94	0.96	0.97	0.66	0.68	0.71	0.73	0.76	0.81	0.50	0.50	0.50	0.50	0.50	0.50
UAF+Bonferroni	0.95	0.96	0.97	0.97	0.98	0.98	0.94	0.96	0.97	0.97	0.98	0.98	0.81	0.82	0.82	0.82	0.83	0.83	0.50	0.50	0.50	0.50	0.50	0.50
UAF+FDR	0.95	0.96	0.97	0.97	0.98	0.98	0.95	0.96	0.97	0.97	0.97	0.98	0.81	0.82	0.82	0.82	0.83	0.83	0.50	0.50	0.50	0.50	0.50	0.50
HITON-PC	0.95	0.95	0.96	0.97	0.97	0.97	0.92	0.95	0.95	0.96	0.97	0.97	0.69	0.78	0.80	0.81	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50
HITON-PC-FDR	0.94	0.95	0.95	0.97	0.97	0.97	0.94	0.95	0.96	0.96	0.97	0.97	0.80	0.81	0.81	0.81	0.82	0.82	0.49	0.49	0.49	0.49	0.49	0.49
MMPC	0.95	0.95	0.95	0.97	0.97	0.97	0.91	0.95	0.95	0.97	0.97	0.97	0.67	0.77	0.80	0.82	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50
LARS-EN	0.94	0.95	0.97	0.97	0.97	0.97	0.93	0.95	0.96	0.96	0.97	0.97	0.78	0.80	0.81	0.81	0.83	0.82	0.50	0.49	0.50	0.50	0.49	0.49
RFE (reduction 50%)	0.94	0.95	0.96	0.97	0.97	0.97	0.93	0.94	0.95	0.96	0.96	0.97	0.72	0.79	0.81	0.82	0.82	0.82	0.50	0.50	0.50	0.49	0.50	0.50
RFE (reduction 20%)	0.94	0.95	0.96	0.96	0.97	0.97	0.91	0.95	0.95	0.96	0.97	0.96	0.73	0.79	0.81	0.81	0.82	0.83	0.49	0.50	0.50	0.50	0.50	0.50
UAF-KW-SVM (50%)	0.94	0.95	0.96	0.97	0.97	0.97	0.94	0.95	0.96	0.97	0.97	0.97	0.75	0.79	0.81	0.82	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50
UAF-KW-SVM (20%)	0.95	0.96	0.96	0.97	0.97	0.98	0.93	0.95	0.96	0.97	0.97	0.97	0.77	0.79	0.82	0.83	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50
UAF-S2N-SVM (50%)	0.95	0.96	0.96	0.97	0.97	0.97	0.94	0.95	0.96	0.97	0.97	0.97	0.77	0.80	0.81	0.82	0.82	0.83	0.50	0.50	0.50	0.49	0.50	0.50
UAF-S2N-SVM (20%)	0.95	0.96	0.96	0.97	0.97	0.97	0.94	0.95	0.96	0.97	0.97	0.97	0.77	0.78	0.81	0.83	0.82	0.83	0.50	0.50	0.50	0.50	0.50	0.50

Low classification performance — *High classification performance*

Table 6: Classification performance (AUC) of polynomial SVM estimated on 5,000 sample independent testing set for selected features. HITON-PC, HITON-PC-FDR, and MMPC are applied with *max-k*=2. The color of each table cell denotes strength of predictivity with yellow (light) corresponding to low classification performance and red (dark) to high classification performance.

Lung_Cancer

FS method	Version 1 (original network) sample size 100	200	500	1000	2000	5000	Version 2 (original network + irrelevant variables) sample size 100	200	500	1000	2000	5000	Version 3 (weakened signal + irrelevant variables) sample size 100	200	500	1000	2000	5000
UAF	3.3	1.2	0.8	0.3	0.3	0.0	3.3	1.2	0.8	0.3	0.3	0.0	9.4	4.4	1.0	0.8	0.7	0.3
UAF+Bonferroni	13.9	6.1	1.5	1.0	0.9	0.2	17.6	8.4	1.8	1.0	1.0	0.5	24.9	19.9	6.7	2.4	1.0	1.0
UAF+FDR	9.2	2.5	0.9	0.5	0.4	0.0	13.4	4.8	1.3	0.9	0.8	0.0	24.0	16.2	3.5	1.3	1.0	0.8
HITON-PC	18.2	17.7	5.7	1.5	1.0	1.0	18.4	17.7	5.7	1.5	1.0	1.0	23.4	23.2	17.5	6.6	1.8	1.0
HITON-PC-FDR	19.3	18.5	5.7	1.5	1.0	1.0	19.2	18.5	5.7	1.5	1.0	1.0	24.7	23.3	17.9	6.6	1.8	1.0
MMPC	18.5	17.7	5.7	1.5	1.0	1.0	18.9	17.7	5.7	1.5	1.0	1.0	23.4	22.8	17.6	6.6	1.8	1.0
LARS-EN	19.9	14.2	8.8	7.9	3.6	1.0	15.9	18.6	10.0	10.0	3.7	1.6	22.8	21.5	18.3	13.4	9.4	10.7
RFE (reduction 50%)	20.7	15.9	9.4	6.1	4.1	1.0	18.8	14.6	13.3	9.2	3.2	1.6	21.1	15.9	7.6	8.6	14.8	12.8
RFE (reduction 20%)	21.9	17.1	10.5	12.5	4.9	2.6	18.7	18.8	11.0	9.1	3.7	2.3	15.6	18.1	8.3	14.3	16.9	12.3
UAF-KW-SVM (50%)	17.5	16.6	5.9	5.3	1.6	0.7	17.8	15.8	8.6	9.8	5.6	1.5	20.1	14.1	10.9	9.3	8.2	7.3
UAF-KW-SVM (20%)	21.0	18.8	10.5	8.3	2.6	0.7	19.1	18.7	10.7	13.2	6.4	1.2	20.5	14.3	12.4	8.1	6.9	7.2
UAF-S2N-SVM (50%)	20.8	17.1	6.0	7.6	2.5	1.3	17.6	16.7	8.4	7.1	7.0	1.9	16.6	15.4	15.6	11.5	8.3	4.9
UAF-S2N-SVM (20%)	23.1	19.9	9.4	10.5	5.1	1.8	20.5	18.5	10.4	11.3	7.0	0.7	19.4	14.8	15.4	12.3	6.6	5.5

Alarm10

FS method	Version 1 (original network) sample size 100	200	500	1000	2000	5000	Version 2 (original network + irrelevant variables) sample size 100	200	500	1000	2000	5000	Version 3 (weakened signal + irrelevant variables) sample size 100	200	500	1000	2000	5000
UAF	1.7	1.4	0.4	0.1	0.0	0.0	1.7	1.4	0.4	0.1	0.0	0.0	2.2	1.8	0.6	0.8	0.1	0.0
UAF+Bonferroni	4.1	2.7	1.4	1.0	0.5	0.0	4.7	3.2	1.5	1.1	0.7	0.2	5.0	4.4	2.7	1.4	1.0	0.5
UAF+FDR	3.3	2.2	0.8	1.0	0.3	0.0	4.3	2.8	1.4	1.1	0.5	0.0	4.9	3.8	2.4	1.2	0.9	0.2
HITON-PC	4.1	4.0	2.7	2.1	1.5	1.1	4.2	4.0	2.9	2.2	1.5	1.1	5.0	4.7	4.4	3.9	3.6	1.7
HITON-PC-FDR	4.6	4.2	3.2	2.3	1.7	1.0	4.8	4.3	3.2	2.3	1.7	1.0	5.5	4.7	4.4	4.2	3.6	2.1
MMPC	4.1	4.0	3.0	2.4	1.6	1.0	4.3	4.1	3.5	2.4	1.6	1.0	5.0	4.7	4.5	4.2	3.7	2.1
LARS-EN	3.8	3.8	1.7	1.7	1.5	1.4	4.4	4.1	2.5	2.2	1.9	1.4	4.6	4.6	4.6	3.5	2.2	2.0
RFE (reduction 50%)	4.1	3.7	2.1	1.9	2.3	1.5	4.8	4.7	3.2	3.3	2.6	1.8	4.6	4.9	5.2	4.6	4.2	3.6
RFE (reduction 20%)	4.1	3.7	2.4	2.7	2.1	1.8	5.0	4.4	3.4	3.2	2.3	2.0	5.0	5.3	5.0	4.5	3.7	3.3
UAF-KW-SVM (50%)	3.8	3.8	2.2	0.8	0.9	0.4	4.8	3.6	2.4	2.2	1.4	0.1	3.8	4.2	3.4	2.1	2.2	0.8
UAF-KW-SVM (20%)	4.0	3.2	2.4	1.1	0.4	0.0	4.2	3.6	2.4	1.9	1.2	0.0	4.2	4.3	2.7	2.8	1.9	1.2
UAF-S2N-SVM (50%)	3.5	3.6	2.1	1.0	0.8	0.4	4.7	3.8	2.2	2.1	1.5	0.2	5.1	4.4	4.3	3.5	2.7	1.0
UAF-S2N-SVM (20%)	4.3	3.5	2.6	1.3	0.5	0.0	4.9	3.7	2.5	1.9	1.7	0.2	5.0	4.5	3.6	3.0	2.5	1.4

Small number of false negatives — *Large number of false negatives*

Table 7: Number of false negatives in the parents and children set for selected features. HITON-PC, HITON-PC- FDR, and MMPC are applied with *max-k*=2. For Version 4 of the network the parents and children set is empty since there are no relevant variables. The color of each table cell denotes number of false negatives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

separation (i.e., 1-2 false negatives and zero false positives) at sample size 1,000 and higher (Table 8). No other method simultaneously minimizes false positives and false negatives as GLL.

(j) In the setting of strong signal with irrelevant features, simple UAF has the least false negatives in very small samples (Table 7) and the largest number of false positives (Table 8).

(k) When the predictive signal is weaker, false negatives are increased and weakly relevant false positives are decreased for a given sample size compared to the stronger signal case (Tables 7 and 8). Simple UAF is again most sensitive in terms of detecting strongly relevant features in smaller samples until sample size 1,000-2,000 where UAF-Bonferroni and UAF-FDR and GLL match the false negative rates (Table 7). As previously, GLL (with HITON-PC and

Lung_Cancer

FS method	*Version 1* (original network) sample size 100	200	500	1000	2000	5000	*Version 2* (original network + irrelevant variables) 100	200	500	1000	2000	5000	*Version 3* (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000
UAF	65.0	120.5	149.0	202.9	236.1	410.4	65.0	120.5	149.0	202.9	236.1	410.4	62.4	85.6	110.7	123.7	171.1	272.6
UAF+Bonferroni	1.8	8.9	33.6	65.5	91.6	160.3	0.6	4.1	21.2	52.5	80.3	134.3	0.1	0.7	4.8	14.9	43.4	83.6
UAF+FDR	9.4	39.3	78.3	130.5	168.6	359.9	2.7	13.6	46.2	82.6	111.8	230.7	0.1	2.3	13.3	33.5	70.8	123.6
HITON-PC	0.3	0.1	0.0	0.1	0.5	2.6	0.4	0.1	0.0	0.1	0.5	2.6	0.5	0.6	0.4	0.0	0.4	1.1
HITON-PC-FDR	0.2	0.0	0.0	0.1	0.3	1.4	0.1	0.1	0.0	0.1	0.3	1.4	0.1	0.6	0.3	0.0	0.3	0.5
MMPC	0.3	0.1	0.0	0.1	0.5	2.7	0.3	0.1	0.0	0.1	0.5	2.7	0.7	0.8	0.4	0.0	0.4	1.1
LARS-EN	7.5	15.7	5.7	3.7	39.2	59.0	4.6	2.1	4.9	1.1	4.0	25.7	5.4	2.9	3.4	4.4	7.2	3.2
RFE (reduction 50%)	0.7	7.1	13.1	22.0	79.1	123.2	3.1	5.5	1.7	5.8	20.3	24.1	82.9	43.5	170.5	108.2	152.6	96.8
RFE (reduction 20%)	0.4	3.2	12.1	3.0	73.1	167.9	4.8	1.3	5.5	1.9	14.0	22.2	141.5	28.1	115.1	18.8	122.6	112.9
UAF-KW-SVM (50%)	2.0	1.5	76.5	6.8	124.9	172.8	1.7	3.3	14.9	2.6	37.7	120.2	8.8	83.0	24.1	257.0	83.5	97.3
UAF-KW-SVM (20%)	0.6	1.1	4.8	2.5	91.4	179.9	1.0	2.1	14.1	0.7	10.3	124.4	6.4	82.5	22.4	137.8	19.1	46.9
UAF-S2N-SVM (50%)	1.3	1.4	43.1	2.7	114.3	139.8	3.5	2.1	7.1	5.0	26.9	109.5	228.9	98.4	25.4	102.6	86.6	180.0
UAF-S2N-SVM (20%)	0.2	0.4	12.7	1.2	70.1	128.1	1.0	1.5	5.3	1.6	22.3	120.8	153.4	117.5	19.5	53.8	93.1	175.8

Alarm10

FS method	*Version 1* (original network) sample size 100	200	500	1000	2000	5000	*Version 2* (original network + irrelevant variables) 100	200	500	1000	2000	5000	*Version 3* (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000
UAF	22.1	26.5	32.2	30.2	33.5	38.0	22.1	26.5	32.2	30.2	33.5	38.0	22.5	25.2	32.0	27.1	32.4	37.3
UAF+Bonferroni	4.4	4.8	7.4	8.6	10.7	14.6	3.3	4.4	6.0	8.0	9.2	13.1	1.5	3.1	4.9	6.7	7.7	10.3
UAF+FDR	5.0	6.2	9.7	10.1	14.3	20.1	3.9	4.8	7.2	8.6	10.7	14.6	1.8	3.8	5.4	7.3	8.7	12.2
HITON-PC	3.7	0.8	0.1	0.0	0.3	0.3	2.4	0.5	0.1	0.0	0.3	0.3	1.8	0.9	0.2	0.1	0.6	0.2
HITON-PC-FDR	0.9	0.5	0.0	0.1	0.1	0.0	0.7	0.4	0.1	0.1	0.1	0.0	0.7	0.6	0.2	0.2	0.2	0.3
MMPC	3.7	0.8	0.2	0.3	0.4	0.1	2.6	0.5	0.2	0.2	0.4	0.1	2.6	0.7	0.3	0.4	0.5	0.3
LARS-EN	20.7	9.4	56.1	24.7	17.2	36.7	3.2	3.0	3.9	4.1	3.9	9.1	1.0	1.6	2.3	3.3	3.4	4.9
RFE (reduction 50%)	16.7	18.6	114.9	68.9	23.7	36.9	2.0	1.3	3.5	2.9	1.5	3.7	19.7	1.4	1.3	1.6	1.9	2.9
RFE (reduction 20%)	11.3	18.1	56.0	9.8	19.7	38.7	2.5	0.9	1.9	2.5	1.7	3.3	11.6	0.9	0.8	1.1	1.5	2.7
UAF-KW-SVM (50%)	13.5	4.0	32.6	51.4	49.7	35.9	3.4	3.4	5.6	5.4	9.1	15.4	13.7	3.7	4.4	5.7	7.6	10.6
UAF-KW-SVM (20%)	5.7	5.4	10.2	42.3	37.5	58.7	3.3	3.1	5.4	5.7	8.8	14.7	5.6	3.3	4.9	5.2	7.3	9.0
UAF-S2N-SVM (50%)	18.6	4.3	72.3	55.0	37.5	38.2	2.0	3.3	8.1	5.9	8.9	14.6	1.4	2.3	2.7	4.2	6.0	9.8
UAF-S2N-SVM (20%)	7.1	4.1	44.6	17.8	38.2	40.1	1.9	3.8	5.0	6.1	8.1	13.1	1.4	2.8	3.2	4.6	6.5	8.8

Small number of false positives *Large number of false positives*

Table 8: Number of false positives (within weakly relevant variables) in the parents and children set for selected features. HITON-PC, HITON-PC-FDR, and MMPC are applied with *max-k*=2. For Version 4 of the network there are no weakly relevant variables. The color of each table cell denotes number of false positives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

MMPC performing similarly) achieves excellent false positive rates better than those by FDR not only for weakly relevant but also for irrelevant features.

(l) HITON-PC augmented with FDR pre-filtering behaves almost identically as regular HITON-PC except for the case with only irrelevant features in the data where HITON-PC without FDR admits a few false positives (Table 9).

(m) State-of-the-art feature selection methods are prone to select very large numbers of irrelevant features (Table 9).

In conclusion, HITON-PC and by extension GLL algorithms (since the same fundamental mechanisms for variable inclusion and elimination are shared because of the GLL-PC template and admissibility requirements), have a very strong built-in capacity to control for false positives due to multiple comparisons. False positives due to multiple comparisons quickly vanish for *max-k* 1 or

Lung_Cancer

FS method	Version 1 (original network) sample size 100	200	500	1000	2000	5000	Version 2 (original network + irrelevant variables) 100	200	500	1000	2000	5000	Version 3 (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000	Version 4 (only irrelevant variables) 100	200	500	1000	2000	5000
UAF	0.2	1.5	0.2	0.5	0.8	0.7	411.6	488.6	446.0	422.7	409.0	403.1	488.8	471.6	424.9	413.2	407.9	397.8	411.6	488.6	446.0	422.7	409.0	403.1
UAF+Bonferroni	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.1	0.1	0.0
UAF+FDR	0.0	0.1	0.1	0.0	0.1	0.2	0.5	1.1	3.7	4.8	6.7	12.5	0.2	0.7	1.4	3.2	4.8	8.0	0.1	0.0	0.0	0.1	0.1	0.1
HITON-PC	0.0	0.0	0.0	0.0	0.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	8.6	2.9	0.9	0.2	0.4	0.0	9.8	5.8	6.4	6.9	6.1	6.2
HITON-PC-FDR	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.2	0.4	0.3	0.1	0.2	0.0	0.1	0.0	0.0	0.1	0.1	0.1
MMPC	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.0	0.0	0.0	0.0	0.0	8.7	3.4	0.8	0.2	0.4	0.0	8.8	6.4	7.2	6.3	6.7	6.3
LARS-EN	0.4	0.1	0.1	0.0	0.9	0.5	32.8	11.5	29.9	1.4	6.2	52.2	53.2	20.8	32.9	47.3	82.0	39.2	35.9	33.5	69.3	63.0	69.6	38.7
RFE (reduction 50%)	0.0	0.0	0.2	0.5	1.4	1.8	24.8	20.5	4.4	10.7	68.5	78.4	868.9	449.1	1741.5	1132.5	1600.0	1005.6	462.4	1084.1	971.3	918.2	1844.9	223.7
RFE (reduction 20%)	0.0	0.0	0.2	0.0	1.4	3.2	28.3	3.5	21.0	4.9	49.3	78.1	1548.4	252.6	1145.1	183.0	1282.2	1171.0	531.2	106.0	1488.3	103.8	849.0	112.7
UAF-KW-SVM (50%)	0.0	0.0	0.7	0.0	0.5	1.6	1.8	0.2	13.9	0.0	23.5	95.5	56.3	798.3	111.0	2593.7	800.6	801.1	809.3	319.6	1161.8	193.0	1676.1	886.0
UAF-KW-SVM (20%)	0.0	0.0	0.0	0.0	0.5	0.4	1.3	0.1	11.9	0.0	0.0	76.9	47.4	816.0	108.9	1215.2	3.4	12.4	971.0	346.6	1061.1	72.3	1283.0	870.6
UAF-S2N-SVM (50%)	0.0	0.0	0.5	0.0	0.7	0.2	29.8	5.8	3.5	0.0	5.6	49.9	2420.2	911.2	193.5	993.5	803.4	1604.8	676.2	1414.0	1666.8	2540.1	2491.3	1032.9
UAF-S2N-SVM (20%)	0.0	0.0	0.1	0.0	0.3	0.3	7.7	3.3	2.8	0.0	16.5	95.9	1624.5	1077.3	128.1	416.3	805.6	1608.2	1036.1	537.2	819.0	236.2	1279.7	990.0

Alarm10

FS method	Version 1 (original network) sample size 100	200	500	1000	2000	5000	Version 2 (original network + irrelevant variables) 100	200	500	1000	2000	5000	Version 3 (weakened signal + irrelevant variables) 100	200	500	1000	2000	5000	Version 4 (only irrelevant variables) 100	200	500	1000	2000	5000
UAF	0.0	0.0	0.0	0.0	0.0	0.0	392.1	412.9	411.6	414.1	382.0	381.0	408.7	427.8	417.9	399.9	380.0	367.1	392.1	412.9	411.6	414.1	382.0	381.0
UAF+Bonferroni	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.1	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.1	0.0
UAF+FDR	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.4	0.7	0.8	1.0	1.2	0.4	0.2	1.0	1.0	0.6	1.4	0.0	0.0	0.2	0.0	0.1	0.0
HITON-PC	0.0	0.0	0.0	0.0	0.0	0.0	22.8	3.8	0.8	0.9	0.0	0.1	26.4	6.5	4.4	3.3	1.8	1.0	23.4	9.7	6.8	7.2	8.8	6.1
HITON-PC-FDR	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.0	0.3	0.2	0.5	0.2	0.1	0.0	0.0	0.0	0.2	0.0	0.1	0.0
MMPC	0.0	0.0	0.0	0.0	0.0	0.0	29.0	3.4	1.2	1.0	0.0	0.1	37.2	5.9	4.4	3.2	2.8	1.0	26.8	10.4	8.5	7.8	9.3	5.8
LARS-EN	0.0	0.0	0.0	0.0	0.0	0.0	34.2	31.7	32.0	21.3	7.7	54.6	16.3	12.9	19.2	45.2	44.6	38.3	55.6	23.8	32.9	9.8	31.9	73.5
RFE (reduction 50%)	0.0	0.0	0.0	0.0	0.0	0.0	12.2	2.9	32.0	18.7	6.4	25.3	385.9	13.1	5.1	7.3	16.2	33.0	502.6	1819.2	80.6	1940.8	965.3	1090.3
RFE (reduction 20%)	0.0	0.0	0.0	0.0	0.0	0.0	11.7	2.6	8.3	23.3	7.2	20.8	201.2	7.4	2.2	8.5	16.7	30.8	264.8	732.4	735.0	633.0	790.6	237.2
UAF-KW-SVM (50%)	0.0	0.0	0.0	0.0	0.0	0.0	16.2	4.0	1.9	2.0	16.0	39.3	240.9	34.1	3.0	8.6	11.8	29.7	204.9	142.9	447.9	2604.5	811.1	1504.0
UAF-KW-SVM (20%)	0.0	0.0	0.0	0.0	0.0	0.0	15.3	2.1	0.9	5.5	13.6	35.8	74.0	23.4	3.0	3.2	8.8	11.3	219.5	102.5	124.6	1554.9	784.1	1230.1
UAF-S2N-SVM (50%)	0.0	0.0	0.0	0.0	0.0	0.0	3.1	5.2	30.2	2.5	20.0	37.2	9.6	2.7	1.1	2.9	3.3	26.4	615.1	32.7	265.8	1901.0	759.3	721.5
UAF-S2N-SVM (20%)	0.0	0.0	0.0	0.0	0.0	0.0	3.1	7.2	3.9	2.4	14.5	23.1	12.3	17.1	2.9	3.7	4.1	10.1	291.8	772.0	392.7	701.5	120.9	870.6

Small number of false positives — *Large number of false positives*

Table 9: Number of false positives(within irrelevant variables) in the parents and children set for selected features. HITON-PC, HITON-PC-FDR, and MMPC are applied with *max-k*=2. The color of each table cell denotes number of false positives with yellow (light) corresponding to smaller values and red (dark) to larger ones.

higher *regardless of sample size*. Given enough sample size (∼1,000 or more in the data tested), and by choosing 5% as the nominal α for all conditioning independence tests executed, the algorithm fully eliminates irrelevant features from its output without incurring a penalty in false negatives, even when irrelevant features are the majority among observed features. Parameter *max-k* controls the false positives due to both weakly relevant and irrelevant features. The false positive rate in this worst-case situation is in the presented experiments $\sim 5/8{,}000 = 0.000625$ which is much better than what the conservative Bonferroni-adjusted α guarantees, and *without* incurring false negatives (as both Bonferroni and FDR methods do). Both established feature selectors such as variants of UAF and newer ones are very sensitive to irrelevant features and produce large numbers of false positives. Given the attractive characteristics of FDR-augmented HITON-PC, we evaluate it with real data sets in Section 5.

4. Theoretical Analysis of GLL

In the present section we provide a theoretical analysis of the Generalized Local Learning algorithms.

4.1 Determinants of Quality of Statistical Decisions and Computational Tractability. Parameters *max-k* and *h-ps*

On a rather superficial level when conditioning sets are large enough, statistical tests become less reliable. For example, as explained in Aliferis et al. (2010), cells in contingency tables used to calculate p-values of discrete tests of independence (such as the widely-used G^2 or X^2 test) become scarcely populated and this leads to unreliable test results. This motivates the heuristic practice of considering as unreliable and not executing a test in which the sample size is less than: ("number of cells to be fitted" $\cdot$ *h-ps*), with parameter *h-ps* set to 10 by default in the PC algorithm (Spirtes et al., 2000) and 5 in GLL instantiations. Recall from Aliferis et al. (2010) that *h-ps* stands for "heuristic power size" and denotes the smallest sample size per cell in the contingency table of a reliable conditional test of independence. Moreover, when the conditioning set size is large enough to block all paths between a weekly relevant variable and the target, there is no need to exceed this conditioning set size because the resulting tests are redundant and the operation of the algorithm becomes unnecessarily slow. Thus it seems reasonable that we would wish to restrict the conditioning set size to not exceed this sufficient blocking size. This is accomplished by setting the value of parameter *max-k*. We will see however that *max-k* has a much more elaborate function than simply "trimming away" excessive computations.

In reality things are significantly more complicated because, as first pointed out by Spirtes et al. (2000), statistical reliability of a single test is a misleading concept in the context of complex constraint-based algorithms such as GLL. Standard statistical considerations of the type of testing a hypothesis once do not carry over well to the constraint-based algorithm setting. Similarly, running time is also a complex function of direct or indirect restrictions placed on number of tests and the number of variables with which to build such tests (i.e., the size of $TPC(T)$).

We first explain what happens when running semi-interleaved HITON-PC in faithful distributions (same arguments can be generalized to other GLL-PC and GLL-MB versions). Consider first that in the case of a strongly relevant feature S, when conducting just one test $\mathrm{I}(S, T|\emptyset)$ for the purposes of inclusion of S in $TPC(T)$, regardless of how small power is, we should always execute this test because the worst that can happen is that we fail to include S in $TPC(T)$, whereas if we do not

execute the test and assume independence by default, we will surely miss it. In the context of *many tests* however, the notion of single-test reliability for S no longer applies. For example, when we consider a test that has the potential to reject S from $TPC(T)$ (where it was placed previously by a *different* test), by allowing the conditioning test size to grow large, the power is reduced (assuming monotonic association of S through the potentially multiple paths connecting S with T). Hence, we need to preserve the combined power (i.e., combination of individual powers of all tests applied to S) in order to not eliminate S from $TPC(T)$. Although these tests are highly correlated and combined power is larger than the product of powers of the same set of tests performed on independent samples, still the more tests are executed the smaller the combined power and the larger the possibility of falsely eliminating S becomes. The parameter *h-ps* partially controls power because the larger it is, the smaller number of tests (that would eliminate S) are executed. However *h-ps* should not be too large either because a strongly relevant S will not be included in $TPC(T)$ in the first place. Parameter *max-k* also controls in part the number of tests allowed. *Max-k* does not fully determine the number of tests because it specifies the dimensionality of allowed tests, not their total number. As *max-k* grows, more tests for eliminating S from $TPC(T)$ are executed, thus the combined power drops. In summary, for a given distribution the number of tests performed is affected by *h-ps*, *max-k* and the size of $TPC(T)$.

So far the discussion has centered on one type of conditional independence test, that is, tests where the candidate member of $PC(T)$, X, is a strongly relevant feature (type 1). This is the first of four types of conditional tests. The other three are: conditional independence tests where the candidate member of $PC(T)$, X, is a weakly relevant feature and some paths with T are not blocked by the conditioning set (type 2a), conditional independence tests where the candidate member of $PC(T)$, X, is a weakly relevant feature and all paths with T are blocked by the conditioning set (type 2b), and finally conditional independence tests where the candidate member of $PC(T)$, X, is an irrelevant feature (type 3).

The quality of conditional tests of the first type is determined by the *power* of the association of X with T given the conditioning set. Since not one but potentially many such tests are conducted, the combined power of all such tests determines whether X will be selected and stay in the $TPC(T)$ set. For example, variable X (a true member of $PC(T)$) will be considered for inclusion in $TPC(T)$ by HITON-PC with probability = power of detecting $\neg \mathrm{I}(X,T)$ given the available sample size and test employed. However for X to stay in $TPC(T)$ until the algorithm terminates, and assuming B, C have entered $TPC(T)$, none of the tests $\mathrm{I}(X,T|B)$, $\mathrm{I}(X,T|C)$, $\mathrm{I}(X,T|\{B,C\})$ must conclude independence. The power or each one of these tests can be lower or higher than the power of $\mathrm{I}(X,T)$ and the combined power can quickly diminish, however several mitigating factors prevent this from happening. First, when using linear tests under common distributional assumptions such as multivariate normality, the necessary sample size to achieve desired level of power grows linearly to number of variables in the conditional set. Second, as explained earlier, conditional independence tests of the same variable and T in the same sample are highly correlated. Third, controlling the number of members of $TPC(T)$ by a good heuristic inclusion function reduces the total number of tests; such control occurs indirectly by putting first the true members of $PC(T)$ or members that block many variables. Fourth, the order of executing the tests and constructing conditioning sets is important for reducing the number of tests performed on strongly relevant variables. This is exemplified in semi-interleaved HITON-PC where new entrants in $TPC(T)$ are tested before current $TPC(T)$ members thus if the heuristic inclusion function is a good one, strongly relevant members are tested a smaller number of times at the elimination phase.

Returning our attention to the quality of statistical decisions for weakly relevant variables, we observe that when a conditioning set *does not* block all paths to/from T either for inclusion or for elimination purposes (type 2a), we are sampling under the alternative hypothesis (i.e., there exists association) and the determining factor for failing to reject the weakly relevant feature is the combined power which is determined by the same factors as elaborated for strongly relevant variables previously. The combined probability for rejection may be small for similar reasons as type 1 conditional independence tests (albeit higher than for strongly relevant features due to the fact that under a good inclusion heuristic weakly relevant features enter $TPC(T)$ later than strongly relevant ones and thus more tests are applied on each weakly relevant than on each strongly relevant feature on average).

However, when the conditioning set blocks all paths from/to T (type 2b), *then we sample under the null hypothesis* and the determining factor shifts from the combined power to the *combined* α (i.e., statistical significance). Given that the α for each conditional test is typically low (i.e., 5% or smaller) and that as the number of tests under the null increases, the combined α drops up to exponentially fast, and eliminating weakly relevant features occurs with high probability as the number of applied tests increases. In HITON-PC, the smaller is *h-ps*, the easier it is to include a weakly relevant feature (based on univariate association heuristic), whereas *max-k* does not affect this function. In terms of rejecting a weakly relevant feature in $TPC(T)$, the larger *max-k* and the smaller *h-ps* become, the easier it is to eliminate a weakly relevant feature.

The quality of statistical decisions for type 3 of conditional independence tests, that is for irrelevant variables, is determined by the combined α since we *always* test under the null hypothesis. Because the combined α drops fast as the number of tests applied to each irrelevant variable (and these tests are abundant when even a handful of variables have been admitted in $TPC(T)$), the combined probability for admitting and not rejecting irrelevant variables is exceedingly small. However when no strongly (and thus no weakly) relevant feature exists, conditioning sets inside the $TPC(T)$ set become smaller as irrelevant variables are eliminated from it with the end result of leaving a small number of "residual" irrelevant features in the final output as evidenced in the simulation experiments of Section 3. By pre-filtering variables with an FDR filter (Benjamini and Yekutieli, 2001; Benjamini and Hochberg, 1995), we not only gain the security that if the data consists exclusively of irrelevant variables fewer or no false positives will be returned, but also we can use *max-k* to control sensitivity and specificity trading weakly relevant false positives for strongly relevant true positives and vice versa (i.e., without worrying about adversely trading off irrelevant features).

Finally, the total number of tests is determined by both parameters *h-ps* and *max-k*, in a non-monotonic manner. That is, whenever *h-ps* is extremely large it effectively disallows most tests and the algorithm quickly terminates returning the empty set regardless of *max-k*. For medium/small values of *h-ps*, more tests are executed, more variables enter $TPC(T)$, and many tests are executed before $TPC(T)$ is finalized. *Max-k* modifies this number by potentially restricting the number of tests. When *h-ps* is very small, tests are allowed with very large conditioning tests and as long as *max-k* does not disallow them, the total number of tests grow very large.

	Number of conditional independence tests			*Cost of conditional independence tests*			*Number of false positives (fp) and false negatives (fn)**	
Lung_Cancer	max-k	**HITON-PC**	**MMPC**	max-k	**HITON-PC**	**MMPC**	max-k	# of fn	# of fp
	1	4,028	5,683	1	7,257	8,900	1	1	13
Target variable #1	2	12,328	14,577	2	33,018	38,892	2	1	0
Number of members in	3	73,554	77,885	3	277,922	294,211	3	1	0
PC set = 26	4	250,560	259,099	4	1,181,889	1,225,682	4	3	0
Alarm10	max-k	**HITON-PC**	**MMPC**	max-k	**HITON-PC**	**MMPC**	max-k	# of fn	# of fp
	1	457	490	1	545	585	1	1	2
Target variable #199	2	470	496	2	608	652	2	1	0
Number of members in	3	491	521	3	692	752	3	1	0
PC set = 6	4	496	527	4	717	782	4	1	0

* Results are same for HITON-PC and MMPC for number of false positives and false negatives

Figure 2: Efficiency of HITON-PC versus MMPC.

4.2 Efficiency and Heuristic Robustness of HITON-PC Versus MMPC

Figure 2 presents the number and cost[2] (proportional to time) of conditional independence tests performed by semi-interleaved HITON-PC versus MMPC in the 2,000-sample data set from the *Alarm10* and *Lung_Cancer* networks. As can be seen, HITON-PC performs fewer tests on average while achieving the same performance as MMPC. We notice that the max-min association heuristic closely reflects the logic behind the combined probability for error for the weakly relevant features. MMPC when testing under the alternative hypothesis (i.e., strongly relevant features, or unblocked weakly relevant ones) requires measuring all relevant associations, whereas HITON requires just the univariate ones *for inclusion purposes*. However semi-interleaved HITON tries to eliminate the newly included variable immediately upon inclusion and thus effectively conducts a similar number of tests as MMPC. Both algorithms when testing under the null hypothesis (irrelevant or fully-blocked weakly relevant features) on average execute the same number of tests. The max-min association inclusion heuristic is a priori more prone to basing its decisions for inclusion in $TPC(T)$ on less statistically reliable criteria. This is because the more associations are considered and the larger the conditioning sets are, the higher variance in the minimum association estimates is expected, making the maximum of such associations over all variables considered more prone to sampling error (i.e., it is likely to be overfitted to the sample). Because of better robustness of the univariate association relative to the weakest association over many conditional associations true members of $PC(T)$ may enter the $TPC(T)$ set earlier. However both HITON-PC and MMPC exhibit similar performance in real and simulated data sets, demonstrating that the theoretical problem with max-min association is in practice very rare.

4.3 Synthesis and Problems for Inclusion Heuristics; Constructing New Inclusion Heuristics

A problem when inducing local neighborhoods and particularly Markov blankets is that of *information synthesis*. The problem consists of a variable X that is not in $PC(T)$ having higher association (univariate or conditional on some subsets) with T than members of $PC(T)$ (for a concrete example see Figure 13). We will call such variables, *synthesis variables*. Synthesis variables were identified

2. The cost of a conditional independence test is calculated as the number of variables participating in it (excluding target variable). For example, univariate tests have cost = 1, tests with conditioning on two variables have cost = 3.

as major problems for algorithms such as IAMB (Tsamardinos and Aliferis, 2003; Tsamardinos et al., 2003a) or GS (Margaritis and Thrun, 1999) that induce Markov blankets and do so by conditioning in their inclusion phase on all variables in the tentative $MB(T)$. Because of the requirement to condition on all variables in the tentative $MB(T)$, the sample requirements grow exponentially fast to the size of the tentative $MB(T)$ and thus it is absolutely imperative to keep out of it synthesis variables since they unnecessarily increase the sample requirements to the point that the algorithm may need to stop executing conditional independence tests (and either halt or output the tentative $MB(T)$ as best but flawed estimate of the true $MB(T)$).

With regards to GLL algorithms, most efficient operation is achieved when the variables that alone or in combination have the property that block the largest fraction of weakly relevant variables, enter first in $TPC(T)$ (even if they are not strongly relevant themselves). Synthesis variables may or may not have this property, so synthesis may or may not be a problem for a specific GLL algorithm based on characteristics of the specific data in hand.

Construction of new inclusion heuristics may be required in difficult cases where the univariate and max-min heuristics do not work well leading to very slow processing time and very large $TPC(T)$ sets, in order to make operation of local learning tractable. In practice, both the univariate and max-min association heuristics work very well with real and simulated data sets, so we do not pursue here implementation and testing new heuristics in artificial problems, although we recognize the possibility of such need in future problematic data distributions. We outline here, in broad strokes, general strategies for creating new inclusion heuristics for such cases:

1. *Random heuristic search informed by standard heuristic values*. This strategy is based on using one of the usual heuristics to rank candidate variables and making selection decisions based on random selection of a candidate variable with probability proportional to the original heuristic value. This enables using the older heuristic as a starting point but allowing occasionally deviations from it to explore the possibility that lower-ranked candidates may have better potential as blocking variables. A simulated-annealing determination of probability of selection (or other efficient stochastic search algorithms) can be pursued as well.

2. *Constructing new heuristic functions by observing blocking capability* (in terms of candidate variables blocked by conditioning sets in which V is a member) or *probability of a variable V to remain in $TPC(T)$*. The empirical observations can be collected from a variety of tractable sources: either from a single incomplete run of the algorithm (i.e., without waiting to terminate), or in other data sets characteristic of the domain, or in multiple runs on smaller (randomly chosen) subsets of the original feature set. The new heuristic function F can be constructed as the conditional probability:

$$F(V_i) = P(V_i \in TPC(T) | h(V_i))$$

where $h(V_i)$ is the original heuristic value of variable V_i, or the proportion of candidates blocked by a conditioning set containing V_i:

$$F(V_i) = \sum_{k=1}^{M} N_k(V_i)/M$$

where $N_k(V_i)$ is the number of candidate variables blocked by a conditioning set that contains variable V_i in trial k.

3. *Exploiting known domain structure*. When properties of the causal structure of the data generating structure and/or distributional characteristics are known, one can use this information alone or in conjunction with the previous two strategies to derive more efficient heuristics.

We note that developing an inclusion heuristic that leads to efficient execution of GLL is not always feasible since the very problem of finding the features with direct edges with the target is intractable in the worst case (e.g., consider a graph that is fully connected). In some cases, as we will show in Section 6, *it is possible to transform an intractable local learning problem into a tractable one by employing a global learning strategy (i.e., exploiting asymmetries in connectivity)*.

4.4 Inductive Bias of GLL

Informally the inductive bias of GLL is that it seeks a balance of false negatives for strongly relevant variables with false positives for weakly relevant and irrelevant variables. The main regulating parameters (for standard inclusion heuristics, elimination and interleaving strategies) are *h-ps* and *max-k*. In practice, the algorithms tested in our work to date reveal higher sensitivity to *max-k* and thus at first approximation we treat optimization of this parameter as having higher priority. Smaller *max-k* empirically decreases false negatives and increases false positives overall. Larger *max-k* increases the false negatives and decreases the false positives. GLL in moderate to large samples achieves small numbers of false negatives and small numbers of false positives. In very small samples GLL prefers false positive errors than false negative ones when *max-k* is small. This occurs because given *some* evidence in favor of $PC(T)$ membership (provided by lower-dimensional and thus more sample efficient) tests of a variable X but *no reliable proof* to the contrary (provided by omitted higher-dimensional and thus unreliable tests), the algorithm outputs X as member of $PC(T)$. A similar behavior exists for the $MB(T)$ versions (with respect to $MB(T)$ membership). Notice that as *max-k* grows many more tests can be executed provided that a liberal *h-ps* is chosen, and these tests can be used to eliminate both weakly relevant as well as strongly relevant features in $TPC(T)$. The choice of a more liberal *h-ps* default value in GLL (compared to the more stringent value in the published implementation of PC algorithm) allows a more effective control of the tradeoff between false positives and false negatives in small samples by changing values of *max-k*.

By contrast, the SGS and PC algorithms (Spirtes et al., 2000) given *no evidence* in favor of membership of X in $PC(T)$ and *no reliable proof* to the contrary, assumes that X has a common edge with T. IAMB (Tsamardinos and Aliferis, 2003; Tsamardinos et al., 2003a) to the contrary, given *some* reliable evidence in favor of a variable X belonging to $MB(T)$ but *no reliable proof* to the contrary, outputs X as member of $MB(T)$ if X is in the tentative Markov blanket $TMB(T)$ and is agnostic with respect to membership in $MB(T)$ if X is outside $TMB(T)$. Bayesian scoring methods in small samples are dominated by their priors and typically they prefer sparse networks which lead to fewer false positives and more false negatives.

4.5 Reasons for Good Performance of Non-Symmetry Corrected Algorithms

The empirical evaluations in part I of this work (Aliferis et al., 2010) have shown that the addition of symmetry correction adds little to quality, while it detracts from computational efficiency. Evidently very often $EPC(T) \approx PC(T)$ in real-life distributions and targets of interest. In addition, due to imperfect power to detect and return strongly relevant features, applying symmetry correction leads to reduced power and increased false negatives.

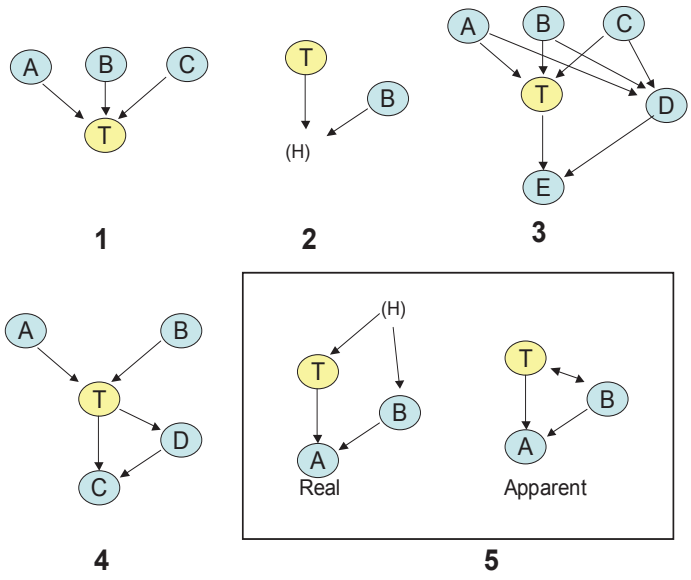


Figure 3: Scenarios explaining good empirical performance of $PC(T)$ set for classification.

4.6 Reasons for Good Performance of the $PC(T)$ Set Instead of the $MB(T)$ Set for Classification

According to the theoretical results summarized in Aliferis et al. (2010), under broad assumptions spouses are needed for optimal classification performance. Given that in the majority of data sets tested in Aliferis et al. (2010) as well as the experiments in Section 2 of the present paper, when the set of parents and children is used instead of $MB(T)$ it produces equal or almost equal performance, more compact feature sets and faster feature selection times than inducting the full $MB(T)$ (i.e., both $PC(T)$ and $MB(T)$ estimated under the same assumptions of the theory that predicts that $MB(T)$ is needed for optimal feature selection). In this sub-section we provide likely explanations for the empirically excellent performance of substituting the set $PC(T)$ in place of $MB(T)$ for classification (apart from the obvious possibility that spouses may be much fewer and with smaller predictive value than parents and children). Figure 3 describes visually five plausible scenarios explaining the phenomenon.

The first scenario corresponds to the situation whereby the target variable T does not have children (and thus no spouses) by virtue of domain constraints. Such situations happen when the target variable is a variable preceded in time by all other variables (e.g., patient outcome on the basis of earlier observations); or when naturally the target variable cannot have children (e.g., the target being meaning category of a text document as a function of patterns of presence/absence of words in the text). The second scenario describes the situation where a child is not observed (hidden) in the data set and thus the spouse B cannot be made informative for the target and thus it can neither be detected nor can it enhance a classifier built from the data. The third scenario describes the situation where a spouse has connecting paths to the target but these cannot be blocked simultaneously because of small sample size and/or choice of *max-k*. Hence GLL-PC could admit the spouse D as a member of $PC(T)$. The fourth scenario simply shows a case where a spouse is also a child (or parent) and thus will be a member of $PC(T)$ as well as $MB(T)$. Finally the fifth

scenario shows that an unmeasured variable may make a spouse appear as having a direct edge to or from the target (and thus are detectable by GLL-PC).

We note that in practical data analysis and evaluations when both $PC(T)$ and $MB(T)$ are induced and are found to have similar classification performance, typically $MB(T)$ is much larger than $PC(T)$. However this may be a reflection of the inductive bias of GLL which prefers to admit potential false positives if they cannot be shown for sample size reasons to be independent of the target.

Finally note that explanations #1, 2, 3, and 4 are special cases of the assumptions of the Markov blanket induction theory and thus they do not refute these assumptions (whereas #5 violates causal sufficiency). In the discussion section we consider additional situations with violations of GLL assumptions.

4.7 Error Estimation Problems in Wrapping and Standard Filters Due to Small Sample Size. GLL Filtering is Less Sensitive to Error Estimation Difficulties and Robust to Small Samples

Wrapping has been praised as a feature selection methodology for its ability to tailor the feature selection to the inductive bias of the classifier(s) of choice as well as to the loss function of interest (Kohavi and John, 1997). Occasionally, this property will work against the analysis (see Section 7 for example for how it can jeopardize causal discovery). On the other hand, wrapping has been criticized for its very large computational cost as well as on the grounds that it is subject to No Free Lunch Theorem limitations (i.e., a priori all wrappers are equally good, making it hard to find the right wrapper for the distribution, loss function and classifier(s) of interest) (Tsamardinos and Aliferis, 2003). In the present section we explain what we believe is perhaps the most serious practical shortcoming of wrapping feature selection methods, namely that *they rely on error estimation procedures that are often unreliable because of small sample sizes*. The difficulties that will be presented here help explain the sometimes poor performance of some of the feature selection algorithms in the evaluation part (Aliferis et al., 2010). In contrast, we will show that GLL filtering is resistant to these problems.

Recall that the critical point when applying error estimators is to have a sufficiently small variance and to be unbiased or to correct for any bias, as for example is the case of the (biased) Bootstrap estimator. Consider an idealized example where a greedy (steepest-descent) backward selection wrapper algorithm is applied on faithful data that contains 5 irrelevant features $I_1, \ldots, I_5$ and one strongly relevant feature S.

Assume that in reality the optimal feature set consisting of only the strongly relevant feature S gives a predictor model with true error measured by AUC is 0.75 in the large sample (i.e., in the distribution where the data is sampled from). For all practical unbiased error estimators, because of variability in the estimates of error due to small sample sizes, and because of potential sensitivity of the classifier employed to irrelevant features, some subsets that contain S will have error estimates in small sample situations that are larger and some smaller than the true AUC of 0.75. The backward wrapping starts by eliminating one variable at a time producing feature sets and corresponding predictor models and by eliminating the feature that decreases error the most relative to the starting model that contains all features. As a result, a feature set can be chosen, not because the error is truly decreased if we remove any more features, but because the error estimates vary and the backward wrapper (naively) does not take this into account. If the wrapper is configured

Action	Decision	Notes
Rank variables according to univariate association with target T	S (association = 0.8) I_1 (association = 0.3) I_2 (association = 0.1) I_3 (association = 0.1) I_4 (association = 0.05) I_5 (association = 0.0)	Some associations of irrelevant variables are non-zero due to sampling variation
Test S for inclusion: $\neg I(S, T)$	Admit S in $TPC(T)$	Assuming S is a strong predictor of the target, the power of the univariate test will be sufficient to reject independence
Test I_1 for inclusion: $I(I_1, T)$	Eliminate I_1	Test will be correct with probability 1-α (typically 0.95)
Test I_2 for inclusion: $I(I_2, T)$	Eliminate I_2	Test will be correct with probability 1-α (typically 0.95)
Test I_3 for inclusion: $\neg I(I_3, T)$	Consider I_3	Assume we were unlucky and had a false positive
Test I_3 for inclusion: $I(I_3, T \mid S)$	Eliminate I_3	Test will be correct with probability 1-α (typically 0.95). Very unlikely (probability = 0.0025) that I_3 will pass through second test
Test I_4 for inclusion: $I(I_4, T)$	Eliminate I_4	Test will be correct with probability 1-α (typically 0.95)
Test I_5 for inclusion: $I(I_5, T)$	Eliminate I_5	Test will be correct with probability 1-α (typically 0.95)
Test S for final elimination: no test to be made	Accept S	
Return $\{S\}$ as final output		

Table 10: Trace of semi-interleaved HITON-PC without symmetry correction (i.e., GLL-PC-nonsym subroutine) showing insensitivity to error estimation difficulties that affect wrappers.

to employ statistical significance tests each time it compares estimates of error between pairs of feature sets and corresponding classifiers, because statistical tests of error estimate differences are often underpowered (which is another manifestation of the large variance in error estimates) such tests will often fail to reveal true differences. Thus the wrapper can falsely conclude that two models have same error when in reality they do not. This will entail choosing wrongly the smallest of the two and eliminating valuable features. Also due to multiple comparisons, such an algorithm will falsely conclude for a proportion of feature sets that a difference in predictor model performance is statistically significant thus continuing removal of relevant features when they should not be removed.

We emphasize that this problem is not present in wrapper methods only. In traditional feature ranking methods, the above problem is also present but often ignored in the sense that many studies on feature ranking algorithms produce a performance-to-feature-number plot, with performance estimated on a single data set. However the practical data analysis problem of how to select a specific number of features that achieves at most some desired error is left unspecified and in fact subject to the same error estimation difficulty that applies to wrapping. Moreover, in recent algorithms such as RFE, the problem is acknowledged implicitly in the applied examples provided by the authors of the method, since feature sets are reduced by for example 50% in each iteration of the

algorithm creating a new subset of features examined by cross-validation by the algorithm (Guyon et al., 2002). This is done to reduce overfitting of selected feature set to the data because of the large variability of error estimates. As evidenced by the evaluations presented in Aliferis et al. (2010), it is possible to improve on traditional wrapping, ranking and RFE selection by applying statistical tests of difference of error estimates, or by increasing/decreasing the granularity of feature selection (i.e., proportion of features removed at each iteration). Still the produced feature sets are not optimal in parsimony. The numbers of strongly relevant, weakly relevant and irrelevant features is not critical to the existence of the problem, neither is the type of wrapper (forward, backward, forward-backward, GA, etc.) as long as some basic requirements are met: error estimation is not perfect but subject to sampling variability due to small sample, and enough features exist in data for enough error estimate comparisons to be spurious.

Contrary to the above, GLL filtering relies little on error estimation[3] and uses robust mechanisms to control false negatives and false positives separately for strongly relevant, weakly relevant and irrelevant features respectively. In Table 10 we give a concrete demonstration of how semi-interleaved HITON-PC (without symmetry correction for simplicity) is less prone to errors in the same example. The critical observation is for an irrelevant feature to enter $TPC(T)$ and stay in it, it has to survive multiple (i.e., $2^{|TPC(T)|}$) tests of conditional independence and each such test has probability $1-\alpha$ to leave the irrelevant feature in $TPC(T)$. The total probability of failing to reject the irrelevant variable thus grows up to exponentially small to the number of tests performed and is independent of the sample size. In our simplified example with just one strongly irrelevant feature inside $TPC(T)$, each irrelevant feature has probability of entering and staying in $TPC(T)$ of at most $\alpha^2 = 0.0025$. This is true regardless of whether sample size is 10,000 samples or just 10 samples.

5. Algorithmic Extensions to GLL

In the present section we introduce algorithmic extensions to the Generalized Local Learning algorithms: parallel and distributed local learning and FDR pre-filtering.

5.1 Parallel and Distributed Local Learning

Following ideas for parallelizing the IAMB algorithm for $MB(T)$ estimation (Aliferis et al., 2002), we introduce a coarse-grain parallelization of GLL-PC that addresses two problems: (a) the data does not fit into fast memory (RAM), and (b) even if the data fits, we wish to speedup execution time by parallel processing. We allow for the possibility that the user may have access to just one node or, alternatively, may have access to several nodes arranged in a parallel cluster. The algorithm presented can return $PC(T)$ and can run with any instantiation of GLL-PC. The algorithm is designed to be correct provided that no symmetry correction is required (i.e., in distributions where $EPC(T) \equiv PC(T)$). Correct parallel/distributed versions in distributions where symmetry correction is needed can also be obtained as can algorithms that parallelize $MB(T)$ induction. In the present paper we only discuss parallel GLL-PC without symmetry correction because of its concep-

3. Notice that some reliance on error estimation exists in domains where a suitable *max-k* and α are not known and need be optimized by cross-validation. The corresponding number of parameterizations is very small however (typically at the order of 10 combined parameter configurations) and thus error estimation is less likely to lead the algorithm astray. The same is true for the optional wrapping step in GLL-MB which selects features from a highly reduced set compared to the original feature set (notice that this wrapping step is seldom needed in practice and is reserved for higher sample settings).

Chunked Parallel GLL-PC Algorithm (not symmetry corrected)

Input: Dataset D, target variable T, desired number of data chunks ch.

1. Split the data D into ch arrays C_i of equal size, such that each array contains a non-overlapping subset of the variables plus T.
2. For all i, compute $ChunkPC_i(T) \leftarrow$ GLL-PC-nonsym(T, C_i)
3. $\boldsymbol{L} \leftarrow$ GLL-PC-nonsym(T, $\cup_i ChunkPC_i(T)$)
4. Return $\boldsymbol{L}$ and exit

Figure 4: Chunked Parallel GLL-PC algorithm (not symmetry corrected).

tual and implementation simplicity and speed, because it can be used for both causal discovery and prediction, and because as demonstrated empirically (Aliferis et al., 2010), many real distributions behave consistently with being "symmetrical" (i.e., $EPC(T) \equiv PC(T)$).

Chunked Parallel GLL-PC algorithm (not symmetry corrected): This algorithm assumes that one has access to several nodes and that the data can fit to the available memory once distributed, while it may or may not fit to a single node. Initially the algorithm divides the input data D into ch chunks C_i such that every C_i includes all cases, but only a subset $\boldsymbol{V}_i$ of the variable set $\boldsymbol{V}$ plus T. For simplicity we assume that each chunk has an equal number of features (that can be determined, for example, by the maximum size that can be processed in fast memory or the number of available computer nodes in a parallel implementation). Variations where unequal variable allocations are employed can be easily obtained in similar fashion. Then GLL-PC-nonsym is run on each chunk (as indicated by the extra input argument C_i) returning $ChunkPC_i(T)$ (i.e., parents and children of T in chunk C_i). Next, GLL-PC-nonsym is run on one node with the union $\cup_i ChunkPC_i(T)$, it obtains a local neighborhood $\boldsymbol{L}$, and terminates by outputting $\boldsymbol{L}$. Figure 4 gives the parallel GLL-PC high-level pseudo-code. Step #2 is the parallel step.

We note that a potential problem with chunked GLL-PC is that the tentative neighborhood in some chunk(s) may grow very large (up to the size of the chunk in the worst case) while the true neighborhood across all variables may be very small. This creates the possibility of overflow both in the sense of data not fitting in a single node and in the sense of not having enough sample size to perform reliable statistical inferences.

Theorem 1 *Chunked parallel GLL-PC without symmetry correction is sound given the sufficient conditions for soundness of GLL-PC and the requirement that in the generating distribution P, $PC(T)$ is the same as the Extended $PC(T)$ (see definition of $EPC(T)$ in Aliferis et al. 2010).*

Proof In each chunk, GLL-PC-nonsym will identify all true members of $PC(T)$ that are in the chunk (because these can never be rendered independent of T, according to Theorem 1 in Aliferis et al. 2010) and some false positives which cannot be eliminated without conditioning on $PC(T)$ members that belong to another chunk. Thus in step #3, GLL-PC-nonsym is executed on a superset of $PC(T)$. By definition, all non-members of $PC(T)$ can be rendered independent of T conditioned on some subset of $PC(T)$ as long as $PC(T) \equiv EPC(T)$. Since $PC(T) \equiv EPC(T)$, the identified $PC(T)$ will be correct. ■

The complexity of Chunked Parallel GLL-PC without symmetry correction is in the worst case exponentially slower than running GLL-PC on all data. This is because the complexity of GLL-PC

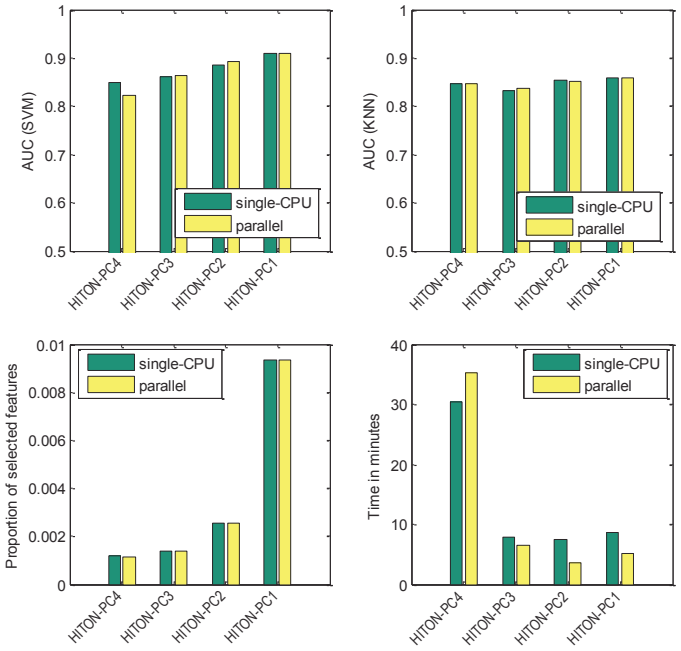


Figure 5: Results of application of single-CPU and parallel versions of semi-interleaved HITON-PC on the four largest real data sets (*Ohsumed, ACPJ_Etiology, Thrombin*, and *Nova*). Average results over 4 data sets are shown. The following versions of HITON-PC are used: HITON-PC4 (*max-k*=4, α=0.05), HITON-PC3 (*max-k*=3, α=0.05), HITON-PC2 (*max-k*=2, α=0.05), HITON-PC1 (*max-k*=1, α=0.05).

is worst-case exponential to the size of $TPC(T)$ and while $TPC(T)$ in all data can be very small, in some chunks $TPC(T)$ can be as large as the chunk itself. When however local neighborhoods in each chunk are smaller than the global $TPC(T)$ and since GLL-PC is worst-case exponential, the algorithm can also be exponentially faster than running GLL-PC on all data. This is in sharp contrast with parallel IAMB where both the speedup is linear to the number of chunks in the best case (upper bound on the speed-up factor is *ch*) and worst-case running time is a small constant multiple of running the algorithm on all data (Aliferis et al., 2002).

Chunked Distributed GLL: When we run the algorithm with data already distributed, the data splitting and transfer step #1 (as well as associated transfer cost) is omitted. Typically we will need to link the distributed data using a suitable common key. For example consider a large organization wishing to analyze data in order to find determinants of production costs overall many and geographically dispersed branches, each with its own local data set and different recorded features. An appropriate key might be time label of observations. Another example is hospital patient data distributed among numerous local databases in different units and labs of the hospital, where patient id is a suitable key.

Chunked GLL with single CPU: This variant assumes access to one CPU only and addresses the problem of data not fitting in the fast memory. By processing parts of the data sequentially and obtaining a small superset of $PC(T)$ each time, a much larger data set than what fits in fast memory can be analyzed.

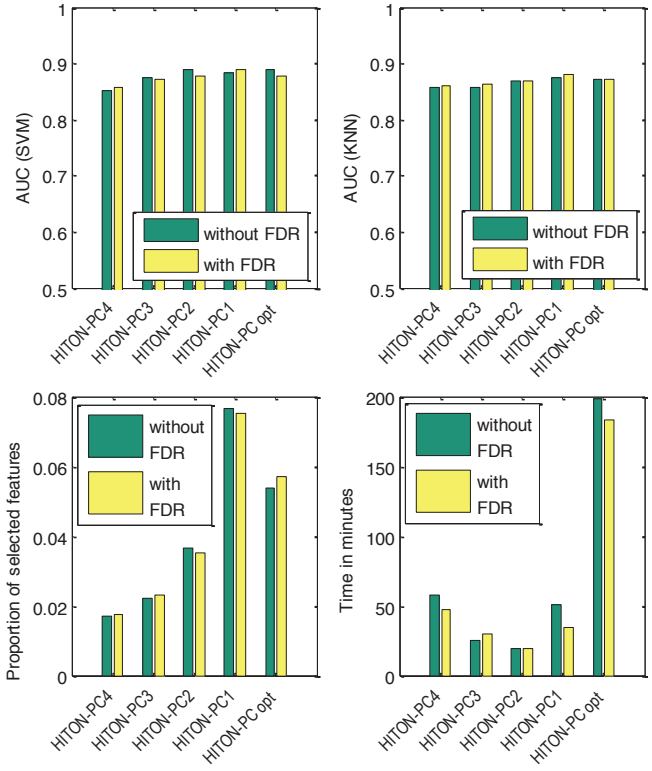


Figure 6: Results of application of semi-interleaved HITON-PC with and without FDR correction on 13 real data sets. Average results over the data sets are shown. The following versions of HITON-PC are used: HITON-PC4 (*max-k*=4, α=0.05), HITON-PC3 (*max-k*=3, α=0.05), HITON-PC2 (*max-k*=2, α=0.05), HITON-PC1 (*max-k*=1, α=0.05), HITON-PC opt (*max-k* and α are optimized over values $\{1,2,3,4\}$ and $\{0.05,0.01\}$, respectively, by cross-validation to maximize SVM classification performance).

We now apply a parallel version of semi-interleaved HITON-PC on the four largest real data sets (*Ohsumed*, *ACPJ_Etiology*, *Thrombin*, and *Nova*) of the empirical evaluation in Aliferis et al. (2010). We use 10 CPU's on the ACCRE cluster described in Aliferis et al. (2010). As can be seen in Figure 5 the parallel version achieves the same parsimony and classification performance as the single-CPU application with speedup for three out of four versions of HITON-PC (see Figure 5). P-values from the permutation test of the null hypothesis that single-CPU and parallel GLL-PC algorithms achieve the same performance are 0.7468 (for SVM classification), 0.4950 (for KNN classification), 0.2408 (for proportion of selected features), and 0.6374 (for running time in minutes). We note that running times for HITON-PC algorithm in this subsection are less than in the remainder of the paper because these experiments were executed on the most recent version of the ACCRE cluster.

5.2 FDR pre-Filtering

As explained in Section 3, in simulated and resimulated data sets with weak-signal/small sample and in all-irrelevant features situations, removing features using false discovery rate control can

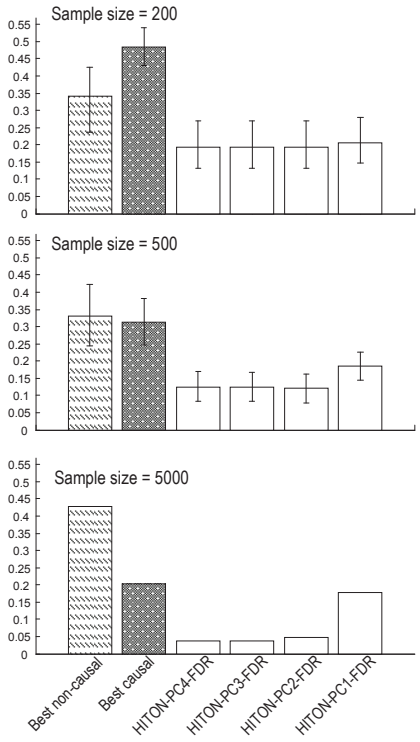


Figure 7: Graph distances averaged over all 9 simulated and resimulated data sets, all selected targets in each data set, and multiple samples of a given size. The following versions of semi-interleaved HITON-PC with FDR correction are used: HITON-PC4-FDR (*max-k*=4, α=0.05), HITON-PC3-FDR (*max-k*=3, α=0.05), HITON-PC2-FDR (*max-k*=2, α=0.05), and HITON-PC1-FDR (*max-k*=1, α=0.05). "Best causal" is the best causal feature selection algorithm among techniques that do not incorporate FDR. "Best non-causal" is the best non-causal feature selection algorithm. See Aliferis et al. (2010) for a detailed list of algorithms.

improve the number of false positives in HITON-PC and MMPC. We applied HITON-PC with FDR pre-filtering in all real data sets of Aliferis et al. (2010). As can be seen in Figure 6, this enhancement does not entail improvements in parsimony, classification performance or running time in the data sets tested. P-values from the permutation test of the null hypothesis that GLL-PC algorithms with and without FDR correction achieve the same performance are 0.5254 (for SVM classification), 0.3698 (for KNN classification), 0.9426 (for proportion of selected features), and 0.3776 (for running time in minutes). Since however the algorithm exhibits small sensitivity to false positives due to multiple comparisons when many irrelevant features are expected and few relevant features are present, we recommend pre-filtering with FDR. Alternatively, if one gets a few variables combined with error estimates consistent with uninformative classifier, then re-running standard GLL with FDR pre-processing can be tried.

When evaluating local causal discovery performance in the simulated data of Aliferis et al. (2010), semi-interleaved HITON-PC with FDR pre-processing achieves dramatically better performance than other algorithms including other HITON and MMPC variants with respect to graph

LGL: Local-to-Global Learning

1. Find *PC(X)* for every variable *X* in the data using an admissible instantiation of GLL-PC and prioritizing which variables to induce *PC(X)* for, according to a prioritization strategy.
2. Piece together the undirected skeleton from the local GLL-PC results.
3. Use any desired arc orientation scheme to orient edges.

Figure 8: Local-to-Global Learning (LGL) algorithmic schema.

MMHC Global Learning Algorithm

1. Find *PC(X)* for every variable *X* in data using MMPC (without symmetry correction) and lexicographic prioritization.
2. Piece together the undirected skeleton using an "OR rule" (an edge exists between *A* and *B* iff *A* is in *PC*(*B*) or *B* is in *PC*(*A*)).
3. Use greedy steepest-ascent TABU search and BDeu score to orient edges.

Figure 9: MMHC global learning algorithm as an instance of LGL.

HHC Global Learning Algorithm

1. Find *PC*(*X*) for every variable *X* in data using semi-interleaved HITON-PC (without symmetry correction) and lexicographic prioritization.
2. Piece together the undirected skeleton using an "OR rule" (an edge exists between *A* and *B* iff *A* is in *PC*(*B*) or *B* is in *PC*(*A*)).
3. Use greedy steepest-ascent TABU search and BDeu score to orient edges.

Figure 10: HHC global learning algorithm as an instance of LGL.

distance score, which indicates average causal proximity to the target of the returned variables. Specifically, in large sample (N=5,000) HITON-PC with FDR correction achieves up to 5-fold reduction in the graph distance score relative to the best non-FDR filtered causal algorithm and up to 9-fold reduction compared to the best non-causal algorithm. In small sample (N=200) the reduction in both cases is 2-fold. P-values from the permutation test of the null hypothesis that the best non-causal algorithm performs the same as the average HITON-PC with FDR correction are <0.0001 for sample sizes 200, 500, and 5,000. P-values for comparison with the best causal algorithm are <0.0001, 0.0030, and <0.0001 for sample sizes 200, 500, and 5000, respectively. See Figure 7. This improvement incurs only a very small decrease in sensitivity as evidenced by small concurrent increases in false negatives.

6. Spanning Local to Global Learning

In the present section we investigate the use of local learning methods (such as GLL) for global learning in a divide-and-conquer fashion. We remind that a major motivation for pursuing local causal learning methods is scaling up causal discovery and causal feature selection as explained in Aliferis et al. (2010). Although similar concepts can be used for region learning, we will not address this type of discovery problem here. The main points of the present section are that (a) the local-to-global framework can be instantiated in several ways with excellent empirical results; (b) an important previously unnoticed factor is the variable order in which to execute local learning, and (c) trying to use non-causal feature selection in order to facilitate global learning (instead of causal local learning) is not as a promising strategy as previously thought.

6.1 General Concepts

A precursor to the main idea behind the local-to-global learning approach can be found in SCA (Friedman et al., 1999), where a heuristic approximation of the local causes of every variable constraints the space of search of the standard greedy search-and-score Bayesian algorithm for global learning increasing thus computational efficiency. Given powerful methods for finding local neighborhoods, provided by the GLL framework, one can circumvent the need for uniform connectivity (as well as user knowledge of that connectivity) and avoid the application of inefficient heuristics employed in SCA thus improving on quality and speed of execution. Figure 8 provides the general algorithmic schema term LGL (for local-to-global learning). Steps #1-3 can be instantiated in numerous ways. If an admissible GLL-PC (as defined in Section 4 of Aliferis et al. 2010) is used in step #1, and step #2 is consistent with the results of GLL-PC for all variables, and a sound orientation scheme in step #3, then the total algorithm is trivially sound under the assumptions of correctness of GLL-PC. These are the admissibility requirements for the LGL template. It follows that:

Proposition 1 *Under the following sufficient conditions we obtain correctly oriented causal graph with any* admissible *instantiation of LGL:*

a. There is a causal Bayesian network faithful to the data distribution P;

b. The determination of variable independence from the sample data D is correct;

c. Causal sufficiency in $\boldsymbol{V}$.

The recently-introduced algorithm MMHC is an instance of the LGL framework (Tsamardinos et al., 2006). Figure 9 shows how MMHC instantiates LGL. MMHC is not sound with respect to orientation because greedy steepest-ascent search is not a sound search strategy for search-and-score global learning. Despite being theoretically not sound the algorithm works very well in practice and in an extensive empirical evaluation it was shown to outperform in speed and quality several state-of-the-art algorithms (Greedy Search, GES, OR, PC, TPDA, and SCA) (Tsamardinos et al., 2006).

6.2 A New Instantiation of LGL: HHC

To demonstrate the generality and robustness of the LGL framework we provide here as an instantiation of LGL, a new global learning algorithm termed HHC (see Figure 10), and compare it empirically to the state-of-the-art MMHC algorithm. We also show that the two algorithms are not identical in edge quality or computational efficiency, with the new algorithm being at least as good on average as MMHC.

Table 11 presents results for missing/extra edges in undirected skeleton, number of statistical tests for construction of skeleton, structural Hamming distance (SHD), Bayesian score, and execution time on 9 of the largest data sets used for the evaluation of MMHC. Since the data sets were simulated from known networks, the algorithm output can be compared to the true structure. As can be seen, in all 9 data sets, HHC performs equally well with MMHC in terms of SHD and Bayesian score. In 8 out of 9 data sets it performs from 10% to 50% fewer tests, and in one data set (*Link*) it performs $>$10 times the tests performed by MMHC resulting in running 35% slower in terms of execution time. Because MMHC was found to be superior to a number of other algorithms for the

HHC

	Dataset								
	Child10	*Insurance10*	*Alarm10*	*Hailfinder10*	*Pigs*	*Munin*	*Lung_Cancer*	*Gene*	*Link*
Extra edges in learned skeleton	95	143	176	1265	276	36	621	601	1456
Missing edges in learned skeleton	25	149	165	359	0	257	91	6	439
Structural Hamming distance for DAG	101	297	344	728	4	273	187	72	1150
Bayesian score for DAG	-188.61	-229.02	-178.56	-738.77	-496.11	-33.14	-559.43	-651.36	-337.74
Number of statistical tests for skeleton construction	28,879	52,757	82,543	217,490	134,244	733	859,348	401,779	7,931,044
Time for building skeleton (in minutes)	0.74	1.59	2.47	8.05	3.98	0.23	24.40	12.32	537.72
Total time for running algorithm (in minutes)	1.21	3.32	6.80	24.84	14.33	0.47	181.97	60.14	563.46

MMHC

	Dataset								
	Child10	*Insurance10*	*Alarm10*	*Hailfinder10*	*Pigs*	*Munin*	*Lung_Cancer*	*Gene*	*Link*
Extra edges in learned skeleton	71	128	184	1220	281	38	567	557	1541
Missing edges in learned skeleton	25	148	164	352	0	258	88	4	396
Structural Hamming distance for DAG	100	296	346	725	4	275	191	69	1145
Bayesian score for DAG	-188.95	-229.03	-179.09	-738.80	-496.11	-33.12	-559.01	-651.12	-337.62
Number of statistical tests for skeleton construction	32,980	67,943	90,117	243,571	177,278	1,023	1,360,493	451,364	644,055
Time for building skeleton (in minutes)	0.81	1.99	2.49	12.81	5.45	0.38	55.16	12.23	382.93
Total time for running algorithm (in minutes)	1.42	3.79	5.21	29.54	13.11	0.46	451.70	51.84	415.69

Table 11: Comparison of HHC and MMHC global learning algorithms. Both algorithms were executed on a random sample of size 1000, using default parameters of MMHC as implemented in *Causal Explorer* (i.e., G^2 test for conditional independence, $\alpha = 0.05$, *max-k* = 10, Dirichlet weight = 10, BDeu priors).

data sets tested, HHC's better performance over MMHC in 8 out of 9 data sets (in terms of number of statistical tests for skeleton construction) and similar performance in 9 out of 9 data sets (in terms

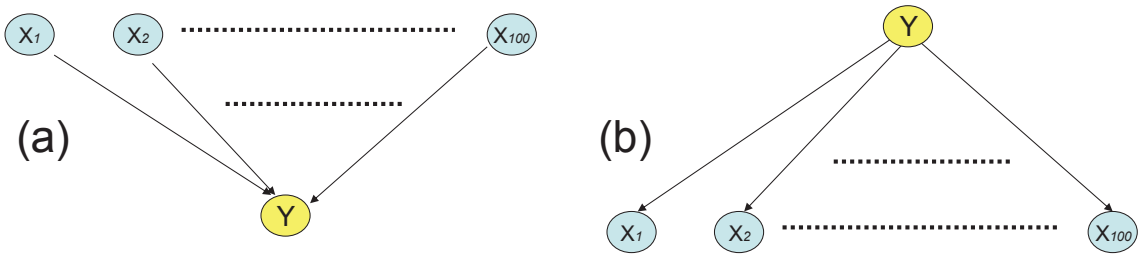


Figure 11: Two examples where the variable ordering for local learning can make execution of the LGL algorithm from quadratic to exponential-time.

of quality metrics) translates also to excellent performance of HHC relative to Greedy Search, GES, OR, PC, TPDA, and SCA (Tsamardinos et al., 2006).

6.3 Importance of Variable Prioritization for Quality and Efficiency

An important parameter of local-to-global learning previously unnoticed in algorithms such as SCA and MMHC is the ordering of variables when executing the local causal discovery variable-by-variable (i.e., not in parallel). We will assume that results are shared among local learning runs of GLL-PC, that is when we start learning $PC(X)$ by GLL-PC rather than starting with an empty $TPC(X)$ set, we start with all variables Y: $X \in PC(Y)$. This constitutes a sound instantiation of the GLL-PC algorithm template as explained in Aliferis et al. (2010). Figure 11 gives two extreme examples where the right order can "make-or-break" an LGL algorithm.

In Figure 11(a) it is straightforward (and left to the reader to verify) that an order of local learning $< X_1, X_2, \ldots, X_{100}, Y >$ without symmetry correction (the latter being a reasonable choice as we have seen) requires a quadratic number of conditional independence tests (CITs) for the unoriented graph to be correctly learned. However, the order of local learning $< Y, X_1, X_2, \ldots, X_{100} >$ requires up to an exponential number of CITs as *max-k* and sample are allowed to grow without bounds. Even with modest *max-k* values, the number of CITs is higher-order polynomial and thus intractable. Even when Y is not in the beginning but as long as a non-trivial number of X's are after it in the ordering, the algorithm will be intractable or at least very slow. The latter setting occurs in the majority of runs of the algorithm with random orderings.

In Table 12 we provide data from a simulation experiment showing the above in concrete terms and exploring the effects of limited sample and connectivity at the same time. As can be seen, under fixed sample, running HHC with order from larger to smaller connectivity, as long as the sample is enough for the number of parents to be learned (i.e., number of parents is ≤ 20), increases run time by more than 100-fold. However because sample is fixed, as the number of parents grows the number of conditional independence tests equalizes between the two strategies because CITs that have too large conditioning sets for the fixed sample size are not executed. Although the number of CITs is self-limiting under these conditions, quality (in terms of number of missing edges, that is, number of undiscovered parents of T) drops very fast as the number of parents increases. The random ordering strategy trades off quality for execution time with the wrong (larger-to-smaller connectivity) ordering, however in all instances the right ordering offers better quality and 2 to 100-fold faster execution that random ordering.

Number of parents of Y	order from low-to-high connectivity			random order (average results over 10 orders)			order from high-to-low connectivity		
	extra edges	missing edges	CITs	extra edges	missing edges	CITs	extra edges	missing edges	CITs
10	2	0	63	2	0	2,461	2	0	4,325
20	4	0	233	4.7	5.2	26,203	5	7	29,774
30	12	0	526	12	12.4	41,499	11	21	9,020
40	13	0	904	16.4	20.1	51,269	19	33	5,626
50	22	7	1,428	28.8	30	16,828	34	43	4,149
60	29	7	2,001	32.9	35.7	36,950	38	54	3,862
70	41	19	2,773	45.7	37.9	24,456	55	63	4,464
80	58	28	3,652	65.4	55.1	12,630	70	74	5,023
90	66	35	4,634	72.3	57.6	16,718	87	85	5,592
100	77	44	5,594	88.7	80	16,266	96	94	7,229

Table 12: Results of simulation experiment with HHC algorithm. The graphical structure is depicted on Figure 11(a). HHC was run on a random sample of size 1,000 with G^2 test for conditional independence, α=0.05, *max-k* = 5, Dirichlet weight = 10, BDeu priors.

A more dramatic difference exists for the structure in Figure 11(b) where Y is a parent of all X's. Here the number of tests required to find the parent (Y) of each X_i is quadratic to the number of variables with the right ordering (low-to-high connectivity) whereas an exponential number is needed with the wrong ordering (large-to-small connectivity). Because the sample requirements are constant to the number of children of Y, quality is affected very little and there is no self-restricting effect of the number of CITs, opposite to what holds for causal structure in Figure 11(a). Hence the number of CITs grows exponentially larger for the large-to-small connectivity ordering versus the opposite ordering and a similar trend is also present for the average random ordering in full concordance with our theoretical expectations. See Table 13 for results of related simulation experiments.

These results show that in some cases, *it is possible to transform an intractable local learning problem into a tractable one by employing a global learning strategy (i.e., by exploiting asymmetries in connectivity)*. Thus the variable order in local-to-global learning may have promise for substantial speedup and improved quality in real-life data sets (assuming the order of connectivity is known or can be estimated). However the optimal order is a priori unknown for some domain. Can we use local variable connectivity as a proxy to optimal order in real data? The next experiment assumes the existence of an oracle that gives the true local connectivity for each variable. The experiment examines empirically the effect of three orders (low-to-high connectivity, lexicographical (random) order, and high-to-low connectivity order) on the quality of learning and number of CITs in the MMHC evaluation data sets. It also compares the sensitivity of HHC to order.

As can be seen in Figure 12, the order does have an effect on computational efficiency however not nearly as dramatic in the majority of these more realistic data sets compared to the simpler structures of Figure 11. An exception is the *Link* data set in which low-to-high connectivity allows HHC to run 17 times faster than lexicographical (random) order and 27 times faster than high-to-low connectivity order. For the majority of cases, running these algorithms with lexicographical (i.e.,

Number of children of Y	order from low-to-high connectivity			random order (average results over 10 orders)			order from high-to-low connectivity		
	extra edges	missing edges	CITs	extra edges	missing edges	CITs	extra edges	missing edges	CITs
10	1	0	106	1	0	2,342	1	0	4,366
20	11	0	489	9.7	0	141,148	9	0	377,448
30	18	0	1,173	16.8	0	2,321,030	17	0	5,020,400
40	24	0	1,968	-	-	-	-	-	-
50	33	0	3,190	-	-	-	-	-	-
60	48	0	5,031	-	-	-	-	-	-
70	53	0	6,899	-	-	-	-	-	-
80	71	0	8,939	-	-	-	-	-	-
90	76	0	11,448	-	-	-	-	-	-
100	95	0	14,677	-	-	-	-	-	-

Table 13: Results of simulation experiment with HHC algorithm. The graphical structure is depicted on Figure 11(b). HHC was run on a random sample of size 1,000 with G^2 test for conditional independence, α=0.05, *max-k*=5, Dirichlet weight = 10, BDeu priors. Empty cells correspond to experiments when the algorithm did not terminate within 10,000,000 CITs.

random) order is very robust and does not affect quality adversely but affects run time and number of CITs to a small degree (details in Table S21 in the online supplement).

Thus, while connectivity affects which variable order is optimal in LGL algorithms, ranking by local connectivity does not exactly correspond to the optimal order. Figure S3 in the online supplement shows the number of CITs plotted against true local connectivity in each one of the 9 data sets used in this section. Related to the above, Figure S4 in the supplement also shows the distribution of true local connectivity in each data set. Consistent trends indicating the shape of the distributions by which the degree of local connectivity may determine an advantage of orderings low-to-high to high-to-low connectivity are not apparent in these data sets.

We hypothesize that more robust criteria for the effect of variable ordering in LGL algorithms can be devised. For example, the number or total cost of CITs required to locally learn the neighborhood of each variable. Such criteria are also more likely to be available or to be approximated well during practical execution of an algorithm than true connectivity. A variant of HHC, algorithm HHC-OO (standing for HHC with optimal order) (Aliferis and Statnikov, 2008) orders variables dynamically according to heuristic approximations to the total number of CITs for each variable. We also conjecture that the strategy for piecing together the local learning results strongly interacts with the local variable ordering to determine the tradeoff between the quality and efficiency of LGL algorithms. Evaluation of these hypotheses is outside the scope of the present paper.

6.4 Using non-Causal Feature Selection for Global Learning

In recent years several researchers have proposed that because modern feature selection methods can deal with large dimensionality/small sample data sets, they could also be used to speed up or approximate large scale causal discovery (e.g., Kohane et al. 2003 use univariate feature selection to build so-called "relevance networks"), or hybrid methods can be employed that use feature selection

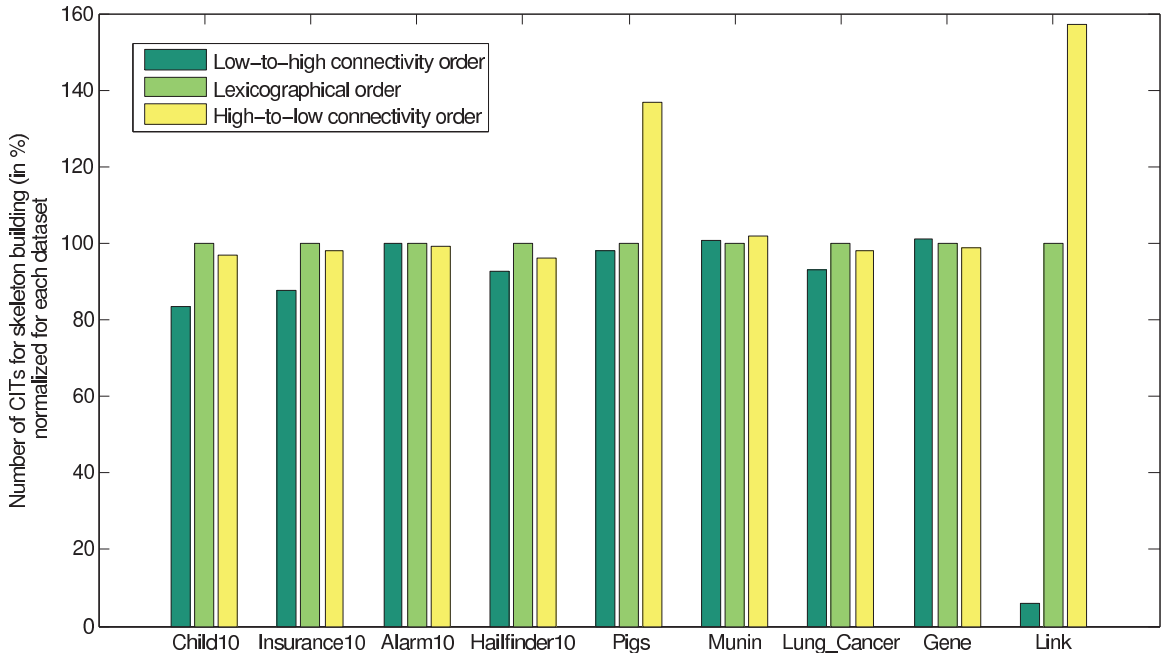


Figure 12: Number of CITs required for skeleton construction during execution of HHC expressed as % points and normalized within each data set to lexicographical order. Data for three orderings of variables is shown on the figure: low-to-high connectivity, lexicographical, and high-to-low connectivity orders. HHC was executed with same parameters as in Table 11. More detailed results are provided in Table 11 and Table S21 in the online supplement.

as a pre-processing to build a skeleton and then an orientation algorithm like Greedy Search in the spirit of MMHC and LGL (Schmidt et al., 2007). The results of Aliferis et al. (2010) contradict this postulate because they show that non-causal feature selection does not give locally correct results. However it is still conceivable that orientation-and-repair post-processing algorithms (e.g., with Bayesian search-and-score) can still provide a high quality final causal graph. We test this hypothesis by examining several such hybrid methods using respectively RFE, LARS-EN and UAF post-processed by Greedy TABU Bayesian search-and-score. We use simulated data sets from 5 out of 9 Bayesian networks employed earlier in the present section. This is because the other 4 networks cannot be used for reliable training and testing of the underlying classifier since they have several variables with very unbalanced distributions. As shown in Table 14, the hypothesis is not corroborated by the experimental results. In particular, Greedy Search with feature selection-based skeleton, exhibits substantial drops in quality of the returned networks (measured by structural hamming distance Tsamardinos et al., 2006) and typically more than one order of magnitude longer running times compared to HHC with lexicographical (random) variable ordering. On the basis of these findings, which are consistent with the results in Aliferis et al. (2010), we do not find encouraging evidence that non-causal feature selection can be used as an adjunct to global causal discovery. Strong evidence exists however in favor of using principled local causal methods instead, within the frameworks of LGL.

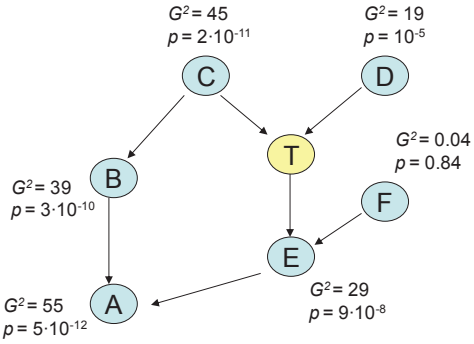
	Child10				*Pigs*				*Hailfinder10*			
	RFE	***LARS***	***UAF***	***HHC***	***RFE***	***LARS***	***UAF***	***HHC***	***RFE***	***LARS***	***UAF***	***HHC***
Extra edges in learned skeleton	2078	7558	3014	95	2262	29570	5593	276	6424	40948	7904	1265
Missing edges in learned skeleton	26	8	20	25	2	0	0	0	461	211	325	359
Structural Hamming distance for DAG	121	117	135	101	76	102	7	4	796	756	733	728
Bayesian score for DAG	-190.0	-189.1	-189.8	-188.61	-497.2	-496.8	-496.4	-496.11	-740.5	-736.4	-737.4	-738.77
Time for building skeleton (in minutes)	41.63	43.57	44.97	0.74	348.44	184.47	355.59	3.98	572.13	365.45	581.34	8.05
Total time for running algorithm (in minutes)	43.23	48.52	47.05	1.21	361.15	265.07	373.54	14.33	603.62	503.63	612.63	24.84

	Gene				*Lung Cancer*			
	RFE	***LARS***	***UAF***	***HHC***	***RFE***	***LARS***	***UAF***	***HHC***
Extra edges in learned skeleton	4039	55384	9834	621	7469	38753	12486	601
Missing edges in learned skeleton	47	8	28	91	120	24	78	6
Structural Hamming distance for DAG	125	156	115	187	220	139	175	72
Bayesian score for DAG	-658.3	-653.1	-655.1	-559.43	-562.4	-555.6	-560.1	-651.36
Time for building skeleton (in minutes)	737.99	513.12	783.97	24.40	493.84	377.85	563.46	12.32
Total time for running algorithm (in minutes)	784.54	912.33	890.63	181.97	708.77	1096.19	855.18	60.14

Table 14: Results for hybrid methods using RFE, LARS-EN and UAF.

7. Using Causal Graphs and Markov Blanket Theory as a Conceptual Analysis Framework for Feature Selection Methods

In the present section we show that by adopting a causal structural perspective founded on the theoretical results outlined in Aliferis et al. (2010), several strengths and weaknesses and general performance characteristics of non-causal feature selection algorithms become apparent and our empirical findings in Aliferis et al. (2010) can be better understood. We review several established and state-of-the-art methods both from a feature selection perspective (e.g., does the algorithm exhibit false positives and false negatives relative to minimal feature set that yields optimal predictivity?) and from a causal discovery perspective (is the output of the algorithm causally sound?). With respect to the latter for reasons elucidated in Aliferis et al. (2010), we focus on localization of causal inferences (i.e., whether the feature selection output is locally causally correct), and when this is not obtained, we examine whether some other useful causal inference can be made.



Select first *k* features:

Selected		FN	FP	Predictivity
A	(k=1)	C,D,E,F	A	Suboptimal
A,C	(k=2)	D,E,F	A	Suboptimal
A,B,C	(k=3)	D,E,F	A,B	Suboptimal
A,B,C,E	(k=4)	D,F	A,B	Suboptimal
A,B,C,D,E	(k=5)	F	A,B	Suboptimal

Figure 13: Limitations of univariate feature selection explained using a causal graph perspective. Strength of univariate association with the target variable T is measured in a fixed sample of size 10,000 by the negative p-value of a G^2-test and depicted next to each variable.

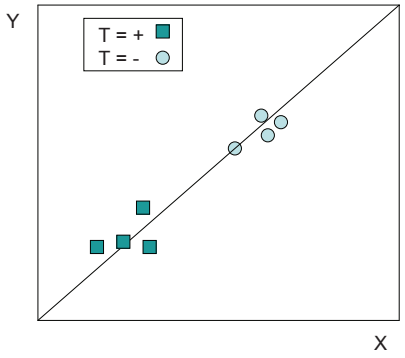


Figure 14: Example showing that Principal Component Analysis yields redundant features.

7.1 Univariate Association Filtering

Figure 13 shows the causal structure of a data-generating process. The causal structure is parameterized as shown in Appendix Figure 19. This structure and parameterization entails that $association(B,T) < association(C,T)$. Because of *synthesis of information along two paths* however, $association(A,T) > association(C,T)$ and $association(A,T) > association(E,T)$. The example illustrates that from the feature selection perspective the optimal predictor set (i.e., the Markov blanket) for predicting or classifying the target T is $\{C,D,E,F\}$. However, because univariate associations of non-$MB(T)$ members can be higher than those of members, false positives are incurred when selecting features using univariate association-based filters. Furthermore, spouses without connecting path to the target will have zero univariate association and thus will not be selected at all by univariate filtering. The embedded table shows the false positives and false negatives (relative to the gold standard set $MB(T)$) at each possible threshold for variable inclusion. In all cases predictivity is suboptimal.

From the causal discovery perspective, the example makes evident that non-causally relevant features such as A and B can be selected with higher ranking than causally relevant ones such as D and E. Association synthesis thus forbids an interpretation of the higher-ranked causal variables as more direct causes (or effects) than lower-ranked features even when all of them are causal. Worse yet, even without synthesis, an arbitrarily large number of non-causal features can be selected before

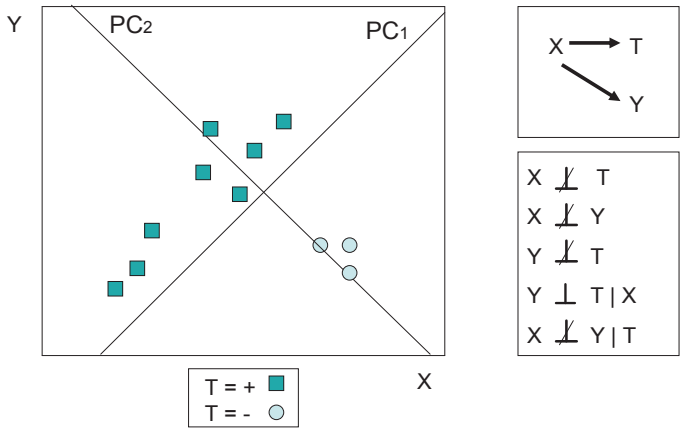


Figure 15: Example showing that Principal Component Analysis yields locally causally inconsistent results.

truly causal ones are selected. To see why this is the case consider that between C and B there may be arbitrarily many variables arranged in a chain so that their association with T is larger than that of both true cause D and true effect E.

7.2 Principal Component Analysis

As can be seen in Figure 14, the principal component defined by the diagonal ($Y - X = 0$) perfectly separates the two target classes and will be chosen by a PCA procedure since it explains maximum proportion of variance in the data. While projecting the original data on this single dimension reduces dimensionality of the classification problem, from the perspective of finding the original features that are important and non-redundant the method leads to false positives (since the coefficients of both Y and X are equal in the depicted Principal Component, indicating that both features are deemed equally necessary).

The example in Figure 15 shows that PCA is not sound for causal discovery. As shown in the figure, X is a direct cause of T and Y is not causal for T but confounded by X. Application of causal learning via the usual assumptions and procedures reveals that X is a direct cause or effect of T and that Y is not directly causally linked with T (the requisite conditional independence tests are depicted). However, an optimal procedure for Principal Component classification will select the second principal component PC_2 which achieves perfect classification. However both X and Y have equal coefficients in each principal component. Hence PCA may select both redundant features and non-causal features.

7.3 Feature Selection Using SVM Weights

A fundamental weakness of the maximum-gap inductive bias, as employed in SVMs, is its local causal inconsistency. Consider a scenario (Figure 16) similar to the previous sub-section where we wish to discover the direct causes of a response variable T, from observations about variables X, Y, T. Assume for simplicity that T is a terminal variable and thus X and Y precede it in time. For example, T can be a clinical phenotype and X, Y can be gene expression values. The causal process that generates the data is seen in the upper right corner of Figure 16. As can be seen in the

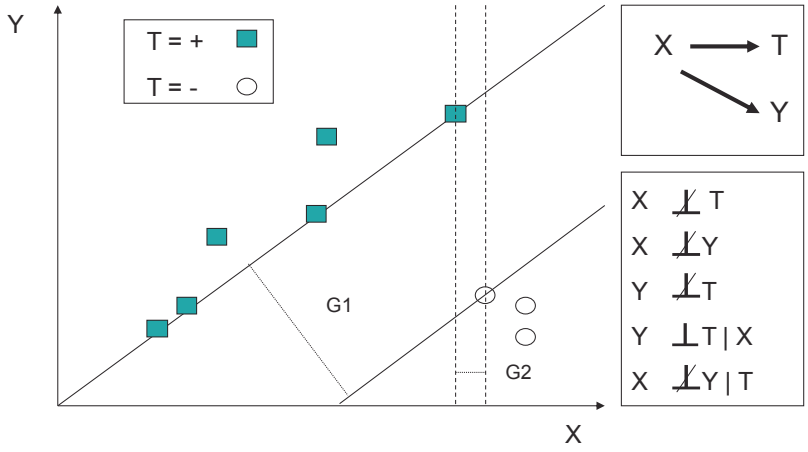


Figure 16: Example showing that SVM weight-based feature selection yields locally causally inconsistent results and redundant features.

left part of the figure, the SVM classifier can perfectly predict T using X and Y as predictors. In doing so it prefers the classifier with gap G1 to the classifier with smaller gap G2. The preferred classifier assigns non-zero (and in fact equal) weights to both X, Y thereby admitting Y in the local causal neighborhood if selected variables are interpreted causally. However, X *renders* Y *independent from* T *and not vice versa*. More generally, in distributions where the Causal Markov Condition holds, SVMs will occasionally fail to detect that Y is not a local cause of T. Sound causal discovery algorithms do not face this problem, however. In addition, the preference for maximum gap classifier biases in favor of assigning non-zero weights to redundant features (Y in the example).

On the positive side, theoretical results show that SVMs in the large sample will assign zero weights to irrelevant variables (Hardin et al., 2004). Despite this theoretical good property, in the experiments of Aliferis et al. (2010) it was found that in realistic finite sample weights of irrelevant variables are non-zero. In the work of Statnikov et al. (2006) it was found that weights of irrelevant features occasionally exceed those of weakly relevant features and furthermore that SVM weights are also susceptible to assigning larger weights to synthesis features rather than direct causes and effects.

7.4 Wrapping

One of the widely-cited advantages of wrapping as a feature selection method is that it allows to tailor the selection of features to the inductive bias of the classifier (Kohavi and John, 1997). We show here how this property when combined with rich connectivity may yield causally misleading results. Consider the generative process of Figure 17. The target T is a quadratic function of its true causes A, B. Variables X, Y are effects of A, B respectively with similar non-linear functional relationships. A causal discovery procedure such as HITON-PC given enough sample and a suitable statistical test of independence will discover $\{A, B\}$ as the correct set of direct causes and direct effects. Consider however a practitioner who attacks the problem of learning a good classifier for T and reducing the necessary feature set using wrapping instead. If, as would normally be the case, the analyst starts with a simpler model class before proceeding to consider more complex

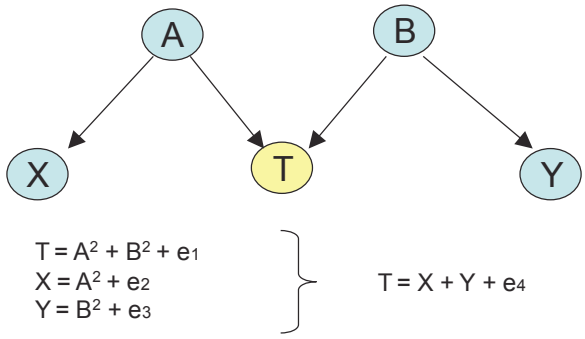
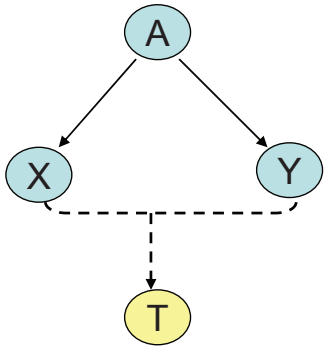


Figure 17: Example showing that wrapping, by tailoring feature selection to the classifier inductive bias may produce causally misleading results.



P(T | X, Y)	(X=0, Y=0)	(X=0, Y=1)	(X=1, Y=0)	(X=1, Y=1)
T=0	1	0	0	1
T=1	0	1	1	0

P(X | A)	A=0	A=1
X=0	0.40	0.75
X=1	0.60	0.25

P(A)	
A=0	0.71
A=1	0.29

P(Y | A)	A=0	A=1
Y=0	0.80	0.10
Y=1	0.20	0.90

Figure 18: Example showing that connectivity may mitigate violations of faithfulness. Dashed line indicates a highly non-linear function (XOR). The left part shows the causal structure, while the right part shows its parameterization.

ones, assuming that noise components e2, and e3 are small enough then the linear classifier would perform very well with $\{X, Y\}$ as predictors and a wrapper tailored to the linear inductive bias would eliminate A and B.

In small networks with a few variables and limited connectivity the above possibility is small, however in large networks with thousands of variables and rich connectivity as well as with massive information redundancy (e.g., biological networks) such "variable replacement" is entirely feasible and thus tailoring feature selection to a classifier's inductive bias (as wrapping does) can be an obstacle to sound causal discovery.

7.5 Connectivity and Priors Compensating for Violations of Faithfulness - Learning XOR Parents Using Univariate Association in GLL and Other Algorithms

A violation of faithfulness where constraint-based algorithms are expected to fail is when the target is an extremely non-linear function of its parents. A prototypical example is when T is the parity (XOR) of its parents A and B. Conventional wisdom, based on the truth table of the XOR function, dictates that first-order effects are zero and, as a result, the parents cannot be detected by the inclusion heuristic of the algorithm (i.e., HITON-PC or MMPC). As shown in Figure 18 however,

connectivity among variables can mitigate this difficulty. In the figure, variables X and Y can have non-zero univariate association with T, even though in textbook descriptions of parity where parents are unconnected and with 50% prior probability each for being 0 or 1, univariate association vanishes. An example parameterization that allows for this effect is given in the figure as well. This counter-intuitive phenomenon occurs because when X and Y are common effects of A, knowing the value of X is informative about A and thus about Y. Therefore the joint values of $\{X,Y\}$ are constrained and this creates univariate association of X and Y with T. Similarly, conditional association of X with T given Y is non zero. The phenomenon is not restricted to parity (or other extremely non-linear) functions in which the parity parents are connected in the network. Figure 20 in the Appendix shows an example where skewed priors on the unconnected parity parents X, Y lead to non-zero univariate association of X and Y with the target T.

The phenomenon described in this sub-section does not only apply to GLL algorithms but extends to other feature selectors as well. For example, the success of univariate filtering as feature selector, which has been documented in many domains (Guyon et al., 2006), can in part be explained via connectivity effects that allow univariate association to detect complex non-linear relationships of selected features with the target variable.

The discussion in this section is complemented by analysis of embedded feature selection in decision tree induction and of RELIEF in the online supplement Figures S5 and S6 (omitted here due to space limitations). It is shown that these algorithms can admit false positives and false negatives both predictively and causally with respect to the target variable neighborhood.

8. Discussion and Open Problems

In this section we present a thorough discussion of results, outline open problems and future directions, and provide a conclusion.

8.1 Discussion of Results

The algorithms presented, and their applied evaluation and theoretical analysis clarify many of the initially open questions discussed in Aliferis et al. (2010) and point to several new research directions. We showed that in empirical tests with 9 simulated data sets, GLL convergence to optimal performance is very fast with respect to sample size both in the sense of producing feature sets that have equal predictivity as the true $MB(T)$ and $PC(T)$ sets, and in the sense of achieving near optimal predictivity even at moderate samples sizes. These results corroborate the empirically good performance of GLL instantiations in real data sets (Aliferis et al., 2010).

An unexpected and important finding was that *GLL algorithms exhibit strong intrinsic control of false positives due not only to weakly relevant but also due to irrelevant features*. This control is empirically better in the tested data sets than what formal state-of-the-art FDR control provides except in the rare case when the data consists exclusively of irrelevant features. In Statnikov et al. (2010) we show that GLL can discover differentially expressed genes when the sample size is so small that FDR does not yield any gene. The same cannot be said for other feature selection methods that were found to be particularly prone to false positives due to both irrelevant and weakly relevant features. On the other hand, it needs to be noted that classical FDR methods do not control at all weakly relevant false positives (as GLL does). A simple pre-filtering of GLL algorithms with an FDR control method eliminates false positives in all cases tested and yields the best algorithm for

local causal learning among tested algorithms. We expect that other algorithms for example PC and MMHC will benefit from such an FDR prefiltering as well.

Within the GLL framework both the *max-k* and *h-ps* parameters control the false positives and false negatives tradeoff, through control of combined power and combined significance levels. We examined via targeted experiments and theoretical discussion the complex determination of quality of statistical decisions in GLL algorithms (aspects of which are shared by previous global constraint-based algorithms). Having two parameters to control quality of statistical decisions confers advantages since they can regulate different aspects of such decisions, and trade-off statistical quality with computational complexity.

Our efforts to explain the good predictive performance of the estimated $PC(T)$ set compared to the estimated $MB(T)$ set focused on producing explanations consistent with sufficient assumptions for Markov blanket optimality so that the good performance of the $PC(T)$ set would not be wrongly construed as entailing rejection of the theoretical assumptions, or as inability to infer the correct $MB(T)$ when the assumptions hold in the data. This is because both the results of our simulated experiments in Aliferis et al. (2010) as well as previously published experiments (Tsamardinos et al., 2003b) show that GLL algorithms estimate very well the $MB(T)$ and $PC(T)$ sets.

We also used a causal graph point of view and Markov blanket concepts to understand a variety of non-causal feature selection algorithms. This approach *provides a cohesive and fresh perspective into the behavior of several algorithms for feature selection*. We made this point by showing that the theory readily reveals why prominent feature selection methods exhibit many false positives and why they cannot be used for sound causal discovery. This complements the findings of Aliferis et al. (2010) that demonstrate empirical feature selection and causal discovery suboptimality for many state-of-the-art non-causal feature selection methods.

We discussed in detail a fundamental statistical weakness of wrapping, namely that it is prone to errors due to imperfect error estimation. This is especially the case when sample size is small whereby practical unbiased error estimators have large variance. The same problem applies implicitly to widely-used feature selection approaches such as ranking by univariate association and selecting the first k features. We showed why GLL algorithms are less sensitive to this shortcoming. In general our results show that GLL instantiations are robust enough to apply across a wide variety of domains.

Established feature selection criteria in statistics such as the AIC (Akaike Information Criterion) bare some resemblance to Markov blanket feature selection in the sense that AIC does not require classification error estimation. Specifically, AIC balances the number of features (parameters) with the likelihood of the data given a model: $\text{AIC} = 2k - 2log(L)$, where k is the number of parameters and L is the likelihood function. Model selection is driven by optimizing AIC. A critical difference however is that Markov blanket induction does not require a generative model of the data to be calculated (but relies on conditional independence tests). Given that inducing a generative model is in general harder than finding features that cannot be rendered independent of the target, and given that many recent powerful classifiers do not build generative models (e.g., SVMs) it follows that the Markov blanket induction approach has a corresponding advantage over AIC. Markov blanket induction is less model-dependent than AIC for the same reason. Note that similarly the GLL algorithms by not attempting to induce edge directionality (a task harder than edge detection, Ramsey et al., 2006) except when absolutely necessary they avoid incurring errors in edge detection produced by false conclusions about directionality (since one type of discovery affects the other). As

a result, Markov blanket induction via the GLL framework has advantages over eliciting Markov blankets by using methods that require global or local orientation.

The extensive evaluation of GLL algorithms in Aliferis et al. (2010) shows that the sufficient conditions stated in the proofs for correctness are likely to hold often, or that violations may be small. In some cases we showed that the algorithms may not fail when the assumptions are violated. Due to the critical role of non-faithfulness as a major source of possible failure we discuss it here in more detail. Faithfulness is violated in a variety of situations (Spirtes et al., 2000), notably in practice when (a) extremely non-linear or deterministic functions exist, when (b) causality cannot be localized, and when (c) variables share the same information for a response (target variable). Practical examples, respectively, are extreme epistasis in genetics, non-local causation in quantum mechanics, and gene-phenotype information redundancy in gene expression microarrays. For many additional reasons see Spirtes et al. (2000) and Meek (1995).

However, we showed that even in prototypical non-faithful functions such as XOR, the existence of unbalanced priors or the existence of connectivity among XOR parent variables of the target can make such parent variables visible again to the GLL algorithms as well as other feature selectors (e.g., univariate association filtering). We believe that this finding may have broad implications of which we mention a few. First, it explains in part the success of univariate feature selection methods in many domains since univariate filtering can pick up features that are involved in extremely non-linear functions. Second, other algorithms that are typically thought to not be able to learn such functions, such as Genetic Algorithms (Sharpe, 2000) in many situations may be able to do just that. In addition, to the extent that biological systems have evolved by evolutionary processes similar to genetic algorithms, truly extreme epistatic functions may not be as rare as previously thought. Recent proposals that suggest that such functions (i.e., biological systems) can be learned (i.e., evolved) by GAs (i.e., by evolution) through multiple objective optimization may be too pessimistic (Lenski et al., 2003). Third, previous postulates that randomized experiments (e.g., in biology, medicine and psychology) because they examine one causal factor at a time are thus unable to detect parity-like functions, may also be pessimistic (Aliferis and Cooper, 1998).

Returning to non-local causality, we point out that cognitively it is advantageous to modularize causal knowledge in order to reduce the connectivity of causal graphs and thus to control learning complexity (as well as to increase ability to store and process such knowledge with limited cognitive resources). We may thus be facing in both natural as well as artificial systems a selection bias (relative to all possible theoretical distributions) where causal systems and models of those are highly modular because it is easier to create and handle such systems and their models. Indeed in most known macroscopic causal processes (e.g., biological pathways, medicine, engineering, economics, social networks) causal systems are highly modular and thus local.

For all of the above reasons faithfulness is a very reasonable a priori, and powerful in practice, distributional assumption. At the same time at least some violations can be tolerated well by causal algorithms that are designed to use it and existing research addresses violations systematically, for example extensions of standard causal discovery algorithms capable of addressing target information equivalency (Statnikov, 2008).

The exploration of parallel and distributed techniques in the present paper showed that *GLL is amenable to parallelized and distributed local causal discovery and feature selection*. We established empirically the potential of parallelization for speeding up processing time without loss of quality. The presented parallel algorithm can also be used for distributed feature selection and causal discovery in a principled manner. Many more algorithms (namely that induce Markov blankets and

admit symmetry correction when needed) can be constructed following the approach introduced in parallel and distributed IAMB for Markov blanket induction (Aliferis et al., 2002). In contrast to parallel IAMB however, parallel GLL-PC can be exponentially faster (or slower) than induction in the full data. This is a very interesting future research direction.

In exploring the transition from local-to-global strategies we showed that the local-to-global learning framework LGL can be instantiated in several ways. We examined one new instantiation of local-to-global learning, algorithm HHC. Although in most real data tested a random variable order is as good as perfectly-informed ordering by local connectivity, we showed in the present paper something previously unnoticed, namely that in some cases the right order of local neighborhood learning can entail exponential time vs. low-order polynomial time execution of local-to-global algorithms. This finding has a subtle implication: if the right ordering can be found for local learning, the resulting global learning of all variables can be faster than the local learning targeted at just one variable. Thus, just as local learning can speed up global learning the reverse may also be true.

On the other hand, our results showed that the idea that non-causal feature selection methods could help in addressing scalability of formal causal algorithms may be misplaced in light of the failure of non-causal feature selection methods to induce causality and given that highly scalable and sound methods such as GLL algorithms do exist. Several tested algorithms where non-causal feature selection is used to elicit a skeleton which is then oriented and refined by formal causal global methods are very slow and typically produce lower-quality graphs than LGL instantiations relying on sound local causal methods.

8.2 Open Problems and Future Directions

The results presented in Aliferis et al. (2010) and in the present paper merely scratch the surface of causal feature selection algorithms, local causal learning, and local-to-global learning. We briefly discuss here a few salient opportunities for moving this exciting area forward.

An assumption that is probably too strong for soundness of $MB(T)$ induction is that of causal sufficiency. For example, we conjecture without formal proof, that the algorithms should attain soundness even if the causal sufficiency is localized among the target and the members of its Markov blanket. Even when this local causal sufficiency is violated, predictive optimality among measured variables may not be compromised in many practical situations (although the usual causal interpretation of the found features is affected). Characterizing localized versions of faithfulness and causal sufficiency is an area that is likely to give a better understanding of existing algorithms and possibly lead to improvements. Examining and dealing with the effects of temporal aggregation, sampling (e.g., cellular) aggregation, feedback loops, and limited local causality on feasibility of local causal discovery will be helpful in determining the space of practical usefulness of the GLL framework.

A previously underemphasized important parameter for false negatives control is the order of conditional independence tests used for elimination (i.e., part of the elimination strategy in the GLL-PC schema). In general, the earlier time that strongly relevant variables are being examined for elimination, the better the chances for avoiding a false negative conditional independence test result since the combined power is larger. This is accomplished implicitly in HITON-PC and MMHC by using heuristics that include strongly relevant features first in $TPC(T)$ and then in both semi-interleaved HITON-PC and MMHC, where new candidates are considered for elimination *first* and where conditioning sets are constructed with stronger candidates for $PC(T)$ *first*. Systematic study

of such prioritization schemes may yield performance benefits over existing GLL instantiations. Other areas that may yield improved performance is selective or full model averaging to address instability of $MB(T)$ estimation in small samples and optimizing alpha thresholds and FDR thresholds either for a domain or a data set, possibly separately for each variable.

In general, the treatment of determination of unreliable tests by means of the heuristic rule and parameter *h-ps* in GLL instantiations can be improved by incorporating formal power-size analysis whenever possible. More broadly, removing the requirement for a uniform sample size requirement across independence tests of same order (but different response function) is likely to yield improved algorithms. Other statistical issues such as improved statistical handling of structural zeros for discrete statistics, improved statistical tests that combine discrete and continuous data, handling "forced" covariates (i.e., variables that need to remain in $TPC(T)$ or $TMB(T)$ so that a particular effect is controlled for) are also worth exploring. Related to proper statistical testing is the issue of optimal discretization, not for classification as has been explored before in the literature, but for causal discovery (for a study toward that direction see Fu 2005). Other statistical extensions are to adapt the GLL method for survival analysis, or other time-to-event analyses without discretizing outcomes and with ability to handle observation censoring.

Exploitation of prior knowledge and development of methods to exploit prior causal knowledge (e.g., variable ordering, forced edges, forbidden edges, known size of local neighborhoods, known directionalities/structure and degree of connectivity, etc.) may yield greatly improved methods. Comparisons of knowledge-enhanced to pure data-driven instantiations will then be very informative.

An obvious possibility not examined in the present work is using GLL methods for regression. Another natural line of future research is to study situations where a loss function does not require exact knowledge of the conditional probability $P(T|MB(T))$ in which a promising strategy is to use a wrapping post-processing step to remove unnecessary features thus tailoring the final feature set to a loss function less stringent than the ones that typically guarantee soundness for GLL-MB algorithms.

Different distributional assumptions, for example monotone DAG faithfulness to make GLL and LGL algorithms faster (for a first attempt see Brown et al. 2005) may provide algorithms that tradeoff well quality for speed in specific domains.

Although we did not address the issue in this work, post-processing the results of GLL and LGL output using algorithms that detect hidden variables and orient edges is an obvious direction for research.

The study of convergence behavior of GLL and of false discovery rate control were either empirical or qualitative in the present paper. Derivation of mathematical analyses of convergence to the optimal $MB(T)$ and optimal classifier (as function of sample size), of effects of synthesis, of how common synthesis is, of combined power and alpha for specific distributions will be very interesting, especially as other components of the framework (for example handling of unreliable tests) are also formalized.

Developing methods that handle efficiently very large neighborhoods with hundreds of features and small sample size, as well as developing methods for special-purpose causal structures (e.g., genome-wide association studies) is also an area where significant improvements can be made.

The skeleton phase of LGL is a form of dynamic programming and this explains its efficiency and soundness and probably leaves reduced opportunity for dramatic efficiency improvements. One possible avenue would be the exploration of different strategies for linking together the local skele-

P(T | C, D)	(D=0, C=0)	(D=0, C=1)	(D=1, C=0)	(D=1, C=1)
T=0	0.55	0.45	0.48	0.45
T=1	0.45	0.55	0.52	0.55

P(E | T, F)	(T=0, F=0)	(T=0, F=1)	(T=1, F=0)	(T=1, F=1)
E=0	0.6	0.4	0.55	0.55
E=1	0.4	0.6	0.45	0.45

P(A | B, E)	(B=0, E=0)	(B=0, E=1)	(B=1, E=0)	(B=1, E=1)
A=0	0.90	0.03	0.04	0.03
A=1	0.03	0.90	0.03	0.03
A=2	0.03	0.04	0.90	0.04
A=3	0.04	0.03	0.03	0.90

P(C)	
C=0	0.50
C=1	0.50

P(F)	
F=0	0.50
F=1	0.50

P(D)	
F=0	0.50
F=1	0.50

P(B | C)	C=0	C=1
B=0	0.98	0.02
B=1	0.02	0.98

Figure 19: Parameterization of the network in Figure 13.

ton results (step #2 in LGL schema). Both MMHC and HHC use an "OR" strategy but many alternative approaches can be devised. Furthermore, the edge orientation step may be greatly improved over the use of greedy search-and-score. Numerous other obvious instantiations of LGL (for instance combining GLL-PC versions with global algorithms such as GES, and TPDA) can also be implemented with substantial potential for good empirical performance. Moreover, methods to automatically identify optimal variable prioritization for local learning can yield improvements in certain distributions and we outlined related research directions in Section 6.3.

Finally, extending the framework to address broader definitions of feature selection is particularly important. Examples include finding: all sets that give desired trade-off between feature number and predictivity; all sets with smallest cost that give highest predictivity (i.e., when different observation costs apply for each variable); and all sets that optimize arbitrary multi-attribute utility/loss functions.

8.3 Conclusions

The empirical and theoretical results presented in the present paper and its companion paper (Aliferis et al., 2010) support the notion that local causal learning in the form of Markov blanket and local neighborhood induction is a theoretically well-motivated and empirically robust learning methodology as embodied in the Generalized Local Learning framework. Generalized Local Learning yields algorithms with excellent performance in data analysis geared toward classification and causal discovery. Local-to-global learning strategies have the potential to enhance large-scale causal discovery. Several existing open problems offer possibilities for non-trivial theoretical and practical discoveries, making this an exciting field of research.

Appendix A.

This Appendix provides additional tables and figures referenced in the paper.

Bayesian network	Number of variables	Training samples	Number of selected targets
Child10	200	5 x 200, 5 x 500, 1 x 5000	10
Insurance10	270	5 x 200, 5 x 500, 1 x 5000	10
Alarm10	370	5 x 200, 5 x 500, 1 x 5000	10
Hailfinder10	560	5 x 200, 5 x 500, 1 x 5000	10
Munin	189	5 x 500, 1 x 5000	6
Pigs	441	5 x 200, 5 x 500, 1 x 5000	10
Link	724	5 x 200, 5 x 500, 1 x 5000	10
Lung_Cancer	800	5 x 200, 5 x 500, 1 x 5000	11
Gene	801	5 x 200, 5 x 500, 1 x 5000	11

Table 15: Simulated and resimulated data sets used for experiments. The *Lung_Cancer* network is resimulated from human lung cancer gene expression data (Bhattacharjee et al., 2001) using the SCA algorithm (Friedman et al., 1999). The *Gene* network is resimulated from yeast cell cycle gene expression data (Spellman et al., 1998) using SCA algorithm. More details about data sets are provided in Tsamardinos et al. (2006).

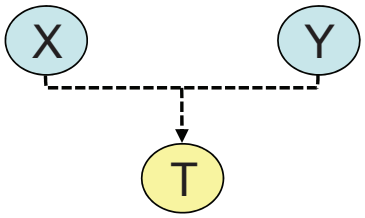
HITON-PC (max k=4)	**Interleaved MMPC** (max k=2)
HITON-PC (max k=3)	**Interleaved MMPC** (max k=1)
HITON-PC (max k=2)	**HITON-MB** (max k=3)
HITON-PC (max k=1)	**MMMB** (max k=3)
Interleaved HITON-PC (max k=4)	**RFE** (reduction of features by 50%)
Interleaved HITON-PC (max k=3)	**RFE** (reduction of features by 20%)
Interleaved HITON-PC (max k=2)	**UAF-KruskalWallis-SVM** (50%)
Interleaved HITON-PC (max k=1)	**UAF-KruskalWallis-SVM** (20%)
MMPC (max k=4)	**UAF-Signal2Noise-SVM** (50%)
MMPC (max k=3)	**UAF-Signal2Noise-SVM** (20%)
MMPC (max k=2)	**L0**
MMPC (max k=1)	**LARS-EN** (for multiclass response)
Interleaved MMPC (max k=4)	**LARS-EN** (one-versus-rest)
Interleaved MMPC (max k=3)	

Table 16: Algorithms used in local causal discovery experiments with simulated and resimulated data.

References

C. F. Aliferis and G. F. Cooper. Aspects of modeling with mtbn's. *Technical report CBMI 1998-3, Center for Biomedical Informatics, University of Pittsburgh*, 1998.

C. F. Aliferis and A. Statnikov. Dynamic ordering-based global learning. *Technical report DSL-08-02*, 2008.



P(T | X, Y)	(X=0, Y=0)	(X=0, Y=1)	(X=1, Y=0)	(X=1, Y=1)
T=0	1	0	0	1
T=1	0	1	1	0

P(X)	
X=0	0.20
X=1	0.80

P(Y)	
Y=0	0.90
Y=1	0.10

Figure 20: In this example, $T = XOR(X,Y)$. The priors of X and Y are given in the table. Both X and Y have very strong univariate association with T despite being XOR parents and in the absence of connectivity.

C. F. Aliferis, I. Tsamardinos, and A. Statnikov. Large-scale feature selection using markov blanket induction for the prediction of protein-drug binding. *Technical Report DSL 02-06*, 2002.

C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification. part i: Algorithms and empirical evaluation. *Journal of Machine Learning Research*, 11:171–234, 2010.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society.Series B (Methodological)*, 57(1):289–300, 1995.

Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Ann.Statist*, 29(4):1165–1188, 2001.

A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. J. Mark, E. S. Lander, W. Wong, B. E. Johnson, T. R. Golub, D. J. Sugarbaker, and M. Meyerson. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc.Natl.Acad.Sci.U.S.A*, 98(24):13790–13795, Nov 2001.

L. E. Brown, I. Tsamardinos, and C. F. Aliferis. A comparison of novel and state-of-the-art polynomial bayesian network learning algorithms. *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, 2005.

G. Casella and R. L. Berger. *Statistical Inference*. Thomson Learning, Australia, 2nd edition, 2002.

N. Friedman, I. Nachman, and D. Pe'er. Learning bayesian network structure from massive datasets: the "sparse candidate" algorithm. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 1999.

L. D. Fu. A comparison of state-of-the-art algorithms for learning bayesian network structure from continuous data. Master's thesis, Vanderbilt University, 2005.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, 2002.

I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction: Foundations and Applications*. Springer-Verlag, Berlin, 2006.

D. Hardin, I. Tsamardinos, and C. F. Aliferis. A theoretical characterization of linear svm-based feature selection. *Proceedings of the Twenty First International Conference on Machine Learning (ICML)*, 2004.

I. S. Kohane, A. T. Kho, and A. J. Butte. *Microarrays for an Integrative Genomics*. MIT Press, Cambridge, Mass, 2003.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2): 273–324, 1997.

R. E. Lenski, C. Ofria, R. T. Pennock, and C. Adami. The evolutionary origin of complex features. *Nature*, 423(6936):139–144, May 2003.

D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. *Advances in Neural Information Processing Systems*, 12:505–511, 1999.

C. Meek. Strong completeness and faithfulness in bayesian networks. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 411–418, 1995.

J. Ramsey, J. Zhang, and P. Spirtes. Adjacency-faithfulness and conservative causal inference. *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, 2006.

M. Schmidt, A. Niculescu-Mizil, and K. Murphy. Learning graphical model structure using l1-regularization paths. *Proceedings of the Twenty-Second National Conference on Artificial Intelligence (AAAI)*, 2007.

O. J. Sharpe. *Towards a Rational Methodology for Using Evolutionary Search Algorithms*. PhD thesis, University of Sussex, 2000.

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol.Biol Cell*, 9(12):3273–3297, Dec 1998.

P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, Prediction, and Search*, volume 2nd. MIT Press, Cambridge, Mass, 2000.

A. Statnikov. Algorithms for discovery of multiple markov boundaries: Application to the molecular signature multiplicity problem. *Ph.D.Thesis, Department of Biomedical Informatics, Vanderbilt University*, 2008.

A. Statnikov, D. Hardin, and C. F. Aliferis. Using svm weight-based methods to identify causally relevant and non-causally relevant variables. *Proceedings of the NIPS 2006 Workshop on Causality and Feature Selection*, 2006.

A. Statnikov, J. Feig, E. Fisher, and C.F. Aliferis. Novel bioinformatics methods for discovery of complex molecular signatures, pathways, and biomarkers in very small sample situations. *Submitted*, 2010.

I. Tsamardinos and C. F. Aliferis. Towards principled feature selection: relevancy, filters and wrappers. *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (AI & Stats)*, 2003.

I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Algorithms for large scale markov blanket discovery. *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pages 376–381, 2003a.

I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Time and sample efficient discovery of markov blankets and direct causal relations. *Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 673–678, 2003b.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

Journal of Machine Learning Research 11 (2010) 285-310 Submitted 6/09; Revised 10/09; Published 1/10

Optimal Search on Clustered Structural Constraint for Learning Bayesian Network Structure

Kaname Kojima KANAME@IMS.U-TOKYO.AC.JP
Eric Perrier PERRIER@IMS.U-TOKYO.AC.JP
Seiya Imoto IMOTO@IMS.U-TOKYO.AC.JP
Satoru Miyano MIYANO@IMS.U-TOKYO.AC.JP
Human Genome Center, Institute of Medical Science
University of Tokyo
4-6-1, Shirokanedai, Minato-ku, Tokyo 108-8639, Japan

Editor: David Maxwell Chickering

Abstract

We study the problem of learning an optimal Bayesian network in a constrained search space; skeletons are compelled to be subgraphs of a given undirected graph called the super-structure. The previously derived constrained optimal search (COS) remains limited even for sparse super-structures. To extend its feasibility, we propose to divide the super-structure into several clusters and perform an optimal search on each of them. Further, to ensure acyclicity, we introduce the concept of ancestral constraints (ACs) and derive an optimal algorithm satisfying a given set of ACs. Finally, we theoretically derive the necessary and sufficient sets of ACs to be considered for finding an optimal constrained graph. Empirical evaluations demonstrate that our algorithm can learn optimal Bayesian networks for some graphs containing several hundreds of vertices, and even for super-structures having a high average degree (up to four), which is a drastic improvement in feasibility over the previous optimal algorithm. Learnt networks are shown to largely outperform state-of-the-art heuristic algorithms both in terms of score and structural hamming distance.

Keywords: Bayesian networks, structure learning, constrained optimal search

1. Introduction

Although structure learning is a fundamental task for building Bayesian networks (BNs), when minimizing a score function, the computational complexity often prevents us from finding optimal BN structures (Perrier et al., 2008). With currently available exact algorithms (Koivisto et al., 2004; Ott et al., 2004; Silander et al., 2006; Singh et al., 2005) and a decomposable score like BDeu, the computational complexity remains exponential, and therefore, such algorithms are intractable for BNs with more than around 30 vertices given our actual computational capacity. For larger systems, heuristic searches like greedy hill-climbing search (HC) or customized versions of this search are employed in practice (Tsamardinos et al., 2006).

Recently, Tsamardinos et al. (2006) proposed an algorithm called max-min hill-climbing (MMHC) that combines an independence test (IT) approach with a score-based search strategy: first, an undirected graph is built based on an IT approach, and then, a constrained greedy hill-climbing search returns a local optimum of the score function. Thus, MMHC can be considered as a constrained search, a concept introduced by Friedman et al. (1999) together with the sparse can-

didate (SC) algorithm. Such algorithms have been empirically shown to outperform unconstrained greedy hill-climbing (Friedman et al., 1999; Tsamardinos et al., 2006). Based on the success of constrained approaches, Perrier et al. (2008) proposed an algorithm that can learn an optimal BN when an undirected graph is given as a structural constraint. Perrier et al. (2008) defined this undirected graph as a super-structure; the skeleton of every graph considered is compelled to be a subgraph of the super-structure. This algorithm can learn optimal BNs containing up to 50 vertices when the average degree of the super-structure is around two, that is, a sparse structural constraint is assumed. However, its feasibility remains limited.

Independently, Friedman et al. (1999) suggested that when the structural constraint is a directed graph (in the case of SC), an optimal search can be carried out on the cluster tree extracted from the constraint. This cluster-based approach could potentially increase the feasibility of optimal searches; nevertheless, the algorithm proposed in Friedman et al. (1999) requires to be given a directed graph-based constraint and to extract a cluster tree. For the latter, a large cluster might be generated, preventing an optimal search from being carried out.

Another potential approach is to search the best BN by checking the network obtained by withdrawing edges in cycles one-by-one, beginning from an initial network which is build by connecting children and their optimal parents with directed edges without checking acyclicity as in B&B algorithm (de Campos et al., 2009). However, children in the best BN are often selected as the best parents without considering acyclicity if the size of a given data set is sufficient. Thus, for the estimation of the best BN of more than hundred vertices and sufficient data samples, the initial network may contain hundreds of small cycles, and it is impossible to check these cycles in the search process.

In this study, we take up the concept of a super-structure constraint and propose a cluster-based search algorithm that can learn an optimal BN given the constraint. Therefore, unlike in Friedman et al. (1999), our algorithm uses an undirected graph as the structural constraint. In addition, we use a different cluster decomposition that enables us to consider more complex cases. As Tsamardinos et al. (2006) and Perrier et al. (2008) showed, good approximations of the true super-structure can be obtained by an IT approach like the max-min parent-children (MMPC) method (Tsamardinos et al., 2006).

If the super-structure is divided into clusters of moderate size (around 30 vertices), a constrained optimal search can be applied on each cluster. Then, to find a globally optimal graph, one could consider all patterns of directions for the edges between clusters and apply a constrained optimal search on each cluster for every pattern of directions independently and return the best result found. We theorize this idea by introducing ancestral constraints; further, we derive the necessary and sufficient ancestral constrains that we must consider to find an optimal network and introduce a pruning technique to skip superfluous cases. Finally, we develop a super-structure constrained optimal algorithm that extends the size of networks that we can consider by more than one order.

The performance of our algorithm is evaluated on the Alarm, Insurance, and Child networks (Beinlich et al., 1989; Binder et al., 1997; Cowell et al., 1999) extended by the tiling method (Tsamardinos et al., 2006) to obtain networks having several hundreds of vertices. Experiments show that our algorithm clearly outperforms MMHC and HC with the TABU search extension.

2. Related Works

Given data D for a set of random variables V, learning an optimal BN using a decomposable score like BDeu involves finding a directed acyclic graph (DAG) N^* such that

$$N^* = \arg\min_N \sum_{v \in V} s(v, Pa_N(v); D), \tag{1}$$

where $Pa_N(v) \subseteq V$ is a set of parents for a vertex v in network N and $s(v, Pa_N(v); D)$ is the value of the score function for v in N. Hereafter, we omit the subscript N for Pa and D for s. In this section, we introduce some structure learning algorithms to show the motivation of our research.

2.1 Optimal search

Although finding a global optimum, that is, a solution of (1), is NP-hard, several optimal algorithms have been developed (Koivisto et al., 2004; Ott et al., 2004; Silander et al., 2006; Singh et al., 2005). The time complexity has been successfully reduced to $O(n2^n)$, where n is the number of vertices in BN (i.e., $|V| = n$).

2.2 Hill-climbing

For learning a larger system, heuristic algorithms must be used. Greedy hill-climbing (HC) is one of the most commonly used algorithms in practice. HC only finds local optima, and upgraded versions of this base algorithm have been extensively studied, leading to some improvement in the score and structure of the results (e.g., by using a TABU list).

2.3 Sparse Candidate

To improve HC, Friedman et al. (1999) limited the maximum number of parents and restricted the set of candidate parents for each vertex. They established SC algorithm and introduced the concept of constraining the search space of score-based approaches.

2.4 Max-min Hill-climbing

MMHC is a hybrid method combining an IT approach and a score-based search strategy. Tsamardinos et al. (2006) showed that on average, MMHC outperforms other heuristic approaches including SC and HC.

2.5 Constrained Optimal Search

In SC and MMHC, the learnt structures are local optima. Perrier et al. (2008) extended the optimal algorithm of Ott et al. (2004) and established a constrained optimal search (COS) that learns an optimal BN structure whose skeleton is a subgraph of a given undirected graph $G = (V, E)$ called the super-structure, that is, COS aims to find N^*_G, the solution of (1), while constraining $Pa_N(v)$ to be included in $\mathcal{N}(v)$, where $\mathcal{N}(v)$ is the neighborhood of v in G. Although using a super-structure increases the feasibility of optimal searches, COS is still limited when the super-structure is dense (high average degree).

2.6 Optimal Search with Cluster Tree

Friedman et al. (1999) suggested the possibility of optimally searching acyclic subgraphs of a digraph constructed by connecting each vertex and its pre-selected candidate parents with directed edges without checking acyclicity. Here, unlike MMHC and COS, the structural constraint is represented by a directed graph. An algorithm would proceed by converting the digraph into a cluster tree, where clusters are densely connected subgraphs. Then, it would perform an optimal search on each cluster for every ordering of the vertices contained in the separators of clusters. However, due to the difficulty of building a minimal cluster tree, large clusters can make the search impractical.

2.7 B&B Algorithm

Recently, de Campos et al. (2009) proposed an optimal branch-and-bound algorithm. This algorithm constructs an initial directed graph by linking every vertex to its optimal parents although this might create directed cycles. Then, it tries to search every possible case in which the direction of one edge comprising each directed cycle is constrained for keeping acyclicity, and finds optimal parents under the constraints iteratively until DAGs are obtained. After the completion of the full search, the optimal solution is finally given by the best DAG found. In addition, for score functions decomposable into penalization and fitting components, optimal parents under the constraints are further effectively computed using a branch-and-bound technique that was originally proposed by Suzuki et al. (1996). This method is interesting in that it is original and allows the development of an anytime search that returns the best current solution found and an upper bound to the global optimum. When the sample size is small, few directed cycles occur in the initial directed graph and updated graphs because information criteria tend to select a smaller parent set for each vertex in small sample data (Dojer, 2006). However, for a large sample size, due to the occurrence of a large number of directed cycles, the complexity of this method can be practically worse than classic optimal searches.

Thereafter, we will consider the same problem as in the COS approach, that is, to find N_G^*, an optimal BN constrained by the super-structure $G = (V,E)$, an undirected graph. In our case, we propose a cluster-based search to reduce the complexity drastically; here, clusters are of a different nature from the ones in Friedman et al. (1999), as shown in the next section.

3. Edge Constrained Optimal Search

In this section, we describe procedures of the proposed algorithm in a bottom-up manner. Under the assumption that the skeleton is separated to small subgraphs, we first describe the definition of ancestral constraints for each subgraph and consider an algorithm to learn an optimal BN on a subgraph under some ancestral constrains. We then explain the procedures in order to efficiently build up an optimal BN on the skeleton by using information of optimal BN on each subgraph under the conditions of ancestral constraints to be considered.

3.1 Ancestrally Constrained Optimal Search for A Cluster

Hereafter, we assume that we are given a set of edges $E^- \subset E$ such that the undirected graph $G^- = (V, E \setminus E^-)$ is not connected.

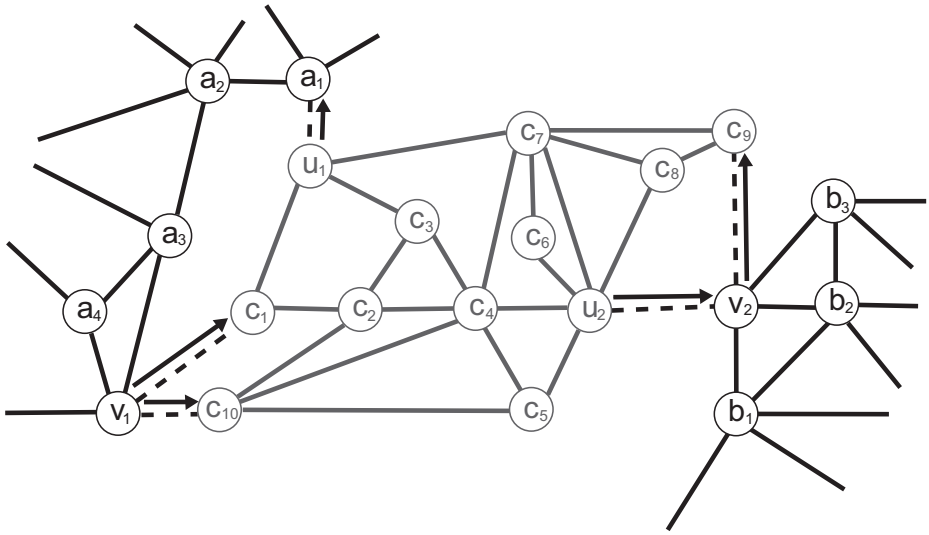


Figure 1: An example of a super-structure to illustrate the definitions we have introduced. The edges of E^- are dashed and they define a cluster C (gray). δ indicated by arrows is one of the 32 TDMs possible over E^C that defines $V^{in}_{C,\delta} = \{v_1, v_2\}$ and $V^{out}_{C,\delta} = \{u_1, u_2\}$.

Definition 1 (cluster and cluster edges) *Let $C = (V_C, E_C)$ be a maximal connected component of G^-. We refer to C as a* cluster *and call $E^C \subset E^-$ containing all and only the edges incident to a vertex in V_C the* set of cluster edges *for C.*

Definition 2 (tentative direction map) *Given a set of edges E^C, we define a* tentative direction map (TDM) δ *on E^C as a set of pairs $\{(e,d), e \in E^C$ and $d \in \{\leftarrow, \rightarrow\}\}$ such that for $\forall e \in E^C$, there uniquely exists d such that $(e,d) \in \delta$. In other words, δ associates a unique direction with each edge in E^C.*

In the following sections, we show that by successively considering every possible δ on E^- and learning the optimal BN on each cluster independently we can reconstruct N^*_G. However, to avoid creating cycles over several clusters, our method has to consider all the possible ancestral constraints for a cluster, a notion that we introduce hereafter.

Definition 3 (in-vertex and out-vertex) *Considering a cluster C and a TDM δ on E^C, we define $V^{in}_{C,\delta}$ and $V^{out}_{C,\delta}$ as $V^{in}_{C,\delta} = \{v \in V \setminus V_C \mid \exists v_a \in V_C, (\{v, v_a\}, \rightarrow) \in \delta\}$ and $V^{out}_{C,\delta} = \{v \in V_C \mid \exists v_a \in V \setminus V_C, (\{v, v_a\}, \rightarrow) \in \delta\}$, respectively. We drop the subscripts C and δ when there is no ambiguity. We call $v \in V^{in}_{C,\delta}$ an* in-vertex *and $v \in V^{out}_{C,\delta}$, an* out-vertex.

Figure 1 illustrates the previously introduced definitions. In this section, we assume C and δ to remain constant.

Definition 4 (ancestral constraints) *An* ancestral constraint (AC) *is a pair (v,u) with $v \in V^{in}_{C,\delta}$ and $u \in V^{out}_{C,\delta}$ that is used to disable v as an ancestor of u. Let $\mathcal{A}$ be a set of ancestral constraints* (ACS),

and $\mathcal{A}(v)$ be the set of all out-vertices u_i such that $(v,u_i) \in \mathcal{A}$. We say that $\mathcal{A}$ is a nested ancestral constraint set (NACS) *if and only if for any v_a and v_b in $V^{in}_{C,\delta}$, $\mathcal{A}(v_a) \subseteq \mathcal{A}(v_b)$ or $\mathcal{A}(v_a) \supseteq \mathcal{A}(v_b)$ holds. Finally, given two ACSs $\mathcal{A}$ and $\mathcal{B}$, if $\forall v \in V^{in}_{C,\delta}$, we have that $\mathcal{A}(v) \subseteq \mathcal{B}(v)$, and we say that $\mathcal{B}$ is* stronger than or equal to *$\mathcal{A}$ and denote this relation as $\mathcal{A} \leq \mathcal{B}$.*

Finally, we recall the definition of a topological ordering and some notions related to it.

Definition 5 (π-linearity and $\mathcal{A}$-linearity) *Given an ordering π (i.e., a bijection of $[|1,n|]$ in V), we say that it is a topological ordering of a BN N if for every v, $Pa(v)$ is included in $P_\pi(v) = \{u \in V \mid \pi^{-1}(u) < \pi^{-1}(v)\}$, the set of the predecessors of v; in such a case, N is said to be* π-linear. *Given an ACS $\mathcal{A}$ and a BN N, we say that N is* $\mathcal{A}$-linear *if and only if it respects all ACs in $\mathcal{A}$. In addition, in the case of a NACS, there exists a topological ordering π of N such that $\forall v \in V^{in}_{C,\delta}$ and $\forall u \in \mathcal{A}(v)$, $\pi^{-1}(u) < \pi^{-1}(v)$ holds.*

For notational brevity, if $\pi^{-1}(u) < \pi^{-1}(v)$ holds for two vertices v and u, we hereafter write $u \prec_\pi v$. Using the previous definitions, we can now prove the validity of our approach.

Theorem 1 *There exists δ^* on E^- and NACSs $\mathcal{A}^*_i$ for every cluster C_i of G^- coherent with a global optimal BN N^*_G. In other words, we can obtain N^*_G by considering the separately obtained optimal BN for every cluster, NACS, and TDM possible.*

Proof We consider an optimal DAG N^*_G and one of its topological orderings π^*. There exists a unique TDM δ^* coherent with π^*. From δ^*, we define for every cluster C_i the ACS $\mathcal{A}^*_i$ such that $\forall v \in V^{in}_{C_i,\delta^*}, \mathcal{A}^*_i(v) = P_{\pi^*}(v) \cap V^{out}_{C_i,\delta^*}$. $\mathcal{A}^*_i$ are definitely NACSs since for every v_a and $v_b \in V^{in}_{C_i,\delta^*}$ if $v_a \prec_{\pi^*} v_b$, then by definition, $\mathcal{A}^*_i(v_a) \subseteq \mathcal{A}^*_i(v_b)$. Further, the subgraphs N^*_i of N^*_G on the sets $V_i = V_{C_i} \cup V^{in}_{C_i,\delta^*}$ are $\mathcal{A}^*_i$-linear (because the definition of $\mathcal{A}^*_i$ is based on π^*). Furthermore, each N^*_i is an optimal graph on $V_{C_i} \cup V^{in}_{C_i,\delta^*}$ given the constraints of δ^* and $\mathcal{A}^*_i$ (otherwise, we could build a DAG having a lower score than N^*_G). Therefore, if we independently compute an optimal BN for every cluster, for every TDM and every NACS, and return the best combination, we can build a globally optimal BN on V. ■

From Theorem 1, the ACS to be considered is limited to only NACS, and N^*_G can be obtained by searching the best combination of an optimal BN separately obtained on every cluster and for every NACS and every TDM. Figure 2 shows the flowchart of the search strategy of our approach. First, an optimal BN and its score on every cluster for every NACS and every TDM are computed. Then, by using this information, an optimal BN and its score on a cluster obtained by merging two clusters are computed for every NACS and every TDM. After the repeated computation of optimal BNs and scores on merged clusters, an optimal BN and its score on a single cluster covering the super-structure are finally obtained. The details and validity of the algorithms shown in the flowchart are discussed in later sections.

The fundamental step involves learning an optimal BN on a cluster C for a given δ and $\mathcal{A}$; we call this algorithm ancestrally constrained optimal search (ACOS). To describe it, we need to recall some functions defined in optimal search algorithm by Ott et al. (2004); however, we prefer to use the notations introduced by Perrier et al. (2008) for the sake of simplicity.

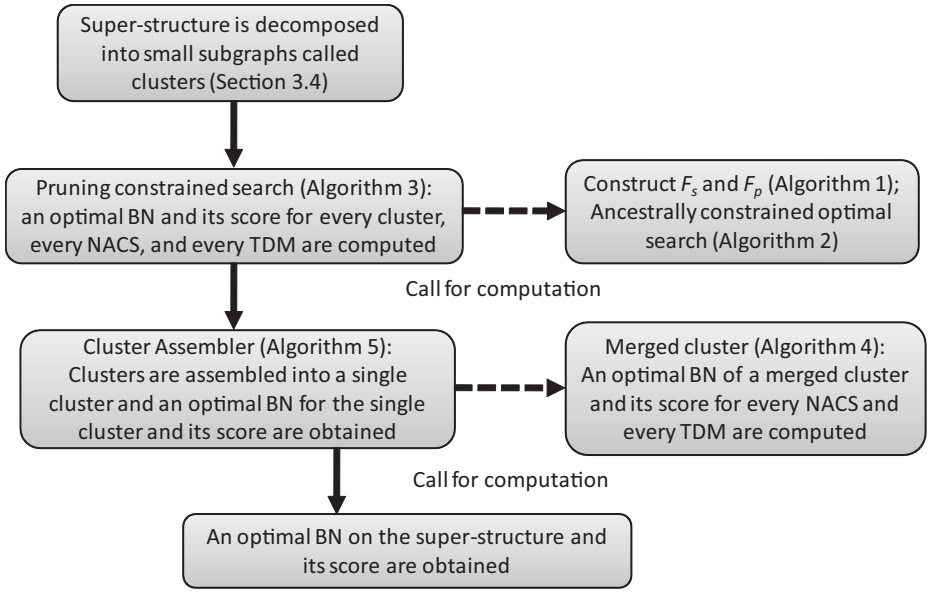


Figure 2: Flowchart of algorithms for the computation of an optimal BN on a super-structure. The details of these algorithms are discussed in later sections.

Definition 6 (Perrier et al. 2008) *For every $v \in V$ and every $X \subseteq \mathcal{N}(v)$, we define $F_p(v,X)$ as being the best parent set for v included in X and $F_s(v,X)$ as the associated score:*

$$\begin{aligned} F_s(v,X) &= \min_{Pa \subseteq X} s(v,Pa), \\ F_p(v,X) &= \arg\min_{Pa \subseteq X} s(v,Pa). \end{aligned}$$

Further, considering a cluster C, δ, and $\mathcal{A}$, for every $X \subseteq V_C$, we define $\hat{s}^{\mathcal{A}}(X)$ to be the best score possible for a DAG over $X \cup V^{in}_{C,\delta}$ that satisfies $\mathcal{A}$ (the scores of the vertices in V^{in} are not considered), and $\ell^{\mathcal{A}}(X)$ to be the last element of a topological ordering (restricted to X) of an optimal DAG over $X \cup V^{in}_{C,\delta}$ that satisfies $\mathcal{A}$ (given the constraints). Later, the subscript $\mathcal{A}$ is omitted.

We first introduce an algorithm described in Perrier et al. (2008) that calculates F_s and F_p using dynamic programming and then explain how we adapt the calculation of $\hat{s}$ and ℓ to satisfy the TDM and the NACS.

Algorithm 1: Calculate F_s and F_p (Perrier et al., 2008)
Input: Score function s and super-structure G
Output: Functions F_p and F_s

1. Set $F_s(v,\emptyset) = s(v,\emptyset)$ and $F_p = \emptyset$ for all $v \in V$.

2. For all $v \in V$ and all $X(\neq \emptyset) \subseteq \mathcal{N}(v) \setminus \{v\}$, compute

$$\begin{aligned} u^* &= \arg\min_{u \in X} F_s(v, X \setminus \{u\}), \\ F_s(v,X) &= \min\{s(v,X), F(v, X \setminus \{u^*\})\}, \\ F_p(v,X) &= \begin{cases} X & \text{if } s(v,X) \leq F_s(v, X \setminus \{u^*\}) \\ F_p(v, X \setminus u^*) & \text{otherwise} \end{cases}. \end{aligned}$$

Note that the in-vertices are considered differently in our algorithm; they either have no parents or fixed parents, and can only be parents of few vertices in V_C depending on $\mathcal{A}$. Thus, although the DAGs considered in the following algorithm are optimal on $X \cup V_{C,\delta}^{in}$, the score of $v \in V_{C,\delta}^{in}$ (that is fixed depending on δ) is not counted in $\hat{s}(X)$ since it is the same for every DAG irrespective of whether it is optimal or not.

From Theorem 1, the only orderings π such that $u \prec_\pi v$ for every $v \in V_{C,\delta}^{in}$ and $u \in \mathcal{A}(v)$ are to be considered. Therefore, for $w \in V_C$ and $v \in V_{C,\delta}^{in}$, v can be a parent of w if and only if $v \prec_\pi w$, implying $u \prec_\pi w$ for every $u \in \mathcal{A}(v)$. Therefore, we do not need to consider $v \in V_{C,\delta}^{in}$ in our ordering since we can infer whether v can be a parent of w by checking if it is ordered after all nodes in $\mathcal{A}(v)$. We define for every $X \subseteq V_C$ the associate set $PC(X) = X \cup \{v \in V_{C,\delta}^{in} \mid \mathcal{A}(v) \subseteq X\}$; in other words, for a given topological ordering π over X, the possible parents in $X \cup V_{C,\delta}^{in}$ of $w \in X$ are in $PC(P_\pi(w)) \cap \mathcal{N}(w)$. Using this result, we can present ACOS:

Algorithm 2: AncestrallyConstrainedOptimalSearch
Input: Cluster C, TDM δ, NACS $\mathcal{A}$, super-structure G, and functions F_p and F_s (previously computed)
Output: Optimal BN under $\mathcal{A}$, δ, G, and its score, $\hat{s}(V_C)$

1. Set $\hat{s}(\emptyset) = \infty$ and $\ell(\emptyset) = \emptyset$.
2. For all X $(\neq \emptyset) \subseteq V_C$, do:
   (a) Compute $\ell(X) = \arg\min_{v \in X}\{F_s(v, PC(X \setminus \{v\}) \cap \mathcal{N}(v)) + \hat{s}(X \setminus \{v\})\}$.
   (b) Define $\hat{s}(X)$ as the minimal score obtained during the previous step.
3. Construct N^*, an optimal $\mathcal{A}$-linear BN over V_C, using F_p and ℓ, and return N^* and its score $\hat{s}(V_C)$.

In Algorithm 2, the computation of F_s and F_p is carried out during preprocessing because it does not depend on δ and $\mathcal{A}$. Step 3 can be completed in linear time in n and is presented in Perrier et al. (2008). To prove the correctness of ACOS, we explain the computation of ℓ in step (a).

Theorem 2 *Given C, δ, and $\mathcal{A}$, ACOS constructs an optimal constrained BN over $V_C \cup V_{C,\delta}^{in}$.*

Proof First, we recursively show on the size of X that the computation of ℓ and $\hat{s}$ in Algorithm 2 respects their definition. Since the initialization in step 1 is correct, let us consider $X \neq \emptyset$ such that for $\forall Y \subset X$, $\hat{s}(Y)$ and $\ell(Y)$ are well defined. For any $v \in X$, we would like to find the score of the best DAG having v as a sink (i.e., v is the last element of a topological ordering over X). In that configuration, all nodes in $X \setminus \{v\}$ are predecessors of v, and therefore, they are potential

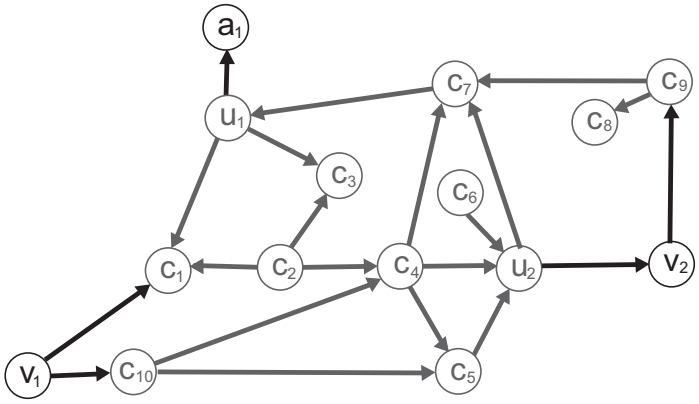


Figure 3: If this graph is the best $\mathcal{A}$-linear DAG with $\mathcal{A} = \{(v_1, u_1)\}$, since both v_1 and v_2 are not ancestors of u_1 and u_2, its strongest NACS is $\mathcal{B}$ with $\mathcal{B}(v_1) = \mathcal{B}(v_2) = \{u_1, u_2\}$. Therefore, this graph is optimal for all seven NACSs included between $\mathcal{A}$ and $\mathcal{B}$.

parents. Moreover, as stated previously, $w \in V_{C,\delta}^{in}$ such that $\mathcal{A}(w) \subseteq X \setminus \{v\}$ can also be a parent of v while satisfying ACs. Consequently, after adding the structural constraint G, the best score for v is $F_s(v, PC(X \setminus \{v\}) \cap \mathcal{N}(v))$. Finally, since v cannot be a parent of any nodes in $X \setminus \{v\}$, the best score over this set is $\hat{s}(X \setminus \{v\})$. Thus, step (a) of Algorithm 2 finds the best sink for X and correctly defines $\ell(X)$ and $\hat{s}(X)$. Finally, as explained in Perrier et al. (2008), we can rebuild an optimal ordering π^* over V_C by using ℓ and obtain an optimal DAG by assigning $\forall v \in V_C$, $Pa(v) = F_p(v, PC(P_{\pi^*}(v)) \cap \mathcal{N}(v))$. ■

3.2 Pruning

Following Theorem 1, we know that ACOS has to be computed only for all NACSs. Although the number of NACSs can be shown to be less than $O(|E^C|!)$ (because all NACSs can be generated through orderings of $V_{C,\delta}^{in} \cup V_{C,\delta}^{out}$), it is experimentally worse than exponential in the number of cluster edges. Fortunately, different NACSs frequently lead to the same optimal networks, and many NACS do not need to be considered. For the cluster C and TDM δ shown in Figure 1, Figure 3 shows an optimal BN N under NACS $\mathcal{A} = \{(v_1, u_1)\}$. Since both v_1 and v_2 are not ancestors of u_1 and u_2, its strongest NACS is $\mathcal{B} = \{(v_1, u_1), (v_1, u_2), (v_2, u_2), (v_2, u_2)\}$. Therefore, N is an optimal BN of seven NACSs between $\mathcal{A}$ and $\mathcal{B}$: $\{(v_1, u_1)\}$, $\{(v_1, u_1), (v_2, u_1)\}$, $\{(v_1, u_1), (v_1, u_2)\}$, $\{(v_1, u_1), (v_1, u_2), (v_2, u_1)\}$, $\{(v_1, u_1), (v_1, u_2), (v_2, u_2)\}$, $\{(v_1, u_1), (v_2, u_1), (v_2, u_2)\}$, $\{(v_1, u_1), (v_1, u_2), (v_2, u_2), (v_2, u_2)\}$. The next lemma formally describes this observation.

Lemma 1 *Let $\mathcal{A}$ be a NACS and $\mathcal{B}$, an ACS such that $\mathcal{B} \geq \mathcal{A}$ and that an optimal $\mathcal{A}$-linear DAG N^* is also $\mathcal{B}$-linear. Then, $\forall \mathcal{A}'$ such that $\mathcal{A} \leq \mathcal{A}' \leq \mathcal{B}$, N^* is also an optimal $\mathcal{A}'$-linear DAG.*

Proof Since $\mathcal{A}'$ is more restrictive than $\mathcal{A}$, an optimal $\mathcal{A}'$-linear DAG N'^* verifies that $s(N'^*) \geq s(N^*)$. However, since N^* is $\mathcal{B}$-linear, it is also $\mathcal{A}'$-linear; therefore, it is optimal and $s(N'^*) = s(N^*)$. ■

By browsing the space of NACS in an order verifying that $\forall i$ and j if $\mathcal{A}_i \leq \mathcal{A}_j$ then $i \leq j$, and by using the previous lemma as a pruning criterion, we can considerably reduce the number of NACSs to which ACOS is applied. For a given C and δ, we consider a score-and-network (SN) map $S_{C,\delta}$ as a list containing pairs of optimal scores and networks generated by every NACS, not to be pruned. We denote the set of $S_{C,\delta}$ for all TDM as S_C.

Algorithm 3: PruningConstrainedSearch
Input: Cluster C and TDM δ
Output: SN maps $S_{C,\delta}$

1. Initialize an empty set of NACSs U and an empty SN map $S_{C,\delta}$.
2. For every NACS $\mathcal{A}_i$ (ordered such that $j \leq k$ if $\mathcal{A}_j \leq \mathcal{A}_k$), do:
   (a) If $\mathcal{A}_i \in U$, $i++$ and restart step (a).
   (b) Otherwise, learn N^*, an optimal $\mathcal{A}_i$-linear DAG of score s^*, using Algorithm 2.
   (c) Let $\mathcal{B}$ be the ACS containing all ACs satisfied in N^*.
   (d) $\forall \mathcal{A}'$ such that $\mathcal{A}_i \leq \mathcal{A}' \leq \mathcal{B}$, add $\mathcal{A}'$ in U.
   (e) Add the pair (N^*, s^*) to $S_{C,\delta}$.

For enumerating ordered NACSs, see Appendix A. The following theorem shows the correctness of PruningConstrainedSearch.

Theorem 3 *For every NACS $\mathcal{A}$, there is an optimal DAG in $S_{C,\delta}$.*

Proof This is trivial since from Lemma 1, we have already found an optimal DAG N^* for the NACS $\mathcal{A}'$ that are pruned (added to U in step (d)). ■

3.3 Assembling Clusters

Next, we describe how the results of two clusters C_1 and C_2 are combined. The algorithm given below builds a set of SN maps S_C for the merged cluster C out of S_{C_1} and S_{C_2} (with $V_C = V_{C_1} \cup V_{C_2}$).

Algorithm 4: MergeCluster
Input: Clusters C_1 and C_2 and sets of SN maps S_{C_1} and S_{C_2}
Output: Merged cluster C and set of SN maps S_C

1. Define $C = (V_{C_1} \cup V_{C_2}, E_{C_1} \cup E_{C_2} \cup (E^{C_1} \cap E^{C_2}))$
2. For every TDM δ of E^C, do:
   (a) For every pair of TDM δ_1 and δ_2 of E^{C_1} and E^{C_2} that satisfy the following conditions:
   Condition i $\forall (e,d) \in \delta$, then $(e,d) \in \delta_i$ ($i = 1$ or 2),
   Condition ii $\forall e \in E^{C_1} \cap E^{C_2}$, $(e,d) \in \delta_1$ if and only if $(e,d) \in \delta_2$,
      i. For every pair of optimal networks and scores (N_1^*, s_1^*) and (N_2^*, s_2^*) of S_{C_1,δ_1} and S_{C_2,δ_2}, respectively, do:

A. Define $N^* = N_1^* \cup N_2^*$ and $s^* = s_1^* + s_2^*$
B. If there exists a directed cycle in N^*, restart step i with the next pair.
C. Let $\mathcal{A}$ be the ACS containing all ACs satisfied in N^*.
D. If there exists an optimal $\mathcal{A}$-linear network in $S_{C,\delta}$ that has a score smaller than s^*, restart step i with the next pair.
E. Add the pair (N^*, s^*) to $S_{C,\delta}$.
F. Remove every pair (N', s') of $S_{C,\delta}$ such that N' is $\mathcal{A}$-linear and $s' > s^*$.

The next theorem shows that $S_{C,\delta}$ contains an optimal BN and its score for every NACS on C.

Theorem 4 *If for every pair of TDMs δ_1 and δ_2 and for every NACS, S_{C_1,δ_1} and S_{C_2,δ_2} contain pairs of an optimal BN and its score, then $S_{C,\delta}$ constructed by Algorithm 4 contains a pair of an optimal BN and its score for every NACS.*

Proof First, we show that for every NACS $\mathcal{A}$ over C, we can build an optimal $\mathcal{A}$-linear BN by merging two optimal networks on C_1 and C_2 for some NACSs $\mathcal{A}_1$ and $\mathcal{A}_2$. To do so, for a given TDM δ, let us consider a NACS $\mathcal{A}$ for C, an optimal $\mathcal{A}$-linear BN N^* of score s^* and one of its topological orderings π^* defined over V_C (that is also in agreement with δ and $\mathcal{A}$). i is used instead of 1 and 2. We define π_i^* the ordering of the vertices in V_{C_i} derived from π^*. Further, we call δ_i the TDM of E^{C_i} derived from π_i^*; we have that δ_1 and δ_2 verify trivially conditions (i) and (ii) stated in step (a) of Algorithm 4. Finally, we define $\mathcal{A}_i$ as a NACS for C_i such that for $\forall v \in V_{C_i,\delta_i}^{in}$, $\mathcal{A}_i(v) = P_{\pi_i}(v) \cap V_{C_i,\delta_i}^{out}$. Given an optimal $\mathcal{A}_i$-linear network of S_{C_i,δ_i} N_i^* and its score s_i^*, let us consider the graph $N' = N_1^* \cup N_2^*$. This graph is acyclic since it is π^*-linear by construction. Further, its score $s' = s_1^* + s_2^*$ is minimal for $\mathcal{A}$-linear; otherwise, one of the N_i^* graphs would not be optimal. Therefore, although N' might be different from N^*, they both have the same score. Therefore, since Algorithm 4 considers every coherent pair δ_1 and δ_2 (that verify conditions (i) and (ii)) and every pair of NACS, $S_{C,\delta}$ is correctly constructed. ■

Given the previous algorithm, we simply need to merge all the clusters to obtain an optimal DAG N_G^* and its score, as explained in the following algorithm.

Algorithm 5: ClusterAssembler
Input: Set of all clusters $\mathcal{C}$
Output: Optimal BN N_G^* and its score s^*

1. $\forall C \in \mathcal{C}$, compute S_C using Algorithm 3 for every δ.
2. While $|\mathcal{C}| > 1$, do:
   (a) Select a pair of clusters C_1 and C_2 such that $|(E^{C_1} \cup E^{C_2}) \setminus (E^{C_1} \cap E^{C_2})|$ is minimal.
   (b) Compute the cluster C and S_C by merging C_1 and C_2 using Algorithm 4.
   (c) Remove C_1 and C_2 from $\mathcal{C}$, and add C to $\mathcal{C}$.
3. Since G is the last element in $\mathcal{C}$, return N_G^* and s^*, the sole pair stored in $S_{G,\emptyset}$.

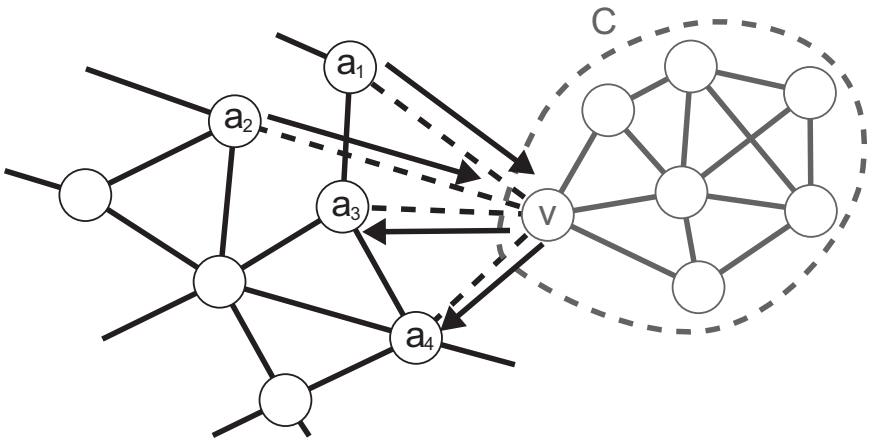


Figure 4: An example of super-structure shrinkage. A block $C = (V_C, E_C)$ (gray) can be separated from the rest of the super-structure by the removal of a cut-vertex $v \in V_C$. Arrows indicate the unique TDM δ_X for $X = \{a_1, a_2\}$.

The correctness of Algorithm 5 is directly derived from Theorem 4. In (a), although we do not prove that the complexity is minimal by merging the clusters that imply less cluster edges for the merged cluster at each step, we decided to use this heuristic. This is because the complexity depends on the number of cluster edges in Algorithm 4; therefore, it is faster to always manipulate a cluster with a small number of cluster edges.

3.4 Preprocessings

In this section, we describe a preprocessing that can drastically reduce the time complexity of our method and the heuristic we used to select the edges in E^-.

3.4.1 SUPER-STRUCTURE SHRINKAGE

First, we introduce the notions of a block and a block tree of an undirected graph. Their formal definitions are described in Diestel (2005). A block is a biconnected subgraph of the undirected graph, and vertices in the intersection of blocks are called cut-vertices, that is, the removal of cut-vertices separates blocks. A block tree is a tree comprised of blocks as vertices and cut-vertices as edges. We here show leaves of a block tree of the super-structure can be removed if their size is small. Let us consider the case shown in Figure 4 where a block $C = (V_C, E_C)$ of the super-structure G can be separated by withdrawing a cut-vertex $v \in V_C$ and that C is of a moderate size ($|V_C| < 30$). Then, all edges (v, w), where $w \notin V_C$, are considered as cluster edges; because only v is connected to cluster edges, no cycle can be created while merging an optimal DAG N_C^* over V_C with another cluster; otherwise, it would imply that there is a cycle in N_C^*. Therefore, there is no need of AC and we propose to process this case in a different manner. For every TDM δ, we learn an optimal DAG N_δ^* and its score s_δ^* over V_C. Then, we replace C by a single vertex $\hat{v}$ in G to obtain the condensed super-structure $\hat{G}$. For every candidate parent set X of $\hat{v}$ in $\hat{G}$ (i.e., $\forall X \subseteq \mathcal{N}(v) \setminus V_C$), there exists the unique TDM δ_X corresponding to C. For example, if a candidate parent set X is set to $\{a_1, a_2\}$ in

Figure 4, unique TDM δ_X for cluster edges (v,a_1), (v,a_2), (v,a_3), and (v,a_4) is indicated by arrows. Using this observation, we redefine F_s on $\hat{v}$ to be $\hat{F}_s(\hat{v},X) = s^*_{\delta_X}$ and F_p to be $\hat{F}_p(\hat{v},X) = N^*_{\delta_X}$; here, $\hat{F}_p$ is used not only to store the optimal parent set of v in $X \cup (V_C \cap \mathcal{N}(v))$ but also to save the optimal network over C. We can repeat this technique to shrink every small subgraph separated by the removal of a single vertex in G during preprocessing. This can lead to a drastic reduction of complexity in some real cases, as discussed later.

3.4.2 PARTITIONING THE SUPER-STRUCTURE INTO CLUSTERS

To apply our algorithm, we need to select a set of edges E^- that separates the super-structure into small strongly connected subgraphs (clusters) having balanced numbers of vertices while minimizing the number of cluster edges for each cluster. Such a problem is called graph partitioning. In our case, we employed an algorithm based on edge betweenness centrality that works efficiently for practical networks (Newman et al., 2004).

3.5 Resulting Algorithm

We summarize all results presented thus far in the following algorithm that learns a super-structure constrained optimal network and its score.

Algorithm 6: EdgeConstrainedOptimalSearch (ECOS)
Input: Super-structure $G = (V,E)$ and data D
Output: Optimal constrained BN N^*_G and its score s^*

1. $\forall v \in V$ and $\forall X \subseteq \mathcal{N}(v)$ compute $F_s(v,X)$ and $F_p(v,X)$.
2. Shrink every block possible in G to obtain a shrunk super-structure $\hat{G}$ and the functions $\hat{F}_s$ and $\hat{F}_p$.
3. Select E^- using the graph partitioning algorithm and obtain the set of all clusters $\mathcal{C}$.
4. $\forall C \in \mathcal{C}$ and $\forall \delta$; apply Algorithm 3 and obtain the set of SN maps S_C.
5. Merge all clusters using Algorithm 5 to obtain $\hat{N}^*_G$ and its score s^*.
6. Expand the subgraphs shrunk during step 2 to obtain N^*_G.

Note that after the expansion of shrunk subgraphs, s^* does not change as the scores for these subgraphs are packed in $\hat{F}_s$.

3.6 Complexity

In this last section, although it is hardly feasible to derive the complexity of Algorithm 6 in a general case because it strongly depends on the topology of the super-structure used, we propose an upper bound of the complexity depending on a few characteristics of G. Subsequently, we describe some practical generic structures to which ECOS can or cannot be profitably applied. We then present an empirical evaluation of the algorithm over randomly generated networks and real networks, with promising results being found for the latter.

Considering step 1 of ECOS, after defining the maximal degree of G as $m = \max_{v \in V} |\mathcal{N}(v)|$, we obtain that the number of scores calculated is upper bounded by $O(n2^m)$. This is actually the main

reason for using a structural constraint because the functions F_p and F_s can be computed in a linear time for bounded degree structures ($m < 30$). Actually, this is feasible even for large m if an additional constraint on the number of parents c is added, the complexity becoming $O(nm^c)$.

Next, if n_1 is the size of the largest cluster that has been shrunk (this is a tunable parameter), and considering that at maximum, the number of cluster edges of a shrunk block m' is $m-1$ and the number of TDM is $2^{m'}$, given the exponential complexity of calculating $\hat{s}$ and ℓ, we find that the complexity of step 2 is bounded by $O(b2^{m-1}2^{n_1})$, where b is the number of blocks shrunk. In other words, if n_1 is tuned suitably, step 2 has negligible complexity as compared to the subsequent steps. Similarly, step 3 is negligible since its complexity is only polynomial in n ($O(mn^3)$).

However, step 4 requires a more detailed analysis. Given E^-, we define $n_2 = \max_{C \in \mathcal{C}} |V_C|$, the size of the largest cluster, and $k = \max_{C \in \mathcal{C}} |E^C|$, the largest number of cluster edges. The complexity of ACOS is trivially bounded by $O(2^{n_2})$. Further, because the number of NACS is less than the number of permutations over $V^{in} \cup V^{out}$ for a given TDM, we have that for every cluster, ACOS is applied at maximum $k!2^k$ times. We derive an upper bound complexity for step 4 as $O(q2^{n_2}k!2^k)$, where q is the number of obtained clusters. Note, however, that the factorial term experimentally appears to be largely overestimated and that ACOS may actually be computed only $O(\beta^k)$ times for some $\beta > 2$.

Finally, at worst, step 5 involves trying every pair of entries in two SN map sets; with the maximum size of cluster edges of merged clusters K, the complexity might theoretically be as bad as $O(q(K!2^K)^2)$. However, in practice, because a major part of NACS was pruned in step 4, many pairs are pruned in step 5, and because all superfluous values of the SN maps are eliminated in Algorithm 4, its complexity is closer to $O(q\beta^k)$.

Following those rough upper bounds, we can derive some generic super-structures that are feasible for any number of vertices while not being naïve. For example, considering step 2, any super-structure whose block tree contains only small blocks (less than 30 vertices) is feasible. Otherwise, we can consider all the networks that can be generated by the following method:

- Generate an undirected graph G_0 of low maximal degree ($m < 10$).
- Replace every vertex v_i by a small graph C_i (up to 20 or slightly more) and randomly connect all edges connected to v_i in G_0 to vertices in C_i.

If ECOS can select all edges between clusters for such networks while defining E^-, the search should finish in reasonable time even for larger networks (up to several hundreds of vertices). Conversely, if a super-structure contains a large clique (containing more than 30 vertices), ECOS cannot finish as other optimal searches. To conclude, our algorithm may be a decisive breakthrough in some real cases where neither optimal searches nor COS can be applied because of a large number of vertices or a high average degree.

4. Experimental Evaluation

We conduct two types of numerical experiments for evaluating the performance of ECOS. In the former experiment, the practical time complexity of ECOS is estimated by the comparison with COS, using random networks of various sizes. In order to show the performance on practically structured networks, we then apply ECOS to the synthetically generated large scale network from Alarm, Insurance, and Child networks in the latter experiment. The performance of ECOS is compared with

$n \setminus \tilde{m}$	2	2.5	3	3.5
10	100	100	100	100
20	100	100	100	100
30	100	100	100	100
50	100	100	95	40
75	100	100	61	0
100	100	100	4	0

Table 1: Number of times the computation finished within one day for a random graph of n vertices and average degree $\tilde{m}$.

Algorithm	$\tilde{m}$	2	2.5	3	3.5
ECOS	$\delta_{\tilde{m}}$	1.06	1.08	1.15	1.25
	$n_{\max}(\tilde{m})$	355	273	151	93
COS	$\delta_{\tilde{m}}$	1.50	1.63	1.74	1.81
	$n_{\max}(\tilde{m})$	51	43	38	35

Table 2: Values of coefficients $\delta_{\tilde{m}}$ and $n_{\max}(\tilde{m})$ of ECOS and COS for average degree of super-structure $\tilde{m}$. $\delta_{\tilde{m}}$ is the estimated base of exponential time complexity and $n_{\max}(\tilde{m})$ is the feasible size of the super-structure for computation.

those of MMHC and greedy hill-climbing. All the computations in the following experiments were performed on machines having 3.0 GHz Intel Xeon processors with a Core microarchitecture (only one core was used for each experiment).

4.1 Benefit in Terms of Complexity

In the first series of experiments, we aimed to evaluate the average complexity of ECOS depending on n and $\tilde{m}$, the average degree of G. Since the feasibility of ECOS depends on the pruning of the search space, the theoretical derivation of the practical time complexity is difficult. Here, we hypothesize that the average complexity is in the form of $O(\delta_{\tilde{m}}^n)$, and then estimate $\delta_{\tilde{m}}$. Let $t_{\tilde{m},n}$ be the time required for a network of n vertices and average degree $\tilde{m}$. Under our assumption of time complexity, $t_{\tilde{m},n}$ is given by

$$t_{\tilde{m},n} = const \cdot \delta_{\tilde{m}}^n, \tag{2}$$

where *const* indicates the dependency of the implementation and machine specifications. From Equation (2), we have that $\delta_{\tilde{m}} = \exp(\frac{1}{n}(\log t_{\tilde{m},n} - \log const))$. Because $\frac{\log const}{n}$ can be ignored for large n, $\delta_{\tilde{m}}$ can be estimated by $\exp(\frac{\log t_{\tilde{m},m}}{n})$. For $\forall \tilde{m} \in \{2, 2.5, 3, 3.5\}$ and $\forall n \in \{10, 20, 30, 50, 75, 100\}$, we generate 100 random networks and we apply ECOS using 1,000 artificially generated samples in each case. We compute the average time $\tilde{t}_{\tilde{m},n}$ that is required and calculate $\delta_{\tilde{m},n} = \exp(\frac{\log(\tilde{t}_{\tilde{m},n})}{n})$. If our hypothesis is correct, $\delta_{\tilde{m},n}$ should converge to $\delta_{\tilde{m}}$ while n increases. However, to keep the computation manageable, we stop the calculation if it requires more

Network	No. of vertices	No. of edges
Alarm1	37	46
Alarm3	111	149
Alarm5	185	253
Alarm10	370	498
Insurance1	27	52
Insurance3	81	163
Insurance5	135	268
Insurance10	270	536
Child1	20	25
Child3	60	79
Child5	100	126
Child10	200	257

Table 3: Characteristics of the real networks considered in the computational experiment.

than one day. Hence, we probably underestimate $\delta_{\tilde{m}}$ slightly; nevertheless, here, we attempt to derive the exponential nature of the average complexity and not the real value of the constants. Further, the following results are sufficient to obtain a rough estimate. The number of times we finished the calculation for each pair of parameters is listed in Table 1. Due to the small ratio of finished experiments for $\tilde{m} = 3$ and 3.5, we selected the values $\delta_{3,75}$, $\delta_{3.5,50}$ for δ_3 and $\delta_{3.5}$, respectively. Further, for every average degree, we evaluated the maximal number of vertices $n_{\max}(\tilde{m})$ feasible from the value of $\delta_{\tilde{m}}$ calculated as proposed in Perrier et al. (2008).

Table 2 lists the values of $\delta_{\tilde{m}}$ and $n_{\max}(\tilde{m})$ for ECOS and COS. We should note that $n_{\max}(\tilde{m})$ of ECOS for $\tilde{m} = 3$ and 3.5 is overestimated since in this case, $\delta_{\tilde{m}}$ is underestimated because only the computations that finished were used to calculate it. In practice, $n_{\max}(\tilde{m})$ of ECOS for $\tilde{m} = 3$ and 3.5 are respectively around 75 and 50 from the results listed in Table 1; therefore, we can clearly see the practical advantage of ECOS over COS, and the improvement in terms of feasibility achieved by our method. In addition, we should emphasize that random networks penalize the results of ECOS because they do not have a logical partitioning. In real cases, we can hope that super-structures can be efficiently partitioned, enabling better performances for ECOS.

4.2 Case Study

We considered four networks whose characteristics are summarized in Table 3; those networks were generated from Alarm, Insurance, and Child networks by the tiling algorithm (Tsamardinos et al., 2006). We compare the performances of ECOS to those of the following state-of-the-art greedy algorithms: MMHC and greedy hill-climbing (HC), both using a TABU search extension; the TABU list size was set to 100 as in Tsamardinos et al. (2006). COS is not included in this evaluation because COS and ECOS are learning the same networks (or at least networks having the same score, that is, the best one possible given the structural constraint). Further, COS cannot be applied to such large networks when using such high values for α (cf. Perrier et al. 2008). The super-structures were generated in two different ways: the true skeleton was given or a skeleton was inferred by using MMPC (Tsamardinos et al., 2006) implemented in the Causal Explorer System

Model	Sample Size	α	Coverage	Average Degree
Alarm5	1000	true	1.00 ±0.00	2.74±0.00
		0.01	0.77 ±0.00	2.29±0.00
		0.02	0.78 ±0.00	2.40±0.00
		0.05	0.80 ±0.00	2.67±0.00
	10000	true	1.00 ±0.00	2.74±0.00
		0.01	0.94 ±0.00	2.67±0.00
		0.02	0.94 ±0.00	2.77±0.00
		0.05	0.95 ±0.00	3.01±0.00
Alarm10	1000	true	1.00 ±0.00	2.69±0.00
		0.01	0.78 ±0.00	2.34±0.00
		0.02	0.80 ±0.00	2.49±0.00
		0.05	0.81 ±0.00	2.87±0.00
	10000	true	1.00 ±0.00	2.69±0.00
		0.01	0.95 ±0.00	2.70±0.00
		0.02	0.95 ±0.00	2.84±0.00
		0.05	0.96 ±0.00	3.14±0.00
Insurance5	1000	true	1.00 ±0.00	3.97±0.00
		0.01	0.64 ±0.00	2.97±0.00
		0.02	0.66 ±0.00	3.15±0.01
		0.05	0.68 ±0.00	3.52±0.01
	10000	true	1.00 ±0.00	3.97±0.00
		0.01	0.80 ±0.00	3.43±0.00
		0.02	0.81 ±0.00	3.53±0.00
		0.05	0.83 ±0.00	3.73±0.01
Insurance10	1000	true	1.00 ±0.00	3.97±0.00
		0.01	0.64 ±0.00	3.00±0.00
		0.02	0.66 ±0.00	3.22±0.00
		0.05	0.67 ±0.00	3.63±0.01
	10000	true	1.00 ±0.00	3.97±0.00
		0.01	0.80 ±0.00	3.46±0.00
		0.02	0.81 ±0.00	3.57±0.00
		0.05	0.82 ±0.00	3.81±0.01
Child5	1000	true	1.00 ±0.00	2.52±0.00
		0.01	0.84 ±0.00	2.32±0.00
		0.02	0.86 ±0.00	2.39±0.00
		0.05	0.88 ±0.00	2.50±0.00
	10000	true	1.00 ±0.00	2.52±0.00
		0.01	1.00 ±0.00	2.53±0.00
		0.02	1.00 ±0.00	2.55±0.00
		0.05	1.00 ±0.00	2.57±0.00
Child10	1000	true	1.00 ±0.00	2.57±0.00
		0.01	0.82 ±0.00	2.3±0.00
		0.02	0.84 ±0.00	2.38±0.00
		0.05	0.87 ±0.00	2.510±0.00
	10000	true	1.00 ±0.00	2.57±0.00
		0.01	0.99 ±0.00	2.58±0.00
		0.02	0.99 ±0.00	2.61±0.00
		0.05	0.99 ±0.00	2.65±0.00

Table 4: Coverage and average degree of super-structures for each experimental condition (mean ± standard deviation).

(Aliferis et al., 2003) with a significance level $\alpha \in \{0.01, 0.02, 0.05\}$. Ten data sets of 500, 1,000, and 10,000 samples were synthetically generated from each BN considered. Here, we evaluate and discuss the cases of Alarm5, Alarm10, Insurance5, Insurance10, Child5, and Child10 with 1,000 and 10,000 samples. The results of all the cases including the remaining ones are summarized in the supplemental material. To help evaluate the quality of the super-structures learnt by MMPC, we

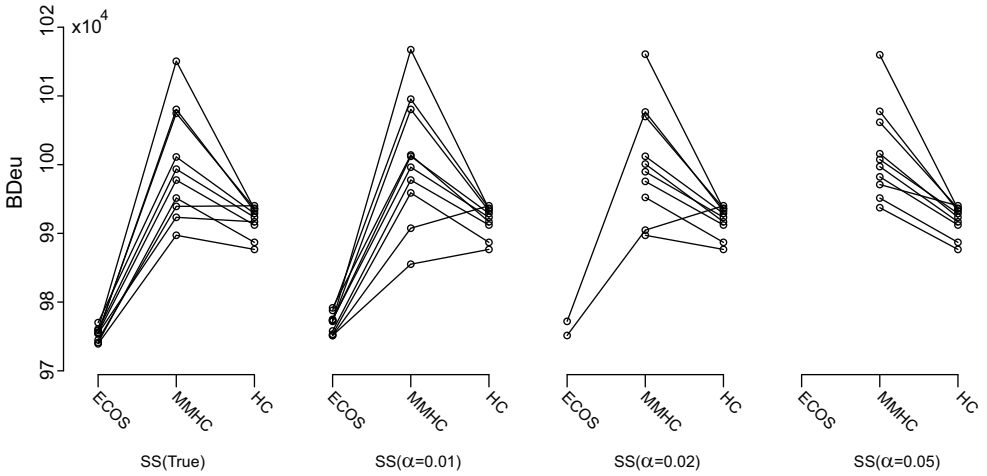


Figure 5: BDeu scores for ten data sets of Alarm10 with 10,000 samples.

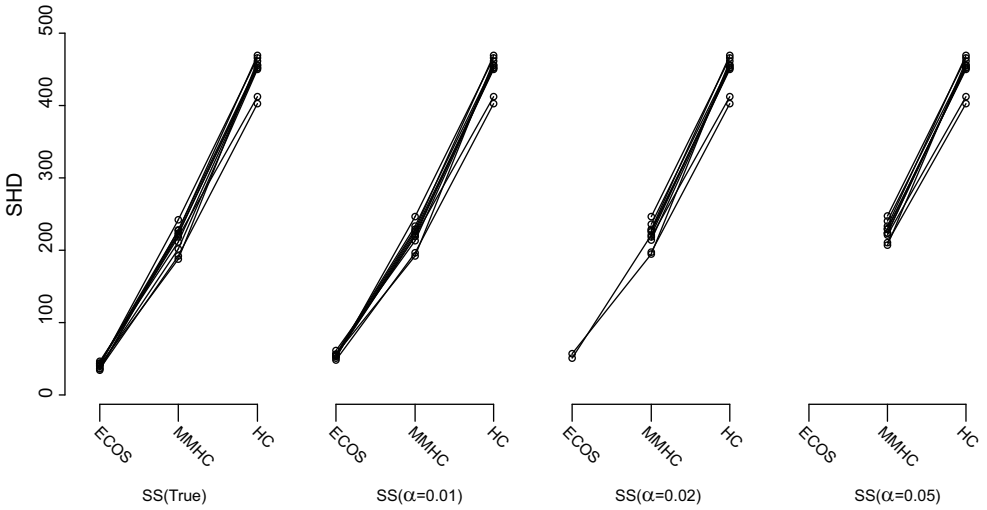


Figure 6: Values of SHD for ten data sets of Alarm10 with 10,000 samples.

summarized the ratio of true edges learnt (this ratio is called coverage) and the average degree of the super-structures in Table 4.

For every experimental condition, the algorithms are compared both in terms of score (we used the negative BDeu score, that is, smaller values are better) and structural hamming distance (SHD, Tsamardinos et al. 2006) that counts the number of differences in the completed partially DAG (CPDAG, Chikdering 2002) of the true network and the learnt one.

Figures 5 and 6 respectively show BDeu and SHD scores of ECOS, MMHC, and HC for ten data sets of Alarm10 with 10,000 samples given the true skeleton and super-structures inferred by MMPC with $\alpha = 0.01$, 0.02, and 0.05. In order to clarify the relation between the results of ECOS, MMHC, and HC for each data set, results from the same data set are linked up. In addition,

the scaling of the plots for the four conditions are the same, and therefore, the results of HC are the same for the four structural constraints considered. For the true skeleton and super-structures obtained with $\alpha = 0.01$, all ten computations of ECOS finished within two days and ECOS gives the best scores in all the data sets both in terms of BDeu and SHD. In terms of BDeu, HC usually performs better than MMHC, whereas it is the opposite in terms of SHD.

For $\alpha = 0.02$, only two computations of ECOS finished within two days; nonetheless, the computed scores are better than those of MMHC and HC both in terms of BDeu and SHD. For the super-structures of $\alpha = 0.05$, no computations of ECOS finished within two days. However, every time ECOS finished, it gave better results than both HC and MMHC. With regard to the structural constraint, the best results were obtained when the true skeleton was known. Further, for $\alpha = 0.01$, MMHC is better than HC for two data sets; but for $\alpha = 0.02$, it is better for one data set; and $\alpha = 0.05$, it is better for no data sets. Because the coverage of super-structures with $\alpha = 0.01$ for Alarm10 with 10,000 samples is already maximal, as shown in Table 4, super-structures for higher α contain more false-positive edges, which worsens the results of MMHC. We observe the same results in terms of SHD as well.

BDeu and SHD scores for all the experiments are summarized in Tables 5 and 6; for each experimental setup, the best result is in bold and the best result without the knowledge of the true skeleton is underlined. The numbers in parentheses for ECOS represent the number of times ECOS could finish within ten data sets in two days. The cases in which no computation finished are indicated by "none". Note that all computations of MMHC and HC finished in two days. BDeu and SHD scores of the finished computations are averaged and rounded off to the nearest integer.

While HC outperforms MMHC in BDeu, MMHC outperforms HC in SHD, which agrees with the results in Tsamardinos et al. (2006). A comparison of the results of MMHC and HC suggests that a structural constraint helps to find networks with smaller SHD. However, this should not mislead us into thinking that minimizing a score function is not a good method to learn a graph having a small SHD. In fact, ECOS returns considerably better results than both MMHC and HC in terms of SHD, strongly illustrating the validity of score-based approaches and also the use of a structural constraint.

One could argue that it is possible to increase the quality of the results returned by MMHC by using a larger α. However, as we can see in Table 6, although the score improves with higher α, it is not always the case with the SHD. This is expected because MMHC converges to HC with an increasing α; hence, it is essential in the greedy case to properly select α. On the other hand, ECOS converges to OS for increasing significance levels. Although in rare cases, SHD slightly worsens with an increasing α, we should generally use as large a significance level as possible when applying ECOS, while ensuring that the algorithm finishes.

The average running time for the experiments in seconds (rounded off to the nearest integer) are summarized in Table 7. All the algorithms except for MMPC are implemented in Java. For ECOS and MMHC, the running time of MMPC is also included. Among the experiments performed till the end, ECOS requires the maximum computational time (around 34 hours) in Insurance10 with 10,000 samples and the true skeleton. The maximum memory space (10 GB) was required by ECOS during Insurance10 with 1,000 samples and $\alpha = 0.01$. Fortunately, for all experimental setups, ECOS outperformed the other two methods both in BDeu and SHD. Although it stopped for some $\alpha > 0.01$, our algorithm is still better with $\alpha = 0.01$ than MMHC with $\alpha = 0.05$.

Model	Sample Size	α	Algorithm		
			ECOS	MMHC	HC
Alarm5	1000	true	**61753 ± 376(10)**	63294 ± 477	62380 ± 420
		0.01	**62111 ± 347(10)**	63128 ± 428	62380 ± 420
		0.02	**62015 ± 364(10)**	63041 ± 489	62380 ± 420
		0.05	**61920 ± 387(10)**	62959 ± 458	62380 ± 420
	10000	true	**487734 ± 1109(10)**	494611 ± 1986	496108 ± 1104
		0.01	**488807 ± 1312(10)**	495708 ± 2212	496108 ± 1104
		0.02	**488418 ± 1137(10)**	495373 ± 2269	496108 ± 1104
		0.05	**488182 ± 1132(10)**	495359 ± 2638	496108 ± 1104
Alarm10	1000	true	**123725 ± 481(10)**	126863 ± 857	125156 ± 585
		0.01	**124406 ± 706(10)**	126673 ± 833	125156 ± 585
		0.02	**124243 ± 750(10)**	126456 ± 869	125156 ± 585
		0.05	none	126375 ± 831	**125156 ± 585**
	10000	true	**975343 ± 943(10)**	999973 ± 8022	991888 ± 2144
		0.01	**976860 ± 1464(10)**	1000644 ± 9143	991888 ± 2144
		0.02	**976150 ± 1404(2)**	1000388 ± 8109	991888 ± 2144
		0.05	none	1001610 ± 6726	**991888 ± 2144**
Insurance5	1000	true	**81148 ± 341(10)**	81551 ± 427	81955 ± 407
		0.01	**81529 ± 351(10)**	81973 ± 458	81955 ± 407
		0.02	**81449 ± 358(10)**	81911 ± 458	81955 ± 407
		0.05	**81376 ± 338(10)**	81862 ± 433	81955 ± 407
	10000	true	**677374 ± 864(10)**	684990 ± 3162	681516 ± 1892
		0.01	681584 ± 1707(10)	688579 ± 3521	**681516 ± 1892**
		0.02	**680666 ± 1418(10)**	687627 ± 2945	681516 ± 1892
		0.05	**680201 ± 1452(8)**	686908 ± 3196	681516 ± 1892
Insurance10	1000	true	**162311 ± 453(10)**	164039 ± 455	164034 ± 571
		0.01	**162949 ± 517(10)**	164557 ± 711	164034 ± 571
		0.02	**162890 ± 505(10)**	164541 ± 712	164034 ± 571
		0.05	**162309 ± 46(2)**	164541 ± 759	164034 ± 571
	10000	true	**1354655 ± 768(10)**	1371312 ± 3374	1364486 ± 2957
		0.01	**1361266 ± 1222(10)**	1376795 ± 2990	1364486 ± 2957
		0.02	**1360571 ± 1040(10)**	1376111 ± 2542	1364486 ± 2957
		0.05	**1360627 ± 955(2)**	1375645 ± 3224	1364486 ± 2957
Child5	1000	true	**71622 ± 324(10)**	72057 ± 344	72335 ± 396
		0.01	**71651 ± 333(10)**	72096 ± 368	72335 ± 396
		0.02	**71648 ± 330(10)**	72114 ± 376	72335 ± 396
		0.05	**71644 ± 329(10)**	72059 ± 385	72335 ± 396
	10000	true	**637783 ± 321(10)**	640908 ± 941	644218 ± 2614
		0.01	**637783 ± 321(10)**	640908 ± 941	644218 ± 2614
		0.02	**637783 ± 321(10)**	640908 ± 941	644218 ± 2614
		0.05	**637783 ± 321(10)**	640908 ± 941	644218 ± 2614
Child10	1000	true	**142541 ± 250(10)**	143847 ± 409	143905 ± 398
		0.01	**142604 ± 256(10)**	143902 ± 415	143905 ± 398
		0.02	**142590 ± 252(10)**	143885 ± 407	143905 ± 398
		0.05	**142582 ± 255(10)**	143896 ± 391	143905 ± 398
	10000	true	**1271035 ± 684(10)**	1277877 ± 2824	1283630 ± 3796
		0.01	**1271089 ± 690(10)**	1278050 ± 2987	1283630 ± 3796
		0.02	**1271072 ± 683(10)**	1277878 ± 2823	1283630 ± 3796
		0.05	**1271054 ± 685(10)**	1277931 ± 2863	1283630 ± 3796

Table 5: Comparison of ECOS, MMHC, and HC in terms of BDeu score (mean ± standard deviation; smaller value is better). "none" refers to the case in which no computations finished. The best score in each case is in bold font; the best score for each data sample without the knowledge of the true skeleton is underlined. The numbers in parentheses for ECOS represent the number of times ECOS could finish within ten data sets in two days.

Model	Sample Size	α	Algorithm		
			ECOS	MMHC	HC
Alarm5	1000	true	**125 ± 10(10)**	163 ± 6	215 ± 8
		0.01	**164 ± 6(10)**	177 ± 4	215 ± 8
		0.02	**164 ± 5(10)**	177 ± 4	215 ± 8
		0.05	**161 ± 5(10)**	176 ± 5	215 ± 8
	10000	true	**21 ± 2(10)**	96 ± 15	231 ± 12
		0.01	**31 ± 2(10)**	100 ± 10	231 ± 12
		0.02	**30 ± 2(10)**	101 ± 10	231 ± 12
		0.05	**30 ± 2(10)**	102 ± 9	231 ± 12
Alarm10	1000	true	**248 ± 9(10)**	321 ± 8	421 ± 15
		0.01	**317 ± 8(10)**	355 ± 13	421 ± 15
		0.02	**313 ± 9(10)**	353 ± 10	421 ± 15
		0.05	none	**354 ± 11**	421 ± 15
	10000	true	**40 ± 3(10)**	215 ± 17	447 ± 22
		0.01	**54 ± 3(10)**	219 ± 16	447 ± 22
		0.02	**54 ± 4(2)**	220 ± 15	447 ± 22
		0.05	none	**226 ± 12**	447 ± 22
Insurance5	1000	true	**196 ± 8(10)**	205 ± 9	246 ± 8
		0.01	**207 ± 7(10)**	217 ± 5	246 ± 8
		0.02	**206 ± 7(10)**	217 ± 4	246 ± 8
		0.05	**206 ± 8(10)**	217 ± 4	246 ± 8
	10000	true	**88 ± 1(10)**	137 ± 9	184 ± 11
		0.01	**99 ± 4(10)**	146 ± 11	184 ± 11
		0.02	**99 ± 6(10)**	146 ± 9	184 ± 11
		0.05	**100 ± 6(8)**	146 ± 11	184 ± 11
Insurance10	1000	true	**378 ± 21(10)**	424 ± 11	502 ± 10
		0.01	**398 ± 21(10)**	441 ± 11	502 ± 10
		0.02	**399 ± 22(10)**	441 ± 11	502 ± 10
		0.05	**376 ± 24(2)**	444 ± 9	502 ± 10
	10000	true	**174 ± 5(10)**	280 ± 22	381 ± 23
		0.01	**198 ± 9(10)**	296 ± 21	381 ± 23
		0.02	**198 ± 12(10)**	295 ± 20	381 ± 23
		0.05	**197 ± 17(2)**	295 ± 22	381 ± 23
Child5	1000	true	**60 ± 7(10)**	70 ± 7	82 ± 8
		0.01	**71 ± 6(10)**	80 ± 5	82 ± 8
		0.02	**70 ± 6(10)**	79 ± 5	82 ± 8
		0.05	**70 ± 6(10)**	78 ± 4	82 ± 8
	10000	true	**1 ± 0(10)**	31 ± 9	43 ± 10
		0.01	**1 ± 0(10)**	31 ± 9	43 ± 10
		0.02	**1 ± 0(10)**	31 ± 9	43 ± 10
		0.05	**1 ± 0(10)**	31 ± 9	43 ± 10
Child10	1000	true	**145 ± 9(10)**	171 ± 6	182 ± 5
		0.01	**167 ± 8(10)**	188 ± 5	182 ± 5
		0.02	**165 ± 8(10)**	186 ± 6	182 ± 5
		0.05	**163 ± 8(10)**	184 ± 6	182 ± 5
	10000	true	**8 ± 0(10)**	88 ± 16	128 ± 17
		0.01	**10 ± 3(10)**	87 ± 18	128 ± 17
		0.02	**9 ± 3(10)**	87 ± 17	128 ± 17
		0.05	**8 ± 0(10)**	87 ± 18	128 ± 17

Table 6: Comparison of ECOS, MMHC, and HC in terms of SHD (mean ± standard deviation; smaller value is better). "none" refers to the case in which no computations finished. The best score in each case is in bold font; the best score for each data sample without the knowledge of the true skeleton is underlined. The numbers in parentheses for ECOS represent the number of times ECOS could finish within ten data sets in two days.

Model	Sample Size	α	Algorithm		
			ECOS	MMHC	HC
Alarm5	1000	true	3231 ±1184 (10)	5 ±0	280 ±31
		0.01	532 ±441 (10)	36 ±0	280 ±31
		0.02	7617 ±22514 (10)	81 ±0	280 ±31
		0.05	8830 ±13741 (10)	100 ±0	280 ±31
	10000	true	19020 ±4363 (10)	10 ±1	517 ±47
		0.01	22921 ±43349 (10)	116 ±1	517 ±47
		0.02	37424 ±48662 (10)	74 ±1	517 ±47
		0.05	92718 ±117691 (10)	91 ±1	517 ±47
Alarm10	1000	true	11653 ±584 (10)	24 ±3	3620 ±339
		0.01	1920 ±1330 (10)	63 ±1	3620 ±339
		0.02	51627 ±91273 (10)	99 ±1	3620 ±339
		0.05	none (0)	61 ±1	3620 ±339
	10000	true	22914 ±1088 (10)	30 ±2	6009 ±1422
		0.01	5800 ±4545 (10)	138 ±3	6009 ±1422
		0.02	122665 ±155955 (2)	131 ±5	6009 ±1422
		0.05	none (0)	112 ±3	6009 ±1422
Insurance5	1000	true	7474 ±452 (10)	5 ±0	84 ±28
		0.01	2117 ±1778 (10)	95 ±0	84 ±28
		0.02	2841 ±3421 (10)	41 ±0	84 ±28
		0.05	47408 ±76941 (10)	88 ±0	84 ±28
	10000	true	9807 ±426 (10)	20 ±4	286 ±99
		0.01	8281 ±14925 (10)	22 ±1	286 ±99
		0.02	38154 ±43336 (10)	70 ±3	286 ±99
		0.05	67546 ±80972 (8)	93 ±2	286 ±99
Insurance10	1000	true	30976 ±4846 (10)	15 ±2	832 ±72
		0.01	5679 ±8074 (10)	77 ±1	832 ±72
		0.02	22177 ±19628 (10)	18 ±1	832 ±72
		0.05	88391 ±113694 (2)	94 ±2	832 ±72
	10000	true	123589 ±6232 (10)	48 ±12	2479 ±570
		0.01	63683 ±66898 (10)	64 ±6	2479 ±570
		0.02	85244 ±101769 (10)	82 ±1	2479 ±570
		0.05	14093 ±12510 (2)	103 ±7	2479 ±570
Child5	1000	true	4 ±0 (10)	3 ±0	71 ±14
		0.01	67 ±0 (10)	68 ±0	71 ±14
		0.02	8 ±0 (10)	8 ±0	71 ±14
		0.05	110 ±1 (10)	109 ±0	71 ±14
	10000	true	9 ±0 (10)	8 ±0	217 ±16
		0.01	90 ±0 (10)	89 ±0	217 ±16
		0.02	51 ±1 (10)	50 ±0	217 ±16
		0.05	25 ±1 (10)	23 ±0	217 ±16
Child10	1000	true	16 ±0 (10)	9 ±1	799 ±129
		0.01	102 ±1 (10)	97 ±0	799 ±129
		0.02	77 ±2 (10)	73 ±0	799 ±129
		0.05	94 ±2 (10)	89 ±0	799 ±129
	10000	true	26 ±0 (10)	22 ±1	1729 ±147
		0.01	73 ±2 (10)	71 ±1	1729 ±147
		0.02	118 ±2 (10)	118 ±1	1729 ±147
		0.05	73 ±1 (10)	74 ±0	1729 ±147

Table 7: Running time of ECOS, MMHC, and HC for each experimental condition in seconds (mean ± standard deviation). The numbers in the second parentheses for ECOS represent the number of times ECOS could finish in ten data sets within two days.

5. Discussion and Conclusion

We presented a new BN learning algorithm that finds the optimal network given a structural constraint, the super-structure. Our algorithm decomposes the super-structure into several clusters and computes optimal networks on each cluster for every ancestral constraint in order to ensure acyclic networks. Experimental evaluations using extended Alarm, Insurance, and Child networks show that our algorithm outperforms state-of-the-art greedy algorithms such as MMHC and HC with a TABU search extension both in terms of the BDeu score and SHD.

It may be possible to develop methods to further increase the feasibility of ECOS; we suggest some ready-to-apply tunings that should make our algorithm faster. First, the algorithm is highly parallelizable since each call to ACOS can be made independently. Further, one could apply a similar shrinkage technique to the interior nodes of the block tree as well if they are of a small size. The algorithm may also benefit from using ECOS recursively instead of ACOS or applying a COS search on the unbreakable clusters rather than a full OS (as is the current case with ACOS). Finally, limiting the maximal size of parent sets by a constant c, as proposed in Section 3.6, improves the feasibility of ECOS because it may also reduce the number of TDMs to consider in some cases. In terms of quality of the learnt network, the best improvements may be realized by developing better algorithms to learn super-structures; in fact, improvements in speed may also be obtained.

If there exist many edges between strongly connected components in the skeleton, the clusters obtained have too many cluster edges; this is usually the main bottleneck of our algorithm that limits its the feasibility. We should emphasize the fact that IT approaches add false-positive edges to the skeleton according to the specified significance level when learning the super-structure. Therefore, although the true skeleton may be well structured (i.e., the true skeleton can be clustered with a small number of cluster edges), the false-positive edges tend to be cluster edges and limit the feasibility of ECOS. For example, the super-structure of Insurance10 obtained with $\alpha = 0.05$ has a smaller average degree than the true skeleton; however, our algorithm did not finish the computation in two days. In addition, since the current version of ECOS depends on a super-structure estimated by the conditional independence test (MMPC), a causal relationship violating faithfulness, for example, XOR parents, can be missed.

We now consider whether our method is practically different from the one suggested by Friedman et al. (1999) and whether it is more efficient. Although we can argue that our method has been implemented and tested practically, it is also obvious that both strategies are different since ECOS can be applied on a skeleton that is not only decomposable in cluster trees but also decomposable in cluster graphs. For example, we converted the true skeleton of Alarm5 into a cluster tree. The true skeleton was decomposed into nine clusters, the largest of them containing 34 vertices and 20 cluster edges. This is not computable for an optimal search, and therefore, the method proposed by Friedman et al. (1999) would not be able to consider this case; such a case motivated us to relax the tree structure condition between clusters. Following the experimental results, we conclude that our approach, that is, decomposing the search on every cluster, succeeded in scaling-up the constrained optimal search.

Acknowledgments

We would like to thank André Fujita and Masao Nagasaki for their helpful comments in the initial stage of this work. Computational resources were provided by the Human Genome Center, Institute of Medical Science, University of Tokyo.

Appendix A.

Given a cluster C and a TDM δ (i.e., the corresponding $V_{C,\delta}^{in}$ and $V_{C,\delta}^{out}$ sets), we define for every possible NACS $\mathcal{A}$.

Definition 7 (NACS template) *The NACS template of $\mathcal{A}$ is a finite list of pairs of positive integers $\{(I_1,O_1),\cdots,(I_l,O_l)\}$ such that $|V_{C,\delta}^{out}| \geq O_1 > \cdots > O_l \geq 0$ and there are exactly I_k in-vertices v_j^{in} such that $|\mathcal{A}(v_j^{in})| = O_k$.*

For example, let us consider the following NACS $\mathcal{A}$ with $V_{C,\delta}^{in} = \{v_1^{in},\ldots,v_6^{in}\}$ and $V_{C,\delta}^{out} = \{v^{out},\ldots,v_3^{out}\}$ defined by

$$\begin{aligned}
\mathcal{A}(v_1^{in}) &= \{v_1^{out},v_2^{out},v_3^{out}\},\\
\mathcal{A}(v_6^{in}) &= \{v_1^{out},v_3^{out},v_3^{out}\},\\
\mathcal{A}(v_2^{in}) &= \{v_1^{out},v_3^{out}\},\\
\mathcal{A}(v_4^{in}) &= \{v_3^{out}\},\\
\mathcal{A}(v_3^{in}) &= \{v_3^{out}\},\\
\mathcal{A}(v_5^{in}) &= \emptyset.
\end{aligned}$$

Here, we arranged the AC sets by size to illustrate the fact that due to the definition of NACS, two in-vertices having the same size of ancestral constraint actually have the same ancestral constraint. The NACS template of $\mathcal{A}$ is

$$\{(2,3),(1,2),(2,1),(1,0)\}.$$

Further, following the definition of the NACS template, we have trivially that $\sum_{i=1}^{l} I_i = |V_{C,\delta}^{in}|$. Because every NACS admits a unique NACS template, templates define a partition of the space of NACS. Given the template $\{(I_1,O_1),\cdots,(I_l,O_l)\}$, there exist

$$\begin{pmatrix} |V_{C,\delta}^{out}| \\ O_1 \end{pmatrix} \times \begin{pmatrix} O_1 \\ O_2 \end{pmatrix} \times \cdots \times \begin{pmatrix} O_{l-1} \\ O_l \end{pmatrix}$$

different ways to assign the out-vertices and

$$\begin{pmatrix} |V_{C,\delta}^{in}| \\ I_1 \end{pmatrix} \times \begin{pmatrix} |V_{C,\delta}^{in}| - I_1 \\ I_2 \end{pmatrix} \times \cdots \times \begin{pmatrix} |V_{C,\delta}^{in}| - \sum_{i=1}^{l-1} I_i \\ I_l \end{pmatrix}$$

different ways to assign in-vertices. The product of these two values gives the number of NACSs corresponding to this template. All these NACSs can be generated by a brute force method that considers every possibility. Next, we introduce an algorithm that generates an ordered list of all NACSs by considering each template successively, starting from $(|V_{C,\delta}^{in}|,0)$ and ending with $(|V_{C,\delta}^{in}|,|V_{C,\delta}^{out}|)$.

Algorithm 7: NACS Enumerator
Input: A set of in-vertices $V_{C,\delta}^{in}$ and a set of out-vertices $V_{C,\delta}^{out}$
Onput: List of NACSs, L

1. Initialize an empty list of NACSs L.
2. Initialize a NACS template NT to $(|V^{in}_{C,\delta}|, 0)$ and l to 1.
3. While true, do:
   (a) Put all the NACSs having NT as template at the end of L.
   (b) If $NT = \{(|V^{in}_{C,\delta}|, |V^{out}_{C,\delta}|)\}$, return L.
   (c) If $l > 1$ and $O_{l-1} = O_l + 1$, increment I_{l-1}, otherwise insert $(1, O_l + 1)$ in NT immediately before (I_l, O_l) and increment l.
   (d) Decrement I_l and set O_l to 0.
   (e) If $I_l = 0$, remove (I_l, O_l) from NT and decrement l.

First, we define an order relation for the templates; we say that $(I_i, O_i) > (I'_j, O'_j)$ if $O_i > O'_j$ or $O_i = O'_j$ and $I_i > I'_j$. Then, we say that a NACS template NT is greater than another NT' if $(I_1, O_1) > (I'_1, O'_1)$ or $(I_j, O_j) = (I'_j, O'_j)$ for every $j < k$ and $(I_k, O_k) > (I'_k, O'_k)$. We note that if $\mathcal{A}_1 > \mathcal{A}_2$, then the template of $\mathcal{A}_1$ is greater than that of $\mathcal{A}_2$. Therefore, to prove that Algorithm 7 is correct, we simply need to prove that it considers every template in increasing order, that is, that the loop of step 3 generates the successor of a given template NT. Given the constraint that the sum of all I_k is $|V^{in}_{C,\delta}|$, there exist three different cases. If NT is of a size $l = 1$, then $NT = \{(|V^{in}_{C,\delta}|, O_1)\}$ and the following template is $\{(1, O_1 + 1), (|V^{in}_{C,\delta}| - 1, 0)\}$, as done in steps (c) and (d). Otherwise, in a similar manner to the previous case, if $O_{l-1} \neq O_l + 1$, the next template will be $\{(I_1, O_1), \cdots, (I_{l-1}, O_{l-1}), (1, O_l + 1), (I_l - 1, 0)\}$, as in Algorithm 7. Finally, in general, $O_{l-1} = O_l + 1$ holds, and we simply need to increment I_{l-1}, decrement I_l, and reset O_l to 0. After decrementing I_l, it is possible that $I_l = 0$, in which case the last element should be removed. Since Algorithm 7 accesses the direct successor of NT at each step starting from the smaller template $(|V^{in}_{C,\delta}|, 0)$ until the largest one, we can assert its correctness.

References

C.F. Aliferis, I. Tsamardinos, and A. Statnikov. Causal Explorer: A probabilistic network learning toolkit for discovery. *The 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences*, 2003.

I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper. The ALARM monitoring system: a case study with two probabilistic inference techniques for belief networks. *Proceedings of 2nd European Conference on Artificial Intelligence in Medicine*, 247–256, 1989.

J. Binder, S. Russell, and P. Smyth. Adaptive probabilistic networks with hidden variables. *Machine Learning*, **29**:213–244, 1997.

D.M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, **2**:445–498, 2002.

R.G., Cowell, A.P. Dawid, S.L. Lauritzen, and D.J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, Berlin, 1999.

C.P. de Campos, Z. Zheng, and Q. Ji. *The 26th International Conference on Machine Learning*, 2009 (in press).

R. Diestel. *Graph Theory 3rd edition*. Springer, Berlin, 2005.

N. Dojer. Learning Bayesian networks does not have to be NP-hard. *Proceedings of Mathematical Foundations of Computer Science*, 305–314, 2006.

N. Friedman, I. Nachman, and D. Pe'er. Learning Bayesian network structure from massive dataset: The "sparse candidate" algorithm. *The 15th Conference on Uncertainty in Artificial Intelligence*, 196–205, 1999.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, **5**:549–573, 2004.

M.E.J. Newman and M. Girvan (2004). Finding and evaluating community structure in networks. *Physical Review E*, **69**(22):26113–26113.

S. Ott, S. Imoto, and S. Miyano. Finding optimal models for small gene networks. *Pacific Symposium on Biocomputing*, **9**:557–567, 2004.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research*, **9**:2251–2286, 2008.

T. Silander and P. Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. *The 22nd Conference on Uncertainty in Artificial Intelligence*, 445–452, 2006.

A. Singh and A. Moore. Finding optimal Bayesian networks by dynamic programming (Technical Report). Carnegie Mellon University, 2005.

J. Suzuki. Learning Bayesian belief networks based on the MDL principle: An efficient algorithm using the branch and bound technique. *IEICE Transactions on Information and Systems*, 356–367, 1999.

I. Tsamardinos, L.E. Brown, and C.F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, **65**(1):31–78, 2006.

I. Tsamardinos, A. Stantnikov, L.E. Brown, and C.F. Aliferis. Generating realistic large Bayesian networks by tiling. *The 19th International FLAIRS Conference*, 592–597, 2006.

Journal of Machine Learning Research 11 (2010) 311-365 Submitted 2/09; Revised 7/09; Published 1/10

Bundle Methods for Regularized Risk Minimization

Choon Hui Teo* CHOONHUI.TEO@ANU.EDU.AU
College of Engineering and Computer Science
Australian National University
Canberra ACT 0200, Australia

S.V. N. Vishwanathan VISHY@STAT.PURDUE.EDU
Departments of Statistics and Computer Science
Purdue University
West Lafayette, IN 47907-2066 USA

Alex Smola ALEX@SMOLA.ORG
Yahoo! Research
Santa Clara, CA USA

Quoc V. Le QUOCLE@STANFORD.EDU
Department of Computer Science
Stanford University
Stanford, CA 94305 USA

Editor: Thorsten Joachims

Abstract

A wide variety of machine learning problems can be described as minimizing a regularized risk functional, with different algorithms using different notions of risk and different regularizers. Examples include linear Support Vector Machines (SVMs), Gaussian Processes, Logistic Regression, Conditional Random Fields (CRFs), and Lasso amongst others. This paper describes the theory and implementation of a scalable and modular convex solver which solves all these estimation problems. It can be parallelized on a cluster of workstations, allows for data-locality, and can deal with regularizers such as L_1 and L_2 penalties. In addition to the unified framework we present tight convergence bounds, which show that our algorithm converges in $O(1/\varepsilon)$ steps to ε precision for general convex problems and in $O(\log(1/\varepsilon))$ steps for continuously differentiable problems. We demonstrate the performance of our general purpose solver on a variety of publicly available data sets.

Keywords: optimization, subgradient methods, cutting plane method, bundle methods, regularized risk minimization, parallel optimization

∗. Also at Canberra Research Laboratory, NICTA.

1. Introduction

At the heart of many machine learning algorithms is the problem of minimizing a regularized risk functional. That is, one would like to solve

$$\min_w J(w) := \lambda\Omega(w) + R_{\text{emp}}(w), \tag{1}$$

$$\text{where } R_{\text{emp}}(w) := \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, w) \tag{2}$$

is the empirical risk. Moreover, $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$ are referred to as training instances and $y_i \in \mathcal{Y}$ are the corresponding labels. l is a (surrogate) convex loss function measuring the discrepancy between y and the predictions arising from using w. For instance, w might enter our model via $l(x,y,w) = (\langle w,x\rangle - y)^2$, where $\langle\cdot,\cdot\rangle$ denotes the standard Euclidean dot product. Finally, $\Omega(w)$ is a convex function serving the role of a regularizer with regularization constant $\lambda > 0$. Typically Ω is differentiable and cheap to compute. In contrast, the empirical risk term $R_{\text{emp}}(w)$ is often non-differentiable, and almost always computationally expensive to deal with.

For instance, if we consider the problem of predicting binary valued labels $y \in \{\pm 1\}$, we can set $\Omega(w) = \frac{1}{2}\|w\|_2^2$ (i.e., L_2 regularization), and the loss $l(x,y,w)$ to be the binary hinge loss, $\max(0, 1 - y\langle w,x\rangle)$, thus recovering linear Support Vector Machines (SVMs) (Joachims, 2006). Using the same regularizer but changing the loss function to $l(x,y,w) = \log(1+\exp(-y\langle w,x\rangle))$, yields logistic regression. Extensions of these loss functions allow us to handle structure in the output space (Bakir et al., 2007) (also see Appendix A for a comprehensive exposition of many common loss functions). On the other hand, changing the regularizer $\Omega(w)$ to the sparsity inducing $\|w\|_1$ (i.e., L_1 regularization) leads to Lasso-type estimation algorithms (Mangasarian, 1965; Tibshirani, 1996; Candes and Tao, 2005).

If the objective function J is differentiable, for instance in the case of logistic regression, we can use smooth optimization techniques such as the standard quasi-Newtons methods like BFGS or its limited memory variant LBFGS (Nocedal and Wright, 1999). These methods are effective and efficient even when m and d are large (Sha and Pereira, 2003; Minka, 2007). However, it is not straightforward to extend these algorithms to optimize a non-differentiable objective, for instance, when dealing with the binary hinge loss (see, e.g., Yu et al., 2008).

When J is non-differentiable, one can use nonsmooth convex optimization techniques such as the cutting plane method (Kelly, 1960) or its *stabilized* version the bundle method (Hiriart-Urruty and Lemaréchal, 1993). The bundle methods not only stabilize the optimization procedure but make the problem a well-posed one, that is, with unique solution. However, the amount of *external* stabilization that needs to be added is a parameter that requires careful tuning.

In this paper, we bypass this stabilization parameter tuning problem by taking a different route. The resultant algorithm – Bundle Method for Regularized Risk Minimization (BMRM) – has certain desirable properties: a) it has no parameters to tune, and b) it is applicable to a wide variety of regularized risk minimization problems. Furthermore, we show that BMRM has an $O(1/\varepsilon)$ rate of convergence for nonsmooth problems and $O(\log(1/\varepsilon))$ for smooth problems. This is significantly tighter than the $O(1/\varepsilon^2)$ rates provable for standard bundle methods (Lemaréchal et al., 1995). A related optimizer, SVM$^{\text{struct}}$ (Tsochantaridis et al., 2005), which is widely used in machine learning applications was also shown to converge at $O(1/\varepsilon^2)$ rates. Our analysis also applies to SVM$^{\text{struct}}$, which we show to be a special case of our solver, and hence tightens its convergence rate to $O(1/\varepsilon)$.

Very briefly, we highlight the two major advantages of our implementation. First, it is completely modular; new loss functions, regularizers, and solvers can be added with relative ease. Second, our architecture allows the empirical risk computation (2) to be easily parallelized. This makes our solver amenable to large data sets which cannot fit into the memory of a single computer. Our open source C/C++ implementation is freely available for download.[1]

The outline of our paper is as follows. In Section 2 we describe BMRM and contrast it with standard bundle methods. We also prove rates of convergence. In Section 3 we discuss implementation issues and present principled techniques to control memory usage, as well as to speed up computation via parallelization. Section 4 puts our work in perspective, and discusses related work. Section 5 is devoted to extensive experimental evaluation, which shows that our implementation is comparable to or better than specialized state-of-the-art solvers on a number of publicly available data sets. Finally, we conclude our work and discuss related issues in Section 6. In Appendix A we describe various *classes* of loss functions organized according to their common traits in computation. Long proofs are relegated to Appendix B. Before we proceed a brief note about our notation:

1.1 Notation

The indices of elements of a sequence or a set appear in subscript, for example, u_1, u_2. The i-th component of a vector u is denoted by $u^{(i)}$. $[k]$ is the shorthand for the set $\{1, 2, \ldots, k\}$. The L_p norm is defined as $\|u\|_p = (\sum_{i=1}^{d} |u^{(i)}|^p)^{1/p}$, for $p \geq 1$, and we use $\|\cdot\|$ to denote $\|\cdot\|_2$ whenever the context is clear. $\mathbf{1}_d$ and $\mathbf{0}_d$ denote the d-dimensional vectors of all ones and zeros respectively.

2. Bundle Methods

The precursor to the bundle methods is the cutting plane method (CPM) (Kelly, 1960). CPM uses subgradients, which are a generalization of gradients appropriate for convex functions, including those which are not necessarily smooth. Suppose w' is a point where a convex function J is finite. Then a subgradient is the normal vector of any tangential supporting hyperplane of J at w' (see Figure 1 for geometric intuition). Formally s' is called a subgradient of J at w' if, and only if,

$$J(w) \geq J(w') + \left\langle w - w', s' \right\rangle \quad \forall w. \tag{3}$$

The set of all subgradients at w' is called the subdifferential, and is denoted by $\partial_w J(w')$. If this set is not empty then J is said to be *subdifferentiable at* w'. On the other hand, if this set is a singleton then the function is said to be *differentiable* at w'. Convex functions are subdifferentiable everywhere in their domain (Hiriart-Urruty and Lemaréchal, 1993).

As implied by (3), J is bounded from below by its linearization (i.e., first order Taylor approximation) at w'. Given subgradients $s_1, s_2, \ldots, s_t$ evaluated at locations $w_0, w_1, \ldots, w_{t-1}$, we can state a tighter (piecewise linear) lower bound for J as follows

$$J(w) \geq J_t^{\mathrm{CP}}(w) := \max_{1 \leq i \leq t} \{J(w_{i-1}) + \langle w - w_{i-1}, s_i \rangle\}. \tag{4}$$

This lower bound forms the basis of the CPM, where at iteration t the set $\{w_i\}_{i=0}^{t-1}$ is augmented by

$$w_t := \operatorname*{argmin}_w J_t^{\mathrm{CP}}(w).$$

1. Software available at `http://users.rsise.anu.edu.au/~chteo/BMRM.html`.

This iteratively refines the piecewise linear lower bound J^{CP} and allows us to get close to the minimum of J (see Figure 2 for an illustration).

If w^* denotes the minimizer of J, then clearly each $J(w_i) \geq J(w^*)$ and hence $\min_{0 \leq i \leq t} J(w_i) \geq J(w^*)$. On the other hand, since $J \geq J_t^{\mathrm{CP}}$ it follows that $J(w^*) \geq J_t^{\mathrm{CP}}(w_t)$. In other words, $J(w^*)$ is sandwiched between $\min_{0 \leq i \leq t} J(w_i)$ and $J_t^{\mathrm{CP}}(w_t)$ (see Figure 3 for an illustration). The CPM monitors the monotonically decreasing quantity

$$\varepsilon_t := \min_{0 \leq i \leq t} J(w_i) - J_t^{\mathrm{CP}}(w_t),$$

and terminates whenever ε_t falls below a predefined threshold ε. This ensures that the solution $J(w_t)$ satisfies $J(w_t) \leq J(w^*) + \varepsilon$.

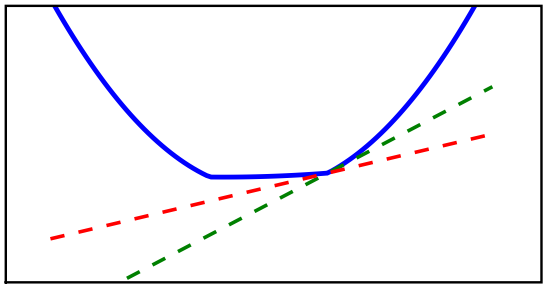


Figure 1: Geometric intuition of a subgradient. The nonsmooth convex function (solid blue) is only subdifferentiable at the "kink" points. We illustrate two of its subgradients (dashed green and red lines) at a "kink" point which are tangential to the function. The normal vectors to these lines are subgradients.

2.1 Standard Bundle Methods

Although CPM was shown to be convergent (Kelly, 1960), it is well known (see, e.g., Lemaréchal et al., 1995; Belloni, 2005) that CPM can be very slow when new iterates move too far away from the previous ones (i.e., causing unstable "zig-zag" behavior in the iterates).

Bundle methods stabilize CPM by augmenting the piecewise linear lower bound (e.g., $J_t^{\mathrm{CP}}(w)$ as in (4)) with a prox-function (i.e., proximity control function) which prevents overly large steps in the iterates (Kiwiel, 1990). Roughly speaking, there are 3 popular types of bundle methods, namely, *proximal* (Kiwiel, 1990), *trust region* (Schramm and Zowe, 1992), and *level set* (Lemaréchal et al., 1995).[2] All three versions use $\frac{1}{2}\|\cdot\|^2$ as their prox-function, but differ in the way they compute the

2. For brevity we will only describe "first-order" bundle methods, and omit discussion about "second-order" variants such as the bundle-Newton method of Lukšan and Vlček (1998).

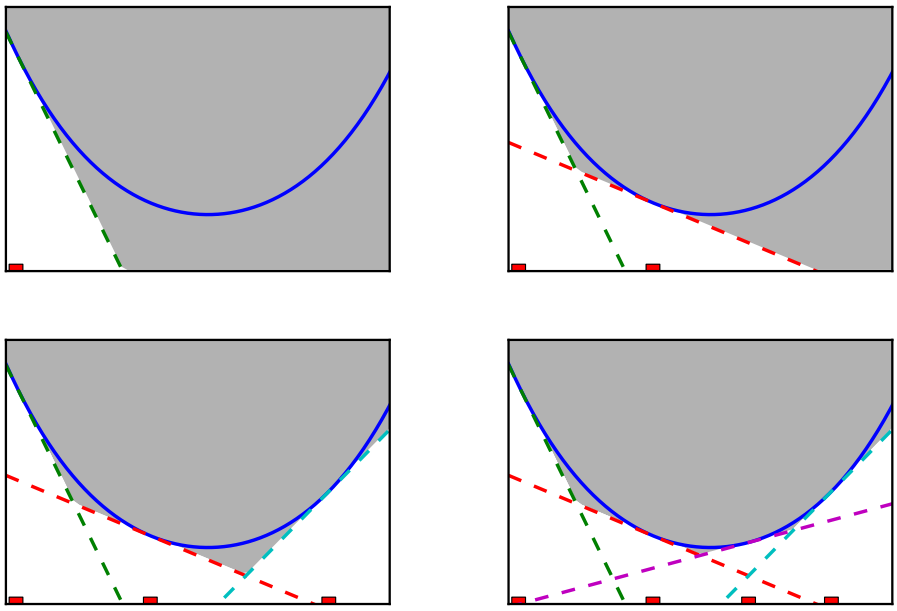


Figure 2: A convex function (blue solid curve) is bounded from below by its linearizations (dashed lines). The gray area indicates the piecewise linear lower bound obtained by using the linearizations. We depict a few iterations of the cutting plane method. At each iteration the piecewise linear lower bound is minimized and a new linearization is added at the minimizer (red rectangle). As can be seen, adding more linearizations improves the lower bound.

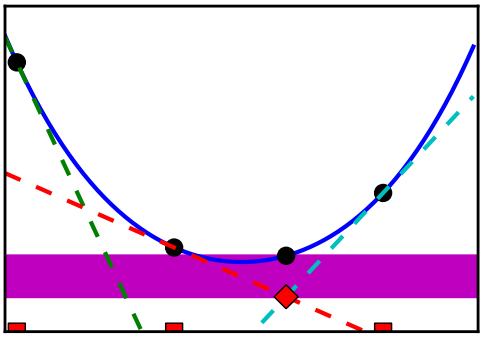


Figure 3: A convex function (blue solid curve) with three linearizations (dashed lines) evaluated at three different locations (red squares). The approximation gap ϵ_3 at the end of third iteration is indicated by the height of the magenta horizontal band, that is, difference between lowest value of $J(w)$ evaluated so far (lowest black circle) and the minimum of $J_3^{\mathrm{CP}}(w)$ (red diamond).

new iterate:

$$\text{proximal:} \quad w_t := \operatorname*{argmin}_w \{\tfrac{\zeta_t}{2}\|w-\hat{w}_{t-1}\|^2 + J_t^{\mathrm{CP}}(w)\}, \tag{5}$$

$$\text{trust region:} \quad w_t := \operatorname*{argmin}_w \{J_t^{\mathrm{CP}}(w) \mid \tfrac{1}{2}\|w-\hat{w}_{t-1}\|^2 \leq \kappa_t\}, \tag{6}$$

$$\text{level set:} \quad w_t := \operatorname*{argmin}_w \{\tfrac{1}{2}\|w-\hat{w}_{t-1}\|^2 \mid J_t^{\mathrm{CP}}(w) \leq \tau_t\},$$

where $\hat{w}_{t-1}$ is the current prox-center, and ζ_t, κ_t, and τ_t are positive trade-off parameters of the stabilization. Although (5) can be shown to be equivalent to (6) for appropriately chosen ζ_t and κ_t, tuning ζ_t is rather difficult while a trust region approach can be used for automatically tuning κ_t. Consequently the trust region algorithm BT of Schramm and Zowe (1992) is widely used in practice.

Since our methods (see Section 2.2) are closely related to the proximal bundle method, we will now describe them in detail. Similar to the CPM the proximal bundle method also builds a piecewise linear lower bound J_t^{CP} (see (4)). In contrast to the CPM, the piecewise linear lower bound augmented with a stabilization term $\frac{\zeta_t}{2}\|w-\hat{w}_{t-1}\|^2$, is minimized to produce the intermediate iterate $\bar{w}_t$. The approximation gap in this case includes the prox-function:

$$\varepsilon_t := J(\hat{w}_{t-1}) - \left[J_t^{\mathrm{CP}}(\bar{w}_t) + \frac{\zeta_t}{2}\|\bar{w}_t - \hat{w}_{t-1}\|^2\right].$$

If ε_t is less than the pre-defined threshold ε the algorithm exits. Otherwise, a line search is performed along the line joining $\hat{w}_{t-1}$ and $\bar{w}_t$ to produce the new iterate w_t. If w_t results in a sufficient decrease of the objective function then it is accepted as the new prox-center $\hat{w}_t$; this is called a serious step. Otherwise, the prox-center remains the same; this is called a null step. Detailed pseudocode can be found in Algorithm 1.

If the approximation gap ε_t is smaller than ε, then this ensures that the solution $J(\hat{w}_{t-1})$ satisfies $J(\hat{w}_{t-1}) \leq J(w) + \frac{\zeta_t}{2}\|w-\hat{w}_{t-1}\|^2 + \varepsilon$ for all w. In particular, if $J(w^*)$ denotes the optimum as before, then $J(\hat{w}_{t-1}) \leq J(w^*) + \frac{\zeta_t}{2}\|w^*-\hat{w}_{t-1}\|^2 + \varepsilon$. Contrast this with the approximation guarantee of the CPM, which does not involve the $\frac{\zeta_t}{2}\|w^*-\hat{w}_{t-1}\|^2$ term.

Although the positive coefficient ζ_t is assumed fixed throughout the algorithm, in practice it must be updated after every iteration to achieve faster convergence, and to guarantee a good quality solution (Kiwiel, 1990). Same is the case for κ_t and τ_t in trust region and level set bundle methods, respectively. Although the update is not difficult, the procedure relies on other parameters which require careful *tuning* (Kiwiel, 1990; Schramm and Zowe, 1992; Lemaréchal et al., 1995).

In the next section, we will describe our method (BMRM) which avoids this problem. There are two key differences between BMRM and the proximal bundle method: Firstly, BMRM maintains a piecewise linear lower bound of $R_{\mathrm{emp}}(w)$ instead of $J(w)$. Secondly, the the stabilizer (i.e., $\|w-\hat{w}_t\|^2$) in proximal bundle method is replaced by the regularizer $\Omega(w)$ hence there is no stabilization parameter to tune. As we will see, not only is the implementation straightforward, but the rates of convergence also improve from $O(1/\varepsilon^3)$ or $O(1/\varepsilon^2)$ to $O(1/\varepsilon)$.

Algorithm 1 Proximal Bundle Method

input & initialization: $\varepsilon \geq 0, \rho \in (0,1), w_0, t \leftarrow 0, \hat{w}_0 \leftarrow w_0$
loop
  $t \leftarrow t+1$
  Compute $J(w_{t-1})$ and $s_t \in \partial_w J(w_{t-1})$
  Update model $J_t^{\mathrm{CP}}(w) := \max_{1 \leq i \leq t} \{J(w_{i-1}) + \langle w - w_{i-1}, s_i \rangle\}$
  $\bar{w}_t \leftarrow \operatorname{argmin}_w J_t^{\mathrm{CP}}(w) + \frac{\zeta_t}{2} \|w - \hat{w}_{t-1}\|^2$
  $\varepsilon_t \leftarrow J(\hat{w}_{t-1}) - \left[J_t^{\mathrm{CP}}(\bar{w}_t) + \frac{\zeta_t}{2} \|\bar{w}_t - \hat{w}_{t-1}\|^2\right]$
  if $\varepsilon_t < \varepsilon$ **then return** $\bar{w}_t$
  Linesearch: $\eta_t \leftarrow \operatorname{argmin}_{\eta \in \mathbb{R}} J(\hat{w}_{t-1} + \eta(\bar{w}_t - \hat{w}_{t-1}))$ (if expensive, set $\eta_t = 1$)
  $w_t \leftarrow \hat{w}_{t-1} + \eta_t(\bar{w}_t - \hat{w}_{t-1})$
  if $J(\hat{w}_{t-1}) - J(w_t) \geq \rho\varepsilon_t$ **then**
    SERIOUS STEP: $\hat{w}_t \leftarrow w_t$
  else
    NULL STEP: $\hat{w}_t \leftarrow \hat{w}_{t-1}$
  end if
end loop

2.2 Bundle Methods for Regularized Risk Minimization (BMRM)

Define:

$$\begin{aligned}
\text{(subgradient of } R_{\mathrm{emp}}) \quad & a_t \in \partial_w R_{\mathrm{emp}}(w_{t-1}), \\
\text{(offset)} \quad & b_t := R_{\mathrm{emp}}(w_{t-1}) - \langle w_{t-1}, a_t \rangle, \\
\text{(piecewise linear lower bound of } R_{\mathrm{emp}}) \quad & R_t^{\mathrm{CP}}(w) := \max_{1 \leq i \leq t} \{\langle w, a_i \rangle + b_i\}, \\
\text{(piecewise convex lower bound of } J) \quad & J_t(w) := \lambda\Omega(w) + R_t^{\mathrm{CP}}(w), \\
\text{(iterate)} \quad & w_t := \min_w J_t(w), \\
\text{(approximation gap)} \quad & \varepsilon_t := \min_{0 \leq i \leq t} J(w_i) - J_t(w_t).
\end{aligned}$$

We now describe BMRM (Algorithm 2), and contrast it with the proximal bundle method. At iteration t the algorithm builds the lower bound R_t^{CP} to the empirical risk R_{emp}. The new iterate w_t is then produced by minimizing J_t which is R_t^{CP} augmented with the regularizer Ω; this is the key difference from the proximal bundle method which uses the $\frac{\zeta_t}{2} \|w - \hat{w}_{t-1}\|^2$ prox-function for stabilization. The algorithm repeats until the approximation gap ε_t is less than the pre-defined threshold ε. Unlike standard bundle methods there is no notion of a serious or null step in our algorithm. In fact, our algorithm does not even maintain a prox-center. It can be viewed as a special case of standard bundle methods where the prox-center is always the origin and never updated (hence every step is a null step). Furthermore, unlike the proximal bundle method, the approximation guarantees of our algorithm do not involve the $\frac{\zeta_t}{2} \|w^* - w_t\|^2$ term.

Algorithm 2 is simple and easy to implement as it does not involve a line search. In fact, whenever efficient (exact) line search is available, it can be used to achieve faster convergence as

Algorithm 2 BMRM

input & initialization: $\varepsilon \geq 0, w_0, t \leftarrow 0$
repeat
  $t \leftarrow t+1$
  Compute $a_t \in \partial_w R_{\text{emp}}(w_{t-1})$ and $b_t \leftarrow R_{\text{emp}}(w_{t-1}) - \langle w_{t-1}, a_t \rangle$
  Update model: $R_t^{\text{CP}}(w) := \max_{1 \leq i \leq t} \{\langle w, a_i \rangle + b_i\}$
  $w_t \leftarrow \operatorname{argmin}_w J_t(w) := \lambda\Omega(w) + R_t^{\text{CP}}(w)$
  $\varepsilon_t \leftarrow \min_{0 \leq i \leq t} J(w_i) - J_t(w_t)$
until $\varepsilon_t \leq \varepsilon$
return w_t

observed by Franc and Sonnenburg (2008) in the case of linear SVMs with binary hinge loss.[3] We now turn to a variant of BMRM which uses a line search (Algorithm 3); this is a generalization of the optimized cutting plane algorithm for support vector machines (OCAS) of Franc and Sonnenburg (2008). This variant first builds R_t^{CP} and minimizes J_t to obtain an intermediate iterate w_t. Then, it performs a line search along the line joining w_{t-1}^b and w_t to produce w_t^b which acts like the new prox-center. Note that $w_t - w_{t-1}^b$ is not necessarily a direction of descent; therefore the line search might return a zero step. Instead of using w_t^b as the new iterate the algorithm uses the pre-set parameter θ to generate w_t^c on the line segment joining w_t^b and w_t. Franc and Sonnenburg (2008) report that setting $\theta = 0.9$ works well in practice. It is easy to see that Algorithm 3 reduces to Algorithm 2 if we set $\eta_t = 1$ for all t, and use the same termination criterion. It is worthwhile noting that this variant is not applicable for structured learning problems such as Max-Margin Markov Networks (Taskar et al., 2004), because no efficient line search is known for such problems.

A specialized variant of BMRM which handles quadratic regularizers, that is, $\Omega(w) = \frac{1}{2}\|w\|^2$ was first introduced to the machine learning community by Tsochantaridis et al. (2005) as SVM$^{\text{struct}}$. In particular, SVM$^{\text{struct}}$ handles quadratic regularizers $\Omega(w) = \frac{1}{2}\|w\|^2$ and non-differentiable large margin loss functions such as (24). Its 1-slack formulation (Joachims et al., 2009) can be shown to be equivalent to BMRM for this specific type of regularizer and loss function. Somewhat confusingly, these algorithms are called the cutting plane method even though they are closer in spirit to bundle methods.

2.3 Dual Problems

In this section, we describe how the sub-problem

$$w_t = \operatorname*{argmin}_w J_t(w) := \lambda\Omega(w) + \max_{1 \leq i \leq t} \langle w, a_i \rangle + b_i \tag{7}$$

in Algorithms 2 and 3 is solved via a dual formulation. In fact, we will show that we need not know $\Omega(w)$ at all, instead it is sufficient to work with its Fenchel dual (Hiriart-Urruty and Lemaréchal, 1993):

Definition 1 (Fenchel Dual) *Denote by $\Omega : \mathcal{W} \to \mathbb{R}$ a convex function on a convex set $\mathcal{W}$. Then the dual Ω^* of Ω is defined as*

$$\Omega^*(\mu) := \sup_{w \in \mathcal{W}} \langle w, \mu \rangle - \Omega(w). \tag{8}$$

3. A different optimization method but with identical efficient line search procedure is described in Yu et al. (2008).

Algorithm 3 BMRM with Line Search

input & initialization: $\varepsilon \geq 0$, $\theta \in (0,1]$, $w_0^b, w_0^c \leftarrow w_0^b, t \leftarrow 0$
repeat
  $t \leftarrow t+1$
  Compute $a_t \in \partial_w R_{\text{emp}}(w_{t-1}^c)$, and $b_t \leftarrow R_{\text{emp}}(w_{t-1}^c) - \langle w_{t-1}^c, a_t \rangle$
  Update model: $R_t^{\text{CP}}(w) := \max_{1 \leq i \leq t} \{\langle w, a_i \rangle + b_i\}$
  $w_t \leftarrow \operatorname{argmin}_w J_t(w) := \lambda \Omega(w) + R_t^{\text{CP}}(w)$
  Linesearch: $\eta_t \leftarrow \operatorname{argmin}_{\eta \in \mathbb{R}} J(w_{t-1}^b + \eta(w_t - w_{t-1}^b))$
  $w_t^b \leftarrow w_{t-1}^b + \eta_t(w_t - w_{t-1}^b)$
  $w_t^c \leftarrow (1-\theta) w_t^b + \theta w_t$
  $\varepsilon_t \leftarrow J(w_t^b) - J_t(w_t)$
until $\varepsilon_t \leq \varepsilon$
return w_t^b

Several choices of regularizers are common. For $\mathcal{W} = \mathbb{R}^d$ the squared norm regularizer yields

$$\Omega(w) = \frac{1}{2}\|w\|_2^2 \qquad \text{and} \qquad \Omega^*(\mu) = \frac{1}{2}\|\mu\|_2^2.$$

More generally, for L_p norms one obtains (Boyd and Vandenberghe, 2004; Shalev-Shwartz and Singer, 2006):

$$\Omega(w) = \frac{1}{2}\|w\|_p^2 \qquad \text{and} \qquad \Omega^*(\mu) = \frac{1}{2}\|\mu\|_q^2 \text{ where } \frac{1}{p} + \frac{1}{q} = 1.$$

For any positive definite matrix B, we can construct a quadratic form regularizer which allows non-uniform penalization of the weight vector as:

$$\Omega(w) = \frac{1}{2} w^\top B w \qquad \text{and} \qquad \Omega^*(\mu) = \frac{1}{2} \mu^\top B^{-1} \mu.$$

For the *unnormalized* negative entropy, where $\mathcal{W} = \mathbb{R}_+^d$, we have

$$\Omega(w) = \sum_i w^{(i)} \log w^{(i)} \qquad \text{and} \qquad \Omega^*(\mu) = \sum_i \exp \mu^{(i)}.$$

For the *normalized* negative entropy, where $\mathcal{W} = \{w \mid w \geq 0 \text{ and } \|w\|_1 = 1\}$ is the probability simplex, we have

$$\Omega(w) = \sum_i w^{(i)} \log w^{(i)} \qquad \text{and} \qquad \Omega^*(\mu) = \log \sum_i \exp \mu^{(i)}.$$

If Ω is differentiable the w at which the supremum of (8) is attained can be written as $w = \partial_\mu \Omega^*(\mu)$ (Boyd and Vandenberghe, 2004). In the sequel we will always assume that Ω^* is twice differentiable. Note that all the regularizers we discussed above are twice differentiable. The following theorem states the dual problem of (7).

Theorem 2 *Denote by $A = [a_1, \ldots, a_t]$ the matrix whose columns are the (sub)gradients, and let $b = [b_1, \ldots, b_t]$. The dual problem of*

$$w_t = \operatorname*{argmin}_{w \in \mathbb{R}^d} \{J_t(w) := \max_{1 \le i \le t} \langle w, a_i \rangle + b_i + \lambda \Omega(w)\} \quad \text{is} \tag{9}$$

$$\alpha_t = \operatorname*{argmax}_{\alpha \in \mathbb{R}^t} \{J_t^*(\alpha) := -\lambda \Omega^*(-\lambda^{-1} A\alpha) + \alpha^\top b \mid \alpha \ge 0,\ \|\alpha\|_1 = 1\}. \tag{10}$$

Furthermore, w_t and α_t are related by the dual connection $w_t = \partial \Omega^*(-\lambda^{-1} A\alpha_t)$.

Proof We rewrite (9) as a constrained optimization problem: $\min_{w,\xi} \lambda\Omega(w) + \xi$ subject to $\xi \ge \langle w, a_i \rangle + b_i$ for $i = 1, \ldots, t$. By introducing non-negative Lagrange multipliers α and recalling that $\mathbf{1}_t$ denotes the t dimensional vector of all ones, the corresponding Lagrangian can be written as

$$L(w, \xi, \alpha) = \lambda\Omega(w) + \xi - \alpha^\top \left(\xi \mathbf{1}_t - A^\top w - b\right) \text{ with } \alpha \ge 0, \tag{11}$$

where $\alpha \ge 0$ denotes that each component of α is non-negative. Taking derivatives with respect to ξ yields $1 - \alpha^\top \mathbf{1}_t = 0$. Moreover, minimization of L with respect to w implies solving $\max_w \left\langle w, -\lambda^{-1} A\alpha \right\rangle - \Omega(w) = \Omega^*(-\lambda^{-1} A\alpha)$. Plugging both terms back into (11) we eliminate the primal variables ξ and w. ■

Since Ω^* is assumed to be twice differentiable and the constraints of (10) are simple, one can easily solve (10) with standard smooth optimization methods such as the *penalty/barrier* methods (Nocedal and Wright, 1999). Recall that for the square norm regularizer $\Omega(w) = \frac{1}{2}\|w\|_2^2$, commonly used in SVMs and Gaussian Processes, the Fenchel dual is given by $\Omega^*(\mu) = \frac{1}{2}\|\mu\|_2^2$. The following corollary is immediate:

Corollary 3 *For quadratic regularization, that is, $\Omega(w) = \frac{1}{2}\|w\|_2^2$, (10) becomes*

$$\alpha_t = \operatorname*{argmax}_{\alpha \in \mathbb{R}^t} \{-\tfrac{1}{2\lambda} \alpha^\top A^\top A\alpha + \alpha^\top b \mid \alpha \ge 0,\ \|\alpha\|_1 = 1\}.$$

This means that for quadratic regularization the dual optimization problem is a quadratic program (QP) where the number of constraints equals the number of (sub)gradients computed previously. Since t is typically in the order of 10s to 100s, the resulting QP is very cheap to solve. In fact, we do not even need to know the (sub)gradients explicitly. All that is required to define the QP are the inner products between (sub)gradients $\langle a_i, a_j \rangle$.

2.4 Convergence Analysis

While the variants of bundle methods we proposed are intuitively plausible, it remains to be shown that they have good rates of convergence. In fact, past results, such as those by Tsochantaridis et al. (2005) suggest a slow $O(1/\varepsilon^2)$ rate of convergence. In this section we tighten their results and show an $O(1/\varepsilon)$ rate of convergence for nonsmooth loss functions and $O(\log(1/\varepsilon))$ rates for smooth loss functions under mild assumptions. More concretely we prove the following two convergence results:

(a) Assume that $\max_{u \in \partial_w R_{\text{emp}}(w)} \|u\| \le G$. For regularizers $\Omega(w)$ for which $\left\|\partial_\mu^2 \Omega^*(\mu)\right\| \le H^*$ we prove $O(1/\varepsilon)$ rate of convergence, that is, we show that our algorithm converges to within ε of the optimal solution in $O(1/\varepsilon)$ iterations.

(b) Under the above conditions, if furthermore $\left\|\partial_w^2 J(w)\right\| \leq H$, that is, the Hessian of J is bounded, we can show $O(\log(1/\varepsilon))$ rate of convergence.

For our convergence proofs we use a duality argument similar to those put forward in Shalev-Shwartz and Singer (2006) and Tsochantaridis et al. (2005), both of which share key techniques with Zhang (2003). Recall that ε_t denotes our approximation gap, which in turn upper bounds how far away we are from the optimal solution. In other words, $\varepsilon_t \geq \min_{0\leq i\leq t} J(w_i) - J^*$, where J^* denotes the optimum value of the objective function J. The quantity $\varepsilon_t - \varepsilon_{t+1}$ can thus be viewed as the "progress" made towards J^* in iteration t. The crux of our proof argument lies in showing that for nonsmooth loss functions the recurrence $\varepsilon_t - \varepsilon_{t+1} \geq c \cdot \varepsilon_t^2$ holds for some appropriately chosen constant c. The rates follow by invoking a lemma from Abe et al. (2001). In the case of the smooth losses we show that $\varepsilon_t - \varepsilon_{t+1} \geq c' \cdot \varepsilon_t$ thus implying an $O(\log(1/\varepsilon))$ rate of convergence.

In order to show the required recurrence, we first observe that by strong duality the values of the primal and dual problems (9) and (10) are equal at optimality. Hence, any progress in J_{t+1} can be computed in the dual. Next, we observe that the solution of the dual problem (10) at iteration t, denoted by α_t, forms a feasible set of parameters for the dual problem (10) at iteration $t+1$ by means of the parameterization $(\alpha_t, 0)$, that is, by padding α_t with a 0. The value of the objective function in this case equals $J_t(w_t)$.

To obtain a lower bound on the improvement due to $J_{t+1}(w_{t+1})$ we perform a 1-d optimization along $((1-\eta)\alpha_t, \eta)$ in (10). The constraint $\eta \in (0,1)$ ensures dual feasibility. We will then bound this improvement in terms of ε_t. Note that, in general, solving the dual problem (10) results in a increase which is larger than that obtained via the line search. The 1-d minimization is used only for analytic tractability. We now state our key theorem and prove it in Appendix B.

Theorem 4 *Assume that* $\max_{u\in\partial_w R_{\mathrm{emp}}(w)} \|u\| \leq G$ *for all* $w \in \operatorname{dom} J$. *Also assume that* Ω^* *has bounded curvature, that is,* $\left\|\partial_\mu^2 \Omega^*(\mu)\right\| \leq H^*$ *for all* $\mu \in \{-\lambda^{-1}\sum_{i=1}^{t+1}\alpha_i a_i$ *where* $\alpha_i \geq 0,\ \forall i$ *and* $\sum_{i=1}^{t+1}\alpha_i = 1\}$. *In this case we have*

$$\varepsilon_t - \varepsilon_{t+1} \geq \tfrac{\varepsilon_t}{2}\min(1, \lambda\varepsilon_t/4G^2H^*). \tag{12}$$

Furthermore, if $\left\|\partial_w^2 J(w)\right\| \leq H$, *then we have*

$$\varepsilon_t - \varepsilon_{t+1} \geq \begin{cases} \varepsilon_t/2 & \text{if } \varepsilon_t > 4G^2H^*/\lambda \\ \lambda/8H^* & \text{if } 4G^2H^*/\lambda \geq \varepsilon_t \geq H/2 \\ \lambda\varepsilon_t/4HH^* & \text{otherwise.} \end{cases}$$

Note that the error keeps on halving initially and settles for a somewhat slower rate of convergence after that, whenever the Hessian of the overall risk is bounded from above. The reason for the difference in the convergence bound for differentiable and non-differentiable losses is that in the former case the gradient of the risk converges to 0 as we approach optimality, whereas in the former case, no such guarantees hold (e.g., when minimizing $|x|$ the (sub)gradient does not vanish at the optimum). The dual of many regularizers, for example, norm, squared L_p norm, and the entropic regularizer have bounded second derivative. See, for example, Shalev-Shwartz and Singer (2006) for a discussion and details. Thus our condition $\left\|\partial_\mu^2 \Omega^*(\mu)\right\| \leq H^*$ is not unreasonable. We are now in a position to state our convergence results. The proof is in Appendix B.

Theorem 5 *Assume that $J(w) \geq 0$ for all w. Under the assumptions of Theorem 4 we can give the following convergence guarantee for Algorithm 2. For any $\varepsilon < 4G^2H^*/\lambda$ the algorithm converges to the desired precision after*

$$n \leq \log_2 \frac{\lambda J(0)}{G^2 H^*} + \frac{8G^2 H^*}{\lambda \varepsilon} - 1$$

steps. Furthermore if the Hessian of $J(w)$ is bounded, convergence to any $\varepsilon \leq H/2$ takes at most the following number of steps:

$$n \leq \log_2 \frac{\lambda J(0)}{4G^2 H^*} + \frac{4H^*}{\lambda} \max\left[0, H - 8G^2 H^*/\lambda\right] + \frac{4HH^*}{\lambda} \log(H/2\varepsilon).$$

Several observations are in order: First, note that the number of iterations only depends *logarithmically* on how far the initial value $J(0)$ is away from the optimal solution. Compare this to the result of Tsochantaridis et al. (2005), where the number of iterations is linear in $J(0)$.

Second, we have an $O(1/\varepsilon)$ dependence in the number of iterations in the non-differentiable case, as opposed to the $O(1/\varepsilon^2)$ rates of Tsochantaridis et al. (2005). In addition to that, the convergence is $O(\log(1/\varepsilon))$ for continuously differentiable problems.

Note that whenever R_{emp} is the average over many piecewise linear functions, R_{emp} behaves essentially like a function with bounded Hessian as long as we are taking large enough steps not to "notice" the fact that the term is actually nonsmooth.

Remark 6 *For $\Omega(w) = \frac{1}{2}\|w\|^2$ the dual Hessian is exactly $H^* = 1$. Moreover we know that $H \geq \lambda$ since $\left\|\partial_w^2 J(w)\right\| = \lambda + \left\|\partial_w^2 R_{\text{emp}}(w)\right\|$.*

Effectively the rate of convergence of the algorithm is governed by upper bounds on the primal and dual curvature of the objective function. This acts like a condition number of the problem—for $\Omega(w) = \frac{1}{2} w^\top Q w$ the dual is $\Omega^*(z) = \frac{1}{2} z^\top Q^{-1} z$, hence the largest eigenvalues of Q and Q^{-1} would have a significant influence on the convergence.

In terms of λ the number of iterations needed for convergence is $O(\lambda^{-1})$. In practice the iteration count *does* increase with λ, albeit not as badly as predicted. This is likely due to the fact that the empirical risk R_{emp} is typically rather smooth and has a certain inherent curvature which acts as a natural regularizer in addition to the regularization afforded by $\lambda\Omega(w)$.

For completeness we also state the convergence guarantees for Algorithm 3 and provide a proof in Appendix B.3.

Theorem 7 *Under the assumptions of Theorem 4 Algorithm 3 converges to the desired precision ε after*

$$n \leq \frac{8G^2 H^*}{\lambda \varepsilon}$$

steps for any $\varepsilon < 4G^2H^*/\lambda$.

3. Implementation Issues

In this section, we discuss the memory and computational issues of the implementation of BMRM. In addition, we provide two variants of BMRM: one is memory efficient and the other one is parallelized.

3.1 Solving the BMRM Subproblem (7) with Limited Memory Space

In Section 2.3 we mentioned the dual of subproblem (7) (i.e., (10)) which is usually easier to solve when the dimensionality d of the problem is larger than the number of iterations t required by BMRM to reach desired precision ε. Although t is usually in the order of 10^2, a problem with d in the order of 10^6 or higher may use up all memory of a typical machine to store the bundle, that is, linearizations $\{(a_i, b_i)\}$, before the convergence is achieved.[4] Here we describe a principled technique which controls the memory usage while maintaining convergence guarantees.

Note that at iteration t, before the computation for new iterate w_t, Algorithm 2 maintains a bundle of t (sub)gradients $\{a_i\}_{i=1}^{t}$ of R_{emp} computed at the locations $\{w_i\}_{i=0}^{t-1}$. Furthermore, the Lagrange multipliers α_{t-1} obtained in iteration $t-1$ satisfy $\alpha_{t-1} \geq 0$ and $\sum_{i=1}^{t-1} \alpha_{t-1}^{(i)} = 1$ by the constraints of (10). We define the *aggregated* (sub)gradient $\hat{a}_I$, offset $\hat{b}_I$ and Lagrange multiplier $\hat{\alpha}_{t-1}^{(I)}$ as

$$\hat{a}_I := \frac{1}{\hat{\alpha}_{t-1}^{(I)}} \sum_{i \in I} \alpha_{t-1}^{(i)} a_i, \quad \hat{b}_I := \frac{1}{\hat{\alpha}_{t-1}^{(I)}} \sum_{i \in I} \alpha_{t-1}^{(i)} b_i, \quad \text{and} \quad \hat{\alpha}_{t-1}^{(I)} := \sum_{i \in I} \alpha_{t-1}^{(i)},$$

respectively, where $I \subseteq [t-1]$ is an index set (Kiwiel, 1983). Clearly, the optimality of (10) at the end of iteration $t-1$ is maintained when a subset $\left\{(a_i, b_i, \alpha_{t-1}^{(i)})\right\}_{i \in I}$ is replaced by the aggregate $(\hat{a}_I, \hat{b}_I, \hat{\alpha}_{t-1}^{(I)}))$ for any $I \subseteq [t-1]$.

To obtain a new iterate w_t via (10) with memory space for at most k linearizations, we can, for example, replace $\{(a_i, b_i)\}_{i \in I}$ with $(\hat{a}_I, \hat{b}_I)$ where $I = [t-k+1]$ and $2 \leq k \leq t$. Then, we solve a k-dimensional variant of (10) with $A := [\hat{a}_I, a_{t-k+2}, \ldots, a_t]$, $b := [\hat{b}_I, b_{t-k+2}, \ldots, b_t]$, and $\alpha \in \mathbb{R}^k$. The optimum of this variant will be lower than or equal to that of (10) as the latter has higher degree of freedom than the former. Nevertheless, solving this variant with $2 \leq k \leq t$ will still guarantee convergence (recall that our convergence proof only uses $k = 2$). In the sequel we name the aforementioned number k as the "bundle size" since it indicates the number of linearizations the algorithm keeps.

For concreteness, we provide here a memory efficient BMRM variant for the cases where $\Omega(w) = \frac{1}{2}\|w\|_2^2$ and $k = 2$. We first see that the dual of subproblem (7) now reads:

$$\begin{aligned} \eta &= \operatorname*{argmax}_{0 \leq \eta \leq 1} -\tfrac{1}{2\lambda} \left\| \hat{a}_{[t-1]} + \eta(a_t - \hat{a}_{[t-1]}) \right\|_2^2 + \hat{b}_{[t-1]} + \eta(b_t - \hat{b}_{[t-1]}) \\ &\equiv \operatorname*{argmax}_{0 \leq \eta \leq 1} -\tfrac{\eta}{\lambda} \hat{a}_{[t-1]}^\top (a_t - \hat{a}_{[t-1]}) - \tfrac{\eta^2}{2\lambda} \left\| a_t - \hat{a}_{[t-1]}^\top \right\|^2 + \eta(b_t - \hat{b}_{[t-1]}). \end{aligned} \tag{13}$$

Since (13) is quadratic in η, we can obtain the optimal η by setting the derivative of the objective in (13) to zero and clipping η in the range $[0, 1]$:

$$\eta = \min\left(\max\left(0, \frac{b_t - \hat{b}_{[t-1]} + w_{t-1}^\top a_t + \lambda \|w_{t-1}\|^2}{\frac{1}{\lambda}\|a_t + \lambda w_{t-1}\|^2}\right), 1\right) \tag{14}$$

4. In practice, we can remove those linearizations $\{(a_i, b_i)\}$ whose Lagrange multipliers α_i are 0 after solving (10). Although this heuristic works well and does not affect the convergence guarantee, there is no bound on the minimum number of linearizations with non-zero Lagrange multipliers needed to achieve convergence.

where $w_{t-1} = -\frac{1}{\lambda}\hat{a}_{[t-1]}$ by the dual connection. With the optimal η, we obtain the new primal iterate $w_t = (1-\eta)w_{t-1} - (\eta/\lambda)a_t$. Algorithm 4 lists the details. Note that this variant is simple to implement and does not require a QP solver.

Algorithm 4 BMRM with Aggregation of Previous Linearizations

input & initialization: $\varepsilon \geq 0, w_0, t \leftarrow 1$
Compute $a_1 \in \partial_w R_{\text{emp}}(w_0)$, and $b_1 \leftarrow R_{\text{emp}}(w_0) - \langle w_0, a_1 \rangle$
$w_1 \leftarrow -\frac{1}{\lambda}a_1$
$\hat{b}_{[1]} \leftarrow b_1$
repeat
  $t \leftarrow t+1$
  Compute $a_t \in \partial_w R_{\text{emp}}(w_{t-1})$ and $b_t \leftarrow R_{\text{emp}}(w_{t-1}) - \langle w_{t-1}, a_t \rangle$
  Compute η using Eq. (14)
  $w_t \leftarrow (1-\eta)w_{t-1} - (\eta/\lambda)a_t$
  $\hat{b}_{[t]} \leftarrow (1-\eta)\hat{b}_{[t-1]} + \eta b_t$
  $\varepsilon_t \leftarrow \min_{0 \leq i \leq t} \frac{\lambda}{2}\|w_i\|^2 + R_{\text{emp}}(w_i) - \frac{\lambda}{2}\|w_t\|^2 - \hat{b}_{[t]}$
until $\varepsilon_t \leq \varepsilon$

3.2 Parallelization

Algorithms 2, 3, and 4 the evaluation of $R_{\text{emp}}(w)$ (and $\partial_w R_{\text{emp}}(w)$) is cleanly separated from the computation of new iterate and the choice of regularizer. If R_{emp} is additively decomposable over the examples (x_i, y_i), that is, can be expressed as a sum of some independent loss terms $l(x_i, y_i, w)$, then we can parallelize these algorithms easily by splitting the data sets and the computation R_{emp} over multiple machines. This parallelization scheme not only reduces the computation time but also allows us to handle data set with size exceeding the memory available on a single machine.

Without loss of generality, we describe a parallelized version of Algorithm 2 here. Assume there are p slave machines and 1 master machine available. At the beginning, we partition a given data set $D = \{(x_i, y_i)\}_{i=1}^m$ into p disjoint sub-datasets $\{D_i\}_{i=1}^p$ and assign one sub-dataset to each slave machine. At iteration t, the master first broadcasts the current iterate w_{t-1} to all p slaves (e.g., using MPI function `MPI::Broadcast` Gropp et al. 1999). The slaves then compute the losses and (sub)gradients on their local sub-datasets in parallel. As soon as the losses and (sub)gradients computation finished, the master combines the results (e.g., using `MPI::AllReduce`). With the combined (sub)gradient and offset, the master computes the new iterate w_t as in Algorithms 2 and 3. This process repeats until convergence is achieved. Detailed pseudocode can be found in Algorithm 5.

4. Related Research

The *kernel trick* is widely used to transform many existing machine learning algorithms into ones operating on a Reproducing Kernel Hilbert Space (RKHS). One lifts w into an RKHS and replaces all inner product computations with a positive definite kernel function $k(x, x') \leftarrow \langle x, x' \rangle$. Examples of algorithms which employ the kernel trick (but essentially still solve (1)) include Support Vector regression (Vapnik et al., 1997), novelty detection (Schölkopf et al., 2001), Huber's robust regression, quantile regression (Takeuchi et al., 2006), ordinal regression (Herbrich et al., 2000), rank-

Algorithm 5 Parallel BMRM

input: $\varepsilon \geq 0$, w_0, data set D, number of slave machines p
initialization: $t \leftarrow 0$, assign sub-dataset D_i to slave i, $i = 1, \ldots, p$
repeat
  $t \leftarrow t + 1$
  Master: Broadcast w_{t-1} to all slaves
  Slaves: Computes $R^i_{\text{emp}}(w_{t-1}) := \sum_{(x,y)\in D_i} l(x, y, w_{t-1})$ and $a^i_t \in \partial_w R^i_{\text{emp}}(w_{t-1})$
  Master: Aggregate $a_t := \frac{1}{|D|}\sum_{i=1}^{p} a^i_t$ and $b_t := \frac{1}{|D|}\sum_{i=1}^{p} R^i_{\text{emp}}(w_{t-1}) - \langle w_{t-1}, a_t \rangle$
  Master: Update model $R^{\text{CP}}_t(w) := \max_{1\leq j\leq t}\{\langle w, a_j \rangle + b_j\}$
  Master: $w_t \leftarrow \operatorname{argmin}_w J_t(w) := \lambda\Omega(w) + R^{\text{CP}}_t(w)$
  Master: $\varepsilon_t \leftarrow \min_{0\leq i\leq t} J(w_i) - J_t(w_t)$
until $\varepsilon_t \leq \varepsilon$
return w_t

ing (Crammer and Singer, 2005), maximization of multivariate performance measures (Joachims, 2005), structured estimation (Taskar et al., 2004; Tsochantaridis et al., 2005), Gaussian Process regression (Williams, 1998), conditional random fields (Lafferty et al., 2001), graphical models (Cowell et al., 1999), exponential families (Barndorff-Nielsen, 1978), and generalized linear models (Fahrmeir and Tutz, 1994).

Traditionally, specialized solvers have been developed for solving the kernel version of (1) in the dual (see, e.g., Chang and Lin, 2001; Joachims, 1999). These algorithms construct the Lagrange dual, and solve for the Lagrange multipliers efficiently. Only recently, research focus has shifted back to solving (1) in the primal (see, e.g., Chapelle, 2007; Joachims, 2006; Sindhwani and Keerthi, 2006). This spurt in research interest is due to three main reasons: First, many interesting problems in diverse areas such as text classification, word-sense disambiguation, and drug design already employ rich high dimensional data which does not necessarily benefit from the kernel trick. All these domains are characterized by large data sets (with m in the order of a million) and very sparse features (e.g., the bag of words representation of a document). Second, efficient factorization methods (e.g., Fine and Scheinberg, 2001) can be used for a low rank representation of the kernel matrix thereby effectively rendering the problem linear. Third, approximation methods such as the *Random Feature Map* proposed by Rahimi and Recht (2008) can efficiently approximate a infinite dimensional nonlinear feature map associated to a kernel by a finite dimensional one. Therefore our focus on the primal optimization problem is not only pertinent but also timely.

The widely used SVM$^{\text{struct}}$ optimizer of Thorsten Joachims[5] is closely related to BMRM. While BMRM can handle many different regularizers and loss functions, SVM$^{\text{struct}}$ is mainly geared towards square norm regularizers and non-differentiable soft-margin type loss functions. On the other hand, SVM$^{\text{struct}}$ can handle kernels while BMRM mainly focuses on the primal problem.

Our convergence analysis is closely related to Shalev-Shwartz and Singer (2006) who prove mistake bounds for online algorithms by lower bounding the progress in the dual. Although not stated explicitly, essentially the same technique of lower bounding the dual improvement was used by Tsochantaridis et al. (2005) to show polynomial time convergence of the SVM$^{\text{struct}}$ algorithm. The main difference however is that Tsochantaridis et al. (2005) only work with a quadratic ob-

5. Software available at `http://svmlight.joachims.org/svm_struct.html`.

jective function while the framework proposed by Shalev-Shwartz and Singer (2006) can handle arbitrary convex functions. In both cases, a weaker analysis led to $O(1/\varepsilon^2)$ rates of convergence for nonsmooth loss functions. On the other hand, our results establish a $O(1/\varepsilon)$ rate for nonsmooth loss functions and $O(\log(1/\varepsilon))$ rates for smooth loss functions under mild technical assumptions.

Another related work is SVM$^{\text{perf}}$ (Joachims, 2006) which solves the SVM with linear kernel in linear time. SVM$^{\text{perf}}$ finds a solution with accuracy ε in $O(md/(\lambda\varepsilon^2))$ time, where the m training patterns $x_i \in \mathbb{R}^d$. This bound was improved by Shalev-Shwartz et al. (2007) to $\tilde{O}(1/\lambda\delta\varepsilon)$ for obtaining an accuracy of ε with confidence $1-\delta$. Their algorithm, Pegasos, essentially performs stochastic (sub)gradient descent but projects the solution back onto the L_2 ball of radius $1/\sqrt{\lambda}$. Note that Pegasos also can be used in an online setting. This, however, only applies whenever the empirical risk decomposes into individual loss terms (e.g., it is not applicable to multivariate performance scores Joachims 2005).

The third related strand of research considers gradient descent in the primal with a line search to choose the optimal step size (see, e.g., Boyd and Vandenberghe, 2004, Section 9.3.1). Under assumptions of smoothness and strong convexity – that is, the objective function can be upper and lower bounded by quadratic functions – it can be shown that gradient descent with line search will converge to an accuracy of ε in $O(\log(1/\varepsilon))$ steps. Our solver achieves the same rate guarantees for smooth functions, under essentially similar technical assumptions.

We would also like to point out connections to subgradient methods (Nedich and Bertsekas, 2000). These algorithms are designed for nonsmooth functions, and essentially choose an arbitrary element of the subgradient set to perform a gradient descent like update. Let $\max_{u \in \partial_w J(w)} \|u\| \leq G$, and $B(w^*, r)$ denote a ball of radius r centered around the minimizer of $J(w)$. By applying the analysis of Nedich and Bertsekas (2000) to the regularized risk minimization problem with $\Omega(w) := \frac{\lambda}{2}\|w\|^2$, Ratliff et al. (2007) show that subgradient descent with a fixed, but sufficiently small, stepsize will converge linearly to $B(w^*, G/\lambda)$.

Finally, several papers (Keerthi and DeCoste, 2005; Chapelle, 2007) advocate the use of Newton-like methods to solve Support Vector Machines in the "primal". However, they need to take precautions when dealing with the fact that the soft-margin type of loss functions such as the hinge loss is only piecewise differentiable. Instead, our method only requires *subdifferentials*, which always exist for convex functions, in order to make progress. The large number of and variety of implemented problems shows the flexibility of our approach.

5. Experiments

In this section, we examine the convergence behavior of BMRM and show that it is versatile enough to solve a variety of machine learning problems. All our experiments were carried out on a cluster of 24 machines each with a 2.4GHz AMD Dual Core processor and 4GB of RAM. Details of the loss functions, data sets, competing solvers and experimental objectives are described in the following subsections.

5.1 Convergence Behavior

We investigated the convergence rate of our method (Algorithm 2) empirically with respect to regularization constant λ, approximation gap ε, and bundle size k. In addition, we investigated the speedup gained by parallelizing the empirical risk computation. Finally, we examined empirically how generalization performance is related to approximation gap. For simplicity, we focused on the

training of a linear SVM with binary hinge loss:[6]

$$\min_w J(w) := \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m}\max(0, 1 - y_i \langle w, x_i \rangle). \tag{15}$$

The experiments were conducted on 6 data sets commonly used in binary classification studies, namely, adult9, astro-ph, news20-b,[7] rcv1, real-sim, and worm. adult9, news20-b, rcv1, and real-sim are available on the LIBSVM tools website.[8] astro-ph (Joachims, 2006) and worm (Franc and Sonnenburg, 2008) are available upon request from Thorsten Joachims and Soeren Sonnenburg, respectively. Table 1 summarizes the properties of the data sets.

Data Set	#examples m	dimension d	density %
adult9	48,842	123	11.27
astro-ph	94,856	99,757	0.08
news20-b	19,954	1,355,191	0.03
rcv1	677,399	47,236	0.15
real-sim	72,201	20,958	0.25
worm	1,026,036	804	25.00

Table 1: Properties of the binary classification data sets used in our experiments.

5.1.1 REGULARIZATION CONSTANT λ AND APPROXIMATION GAP ε

As suggested by the convergence analysis, the linear SVM with the nonsmooth binary hinge loss should converge in $O(\frac{1}{\lambda\varepsilon})$ iterations, where λ and ε are two parameters which one normally tunes during the model selection phase. Therefore, we investigated the scaling behavior of our method w.r.t. these two parameters. We performed the experiments with unlimited bundle size and with a heuristic that removes subgradients which remained inactive (i.e., Lagrange multiplier = 0) for 10 or more consecutive iterations.[9]

Figure 4 shows the approximation gap ε_t as a function of number of iterations t. As predicted by our convergence analysis, BMRM converges faster for larger values of λ. Furthermore, the empirical convergence curves exhibit a $O(\log(1/\varepsilon))$ rate instead of the (pessimistic) theoretical rate of $O(\frac{1}{\varepsilon})$, especially for large values of λ. Interestingly, BMRM converges faster on high-dimensional text data sets (i.e., astro-ph, news20-b, rcv1, and real-sim) than on lower dimensional data sets (i.e., adult9 and worm).

5.1.2 BUNDLE SIZE

The dual of our method (10) is a concave problem which has dimensionality equal to the number of iterations executed. In the case of linear SVM, (10) is a QP. Hence, as described in Section 3.1, we can trade potentially greater bundle improvement for memory efficiency.

6. Similar behavior was observed with other loss functions.
7. The data set is originally named news20; we renamed it to avoid confusion with the multiclass version of the data set.
8. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.
9. Note that this heuristic does not have any implication in the convergence analysis.

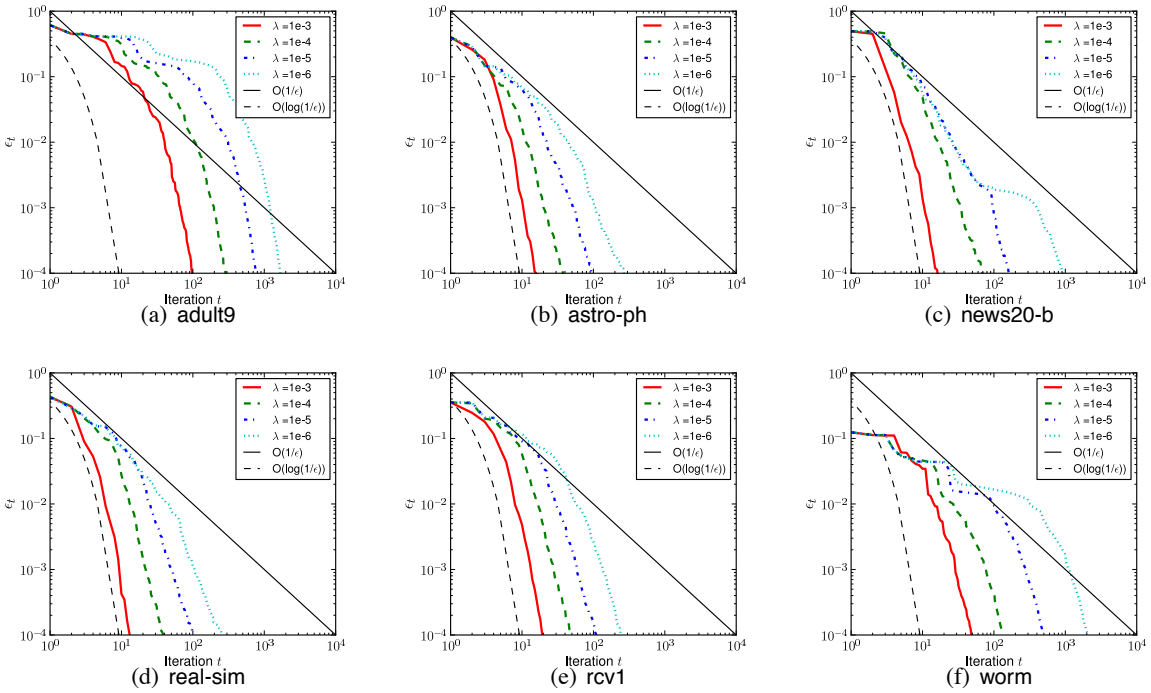


Figure 4: Approximation gap ε_t as a function of number of iterations t; for different regularization constants λ (and unlimited bundle size).

Figure 5 shows the approximation gap ε_t during the training of linear SVM as a function of the number of iterations t, for different bundle sizes $k \in \{2, 10, 50, \infty\}$. In the case of $k = \infty$, we employed the same heuristics which remove inactive linearizations as those mentioned in Section 5.1.1. As expected, the larger k is, the faster the algorithm converges. Although the case $k = 2$ is the slowest, its convergence rate is still faster than the theoretical bound $\frac{1}{\lambda\varepsilon}$.

5.1.3 PARALLELIZATION

When the empirical risk R_{emp} is additively decomposable, the loss and subgradient computation can be executed concurrently on multiple processors for different subsets of data points.[10]

We performed experiments for linear SVMs training with parallelized risk computation on the worm data set. Figure 6(a) shows the wallclock time for the overall training phase (e.g., data loading, risk computation, and solving the QP) and CPU time for just the risk computation as a function of number of processors p. Note that the gap between the two curves essentially tells the runtime upper bound of the sequential part of the algorithm. As expected, both overall and risk computation time decrease as the number of processors p increases. However, in Figure 6(b), we see two different speedups.[11] The speedup for the risk computation is roughly linear as there is no sequential part in

10. This requires only slight modification to the data loading process and the addition of some parallelization related code before and after the code segment for empirical risk computation.

11. Speedup $S_p = \frac{T_1}{T_p}$ where p is the number of processors and T_q is the runtime of the parallelized algorithm on q processors.

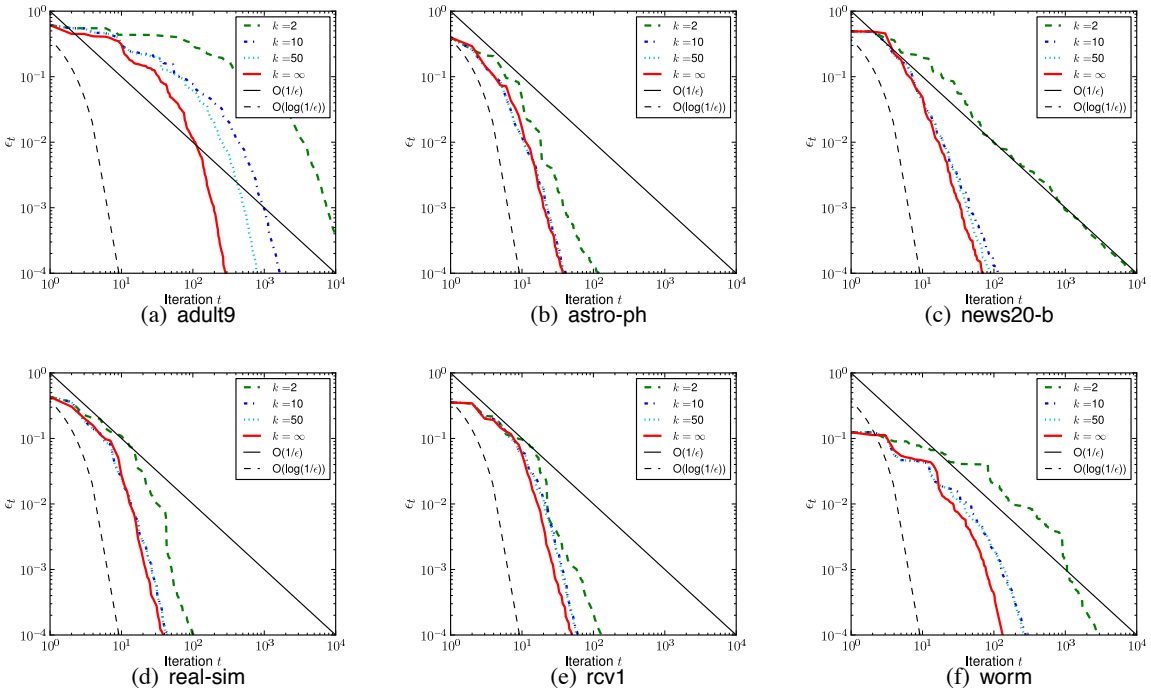


Figure 5: Approximation gap ε_t as a function of number of iterations t; for different bundle sizes k (and fixed regularization constant $\lambda = 10^{-4}$).

it; the speedup of overall computation is approaching a limit[12] as well-explained by Amdahl's law (Amdahl, 1967).

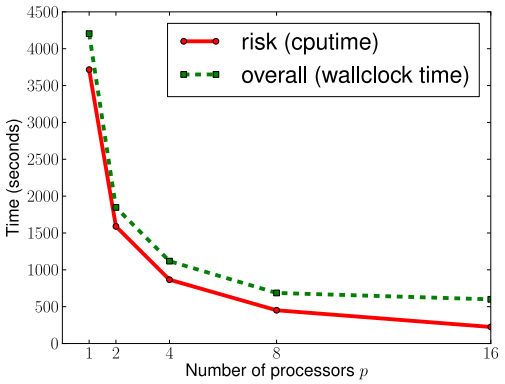
5.1.4 GENERALIZATION VERSUS APPROXIMATION GAP

Since the problems we are considering are convex, all properly convergent optimizers will converge to the same solution. Therefore, comparing generalization performance of the final solution is meaningless. But, in real life one is often interested in the speed with which the algorithm achieves good generalization performance. In this section we study this question. We focus on the generalization (in terms of accuracy) as a function of approximation gap during training. For this experiment, we randomly split each of the data sets into training (60%), validation (20%) and testing (20%) sets.

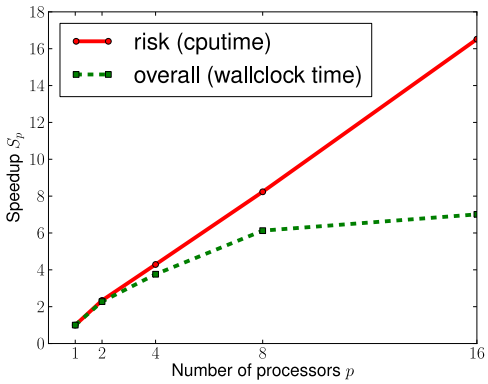
We first obtained the best $\lambda \in \{2^{-20}, \ldots, 2^{0}\}$ for each of the data sets using their corresponding validation sets. With these best λ's, we (re)trained linear SVMs and recorded the testing accuracy as well as the approximation gap at every iteration, with termination criterion $\varepsilon = 10^{-4}$. Figure 7 shows the difference between the testing accuracy evaluated at every iteration and that after training, as a function of approximation gap at each iteration.

From the figure, we see that the testing accuracies for adult9 and worm data sets are less stable in general and the approximation gap must be reduced to at least 10^{-3} to reach the 0.5% regime

12. The limit of speedup is the inverse of the sequential fraction of the algorithm such as the QP.



(a) Risk computation in CPU time (red solid line) and overall computation (i.e., data loading + risk computation + solving the QP) in wallclock time (green dashed line) as a function of number of processors.



(b) Speedup in risk computation (in CPU time) and overall computation (in wallclock time) as a function of number of processors.

Figure 6: CPU and wallclock time for training linear SVM using parallel BMRM on worm data set with varying number of processors $p \in \{1,2,4,8,16\}$. In these experiments, regularization constant $\lambda = 10^{-6}$, and termination criterion $\varepsilon = 10^{-4}$.

of the final testing accuracies; the testing accuracies for the rest of the data sets arrived at the same regime with approximation gap of 10^{-2} or lower.

In general, the generalization improved as the approximation gap decreased. The improvement in generalization became rather insignificant (say, the maximum of changes in testing accuracies is less than 0.1%) when the approximation gap was further reduced to below some effective threshold ε_{eff}; that said, it is not necessary to continue the optimization when $\varepsilon_t \leq \varepsilon_{\text{eff}}$.[13] Since ε_{eff} (or its scale) is not known *a priori* and the asymptotic analysis in Shalev-Schwartz and Srebro (2008) does not reveal the actual scale of ε_{eff} directly applicable in our case, we carried out another set of experiments to investigate if ε_{eff} could be estimated with as little effort as possible: For each data set, we randomly subsampled $10\%,\ldots,50\%$ of the training set as sub-datasets and performed the same experiment on all sub-datasets. We then determined the largest ε_{eff} such that the maximum changes in testing accuracies is less than 0.1%.

Table 5.1.4 shows the (base 10 logarithm of) ε_{eff} for all sub-datasets as well as the full data sets. It seems that the ε_{eff} estimated on a smaller sub-dataset is at most 1 order of magnitude larger than the actual ε_{eff} required on full data set. In addition, we show in the table that the necessary threshold $\varepsilon_{10\%}$ required by the sub-datasets and the full data sets to attain the final testing accuracies attained by the 10% sub-datasets. The observations obey the analysis in Shalev-Schwartz and Srebro (2008) that for a fixed testing accuracy, approximation gap (i.e., optimization error) can be relaxed when more data is given.

13. Heuristically, we could terminate the training phase following the *early stopping* strategy by monitoring the changes in accuracies on validation set evaluated in some most recent iterations.

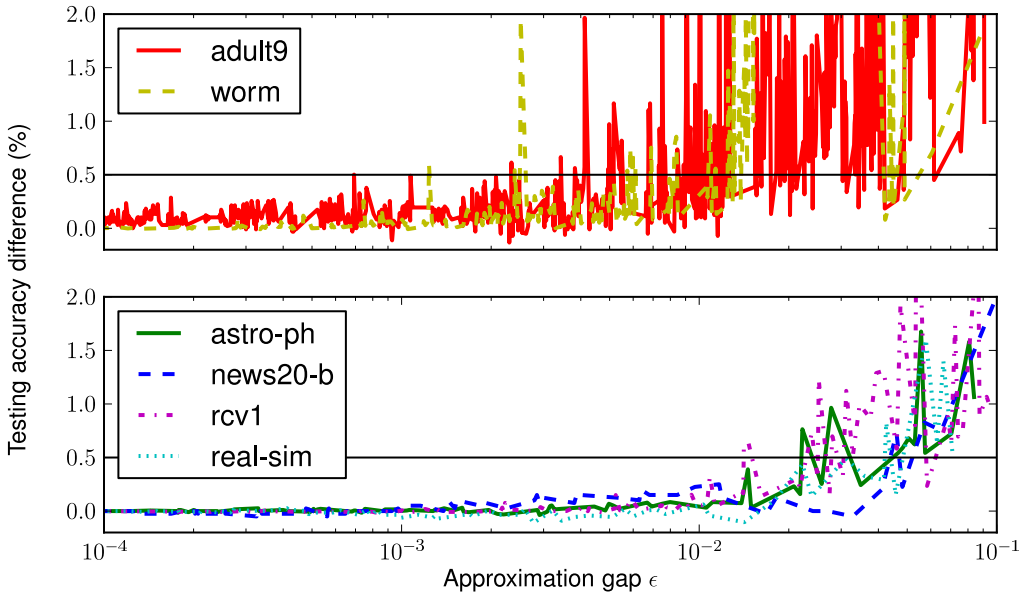


Figure 7: Difference between testing accuracies of intermediate and final models.

5.2 Comparison with Existing Bundle Methods

In this section we compare BMRM with a BT implementation obtained from Schramm and Zowe (1992).[14] We also compare the performance of BMRM (Algorithm 2) and LSBMRM (Algorithm 3). The multiclass line search used in LSBMRM can be found in Yu et al. (2008).

For binary classification, we solve the linear SVM (15) on the data sets: adult9, astro-ph, news20-b, rcv1, real-sim, and worm as mentioned in Section 5.1. For multiclass classification, we solve (Crammer and Singer, 2003):

$$\min_w J(w) := \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max_{y_i' \in [c]} \left\langle w, e_{y_i'} \otimes x_i - e_{y_i} \otimes x_i \right\rangle + \mathbf{I}(y_i \neq y_i'), \tag{16}$$

where c is the number of classes in the problem, e_i is the i-th standard basis for $\mathbb{R}^c$, $\otimes$ denotes Kronecker product; and $\mathbf{I}(\cdot)$ is an indicator function that has value 1 if its argument is evaluated true, and 0 otherwise. The data sets used in multiclass classification experiments were inex, letter, mnist, news20-m,[15] protein, and usps. inex is available for download on the website of Antoine Bordes[16] and the rest can be found on the LIBSVM tools website.[17] Table 3 summarizes the properties of these data sets.

In each of the experiments, we first obtain the optimal weight vector $\bar{w}$ by running BMRM until the termination criteria $J(w_t) - J_t(w_t) \leq 0.01 J(w_t)$ is satisfied. Then we run BT, LSBMRM, and

14. The original FORTRAN implementation was automatically converted into C for use in our library.
15. The data set is originally named news20; we renamed it to avoid confusion with the binary version of the data set.
16. Software available at `http://webia.lip6.fr/~bordes/datasets/multiclass/inex.tar.gz`.
17. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html`.

		10%	20%	30%	40%	50%	100%
adult9	Acc. (%)	84.3	84.7	84.9	85.1	85.1	85.2
	$\log_{10}\varepsilon_{\text{eff}}$	-3.90	-3.72	-3.77	-3.88	-3.64	-4.00
	$\log_{10}\varepsilon_{10\%}$	-4.01	-1.18	-1.07	-1.16	-1.27	-1.04
astro-ph	Acc. (%)	96.1	96.6	96.4	96.6	96.8	97.4
	$\log_{10}\varepsilon_{\text{eff}}$	-1.48	-1.70	-1.57	-1.49	-1.68	-1.84
	$\log_{10}\varepsilon_{10\%}$	-4.00	-1.15	-1.06	-0.98	-1.02	-0.87
news20-b	Acc. (%)	89.9	92.9	94.3	94.5	95.4	96.6
	$\log_{10}\varepsilon_{\text{eff}}$	-2.00	-2.48	-3.87	-1.65	-3.71	-2.84
	$\log_{10}\varepsilon_{10\%}$	-4.02	-0.92	-0.70	-0.80	-0.80	-0.67
rcv1	Acc. (%)	96.9	97.2	97.4	97.2	97.5	97.6
	$\log_{10}\varepsilon_{\text{eff}}$	-2.02	-2.40	-1.99	-2.16	-2.34	-2.28
	$\log_{10}\varepsilon_{10\%}$	-4.07	-1.19	-1.30	-1.29	-1.13	-1.11
real-sim	Acc. (%)	95.0	95.9	96.3	96.6	96.6	97.2
	$\log_{10}\varepsilon_{\text{eff}}$	-1.74	-1.84	-1.71	-1.99	-1.74	-1.75
	$\log_{10}\varepsilon_{10\%}$	-4.02	-1.04	-0.88	-0.87	-0.85	-0.82
worm	Acc. (%)	98.2	98.2	98.2	98.3	98.3	98.4
	$\log_{10}\varepsilon_{\text{eff}}$	-2.43	-2.47	-2.48	-3.62	-2.81	-3.55
	$\log_{10}\varepsilon_{10\%}$	-4.00	-1.38	-1.28	-1.37	-1.28	-1.31

Table 2: The first sub-row in each data set row indicates the testing accuracies of models trained on the corresponding proportions of the training set. The second sub-row indicates the (base 10 logarithm of) effective threshold such that the maximum difference in testing accuracies of models with approximation gap smaller than that is less than 0.1%. The third sub-row indicates the (base 10 logarithm of) threshold necessary for models to attain the testing accuracy attained by the model trained on the 10% sub-dataset with default $\varepsilon = 10^{-4}$.

Data Set	#examples m	#classes c	dimension d	density %
inex	12,107	18	167,295	0.48
letter	20,000	26	16	100.00
mnist	70,000	10	780	19.24
news20-m	19,928	20	62,061	0.13
protein	21,516	3	357	28.31
usps	9,298	10	256	96.70

Table 3: Properties of the multiclass classification data sets used in the experiments.

BMRM until the following termination criteria is satisfied:

$$J(w_t) - J(\bar{w}) \leq 0.01 J(\bar{w}). \tag{17}$$

Figure 8 shows the number of iterations t required by the three methods on each data set to satisfy (17) as a function of regularization constant $\lambda \in \left\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\right\}$. As expected, LSBMRM, which uses an exact line search, outperformed both BMRM and BT on all data sets. BMRM performed better than BT on all high dimensional data sets except news20-m but worse on the rest.

Although BT tunes the stabilization trade-off parameter κ_t automatically, it still does not guarantee superiority over BMRM which is considerably simpler. Nevertheless, external stabilization (in BT) clearly helps speed up the convergence in certain cases.

5.3 Versatility

In the following subsections, we will illustrate some of the applications of BMRM to various machine learning problems with smooth and non-differentiable loss functions, and with different regularizers. Our aim is to show that BMRM is versatile enough to be used in a variety of seemingly different problems. Readers not interested in this aspect of BMRM can safely skip this subsection.

5.3.1 BINARY CLASSIFICATION

In this section, we evaluate the performance of our method BMRM in the training of binary classifier using linear SVMs (15) and logistic loss:

$$\min_w J(w) := \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} \log(1+\exp(-y_i \langle w, x_i \rangle)),$$

on the binary classification data sets mentioned in Section 5.1 with split similar to that in Section 5.1.4. Since we will compare BMRM with other solvers which use different termination criteria, we consider the CPU time used in reducing the relative difference between the current smallest objective function value and the optimum:

$$\frac{\min_{i \le t} J(w_i) - J(w^*)}{J(w^*)},$$

where w_i is the weight vector at time/iteration i, and w^* is the minimizer obtained by running BMRM until the approximation gap $\varepsilon_t < 10^{-4}$. The best $\lambda \in \{2^{-20}, \ldots, 2^0\}$ for each of the data sets was determined by evaluating the performance on the corresponding validation set.[18]

In the case of linear SVMs, we compared BMRM to three publicly available state of the art *batch learning* solvers:

1. OCAS (Franc and Sonnenburg, 2008). Since this method is equivalent to LSBMRM with binary hinge loss, we refer to this software by LSBMRM for naming consistency.
2. LIBLINEAR (Fan et al., 2008) version 1.33 with option "-s 3".
3. SVM$^{\text{perf}}$ (Joachims, 2006) version 2.5 with option "-w 3" and with double precision floating point numbers.

LIBLINEAR solves the dual problem of linear SVM using a coordinate descent method (Hsieh et al., 2008). SVM$^{\text{perf}}$ was chosen for comparison as it is algorithmically identical to BMRM in this case. Both LIBLINEAR and SVM$^{\text{perf}}$ provide a "shrinking" technique to speed up the algorithms by ignoring some data points which are not likely to affect the objective. Since BMRM does not provide such shrinking technique, we excluded this option in both LIBLINEAR and SVM$^{\text{perf}}$ for a fair comparison.

Figure 9 shows the relative difference in objective value as a function of training time (CPU seconds) for three methods on various data sets. BMRM is faster than SVM$^{\text{perf}}$ on all data sets

18. The corresponding penalty parameter C for LIBLINEAR and OCAS is $1/(m\lambda)$, and for SVM$^{\text{perf}}$ is $1/(100\lambda)$.

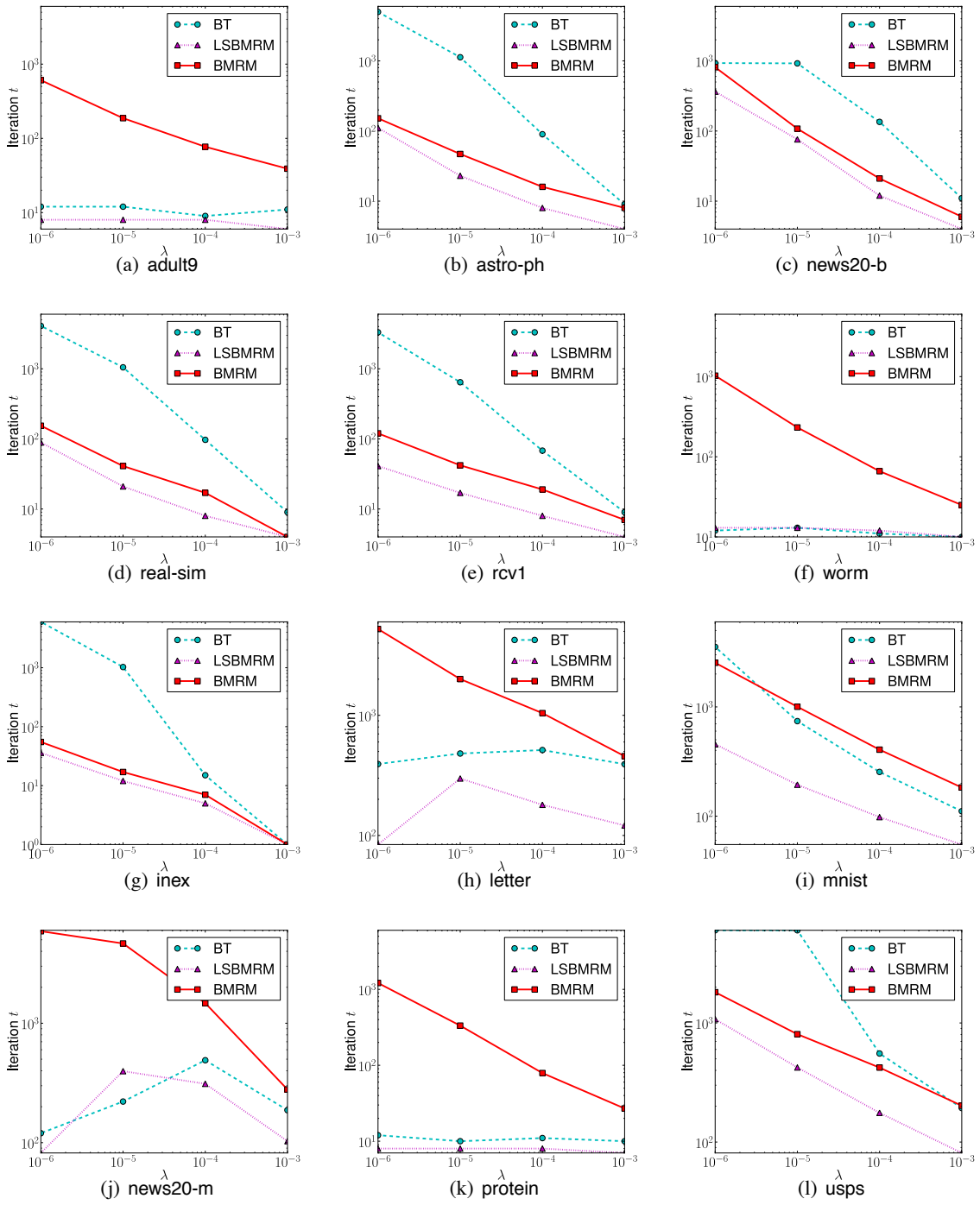


Figure 8: Smallest number of iterations required to satisfy the termination criterion (17) for each data set and various regularization constants. (BT did not satisfy (17) in the inex and usps experiments for $\lambda = 10^{-6}$ after 6000 iterations.)

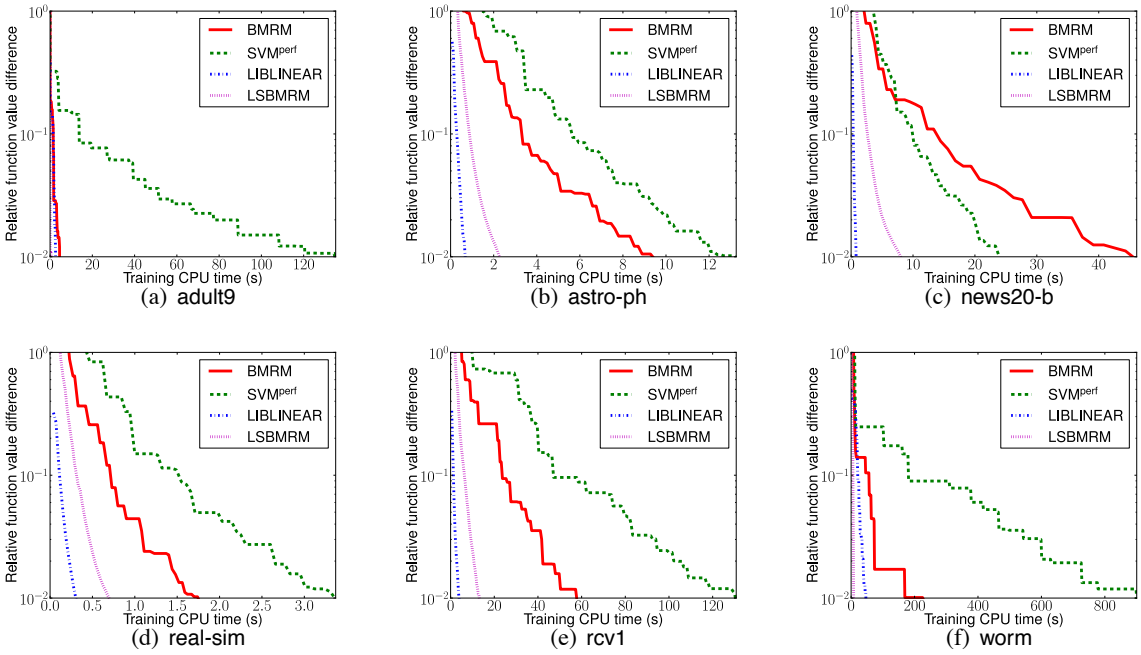


Figure 9: Linear SVMs. Relative primal objective value difference during training.

except news20-b. The performance difference observed here is largely due to the differences in the implementations (e.g., feature vector representation, QP solver, etc.). Nevertheless, both BMRM and SVM^{perf} are significantly outperformed by LSBMRM and LIBLINEAR on all data sets, and LIBLINEAR is almost always faster than LSBMRM. It is clear from the figure that LSBMRM and LIBLINEAR enjoy progression with "strictly" decreasing objective values; whereas the progress of both BMRM and SVM^{perf} are hindered by the "stalling" steps (i.e., the flat line segments in the plots). The fact that LSBMRM is different from BMRM and SVM^{perf} by one additional line search step implies that the "stalling" steps is the time that BMRM and SVM^{perf} improve the approximation at the regions which do not help reducing the primal objective function value.

In the case of logistic regression, we compare BMRM to the state of the art trust region Newton method for logistic regression (Lin et al., 2008) which is also available in the LIBLINEAR package (option "-s 0"). From Figure 10, we see that LIBLINEAR outperforms BMRM on all data sets and that BMRM suffers from the same "stalling" phenomenon as observed in the linear SVMs case.

5.3.2 LEARNING THE COST MATRIX FOR GRAPH MATCHING

In computer vision, there are problems which require matching the objects of interest in a pair of images. These problems are often modeled as attributed graph matching problems where the (extracted) landmark points x_i in the first image x must be matched to the corresponding points $x'_{i'}$ in the second image x'. Note that we represent the *point* x_i or $x'_{i'}$ as d-dimensional feature vectors. The attributed graph matching problem is then cast as a *Linear Assignment Problem* (LAP) which

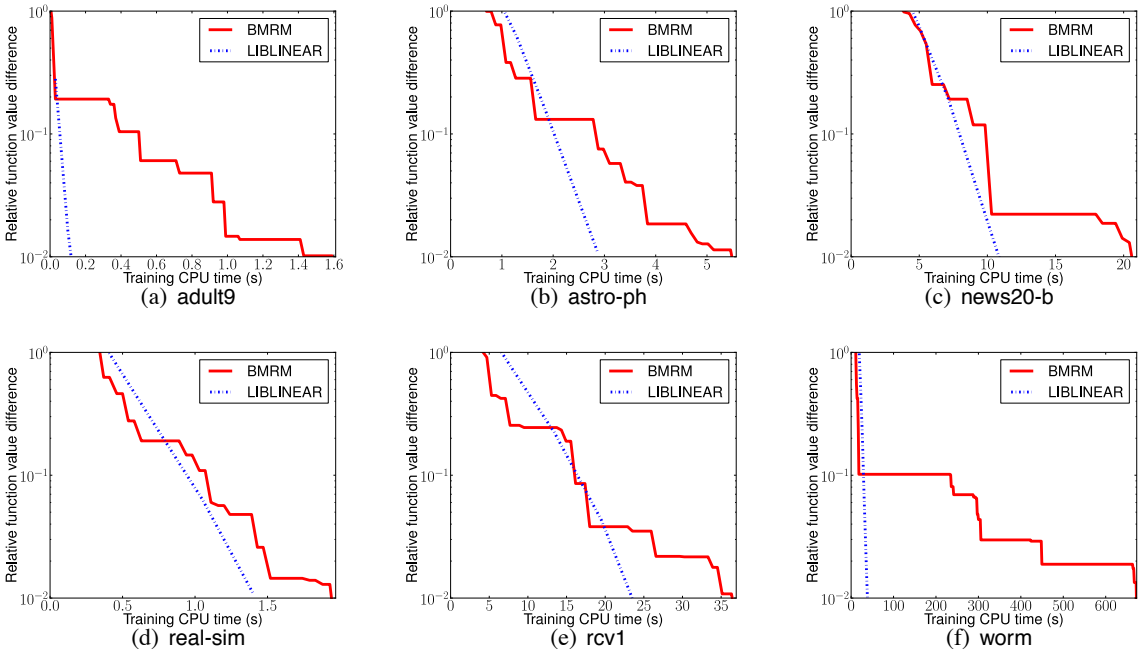


Figure 10: Logistic regression. Relative primal objective value difference during training.

can be solved in worst case $O(n^3)$ time where n is the number of landmark points (Kuhn, 1955).[19] Formally, the LAP reads

$$\max_{y \in \mathcal{Y}} \sum_{i=1}^{n} \sum_{i'=1}^{n} y_{ii'} C_{ii'},$$

where $\mathcal{Y}$ is the set of all $n \times n$ permutation matrices, and $C_{ii'}$ is the cost of matching point x_i to point $x'_{i'}$. In the standard setting of graph matching, one way to determine the cost matrix C is as

$$C_{ii'} := -\sum_{k=1}^{d} \left| x_i^{(k)} - x_{i'}^{\prime(k)} \right|^2.$$

Instead of finding more features to describe the points x_i and $x'_{i'}$ that might improve the matching results, Caetano et al. (2007) propose to learn a weighting to a given set of features that actually improved the matching results in many cases (Caetano et al., 2008).

In Caetano et al. (2007, 2008) the problem of learning the cost matrix for graph matching is formulated as a L_2 regularized risk minimization with loss function

$$l(x, x', y, w) = \max_{\bar{y} \in \mathcal{Y}} \left\langle w, \phi(x, x', \bar{y}) - \phi(x, x', y) \right\rangle + \Delta(\bar{y}, y), \tag{18}$$

where the feature map ϕ is defined as

$$\phi(x, x', y) = -\sum_{i=1}^{n} \sum_{i'=1}^{n} y_{ii'} (|x_i^{(1)} - x_{i'}^{\prime(1)}|^2, \ldots, |x_i^{(d)} - x_{i'}^{\prime(d)}|^2), \tag{19}$$

19. To achieve better matching results, one could further enforce edge-to-edge matching where edge refers to pair of landmark points. This additional matching requirement renders the problem as a *Quadratic Assignment Problem*.

and the label loss Δ is defined as the normalized Hamming loss

$$\Delta(\bar{y}, y) = 1 - \frac{1}{n}\sum_{i=1}^{n}\sum_{i'=1}^{n}\bar{y}_{ii'}y_{ii'}. \tag{20}$$

By (19) and (20), the argument of (18) becomes

$$\left\langle w, \phi(x,x',\bar{y}) - \phi(x,x',y)\right\rangle + \Delta(\bar{y},y) = \sum_{i=1}^{n}\sum_{i'=1}^{n}\bar{y}_{ii'}\tilde{C}_{ii'} + \text{constant},$$

where $\tilde{C}_{ii'} = -\sum_{k=1}^{d} w_k |x_i^{(k)} - x_{i'}^{\prime(k)}|^2 - y_{ii'}/n$. Therefore, (18) is exactly a LAP. We refer interested readers to Caetano et al. (2007, 2008) for more detailed exposition especially on the use of edge matching (in addition to point matching) which leads to much better performance.

We reproduced the experiment in Caetano et al. (2008) that used BMRM with L_2 regularization on the CMU house data set.[20] For this data set, there are 30 hand-labeled landmark points in each image and the features for those points are the 60-dimensional Shape Context features (Belongie et al., 2001; Caetano et al., 2008). The experiments evaluated the performance of the method for training/validation/testing pairs fixed at baselines (separation of frames) $0, 10, \ldots, 90$. Additionally, we ran the same set of experiments with L_1 regularization, that is, $\Omega(w) = \|w\|_1$.[21] The matching performance of the cost matrices augmented with learned weight vectors w's are compared with the original non-learning cost matrix, that is, with uniform weight vector $w = (1, \ldots, 1)$.

Figure 11 shows the results of the experiments. On the left, we see that the matching performance with learned cost matrices are getting more superior to that of non-learning as the baseline increases. The performance of L_1 and L_2 regularized learning are quite similar on average. On the right are the best learned weights for the features using L_1 regularization (top) and L_2 regularization (bottom) for baseline 50. The weights due to L_1 regularization is considerably sparser (i.e., 42 non-zeros) than that due to L_2 regularization (i.e., 52 non-zeros).

5.3.3 HUMAN ACTION SEGMENTATION AND RECOGNITION

In this section, we consider the problem of joint segmentation and recognition of human action from a video sequence using the discriminative Semi-Markov Models (SMM) proposed by Shi et al. (2008). Denote by $x = \{x_i\}_{i=1}^{n} \in \mathcal{X}$ a sequence of n video frames, and by $y = \{(s_i, c_i)\}_{i=1}^{\bar{n}} \in \mathcal{Y}$ the corresponding segment labeling where s_i is the starting location of the i-th segment which ends at $s_{i+1} - 1$, c_i is the frame label for all frames in the segment, and $\bar{n} \leq n$ the number of segments. For ease of presentation, we append a dummy video frame x_{n+1} to x and a dummy segment label $(s_{\bar{n}+1}, c_{\bar{n}+1})$ to y to mark x_{n+1} as the last segment.

In SMM, there exists a *segment* variable for each possible segment (i.e., multiple frames) of x that model the frame label and the boundaries (or length) of a segment jointly; these *segment* variables form a Markov Chain. On the contrary, the Hidden Markov Model (HMM) for the same video x has one *frame label* variable y_i for each video frame x_i. The fact that SMM models multiple frames as one variable allows one to exploit the *structure* and *information* in the problem more efficiently than in HMM. The *structure* exploitation is due to the fact that one human action usually

20. This data set consist of a sequence of 111 images of a toy house. Available at `http://vasc.ri.cmu.edu/idb/html/motion/house/index.html`.

21. Further description on L_1 regularized BMRM can be found in Appendix C.

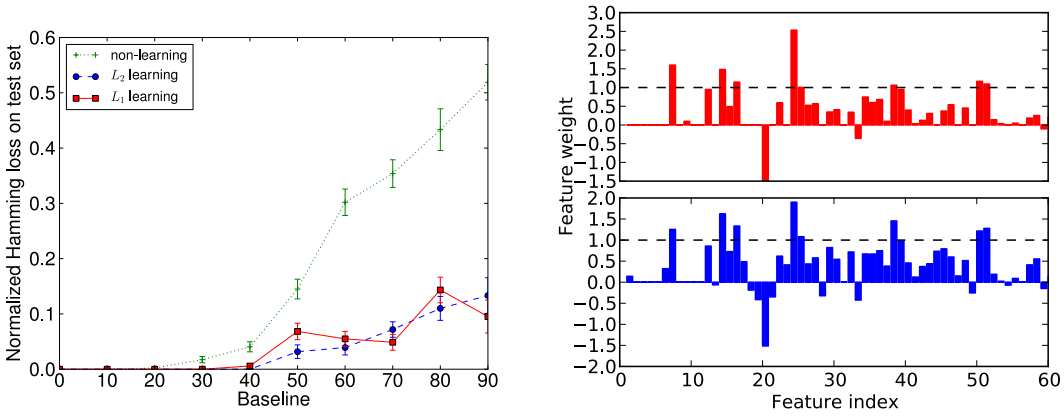


Figure 11: **Left:** Performance on the house data set as the baseline varies. For each baseline, the minimizer of validation loss is evaluated on all testing examples. The corresponding mean normalized Hamming losses (as points) and its standard errors (as error bars) are reported. **Right:** Feature weights for best models trained with L_1 regularization (top) and L_2 regularization (bottom) for baseline 50. Dashed lines indicate the feature weight value 1.

spans several consecutive frames, and the *information* exploitation is due to the possibility to extract features which only become apparent within a segment of several frames.

The discriminative SMM in Shi et al. (2008) is formulated as a regularized risk minimization problem where the loss function is

$$l(x,y,w) = \max_{\bar{y}\in\mathcal{Y}} \langle w, \phi(x,\bar{y}) - \phi(x,y)\rangle + \Delta(\bar{y},y). \tag{21}$$

The feature map ϕ is defined as

$$\phi(x,y) = \left(\sum_{i=1}^{\bar{n}} \phi_1(x,s_i,c_i), \sum_{i=1}^{\bar{n}} \phi_2(x,s_i,s_{i+1},c_i), \sum_{i=1}^{\bar{n}} \phi_3(x,s_i,s_{i+1},c_i,c_{i+1})\right),$$

where ϕ_1, ϕ_2, and ϕ_3 are some feature functions for the segment boundaries, segments, and adjacent segments, respectively. Let y_i be the frame label for x_i according to segment labeling y, the label loss function Δ is defined as

$$\Delta(\bar{y},y) = \sum_{i=1}^{n} \mathbf{I}(\bar{y}_i \neq y_i), \tag{22}$$

where $\mathbf{I}(\cdot)$ is an indicator function as defined in (16). We refer interested readers to Shi et al. (2008) for more details on the features and the dynamic programming to compute (21) and its subgradient.

We followed the experimental setup of Shi et al. (2008) by running BMRM for this problem with L_2 (i.e., $\Omega(w) = \frac{1}{2}\|w\|^2$) and L_1 (i.e., $\Omega(w) = \|w\|_1$) regularization, on the Walk-Bend-Draw (WBD) data sets (Shi et al., 2008) which consists of 18 video sequences with 3 human action classes (i.e.,

walking, *bending*, *drawing*). For this data set, the dimensionality of the image of the feature map ϕ is $d = 9917$.

Table 4 shows the 6 fold cross validation results for our methods (L_1 and L_2 SMM),[22] SVMs and SVM-HMM (Tsochantaridis et al., 2005). The latter two are adopted from Shi et al. (2008). SMM outperforms SVM-HMM and SVM as reported in Shi et al. (2008). Amongst L_1 and L_2 SMMs, the latter performs the best and converged to optimal. Although L_1 SMM failed to satisfy the termination criterion, the performance is comparable to that of L_2 SMM even with a 40 times sparser weight vector (see Figure 12 for the feature weights distributions of L_1 and L_2 SMMs).

Methods	CV mean (std. err.)	#iter.	CPU seconds	nnz(w)
L_2 SMM	0.954 (0.006)	231	1129	3690
L_1 SMM	0.930 (0.010)	500	2659	84
SVM-HMM	0.870 (0.020)	–	–	–
SVM	0.840 (0.030)	–	–	–

Table 4: Experimental results on WBD data set. The second column indicates the mean and standard error of the test accuracy (22). The third and fourth columns indicate the number of iterations and CPU seconds for the training of the final model with the best parameter, and the last column indicates the number of nonzero in the final weight vector w.

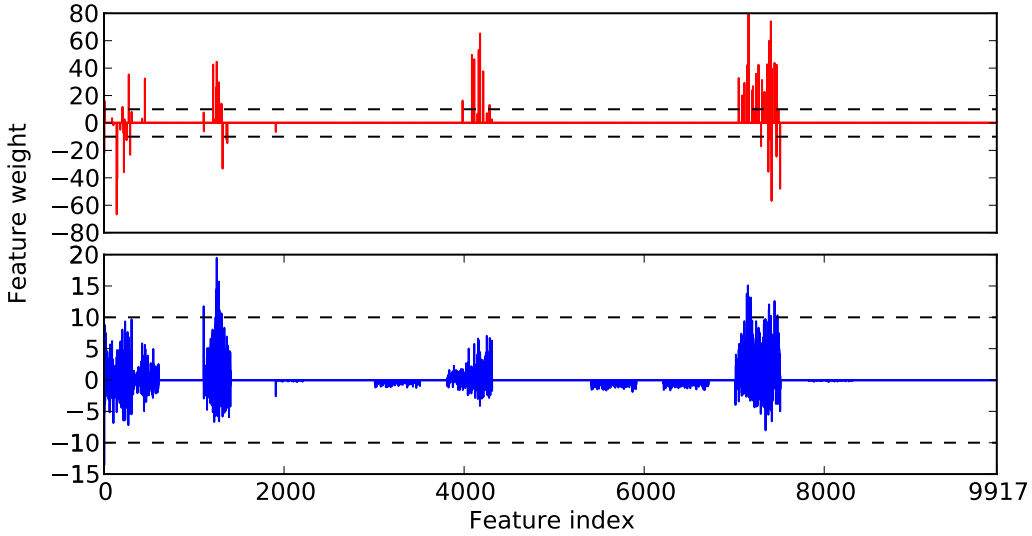


Figure 12: Feature weights for best models trained with L_1 regularization (top) and L_2 regularization (bottom). Dashed lines indicate the feature weight range $[\pm 10]$.

22. We set termination criterion $\varepsilon = 10^{-4}$ and limited the maximum number of iteration to 500.

6. Discussion and Conclusion

The experiments presented in the paper indicate that BMRM is suitable for a wide variety of machine learning problems. In fact, the *modularity* of BMRM not only brings the benefits of parallel and distributed computation but also makes BMRM a natural test bed for trying out new models/ideas on any particular problem with less effort, that is, the user is only required to implement the loss functions and/or regularizers corresponding to different models/ideas.

Nevertheless, we saw in the experiments that BMRM does not guarantee strict improvement in the primal when the dual is solved instead. This phenomenon could significantly hinder the performance of BMRM as seen in some of the experiments. Since efficient line search procedure may not exist for general structured prediction tasks, the *trust region* philosophy used in BT could be a potential strategy to alleviate this problem; we leave this to the future work. We also note that for computationally expensive nonsmooth loss functions, one way to make fuller use of each loss function evaluation is by updating the model R_t^{CP} with two or more linearizations at a non-diffferentiable location (Frangioni, 1997).

In conclusion, we have presented a variant of standard bundle methods, that is, BMRM, which is algorithmically simpler and, in some senses, more straightforward for regularized risk minimization problems than the standard bundle methods. We also showed a $O(1/\varepsilon)$ rate of convergence for nonsmooth objective functions and $O(\log(1/\varepsilon))$ rates for smooth objective functions.

Acknowledgments

NICTA is funded by the Australian Government's Backing Australia's Ability and the Center of Excellence programs. This work is also supported by the IST Program of the European Community, under the FP7 Network of Excellence, ICT-216886-NOE. We thank Jin Yu for many useful discussions and help with implementation issues. Julian McAuley and Qinfeng Shi provided their source code, facilitating some of the experimental comparisons reported in Sections 5.3.2 and 5.3.3. We thank Manfred Warmuth for pointing out Lemma 13. We also thank Sam Roweis, Yasemin Altun, and Nic Schraudolph for insightful discussions. A short, early version of this work appeared in Teo et al. (2007), while the convergence proofs first appeared in Smola et al. (2007).

Appendix A. Loss Functions

A multitude of loss functions are commonly used to derive seemingly different algorithms. This often blurs the similarities as well as subtle differences between them, often for historic reasons: Each new loss is typically accompanied by at least one publication dedicated to it. In many cases, the loss is not spelled out explicitly either but instead, it is only given by means of a constrained optimization problem. A case in point are the papers introducing (binary) hinge loss (Bennett and Mangasarian, 1992; Cortes and Vapnik, 1995) and structured loss (Taskar et al., 2004; Tsochantaridis et al., 2005). Likewise, a geometric description obscures the underlying loss function, as in novelty detection (Schölkopf et al., 2001).

In this section we give an expository yet unifying presentation of many of those loss functions. Many of them are well known, while others, such as multivariate ranking, hazard regression, or Poisson regression are not commonly used in machine learning. Tables 5 and 6 contain a choice subset of simple scalar and vectorial losses. Our aim is to put the multitude of loss functions in

Table 5: Scalar loss functions and their derivatives, depending on $f := \langle w,x \rangle$, and y.

	Loss $\bar{l}(f,y)$	Derivative $\bar{l}'(f,y)$
Hinge (Bennett and Mangasarian, 1992)	$\max(0, 1-yf)$	0 if $yf \geq 1$ and $-y$ otherwise
Squared Hinge (Keerthi and DeCoste, 2005)	$\frac{1}{2}\max(0, 1-yf)^2$	0 if $yf \geq 1$ and $f-y$ otherwise
Exponential (Cowell et al., 1999)	$\exp(-yf)$	$-y\exp(-yf)$
Logistic (Collins et al., 2000)	$\log(1+\exp(-yf))$	$-y/(1+\exp(-yf))$
Novelty (Schölkopf et al., 2001)	$\max(0, \rho - f)$	0 if $f \geq \rho$ and -1 otherwise
Least mean squares (Williams, 1998)	$\frac{1}{2}(f-y)^2$	$f-y$
Least absolute deviation	$\lvert f-y \rvert$	$\operatorname{sgn}(f-y)$
Quantile regression (Koenker, 2005)	$\max(\tau(f-y), (1-\tau)(y-f))$	τ if $f > y$ and $\tau - 1$ otherwise
ε-insensitive (Vapnik et al., 1997)	$\max(0, \lvert f-y \rvert - \varepsilon)$	0 if $\lvert f-y \rvert \leq \varepsilon$, else $\operatorname{sgn}(f-y)$
Huber's robust loss (Müller et al., 1997)	$\frac{1}{2}(f-y)^2$ if $\lvert f-y \rvert \leq 1$, else $\lvert f-y \rvert - \frac{1}{2}$	$f-y$ if $\lvert f-y \rvert \leq 1$, else $\operatorname{sgn}(f-y)$
Poisson regression (Cressie, 1993)	$\exp(f) - yf$	$\exp(f) - y$

Table 6: Vectorial loss functions and their derivatives, depending on the vector $f := Wx$ and on y.

	Loss	Derivative
Soft-Margin Multiclass (Taskar et al., 2004) (Crammer and Singer, 2003)	$\max_{y'}(f_{y'} - f_y + \Delta(y,y'))$	$e_{y^*} - e_y$ where y^* is the argmax of the loss
Scaled Soft-Margin Multiclass (Tsochantaridis et al., 2005)	$\max_{y'} \Gamma(y,y')(f_{y'} - f_y + \Delta(y,y'))$	$\Gamma(y,y')(e_{y^*} - e_y)$ where y^* is the argmax of the loss
Softmax Multiclass (Cowell et al., 1999)	$\log \sum_{y'} \exp(f_{y'}) - f_y$	$\left[\sum_{y'} e_{y'} \exp(f'_y)\right] / \sum_{y'} \exp(f'_y) - e_y$
Multivariate Regression	$\frac{1}{2}(f-y)^\top M(f-y)$ where $M \succeq 0$	$M(f-y)$

an unified framework, and to show how these losses and their (sub)gradients can be computed efficiently for use in our solver framework.

Note that not all losses, while convex, are continuously differentiable. In this situation we give *a* subgradient. While this may not be optimal, the convergence rates of our algorithm do not depend on which element of the subdifferential we provide: in all cases the first order Taylor approximation is a lower bound which is tight at the point of expansion.

In this setion, with little abuse of notation, v_i is understood as the i-th component of vector v when v is clearly not an element of a sequence or a set.

A.1 Scalar Loss Functions

It is well known (Wahba, 1997) that the convex optimization problem

$$\min_{\xi} \xi \text{ subject to } y\langle w,x\rangle \geq 1-\xi \text{ and } \xi \geq 0$$

takes on the value $\max(0,1-y\langle w,x\rangle)$. The latter is a convex function in w and x. Likewise, we may rewrite the ε-insensitive loss, Huber's robust loss, the quantile regression loss, and the novelty detection loss in terms of loss functions rather than a constrained optimization problem. In all cases, $\langle w,x\rangle$ will play a key role insofar as the loss is convex in terms of the *scalar* quantity $\langle w,x\rangle$. A large number of loss functions fall into this category, as described in Table 5. Note that not all functions of this type are continuously differentiable. In this case we adopt the convention that

$$\partial_x \max(f(x),g(x)) = \begin{cases} \partial_x f(x) & \text{if } f(x) \geq g(x) \\ \partial_x g(x) & \text{otherwise .} \end{cases}$$

Since we are only interested in obtaining an arbitrary element of the subdifferential this convention is consistent with our requirements.

Let us discuss the issue of efficient computation. For all scalar losses we may write $l(x,y,w) = \bar{l}(\langle w,x\rangle, y)$, as described in Table 5. In this case a simple application of the chain rule yields that $\partial_w l(x,y,w) = \bar{l}'(\langle w,x\rangle, y)\cdot x$. For instance, for squared loss we have

$$\bar{l}(\langle w,x\rangle, y) = \tfrac{1}{2}(\langle w,x\rangle - y)^2 \text{ and } \bar{l}'(\langle w,x\rangle, y) = \langle w,x\rangle - y.$$

Consequently, the derivative of the empirical risk term is given by

$$\partial_w R_{\text{emp}}(w) = \frac{1}{m}\sum_{i=1}^{m} \bar{l}'(\langle w,x_i\rangle, y_i)\cdot x_i.$$

This means that if we want to compute l and $\partial_w l$ on a large number of observations x_i, represented as matrix X, we can make use of fast linear algebra routines to pre-compute the vectors

$$f = Xw \text{ and } g^\top X \text{ where } g_i = \bar{l}'(f_i, y_i).$$

This is possible for any of the loss functions listed in Table 5, and many other similar losses. The advantage of this unified representation is that implementation of each individual loss can be done in very little time. The computational infrastructure for computing Xw and $g^\top X$ is shared. Evaluating $\bar{l}(f_i,y_i)$ and $\bar{l}'(f_i,y_i)$ for all i can be done in $O(m)$ time and it is not time-critical in comparison to the remaining operations. Algorithm 6 describes the details.

Algorithm 6 ScalarLoss(w, X, y)

1: **input:** Weight vector w, feature matrix X, and labels y
2: Compute $f = Xw$
3: Compute $r = \sum_i \bar{l}(f_i, y_i)$ and $g = \bar{l}'(f, y)$
4: $g \leftarrow g^\top X$
5: **return** Risk r and gradient g

An important but often neglected issue is worth mentioning. Computing f requires us to *right* multiply the matrix X with the vector w while computing g requires the *left* multiplication of X with the vector $g^\top$. If X is stored in a row major format then Xw can be computed rather efficiently while $g^\top X$ is expensive. This is particularly true if X cannot fit in main memory. Converse is the case when X is stored in column major format. Similar problems are encountered when X is a sparse matrix and stored in either compressed row format or in compressed column format.

A.2 Structured Loss

In recent years structured estimation has gained substantial popularity in machine learning (Tsochantaridis et al., 2005; Taskar et al., 2004; Bakir et al., 2007). At its core it relies on two types of convex loss functions: logistic loss:

$$l(x, y, w) = \log \sum_{y' \in \mathcal{Y}} \exp\left(\left\langle w, \phi(x, y')\right\rangle\right) - \left\langle w, \phi(x, y)\right\rangle, \tag{23}$$

and soft-margin loss:

$$l(x, y, w) = \max_{y' \in \mathcal{Y}} \Gamma(y, y') \left\langle w, \phi(x, y') - \phi(x, y)\right\rangle + \Delta(y, y'). \tag{24}$$

Here $\phi(x, y)$ is a *joint* feature map, $\Delta(y, y') \geq 0$ describes the cost of misclassifying y by y', and $\Gamma(y, y') \geq 0$ is a scaling term which indicates by how much the large margin property should be enforced. For instance, Taskar et al. (2004) choose $\Gamma(y, y') = 1$. On the other hand, Tsochantaridis et al. (2005) suggest $\Gamma(y, y') = \Delta(y, y')$, which reportedly yields better performance. Finally, McAllester (2007) recently suggested generic functions $\Gamma(y, y')$.

The logistic loss can also be interpreted as the negative log-likelihood of a conditional exponential family model:

$$p(y|x; w) := \exp(\langle w, \phi(x, y)\rangle - g(w|x)), \tag{25}$$

where the normalizing constant $g(w|x)$, often called the log-partition function, reads

$$g(w|x) := \log \sum_{y' \in \mathcal{Y}} \exp\left(\left\langle w, \phi(x, y')\right\rangle\right).$$

As a consequence of the Hammersley-Clifford theorem (Jordan, 2002) every exponential family distribution corresponds to a undirected graphical model. In our case this implies that the labels y factorize according to an undirected graphical model. A large number of problems have been addressed by this setting, amongst them named entity tagging (Lafferty et al., 2001), sequence alignment (Tsochantaridis et al., 2005), segmentation (Rätsch et al., 2007) and path planning (Ratliff

et al., 2006). It is clearly impossible to give examples of all settings in this section, nor would a brief summary do this field any justice. We therefore refer the reader to the edited volume by Bakir et al. (2007) and the references therein.

If the underlying graphical model is tractable then efficient inference algorithms based on dynamic programming can be used to compute (23) and (24). We discuss intractable graphical models in Section A.2.1, and now turn our attention to the derivatives of the above structured losses.

When it comes to computing derivatives of the logistic loss, (23), we have

$$\begin{aligned}\partial_w l(x,y,w) &= \frac{\sum_{y'} \phi(x,y') \exp\langle w, \phi(x,y')\rangle}{\sum_{y'} \exp\langle w, \phi(x,y')\rangle} - \phi(x,y) \\ &= \mathbf{E}_{y' \sim p(y'|x)}\left[\phi(x,y')\right] - \phi(x,y).\end{aligned}$$

where $p(y|x)$ is the exponential family model (25). In the case of (24) we denote by $\bar{y}(x)$ the argmax of the RHS, that is

$$\bar{y}(x) := \operatorname*{argmax}_{y'} \Gamma(y,y') \left\langle w, \phi(x,y') - \phi(x,y)\right\rangle + \Delta(y,y').$$

This allows us to compute the derivative of $l(x,y,w)$ as

$$\partial_w l(x,y,w) = \Gamma(y,\bar{y}(x))\left[\phi(x,\bar{y}(x)) - \phi(x,y)\right].$$

In the case where the loss is maximized for more than one distinct value $\bar{y}(x)$ we may average over the individual values, since any convex combination of such terms lies in the subdifferential.

Note that (24) majorizes $\Delta(y,y^*)$, where $y^* := \operatorname{argmax}_{y'} \langle w, \phi(x,y')\rangle$ (Tsochantaridis et al., 2005). This can be seen via the following series of inequalities:

$$\Delta(y,y^*) \leq \Gamma(y,y^*)\langle w, \phi(x,y^*) - \phi(x,y)\rangle + \Delta(y,y^*) \leq l(x,y,w).$$

The first inequality follows because $\Gamma(y,y^*) \geq 0$ and y^* maximizes $\langle w, \phi(x,y')\rangle$ thus implying that $\Gamma(y,y^*)\langle w, \phi(x,y^*) - \phi(x,y)\rangle \geq 0$. The second inequality follows by definition of the loss.

We conclude this section with a simple lemma which is at the heart of several derivations of Joachims (2005). While the proof in the original paper is far from trivial, it is straightforward in our setting:

Lemma 8 *Denote by $\delta(y,y')$ a loss and let $\phi(x_i,y_i)$ be a feature map for observations (x_i,y_i) with $1 \leq i \leq m$. Moreover, denote by X,Y the set of all m patterns and labels respectively. Finally let*

$$\Phi(X,Y) := \sum_{i=1}^m \phi(x_i,y_i) \text{ and } \Delta(Y,Y') := \sum_{i=1}^m \delta(y_i,y_i').$$

Then the following two losses are equivalent:

$$\sum_{i=1}^m \max_{y'} \left\langle w, \phi(x_i,y') - \phi(x_i,y_i)\right\rangle + \delta(y_i,y') \text{ and } \max_{Y'} \left\langle w, \Phi(X,Y') - \Phi(X,Y)\right\rangle + \Delta(Y,Y').$$

This is immediately obvious, since both feature map and loss decompose, which allows us to perform maximization over Y' by maximizing each of its m components. In doing so, we showed that aggregating all data and labels into a single feature map and loss yields results identical to minimizing the sum over all individual losses. This holds, in particular, for the sample error loss of Joachims (2005). Also note that this equivalence does *not* hold whenever $\Gamma(y,y')$ is not constant.

A.2.1 INTRACTABLE MODELS

We now discuss cases where computing $l(x,y,w)$ itself is too expensive. For instance, for intractable graphical models, the computation of $\sum_y \exp\langle w,\phi(x,y)\rangle$ cannot be computed efficiently. Wainwright and Jordan (2003) propose the use of a convex majorization of the log-partition function in those cases. In our setting this means that instead of dealing with

$$l(x,y,w) = g(w|x) - \langle w,\phi(x,y)\rangle \text{ where } g(w|x) := \log \sum_y \exp\langle w,\phi(x,y)\rangle$$

one uses a more easily computable convex upper bound on g via

$$\sup_{\mu\in \mathrm{MARG}(x)} \langle w,\mu\rangle + H_{\mathrm{Gauss}}(\mu|x). \tag{26}$$

Here $\mathrm{MARG}(x)$ is an outer bound on the conditional marginal polytope associated with the map $\phi(x,y)$. Moreover, $H_{\mathrm{Gauss}}(\mu|x)$ is an upper bound on the entropy by using a Gaussian with identical variance. More refined tree decompositions exist, too. The key benefit of our approach is that the solution μ of the optimization problem (26) can immediately be used as a gradient of the upper bound. This is computationally rather efficient.

Likewise, note that Taskar et al. (2004) use relaxations when solving structured estimation problems of the form

$$l(x,y,w) = \max_{y'} \Gamma(y,y')\left\langle w,\phi(x,y') - \phi(x,y)\right\rangle + \Delta(y,y'),$$

by enlarging the domain of maximization with respect to y'. For instance, instead of an integer programming problem we might relax the setting to a linear program which is much cheaper to solve. This, again, provides an upper bound on the original loss function.

In summary, we have demonstrated that convex relaxation strategies are well applicable for bundle methods. In fact, the results of the corresponding optimization procedures can be used directly for further optimization steps.

A.3 Scalar Multivariate Performance Scores

We now discuss a series of structured loss functions and how they can be implemented efficiently. For the sake of completeness, we give a concise representation of previous work on multivariate performance scores and ranking methods. All these loss functions rely on having access to $\langle w,x\rangle$, which can be computed efficiently by using the same operations as in Section A.1.

A.3.1 ROC SCORE

Denote by $f = Xw$ the vector of function values on the training set. It is well known that the area under the ROC curve is given by

$$\mathrm{AUC}(x,y,w) = \frac{1}{m_+ m_-} \sum_{y_i<y_j} \mathbf{I}(\langle w,x_i\rangle < \langle w,x_j\rangle),$$

where m_+ and m_- are the numbers of positive and negative observations respectively, and $\mathbf{I}(\cdot)$ is indicator function. Directly optimizing the cost $1-\mathrm{AUC}(x,y,w)$ is difficult as it is not continuous

Algorithm 7 ROCScore(X, y, w)

input: Feature matrix X, labels y, and weight vector w
initialization: $s_- = m_-$ and $s_+ = 0$ and $l = \mathbf{0}_m$ and $c = Xw - \frac{1}{2}y$
$\pi \leftarrow \{1, \ldots, m\}$ sorted in ascending order of c
for $i = 1$ **to** m **do**
  if $y_{\pi_i} = -1$ **then**
    $l_{\pi_i} \leftarrow s_+$ and $s_- \leftarrow s_- - 1$
  else
    $l_{\pi_i} \leftarrow -s_-$ and $s_+ \leftarrow s_+ + 1$
  end if
end for
Rescale $l \leftarrow l/(m_+ m_-)$ and compute $r = \langle l, c \rangle$ and $g = l^\top X$.
return Risk r and subgradient g

in w. By using $\max(0, 1 + \langle w, x_i - x_j \rangle)$ as the surrogate loss function for all pairs (i, j) for which $y_i < y_j$ we have the following convex multivariate empirical risk

$$R_{\text{emp}}(w) = \frac{1}{m_+ m_-} \sum_{y_i < y_j} \max(0, 1 + \langle w, x_i - x_j \rangle) = \frac{1}{m_+ m_-} \sum_{y_i < y_j} \max(0, 1 + f_i - f_j). \tag{27}$$

Obviously, we could compute $R_{\text{emp}}(w)$ and its derivative by an $O(m^2)$ operation. However Joachims (2005) shows that both can be computed in $O(m \log m)$ time using a sorting operation, which we now describe.

Denote by $c = f - \frac{1}{2}y$ an auxiliary variable and let i and j be indices such that $y_i = -1$ and $y_j = 1$. It follows that $c_i - c_j = 1 + f_i - f_j$. The efficient algorithm is due to the observation that there are at most m distinct terms c_k, $k = 1, \ldots, m$, each with different frequency l_k and sign, appear in (27). These frequencies l_k can be determined by first sorting c in ascending order then scanning through the labels according to the sorted order of c and keeping running statistics such as the number s_- of negative labels yet to encounter, and the number s_+ of positive labels encountered. When visiting y_k, we know c_k should appears s_+ (or s_-) times with positive (or negative) sign in (27) if $y_k = -1$ (or $y_k = 1$). Algorithm 7 spells out explicitly how to compute $R_{\text{emp}}(w)$ and its subgradient.

A.3.2 ORDINAL REGRESSION

Essentially the same preference relationships need to hold for ordinal regression. The only difference is that y_i need not take on binary values any more. Instead, we may have an arbitrary number of different values y_i (e.g., 1 corresponding to 'strong reject' up to 10 corresponding to 'strong accept', when it comes to ranking papers for a conference). That is, we now have $y_i \in \{1, \ldots, n\}$ rather than $y_i \in \{\pm 1\}$. Our goal is to find some w such that $\langle w, x_i - x_j \rangle < 0$ whenever $y_i < y_j$. Whenever this relationship is not satisfied, we incur a cost $C(y_i, y_j)$ for preferring x_i to x_j. For examples, $C(y_i, y_j)$ could be constant, that is, $C(y_i, y_j) = 1$ (Joachims, 2006) or linear, that is, $C(y_i, y_j) = y_j - y_i$.

Denote by m_i the number of x_j for which $y_j = i$. In this case, there are $\bar{M} = m^2 - \sum_{i=1}^n m_i^2$ pairs (y_i, y_j) for which $y_i \neq y_j$; this implies that there are $M = \bar{M}/2$ pairs (y_i, y_j) such that $y_i < y_j$.

Normalizing by the total number of comparisons we may write the overall cost of the estimator as

$$\frac{1}{M}\sum_{y_i<y_j} C(y_i,y_j)\mathbf{I}(\langle w,x_i\rangle > \langle w,x_j\rangle) \text{ where } M = \frac{1}{2}\left[m^2 - \sum_i^n m_i^2\right].$$

Using the same convex majorization as above when we were maximizing the ROC score, we obtain an empirical risk of the form

$$R_{\text{emp}}(w) = \frac{1}{M}\sum_{y_i<y_j} C(y_i,y_j)\max(0,1+\langle w,x_i-x_j\rangle).$$

Now the goal is to find an efficient algorithm for obtaining the number of times when the individual losses are nonzero such as to compute both the value and the gradient of $R_{\text{emp}}(w)$. The complication arises from the fact that observations x_i with label y_i may appear in either side of the inequality depending on whether $y_j < y_i$ or $y_j > y_i$. This problem can be solved as follows: sort $f = Xw$ in ascending order and traverse it while keeping track of how many items with a lower value y_j are no more than 1 apart in terms of their value of f_i. This way we may compute the count statistics efficiently. Algorithm 8 describes the details, generalizing the results of Joachims (2006). Again, its runtime is $O(m\log m)$, thus allowing for efficient computation.

A.3.3 PREFERENCE RELATIONS

In general, our loss may be described by means of a set of preference relations $j \succeq i$ for arbitrary pairs $(i,j) \in \{1,\dots m\}^2$ associated with a cost $C(i,j)$ which is incurred whenever i is ranked above j. This set of preferences may or may not form a partial or a total order on the domain of all observations. In these cases efficient computations along the lines of Algorithm 8 exist. In general, this is not the case and we need to rely on the fact that the set P containing all preferences is sufficiently small that it can be enumerated efficiently. The risk is then given by

$$\frac{1}{|P|}\sum_{(i,j)\in P} C(i,j)\mathbf{I}(\langle w,x_i\rangle > \langle w,x_j\rangle).$$

Again, the same majorization argument as before allows us to write a convex upper bound

$$R_{\text{emp}}(w) = \frac{1}{|P|}\sum_{(i,j)\in P} C(i,j)\max\left(0,1+\langle w,x_i\rangle - \langle w,x_j\rangle\right)$$

$$\text{where } \partial_w R_{\text{emp}}(w) = \frac{1}{|P|}\sum_{(i,j)\in P} C(i,j)\begin{cases}0 & \text{if } \langle w,x_j-x_i\rangle \geq 1\\ x_i - x_j & \text{otherwise.}\end{cases}$$

The implementation is straightforward, as given in Algorithm 9.

A.3.4 RANKING

In webpage and document ranking we are often in a situation similar to that described in Section A.3.2, however with the difference that we do not only care about objects x_i being ranked according to scores y_i but moreover that different degrees of importance are placed on different documents.

Algorithm 8 OrdinalRegression(X, y, w, C)

input: Feature matrix X, labels y, weight vector w, and score matrix C
initialization: $l = \mathbf{0}_n$ and $u_i = m_i \; \forall i \in [n]$ and $r = 0$ and $g = \mathbf{0}_m$
Compute $f = Xw$ and set $c = [f - \frac{1}{2}, f + \frac{1}{2}] \in \mathbb{R}^{2m}$ (concatenate the vectors)
Compute $M = (m^2 - \sum_{i=1}^{n} m_i^2)/2$
Rescale $C \leftarrow C/M$
$\pi \leftarrow \{1, \ldots, 2m\}$ sorted in ascending order of c
for $i = 1$ **to** $2m$ **do**
  $j = \pi_i \bmod m$
  if $\pi_i \leq m$ **then**
    for $k = 1$ **to** $y_j - 1$ **do**
      $r \leftarrow r - C(k, y_j) u_k c_j$
      $g_j \leftarrow g_j - C(k, y_j) u_k$
    end for
    $l_{y_j} \leftarrow l_{y_j} + 1$
  else
    for $k = y_j + 1$ **to** n **do**
      $r \leftarrow r + C(y_j, k) l_k c_{j+m}$
      $g_j \leftarrow g_j + C(y_j, k) l_k$
    end for
    $u_{y_j} \leftarrow u_{y_j} - 1$
  end if
end for
$g \leftarrow g^\top X$
return: Risk r and subgradient g

Algorithm 9 Preference(X, w, C, P)

input: Feature matrix X, weight vector w, score matrix C, and preference set P
initialization: $r = 0$ and $g = \mathbf{0}_m$
Compute $f = Xw$
while $(i, j) \in P$ **do**
  if $f_j - f_i < 1$ **then**
    $r \leftarrow r + C(i, j)(1 + f_i - f_j)$
    $g_i \leftarrow g_i + C(i, j)$ and $g_j \leftarrow g_j - C(i, j)$
  end if
end while
$g \leftarrow g^\top X$
return Risk r and subgradient g

The information retrieval literature is full with a large number of different scoring functions. Examples are criteria such as *Normalized Discounted Cumulative Gain (NDCG)*, *Mean Reciprocal Rank (MRR)*, *Precision@n*, or *Expected Rank Utility (ERU)*. They are used to address the issue of evaluating rankers, search engines or recommender sytems (Voorhees, 2001; Jarvelin and Kekalainen, 2002; Breese et al., 1998; Basilico and Hofmann, 2004). For instance, in webpage

Algorithm 10 Ranking(X, y, w)

1: **input:** Feature matrix X, relevances y, and weight vector w
2: Compute vectors a and $b(y)$ according to some ranking measure
3: Compute $f = Xw$
4: Compute elements of matrix $C_{ij} = c_i f_j - b_i a_j$
5: $\pi = \text{LinearAssignment}(C)$
6: $r = c^\top (f(\pi) - f) + (a - a(\pi))^\top b$
7: $g = c(\pi^{-1}) - c$ and $g \leftarrow g^\top X$
8: **return** Risk r and subgradient g

ranking only the first k retrieved documents that matter, since users are unlikely to look beyond the first k, say 10, retrieved webpages in an internet search. Le and Smola (2007) show that these scores can be optimized directly by minimizing the following loss:

$$l(X, y, w) = \max_\pi \sum_i c_i \left\langle w, x_{\pi(i)} - x_i \right\rangle + \langle a - a(\pi), b(y) \rangle . \tag{28}$$

Here c_i is a monotonically decreasing sequence, the documents are assumed to be arranged in order of decreasing relevance, π is a permutation, the vectors a and $b(y)$ depend on the choice of a particular ranking measure, and $a(\pi)$ denotes the permutation of a according to π. Pre-computing $f = Xw$ we may rewrite (28) as

$$l(f, y) = \max_\pi \left[c^\top f(\pi) - a(\pi)^\top b(y) \right] - c^\top f + a^\top b(y)$$

and consequently the derivative of $l(X, y, w)$ with respect to w is given by

$$\partial_w l(X, y, w) = (c(\bar{\pi}^{-1}) - c)^\top X \text{ where } \bar{\pi} = \operatorname*{argmax}_\pi c^\top f(\pi) - a(\pi)^\top b(y).$$

Here π^{-1} denotes the inverse permutation, such that $\pi \circ \pi^{-1} = 1$. Finding the permutation maximizing $c^\top f(\pi) - a(\pi)^\top b(y)$ is a linear assignment problem which can be easily solved by the Hungarian Marriage algorithm, that is, the Kuhn-Munkres algorithm.

The original papers by Kuhn (1955) and Munkres (1957) implied an algorithm with $O(m^3)$ cost in the number of terms. Later, Karp (1980) suggests an algorithm with expected quadratic time in the size of the assignment problem (ignoring log-factors). Finally, Orlin and Lee (1993) propose a linear time algorithm for large problems. Since in our case the number of pages is fairly small (in the order of 50 to 200 *per query*) the scaling behavior per query is not too important. We used an existing implementation due to Jonker and Volgenant (1987).

Note also that training sets consist of a *collection* of ranking problems, that is, we have several ranking problems of size 50 to 200. By means of parallelization we are able to distribute the work onto a cluster of workstations, which is able to overcome the issue of the rather costly computation per collection of queries. Algorithm 10 spells out the steps in detail.

A.3.5 CONTINGENCY TABLE SCORES

Joachims (2005) observed that F_β scores and related quantities dependent on a contingency table can also be computed efficiently by means of structured estimation. Such scores depend in general on

the number of true and false positives and negatives alike. Algorithm 11 shows how a corresponding empirical risk and subgradient can be computed efficiently. As with the previous losses, here again we use convex majorization to obtain a tractable optimization problem.

Given a set of labels y and an estimate y', the numbers of true positives (T_+), true negatives (T_-), false positives (F_+), and false negatives (F_-) are determined according to a contingency table as follows:

	$y > 0$	$y < 0$
$y' > 0$	T_+	F_+
$y' < 0$	F_-	T_-

In the sequel, we denote by $m_+ = T_+ + F_-$ and $m_- = T_- + F_+$ the numbers of positives and negative labels in y, respectively. We note that F_β score can be computed based on the contingency table (Joachims, 2005) as

$$F_\beta(T_+, T_-) = \frac{(1+\beta^2)T_+}{T_+ + m_- - T_- + \beta^2 m_+}.$$

If we want to use $\langle w, x_i \rangle$ to estimate the label of observation x_i, we may use the following structured loss to "directly" optimize w.r.t. F_β score (Joachims, 2005):

$$l(X, y, w) = \max_{y'} \left[(y' - y)^\top f + \Delta(T_+, T_-) \right],$$

where $f = Xw$, $\Delta(T_+, T_-) := 1 - F_\beta(T_+, T_-)$, and (T_+, T_-) is determined by using y and y'. Since Δ does not depend on the specific choice of (y, y') but rather just on which sets they disagree, l can be maximized as follows: Enumerating all possible $m_+ m_-$ contingency tables in a way such that given a configuration (T_+, T_-), T_+ (T_-) positive (negative) observations x_i with largest (lowest) value of $\langle w, x_i \rangle$ are labeled as positive (negative). This is effectively implemented as a nested loop hence run in $O(m^2)$ time. Algorithm 11 describes the procedure in details.

A.4 Vector Loss Functions

Next we discuss "vector" loss functions, that is, functions where w is best described as a matrix (denoted by W) and the loss depends on Wx. Here, we have feature vector $x \in \mathbb{R}^d$, label $y \in \mathbb{R}^k$, and weight matrix $W \in \mathbb{R}^{d \times k}$. We also denote feature matrix $X \in \mathbb{R}^{m \times d}$ as a matrix of m feature vectors x_i, and stack up the columns W_i of W as a vector w.

Some of the most relevant cases are multiclass classification using both the exponential families model and structured estimation, hierarchical models, that is, ontologies, and multivariate regression. Many of those cases are summarized in Table 6.

A.4.1 UNSTRUCTURED SETTING

The simplest loss is multivariate regression, where $l(x, y, W) = \frac{1}{2}(y - x^\top W)^\top M (y - x^\top W)$. In this case it is clear that by pre-computing XW subsequent calculations of the loss and its gradient are significantly accelerated.

A second class of important losses is given by plain multiclass classification problems, for example, recognizing digits of a postal code or categorizing high-level document categories. In this case, $\phi(x, y)$ is best represented by $e_y \otimes x$ (using a linear model). Clearly we may view $\langle w, \phi(x, y) \rangle$

Algorithm 11 $\mathrm{F}_\beta(X, y, w)$

input: Feature matrix X, labels y, and weight vector w
Compute $f = Xw$
$\pi^+ \leftarrow \{i : y_i = 1\}$ sorted in descending order of f
$\pi^- \leftarrow \{i : y_i = -1\}$ sorted in ascending order of f
Let $p_0 = 0$ and $p_i = 2\sum_{k=i}^{m_+} f_{\pi_k^+},\ i = 1, \ldots, m_+$
Let $n_0 = 0$ and $n_i = 2\sum_{k=i}^{m_-} f_{\pi_k^-},\ i = 1, \ldots, m_-$
$y' \leftarrow -y$ and $r \leftarrow -\infty$
for $i = 0$ **to** m_+ **do**
  for $j = 0$ **to** m_- **do**
    $r_{\text{tmp}} = \Delta(i, j) - p_i + n_j$
    if $r_{\text{tmp}} > r$ **then**
      $r \leftarrow r_{\text{tmp}}$
      $T_+ \leftarrow i$ and $T_- \leftarrow j$
    end if
  end for
end for
$y'_{\pi_i^+} \leftarrow 1,\ i = 1, \ldots, T_+$
$y'_{\pi_i^-} \leftarrow -1,\ i = 1, \ldots, T_-$
$g \leftarrow (y' - y)^\top X$
return Risk r and subgradient g

as an operation which chooses a column indexed by y from xW, since all labels y correspond to a different weight vector W_y. Formally we set $\langle w, \phi(x, y)\rangle = [xW]_y$. In this case, structured estimation losses can be rewritten as

$$l(x, y, W) = \max_{y'} \Gamma(y, y') \left\langle W_{y'} - W_y, x \right\rangle + \Delta(y, y') \tag{29}$$

$$\text{and } \partial_W l(x, y, W) = \Gamma(y, y^*)(e_{y^*} - e_y) \otimes x.$$

Here Γ and Δ are defined as in Section A.2 and y^* denotes the value of y' for which the RHS of (29) is maximized. This means that for unstructured multiclass settings we may simply compute xW. Since this needs to be performed for all observations x_i we may take advantage of fast linear algebra routines and compute $f = XW$ for efficiency. Likewise note that computing the gradient over m observations is now a matrix-matrix multiplication, too: denote by G the matrix of rows of gradients $\Gamma(y_i, y_i^*)(e_{y_i^*} - e_{y_i})$. Then $\partial_W R_{\text{emp}}(X, y, W) = G^\top X$. Note that G is very sparse with at most two nonzero entries per row, which makes the computation of $G^\top X$ essentially as expensive as two matrix vector multiplications. Whenever we have many classes, this may yield significant computational gains.

Log-likelihood scores of exponential families share similar expansions. We have

$$l(x, y, W) = \log \sum_{y'} \exp \left\langle w, \phi(x, y')\right\rangle - \langle w, \phi(x, y)\rangle = \log \sum_{y'} \exp \left\langle W_{y'}, x\right\rangle - \langle W_y, x\rangle$$

$$\partial_W l(x, y, W) = \frac{\sum_{y'} (e_{y'} \otimes x) \exp \left\langle W_{y'}, x\right\rangle}{\sum_{y'} \exp \left\langle W_{y'}, x\right\rangle} - e_y \otimes x.$$

The main difference to the soft-margin setting is that the gradients are *not* sparse in the number of classes. This means that the computation of gradients is slightly more costly.

A.4.2 ONTOLOGIES

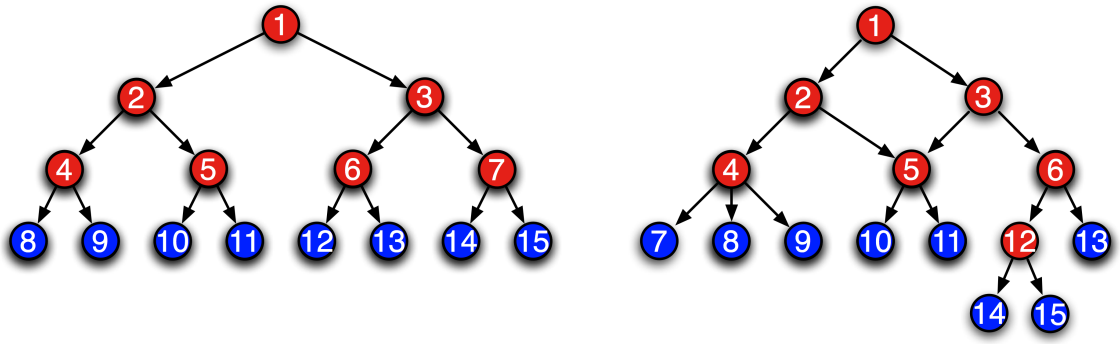


Figure 13: Two ontologies. **Left:** a binary hierarchy with internal nodes $\{1,\dots,7\}$ and labels $\{8,\dots 15\}$. **Right:** a generic directed acyclic graph with internal nodes $\{1,\dots,6,12\}$ and labels $\{7,\dots,11,13,\dots,15\}$. Note that node 5 has two parents, namely nodes 2 and 3. Moreover, the labels need not be found at the same level of the tree: nodes 14 and 15 are one level lower than the rest of the nodes.

Assume that the labels we want to estimate can be found to belong to a directed acyclic graph (DAG). For instance, this may be a gene-ontology graph (Ashburner et al., 2000) a patent hierarchy (Cai and Hofmann, 2004), or a genealogy. In these cases we have a hierarchy of categories to which an element x may belong. Figure 13 gives two examples of such directed acyclic graphs. The first example is a binary tree, while the second contains nodes with different numbers of children (e.g., node 4 and 12), nodes at different levels having children (e.g., nodes 5 and 12), and nodes which have more than one parent (e.g., node 5). It is a well known fundamental property of trees that they have at most as many internal nodes as they have leaf nodes.

It is now our goal to build a classifier which is able to categorize observations according to which leaf node they belong to (each leaf node is assigned a label y). Denote by $k+1$ the number of nodes in the DAG including the root node. In this case we may design a feature map $\phi(y) \in \mathbb{R}^k$ (Cai and Hofmann, 2004) by associating with every label y the vector describing the path from the root node to y, ignoring the root node itself. For instance, for the first DAG in Figure 13 we have

$$\phi(8) = (1,0,1,0,0,0,1,0,0,0,0,0,0,0) \text{ and } \phi(13) = (0,1,0,0,1,0,0,0,0,0,0,1,0,0)$$

Whenever several paths are admissible, as in the right DAG of Figure 13 we average over all possible paths. For example, we have

$$\phi(10) = (0.5,0.5,0,1,0,0,0,0,1,0,0,0,0,0) \text{ and } \phi(15) = (0,1,0,0,1,0,0,0,0,0,0,1,0,0,1).$$

Also note that the lengths of the paths need not be the same (e.g., to reach 15 it takes a longer path than to reach 13). Likewise, it is natural to assume that $\Delta(y,y')$, that is, the cost for mislabeling y

Algorithm 12 Ontology(X, y, W)

input: Feature matrix $X \in \mathbb{R}^{m \times d}$, labels y, and weight matrix $W \in \mathbb{R}^{d \times k}$
initialization: $G = \mathbf{0} \in \mathbb{R}^{m \times k}$ and $r = 0$
Compute $f = XW$ and let $f_i = x_i W$
for $i = 1$ **to** m **do**
  Let D_i be the DAG with edges annotated with the values of f_i
  Traverse D_i to find a path y^* that maximizes the value $z_{y^*} := \sum_{j=1}^{k} [\phi(y^*)]_j f_{ij} + \Delta(y_i, y^*)$
  $G_i = \phi(y^*) - \phi(y_i)$
  $r \leftarrow r + z_{y^*} - z_{y_i}$
end for
$g = G^\top X$
return Risk r and subgradient g

as y' will depend on the similarity of the path. In other words, it is likely that the cost for placing x into the wrong sub-sub-category is less than getting the main category of the object wrong.

To complete the setting, note that for $\phi(x, y) = \phi(y) \otimes x$ the cost of computing all labels is k inner products, since the value of $\langle w, \phi(x, y) \rangle$ for a particular y can be obtained by the sum of the contributions for the segments of the path. This means that the values for *all* terms can be computed by a simple breadth first traversal through the graph. As before, we may make use of vectorization in our approach, since we may compute $xW \in \mathbb{R}^k$ to obtain the contributions on all segments of the DAG before performing the graph traversal. Since we have m patterns x_i we may vectorize matters by pre-computing XW.

Also note that $\phi(y) - \phi(y')$ is nonzero only for those edges where the paths for y and y' differ. Hence we only change weights on those parts of the graph where the categorization differs. Algorithm 12 describes the subgradient and loss computation for the soft-margin type of loss function.

The same reasoning applies to estimation when using an exponential families model. The only difference is that we need to compute a *soft-max* over paths rather than exclusively choosing the best path over the ontology. Again, a breadth-first recursion suffices: each of the leaves y of the DAG is associated with a probability $p(y|x)$. To obtain $\mathbf{E}_{y \sim p(y|x)} [\phi(y)]$ all we need to do is perform a bottom-up traversal of the DAG summing over all probability weights on the path. Wherever a node has more than one parent, we distribute the probability weight equally over its parents.

Appendix B. Proofs

This section contains the proofs of Theorems 4, 5, and 7, along with the technical lemmas required for these.

B.1 Proof of Theorem 4

To show Theorem 4 we need several technical intermediate steps. Let $\gamma_t := J(w_t) - J_t(w_t)$ and recall that $\varepsilon_t := \min_{t' \leq t} J(w_{t'}) - J_t(w_t)$. The following lemma establishes some useful properties of γ_t and ε_t.

Lemma 9 *We have $J_{t'}(w_{t'}) \leq J_t(w_t) \leq J(w^*) \leq J(w_t) = J_{t+1}(w_t)$ for all $t' \leq t$. Furthermore, ε_t is monotonically decreasing with $\varepsilon_t - \varepsilon_{t+1} \geq J_{t+1}(w_{t+1}) - J_t(w_t) \geq 0$. Also, ε_t upper bounds the distance from optimality via $\gamma_t \geq \varepsilon_t \geq \min_{t' \leq t} J(w_{t'}) - J(w^*)$.*

Proof Since $J_{t'}(w) \leq J_t(w) \leq J(w)$ for all $t' \leq t$ this property also applies to their respective minima. Moreover, since w^* minimizes $J(w)$ we have $J(w^*) \leq J(w_t)$. Since Taylor expansions are exact at the point of expansion $J(w_t) = J_{t+1}(w_t)$. The first inequality follows from the definition of ε_t, and the fact that J_t is monotonically increasing. Finally, since $J_{t'+1}(w_{t'}) = J(w_{t'})$ it is easy to see that $\gamma_t \geq \varepsilon_t = \min_{t' \leq t} J(w_{t'}) - J_t(w_t) \geq \min_{t' \leq t} J(w_{t'}) - J(w^*)$. ■

Our second technical lemma allows us to bound the maximum value of a concave function provided that we know its first derivative and a bound on the second derivative.

Lemma 10 *Denote by $f : [0,1] \to \mathbb{R}$ a concave function with $f(0) = 0$, $f'(0) = l$, and $|f''(x)| \leq K \; \forall x \in [0,1]$. Then we have $\max_{x \in [0,1]} f(x) \geq \frac{l}{2} \min(\frac{l}{K}, 1)$.*

Proof We first observe that $g(x) := lx - \frac{K}{2}x^2 \leq f(x) \; \forall x$ implies $\max_{x \in [0,1]} f(x) \geq \max_{x \in [0,1]} g(x)$. g attains the unconstrained maximum $\frac{l^2}{2K}$ at $x = \frac{l}{K}$. Since g is monotonically increasing in $[0, \frac{l}{K}]$, if $l > K$ we pick $x = 1$ which yields constrained maximum $l - \frac{K}{2} > \frac{l}{2}$. Taking the minimum over both maxima proves the claim.

■

To apply the above result, we need to compute the gradient and Hessian of $J^*_{t+1}(\alpha)$ with respect to the search direction $((1-\eta)\alpha_t, \eta)$. The following lemma takes care of the gradient:

Lemma 11 *Denote by α_t the solution of (9) at time instance t. Moreover, denote by $\bar{A} = [A, a_{t+1}]$ and $\bar{b} = [b, b_{t+1}]$ the extended matrices and vectors needed to define the dual problem for step $t+1$, and let $\bar{\alpha} \in \mathbb{R}^{t+1}$. Then the following holds:*

$$\begin{aligned} \partial_{\bar{\alpha}} J^*_{t+1}([\alpha_t, 0]) &= \bar{A}^\top w_t + \bar{b} \text{ and} \\ [-\alpha_t, 1]^\top \left[\bar{A}^\top w_t + \bar{b}\right] &= J_{t+1}(w_t) - J_t(w_t) = \gamma_t. \end{aligned} \tag{30}$$

Proof By the dual connection $\partial \Omega^*(-\lambda^{-1} A \alpha_t) = w_t$. Hence we have that $\partial_{\bar{\alpha}} - \lambda \Omega^*(-\lambda^{-1} \bar{A} \bar{\alpha}) + \bar{\alpha}^\top \bar{b} = \bar{A}^\top w_t + \bar{b}$ for $\bar{\alpha} = [\alpha_t, 0]^\top$. This proves the first claim. To see the second part we eliminate ξ from of the Lagrangian (11) and write the partial Lagrangian

$$L(w, \alpha) = \lambda \Omega(w) + \alpha^\top \left(A^\top w + b\right) \text{ with } \alpha \geq 0.$$

The result follows by noting that at optimality $L(w_t, \alpha_t) = J_t(w_t)$ and $J_{t+1}(w_t) = \lambda \Omega(w_t) + \langle w_t, a_{t+1} \rangle + b_{t+1}$. Consequently we have

$$J_{t+1}(w_t) - J_t(w_t) = \lambda \Omega(w_t) + \langle w_t, a_{t+1} \rangle + b_{t+1} - \lambda \Omega(w_t) - \alpha_t (A^\top w_t + b_t).$$

Rearranging terms proves the claim. ■

To apply Lemma 10 we also need to bound the second derivative.

Lemma 12 *Under the assumptions of Lemma 11 we have*

$$\partial^2_{\bar{\alpha}} J^*_{t+1}(\bar{\alpha}) = -\lambda^{-1}\bar{A}^\top \partial^2\Omega^*(-\lambda^{-1}\bar{A}\bar{\alpha})\bar{A} \tag{31}$$

$$\text{moreover } \bar{A}[-\alpha_t, 1] = s_t \in \partial_w J(w_t). \tag{32}$$

Proof The first equality is immediate from the chain rule. Next note that $\partial_w\Omega(w_t) = -\lambda^{-1}A\alpha_t$ by dual connection. Since $a_{t+1} \in \partial_w R_{\text{emp}}(w_t)$ the claim follows from $J(w) = R_{\text{emp}}(w) + \lambda\Omega(w)$. ■

This result allows us to express the second derivative of the dual objective function (10) in terms of the gradient of the risk functional. The idea is that as we approach optimality, the second derivative will vanish. We will use this fact to argue that for continuously differentiable losses $R_{\text{emp}}(w)$ we enjoy linear convergence throughout.

Proof [Theorem 4] We overload the notation for J^*_{t+1} by defining the following one dimensional concave function

$$J^*_{t+1}(\eta) := J^*_{t+1}([(1-\eta)\alpha_t, \eta]) = -\lambda\Omega^*(-\lambda^{-1}\bar{A}[(1-\eta)\alpha_t^\top, \eta]) + [(1-\eta)\alpha_t^\top, \eta]\bar{b}.$$

Clearly, $J^*_{t+1}(0) = J_t(w_t)$. Furthermore, by (30), (31), and (32) it follows that

$$\begin{aligned}
\partial_\eta J^*_{t+1}(\eta)|_{\eta=0} &= [-\alpha_t, 1]^\top \partial_{\bar{\alpha}} J^*_{t+1}([\alpha_t, 0]) = \gamma_t \text{ and} \\
\partial^2_\eta J^*_{t+1}(\eta) &= -\lambda^{-1}[-\alpha_t, 1]^\top \bar{A}^\top \partial^2\Omega^*(-\lambda^{-1}\bar{A}[(1-\eta)\alpha_t, \eta])\bar{A}[-\alpha_t, 1]^\top \\
&= -\lambda^{-1} s_t^\top \partial^2\Omega^*(-\lambda^{-1}\bar{A}[(1-\eta)\alpha_t, \eta])s_t := r.
\end{aligned}$$

By our assumption on $\|\partial^2\Omega^*\| \le H^*$ we have

$$|r| \le H^*\|s_t\|^2/\lambda.$$

Next we need to bound the gradient of J. For this purpose note that $\partial_w\lambda\Omega(w_t) = -A^\top\alpha_t$ and moreover that $\|\alpha_t\|_1 = 1$. This implies that $\partial_w\lambda\Omega(w_t)$ lies in the convex hull of the past gradients, $a_{t'}$. By our assumption that $\max_{u\in\partial_w R_{\text{emp}}(w)}\|u\| \le G$ it follows that $\|\partial_w\lambda\Omega(w_t)\| \le G$. We conclude that

$$\|s_t\|^2 \le 4G^2 \text{ and } |r| \le 4G^2H^*/\lambda.$$

Invoking Lemma 10 on $J^*_{t+1}(\eta) - J_t(w_t)$ shows that

$$J^*_{t+1}(\eta) - J_t(w_t) \ge \tfrac{\gamma_t}{2}\min(1, \lambda\gamma_t/4G^2H^*).$$

We now upper bound the LHS of the above inequality as follows:

$$\varepsilon_t - \varepsilon_{t+1} \ge J_{t+1}(w_{t+1}) - J_t(w_t) \ge J^*_{t+1}(\eta) - J_t(w_t) \ge \tfrac{\gamma_t}{2}\min(1, \lambda\gamma_t/4G^2H^*). \tag{33}$$

The first inequality follows from Lemma 9 while the second follows by observing that $J_{t+1}(w_{t+1}) = J^*_{t+1}(\alpha_{t+1}) \ge J^*_{t+1}(\eta)$. The RHS of the third inequality on the other hand can be lower bounded by observing that $\gamma_t \ge \varepsilon_t$, which follows from Lemma 9. This in turn obtains (12).

For the second part note that (12) already yields the $\varepsilon_t/2$ decrease when $\varepsilon_t \ge 4G^2H^*/\lambda$. To show the other parts we need to show that the gradient of the regularized risk vanishes as we converge

to the optimal solution. Towards this end, we apply Lemma 10 in the *primal*.[23] This allows us to bound $\|\partial_w J(w_t)\|$ in terms of γ_t. Plugging in the first and second derivative of $J(w_t)$ we obtain

$$\gamma_t \geq \frac{1}{2}\|\partial_w J(w_t)\| \min(1, \|\partial_w J(w_t)\|/H).$$

If $\|\partial_w J(w_t)\| > H$, then $\gamma_t \geq \frac{1}{2}\|\partial_w J(w_t)\|$ which in turn yields $|r| \leq 4\gamma_t^2 H^*/\lambda$. Plugging this into Lemma 10 yields a lower bound on the improvement of $\lambda/8H^*$.

Finally, for $\|\partial_w J(w_t)\| \leq H$ we have $\gamma_t \geq \|\partial_w J(w_t)\|^2/2H$, which implies $|r| \leq 2HH^*\gamma_t/\lambda$. Plugging this into Lemma 10 yields an improvement of $\lambda\gamma_t/4HH^* \geq \lambda\varepsilon_t/4HH^*$.

Since both cases cover the remaining range of convergence, the minimum $\min(\lambda/8H^*, \lambda\varepsilon_t/4HH^*)$ provides a lower bound for the improvement. The crossover point between both terms occurs at $\varepsilon_t = H/2$. Rearranging the conditions leads to the (pessimistic) improvement guarantees of the second claim.

■

Note that a key step in the above analysis involved bounding $r := \partial_\eta^2 J^*_{t+1}(\eta)$. For a number of regularizers tighter bounds can be obtained. The following bounds are essentially due to Shalev-Shwartz and Singer (2006):

- For squared norm regularization, that is, $\Omega^*(\mu) = \frac{1}{2}\|\mu\|_2^2$ we have $r = \|\partial_w J(w_t)\|_2^2$.
- For L_p norm regularization, that is, $\Omega^*(\mu) = \frac{1}{2}\|\mu\|_q^2$ we have $r \leq (q-1)\|\partial_w J(w_t)\|_q^2$.
- For quadratic form regularization with PD matrix B, that is, $\Omega^*(\mu) = \frac{1}{2}\mu B^{-1}\mu$, we have $r = \partial_w J(w_t)^\top B^{-1}\partial_w J(w_t)$.
- For unnormalized entropic regularization we have $\partial_\mu^2\Omega^*(\mu) = \operatorname{diag}\left(e^{\mu^{(1)}}, \ldots, e^{\mu^{(d)}}\right)$. Hence we may bound $r \leq \|\partial_w J(w_t)\|_2^2 \exp(\|\mu\|_\infty)$. Clearly this bound may be very loose whenever μ has only very few large coefficients.
- For normalized entropy regularization, that is, $\Omega^*(\mu) = \log\sum_i \exp\mu^{(i)}$ we have $r \leq \|\partial_w J(w_t)\|_\infty^2$.

B.2 Proof of Theorem 5

We need the following technical lemma for the proof:

Lemma 13 *Let $\langle \rho_1, \rho_2, \ldots\rangle$ be a sequence of non-negative numbers satisfying the following recurrence, for $t \geq 1$: $\rho_t - \rho_{t+1} \geq c(\rho_t)^2$, where $c > 0$ is a constant. Then for all integers $t \geq 1$,*

$$\rho_t \leq \frac{1}{c(t-1+\frac{1}{\rho_1 c})}.$$

Furthermore $\rho_t \leq \rho$ whenever

$$t \geq \frac{1}{c\rho} - \frac{1}{\rho_1 c} + 1.$$

23. Define $\bar{J}(\eta) := J(w_t) - J(w_t + \eta p)$ where $p = -\frac{\partial_w J(w_t)}{\|\partial_w J(w_t)\|}$ is the unit-length gradient. We see that $\frac{d}{d\eta}\bar{J}(\eta)\Big|_{\eta=0} = [-\partial_w J(w_t + \eta p)^\top p]|_{\eta=0} = \|\partial_w J(w_t)\|$, and $\bar{J}(0) = 0$. Hence Lemma 10 is applicable in this case.

This is Sublemma 5.4 of Abe et al. (2001) which is easily proven by induction. Now we can prove the main result.

Proof [Theorem 5] For any $\varepsilon_t > 4G^2H^*/\lambda$ it follows from (12) that $\varepsilon_{t+1} \leq \varepsilon_t/2$. Moreover, $\varepsilon_0 \leq J(0)$, since we know that J is non-negative. Hence we need at most $\log_2[\lambda J(0)/4G^2H^*]$ to achieve this level of precision. Subsequently we have

$$\varepsilon_t - \varepsilon_{t+1} \geq \frac{\lambda}{8G^2H^*}\varepsilon_t^2.$$

Invoking Lemma 13 by setting $c = \frac{\lambda}{8G^2H^*}$ and $\rho_1 = 4G^2H^*/\lambda$ shows that $\varepsilon_t \leq \varepsilon$ after at most $\frac{8G^2H^*}{\lambda\varepsilon} - 1$ more steps. This proves the first claim.

To analyze convergence in the second case we need to study two additional phases: for $\varepsilon_t \in [H/2, 4G^2H^*/\lambda]$ we see constant progress. Hence it takes us $4\lambda^{-2}[8G^2(H^*)^2 - HH^*\lambda]$ steps to cover this interval. Finally in the third phase we have $\varepsilon_{t+1} \leq \varepsilon_t[1-\lambda/4HH^*]$. Starting from $\varepsilon_t = H/2$ we need $\log_2[2\varepsilon/H]/\log_2[1-\lambda/4HH^*]$ steps to converge. Expanding the logarithm in the denominator close to 1 proves the claim. ■

B.3 Proof of Theorem 7

We first note that the termination criterion of Algorithm 3 is slightly different from that of Algorithm 2. In order to apply the convergence results for Algorithm 2 to Algorithm 3 we redefine the following notations:

$$\begin{aligned} \varepsilon_t &:= J(w_t^b) - J_t(w_t) \\ a_{t+1} &\in \partial_w R_{\text{emp}}(w_t^c), \\ b_{t+1} &:= R_{\text{emp}}(w_t^c) - \langle w_t, a_{t+1} \rangle, \end{aligned} \tag{34}$$

where

$$\begin{aligned} \eta_t &:= \operatorname*{argmin}_{\eta} J(w_{t-1}^b + \eta(w_t - w_{t-1}^b)), \\ w_t^b &:= \hat{w}_{t-1} + \eta_t(\bar{w}_t - \hat{w}_{t-1}), \text{ and} \\ w_t^c &:= (1-\theta)w_t^b + \theta w_t. \end{aligned}$$

Then we state and prove the following lemma which is crucial to the application of Lemma 11 in the proof.

Lemma 14 $J_{t+1}(w_t) = \lambda\Omega(w_t) + \langle w_t, a_{t+1} \rangle + b_{t+1}$

Proof w_t^b is the optimal value of J on the line joining w_t and w_{t-1}^b while w_t^c is a convex combination of w_t and w_t^b. Moreover by definition of a_{t+1} and b_{t+1} we have $J(w_t^c) = J_{t+1}(w_t^c)$. Therefore,

$$J(w_t^c) = J_{t+1}(w_t^c) = \lambda\Omega(w_t^c) + \langle a_{t+1}, w_t^c \rangle + b_{t+1} \geq J(w_t^b). \tag{35}$$

But since Ω is convex

$$\Omega(\underbrace{(1-\theta)w_t^b + \theta w_t}_{w_t^c}) \leq (1-\theta)\Omega(w_t^b) + \theta\Omega(w_t),$$

which can be rearranged to

$$\theta(\Omega(w_t^b) - \Omega(w_t)) \leq \Omega(w_t^b) - \Omega(w_t^c).$$

Multiplying by λ and adding and subtracting $\theta R_{\text{emp}}(w_t^b)$ and $\theta R_t(w_t)$ respectively to the above equation

$$\underbrace{\lambda\theta\Omega(w_t^b) + \theta R_{\text{emp}}(w_t^b)}_{\theta J(w_t^b)} - \underbrace{\lambda\theta\Omega(w_t) - \theta R_t(w_t)}_{\theta J_t(w_t)} \leq \underbrace{\lambda\Omega(w_t^b) + R_{\text{emp}}(w_t^b)}_{J(w_t^b)} - \lambda\Omega(w_t^c) - (1-\theta)R_{\text{emp}}(w_t^b) - \theta R_t(w_t).$$

Plugging in (34) obtains

$$\theta\varepsilon_t \leq J(w_t^b) - \lambda\Omega(w_t^c) - (1-\theta)R_{\text{emp}}(w_t^b) - \theta R_t(w_t). \tag{36}$$

Putting (35) and (36) together

$$\langle a_{t+1}, w_t^c\rangle + b_{t+1} \geq J(w_t^b) - \lambda\Omega(w_t^c) \geq (1-\theta)R_{\text{emp}}(w_t^b) + \theta R_t(w_t) + \theta\varepsilon_t.$$

Since $w_t^c = (1-\theta)w_t^b + \theta w_t$ it follows that

$$(1-\theta)\left\langle a_{t+1}, w_t^b\right\rangle + \theta\langle a_{t+1}, w_t\rangle + b_{t+1} \geq (1-\theta)R_{\text{emp}}(w_t^b) + \theta R_t(w_t) + \theta\varepsilon_t.$$

Which can be rearranged to

$$(1-\theta)\left(\left\langle a_{t+1}, w_t^b\right\rangle - R_{\text{emp}}(w_t^b)\right) + \theta(\langle a_{t+1}, w_t\rangle - R_t(w_t)) + b_{t+1} \geq \theta\varepsilon_t.$$

Since $\left\langle w_t^b, a_{t+1}\right\rangle + b_{t+1}$ is the Taylor approximation of the convex function R_{emp} around w_t^c evaluated at w_t^b it follows that $R_{\text{emp}}(w_t^b) \geq \left\langle w_t^b, a_{t+1}\right\rangle + b_{t+1}$. Plugging this into the above equation yields

$$(1-\theta)(-b_{t+1}) + \theta(\langle w_t, a_{t+1}\rangle - R_t(w_t)) + b_{t+1} \geq \theta\varepsilon_t.$$

Dividing by $\theta > 0$ and rearranging yields

$$\langle w_t, a_{t+1}\rangle + b_{t+1} \geq R_t(w_t) + \varepsilon_t.$$

The conclusion of the lemma follows from observing that $R_{t+1}(w_t) = \max(\langle w_t, a_{t+1}\rangle + b_{t+1}, R_t(w_t)) = \langle w_t, a_{t+1}\rangle + b_{t+1}$ and $J_{t+1}(w_t) = \lambda\Omega(w_t) + R_{t+1}(w_t)$. ■

We also need the following two lemmas before we can proceed to the final proof.

Lemma 15 $\varepsilon_t - \varepsilon_{t+1} \geq J_{t+1}(w_{t+1}) - J_t(w_t)$

Proof

$$\begin{aligned}
\varepsilon_t - \varepsilon_{t+1} &= J(w_t^b) - J_t(w_t) - J(w_{t+1}^b) + J_{t+1}(w_{t+1}) \\
&= \underbrace{(J(w_t^b) - J(w_{t+1}^b))}_{\geq 0} + J_{t+1}(w_{t+1}) - J_t(w_t) \quad \text{(by the definition of } w_t^b\text{)} \\
&\geq J_{t+1}(w_{t+1}) - J_t(w_t).
\end{aligned}$$

■

Lemma 16 *Let* α_t, $\bar{A} := [a_1, \ldots, a_{t+1}]$, *and* $\bar{b} := [b_1, \ldots, b_{t+1}]$ *be as defined in Lemma 11. Then under the assumption of Theorem 4 that* $\max_{u \in \partial_w R_{\text{emp}}(w)} \|u\| \leq G$, *we have*

$$[-\alpha_t, 1]^\top \bar{A}^\top \bar{A} [-\alpha_t, 1] \leq 4G^2.$$

Proof By the dual connection, $\partial_w \lambda \Omega(w_t) = -A\alpha_t$. Also, $\alpha_t \geq 0$, and $\|\alpha_t\|_1 = 1$ as it is the optimal solution of (10) at iteration t. It follows that $\partial_w \lambda \Omega(w_t)$ lies in the convex hull of $a_{t'} \in \partial_w R_{\text{emp}}(w_{t'}^c)\ \forall t' \leq t$. Therefore $\|\partial_w \lambda \Omega(w_t)\| \leq G$. Consequently,

$$\begin{aligned}[-\alpha_t, 1]^\top \bar{A}^\top \bar{A} [-\alpha_t, 1] &= \|\partial_w \lambda \Omega(w_t) + a_{t+1}\|^2 \\ &= \|\partial_w \lambda \Omega(w_t)\|^2 + 2\partial_w \lambda \Omega(w_t)^\top a_{t+1} + \|a_{t+1}\|^2 \leq 4G^2,\end{aligned}$$

by Cauchy-Schwarz inequality. ■

Finally, we sketch the proof for Theorem 7.

Proof [Theorem 7] (Sketch) Theorem 4 holds for Algorithm 3 by applying Lemmas 14, 15, and 16 into the first part of the proof. Therefore, for $\varepsilon < 4G^2H^*/\lambda$, (33) reduces to $\varepsilon_t - \varepsilon_{t+1} \geq \lambda \varepsilon_t / 4G^2 H^*$. Applying Lemma 13 yields $\varepsilon_t \leq \frac{1}{c\left(t - 1 + \frac{1}{\varepsilon_1 c}\right)}$, with $c = \lambda / 8G^2 H^*$. Setting $\frac{1}{c\left(t-1+\frac{1}{\varepsilon_1 c}\right)} = \varepsilon$, assuming that $\varepsilon_1 > 0$, and solving for n yields $n \leq \frac{1}{c\varepsilon} = \frac{8G^2H^*}{\lambda\varepsilon}$. ■

Appendix C. L_1 Regularized BMRM

Following our convention, the L_1 norm regularized BMRM reads

$$\min_{\xi, w} \ \xi + \lambda \|w\|_1 \ \text{ subject to } w^\top a_i + b_i \leq \xi,\ i = 1, \ldots, t. \tag{37}$$

An equivalent formulation is

$$\min_{\xi, w} \ \xi \text{ subject to } w^\top a_i + b_i \leq \xi,\ i = 1, \ldots, t \text{ and } \|w\|_1 \leq \tau, \tag{38}$$

where one can show a monotone correspondence between τ and the λ in (37) by comparison of the KKT conditions for the two problems.

Note that our convergence proof does not apply in this case as the Fenchel dual of $\Omega(w) = \|w\|_1$ fails to satisfy the strong convexity assumption. Nevertheless, we see that (38) can be easily solved by CPM where the solution must lie in the L_1 ball of radius τ. Finally, we note that the L_1 regularized BMRM can be written in a rather standard linear programming (LP) formulation:

$$\begin{aligned}\min_{\xi, u, v} \ & \xi + \lambda \mathbf{1}_d^\top (u + v) \\ \text{s.t. } & a_i^\top u - a_i^\top v + b_i \leq \xi,\ i = 1, \ldots, t \\ & u, v \geq 0,\end{aligned}$$

with the variable of interest $w = u - v$.

References

N. Abe, J. Takeuchi, and M. K. Warmuth. Polynomial Learnability of Stochastic Rules with Respect to the KL-Divergence and Quadratic Distance. *IEICE Transactions on Information and Systems*, 84(3):299–316, 2001.

G. Amdahl. Validity of the single processor approach to achieving large-scale computing capabilities. In *AFIPS Conference Proceedings*, volume 30, pages 483–485, 1967.

M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nature Genetics*, 25:25–29, 2000.

G. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. MIT Press, Cambridge, Massachusetts, 2007.

O. E. Barndorff-Nielsen. *Information and Exponential Families in Statistical Theory*. John Wiley and Sons, New York, 1978.

J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the International Conference on Machine Learning*, pages 65–72, New York, NY, 2004. ACM Press.

A. Belloni. Introduction to bundle methods. Technical report, Operation Research Center, M.I.T., 2005.

S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Eighth IEEE International Conference on Computer Vision*, Vancouver, Canada, July 2001.

K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods Software*, 1:23–34, 1992.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.

J. S. Breese, D. Heckerman, and C. Kardie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

T. Caetano, L. Cheng, Q. V. Le, and A. J. Smola. Learning graph matching. In *Proceedings of the 11th International Conference On Computer Vision*, pages 1–8, Los Alamitos, CA, 2007. IEEE Computer Society.

T. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola. Learning graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008. submitted.

L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the Thirteenth ACM conference on Information and knowledge management*, pages 78–87, New York, NY, USA, 2004. ACM Press.

E. Candes and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

C. C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5):1155–1178, 2007.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. In *Proceedings of the 13th Annual Conference on Computational Learning Theory*, pages 158–169. Morgan Kaufmann, San Francisco, 2000.

C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20(3):273–297, 1995.

R. Cowell, A. Dawid, S. Lauritzen, and D. Spiegelhalter. *Probabilistic Networks and Expert Sytems*. Springer, New York, 1999.

K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991, January 2003.

K. Crammer and Y. Singer. Online ranking by projecting. *Neural Computation*, 17(1):145–175, 2005.

N. A. C. Cressie. *Statistics for Spatial Data*. John Wiley and Sons, New York, 1993.

L. Fahrmeir and G. Tutz. *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer, 1994.

R.-E. Fan, J.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, August 2008.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, Dec 2001.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for support vector machines. In A. McCallum and S. Roweis, editors, *Proceedings of the International Conference on Machine Learning*, pages 320–327. Omnipress, 2008.

A. Frangioni. *Dual-Ascent Methods and Multicommodity Flow Problems*. PhD thesis, Dipartimento di Informatica, Universita' di Pisa, 1997. TD 5/97.

W. Gropp, E. Lusk, and R. Thakur. *Using MPI-2*. MIT Press, 1999.

R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 115–132, Cambridge, MA, 2000. MIT Press.

J. B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms, I and II*, volume 305 and 306. Springer-Verlag, 1993.

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 W. Cohen, A. McCallum, and S. Roweis, editors, *icml*, pages 408–415. ACM, 2008.

K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *ACM Special Interest Group in Information Retrieval (SIGIR)*, pages 41–48, New York, 2002. ACM.

T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the International Conference on Machine Learning*, pages 377–384, San Francisco, California, 2005. Morgan Kaufmann Publishers.

T. Joachims. Training linear SVMs in linear time. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2006.

T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

T. Joachims, T. Finley, and C.-N. Yu. Cutting-plane traning of structural SVMs. *Machine Learning*, 76(1), 2009.

R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38:325–340, 1987.

M. I. Jordan. *An Introduction to Probabilistic Graphical Models*. MIT Press, 2002. To Appear.

R. M. Karp. An algorithm to solve the $m \times n$ assignment problem in expected time $O(mn \log n)$. *Networks*, 10(2):143–152, 1980.

S. S. Keerthi and D. DeCoste. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

J. E. Kelly. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, December 1960.

K. C. Kiwiel. An aggregate subgradient method for nonsmooth convex minimization. *Mathematical Programming*, 27:320–341, 1983.

K. C. Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46:105–122, 1990.

R. Koenker. *Quantile Regression*. Cambridge University Press, 2005.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

J. D. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic modeling for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, volume 18, pages 282–289, San Francisco, CA, 2001. Morgan Kaufmann.

Q. V. Le and A. J. Smola. Direct optimization of ranking measures. *Journal of Machine Learning Research*, 2007. submitted.

C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69:111–147, 1995.

C.-J. Lin, R. C. Weng, and S. S. Keerthi. Trust region newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, April 2008.

L. Lukšan and J. Vlček. A bundle-Newton method for nonsmooth unconstrained minimization. *Mathematical Programming*, 83(3):373–391, 1998.

O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

D. McAllester. Generalization bounds and consistency for structured labeling. In *Predicting Structured Data*, Cambridge, Massachusetts, 2007. MIT Press.

T. P. Minka. A comparison of numerical optimizers for logistic regression. Technical report, Microsoft Research, 2007.

K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, editors, *Artificial Neural Networks ICANN'97*, volume 1327 of *Lecture Notes in Comput. Sci.*, pages 999–1004, Berlin, 1997. Springer-Verlag.

J. Munkres. Algorithms for the assignment and transportation problems. *Journal of SIAM*, 5(1): 32–38, 1957.

A. Nedich and D. P Bertsekas. Convergence rate of incremental subgradient algorithms. In S. Uryasev and P. M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications*, pages 263–304. Kluwer Academic Publishers, 2000.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

J. B. Orlin and Y. Lee. Quickmatch: a very fast algorithm for the assignment problem. Working Paper 3547-93, Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA, March 1993.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA, 2008.

N. Ratliff, J. Bagnell, and M. Zinkevich. Maximum margin planning. In *Proceedings of the International Conference on Machine Learning*, July 2006.

N. Ratliff, J. Bagnell, and M. Zinkevich. (online) subgradient methods for structured prediction. In *Eleventh International Conference on Artificial Intelligence and Statistics (AIStats)*, March 2007.

G. Rätsch, S. Sonnenburg, J. Srinivasan, H. Witte, K.-R. Müller, R. J. Sommer, and B. Schölkopf. Improving the Caenorhabditis elegans genome annotation using machine learning. *PLoS Computational Biology*, 3(2):e20 doi:10.1371/journal.pcbi.0030020, 2007.

B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results. *SIAM Journal of Optimization*, 2:121–152, 1992.

F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL*, pages 213–220, Edmonton, Canada, 2003. Association for Computational Linguistics.

S. Shalev-Schwartz and N. Srebro. SVM optimization: inverse dependence on training set size. In W. Cohen, A. McCallum, and S. Roweis, editors, *Proceedings of the International Conference on Machine Learning*. Omnipress, 2008.

S. Shalev-Shwartz and Y. Singer. Online learning meets optimization in the dual. In H.U. Simon and G. Lugosi, editors, *Computational Learning Theory (COLT)*, LNCS. Springer, 2006. extended version.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In *Proceedings of the International Conference on Machine Learning*, 2007.

Q. Shi, L. Wang, L. Cheng, and A. J. Smola. Discriminative human action segmentation and recognition using semi-markov model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, AK, 2008.

V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear SVMs. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 477–484, New York, NY, USA, 2006. ACM Press.

A. J. Smola, S. V. N. Vishwanathan, and Q. V. Le. Bundle methods for machine learning. In D. Koller and Y. Singer, editors, *Advances in Neural Information Processing Systems 20*, Cambridge MA, 2007. MIT Press.

I. Takeuchi, Q. V. Le, T. Sears, and A. J. Smola. Nonparametric quantile estimation. *Journal of Machine Learning Research*, 7, 2006.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 25–32, Cambridge, MA, 2004. MIT Press.

C.-H. Teo, Q. V. Le, A. J. Smola, and S. V. N. Vishwanathan. A scalable modular convex solver for regularized risk minimization. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2007.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Section B (Statistical Methodology)*, 58:267–288, 1996.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

V. Vapnik, S. Golowich, and A. J. Smola. Support vector method for function approximation, regression estimation, and signal processing. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 281–287, Cambridge, MA, 1997. MIT Press.

E. Voorhees. Overview of the TREC 2001 question answering track. In *TREC*, 2001.

G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. Technical Report 984, Department of Statistics, University of Wisconsin, Madison, 1997.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, UC Berkeley, Department of Statistics, September 2003.

C. K. I. Williams. Prediction with Gaussian processes: from linear regression to linear prediction and beyond. In M. I. Jordan, editor, *Learning and Inference in Graphical Models*, pages 599–621. Kluwer Academic, 1998.

J. Yu, S. V. N. Vishwanathan, S. Günter, and N. N. Schraudolph. A quasi-Newton approach to nonsmooth convex optimization. In A. McCallum and S. Roweis, editors, *ICML*, pages 1216–1223. Omnipress, 2008.

T. Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transactions on Information Theory*, 49(3):682–691, 2003.

Journal of Machine Learning Research 11 (2010) 367-410 Submitted 11/08; Revised 8/09; Published 1/10

A Convergent Online Single Time Scale Actor Critic Algorithm

Dotan Di Castro DOT@TX.TECHNION.AC.IL
Ron Meir RMEIR@EE.TECHNION.AC.IL
Department of Electrical Engineering
Technion, Haifa 32000, Israel

Editor: Peter Dayan

Abstract

Actor-Critic based approaches were among the first to address reinforcement learning in a general setting. Recently, these algorithms have gained renewed interest due to their generality, good convergence properties, and possible biological relevance. In this paper, we introduce an online temporal difference based actor-critic algorithm which is proved to converge to a neighborhood of a local maximum of the average reward. Linear function approximation is used by the critic in order estimate the value function, and the temporal difference signal, which is passed from the critic to the actor. The main distinguishing feature of the present convergence proof is that both the actor and the critic operate on a similar time scale, while in most current convergence proofs they are required to have very different time scales in order to converge. Moreover, the same temporal difference signal is used to update the parameters of both the actor and the critic. A limitation of the proposed approach, compared to results available for two time scale convergence, is that convergence is guaranteed only to a neighborhood of an optimal value, rather to an optimal value itself. The single time scale and identical temporal difference signal used by the actor and the critic, may provide a step towards constructing more biologically realistic models of reinforcement learning in the brain.

Keywords: actor critic, single time scale convergence, temporal difference

1. Introduction

In Reinforcement Learning (RL) an agent attempts to improve its performance over time at a given task, based on continual interaction with the (usually unknown) environment (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998). Formally, it is the problem of mapping situations to actions in order to maximize a given average reward signal. The interaction between the agent and the environment is modeled mathematically as a Markov Decision Process (MDP). Approaches based on a direct interaction with the environment, are referred to as *simulation based algorithms*, and will form the major focus of this paper.

A well known subclass of RL approaches consists of the so called actor-critic (AC) algorithms (Sutton and Barto, 1998), where the agent is divided into two components, an actor and a critic. The critic functions as a value estimator, whereas the actor attempts to select actions based on the value estimated by the critic. These two components solve their own problems separately but interactively. Many methods for solving the critic's value estimation problem, for a *fixed* policy, have been proposed, but, arguably, the most widely used is *temporal difference* (TD) learning. TD learning was demonstrated to accelerate convergence by trading bias for variance effectively (Singh and Dayan, 1998), and is often used as a component of AC algorithms.

In general, policy selection may be randomized. When facing problems with a large number of states or actions (or even continuous state-action problems), effective policy selection may suffer from several problems, such as slow convergence rate or an inefficient representation of the policy. A possible approach to policy learning is the so-called *policy gradient method* (Baxter and Bartlett, 2001; Cao, 2007; Cao and Chen, 1997; Konda and Tsitsiklis, 2003; Marbach and Tsitsiklis, 1998). Instead of maintaining a separate estimate for the value for each state (or state-action pair), the agent maintains a parametrized policy function. The policy function is taken to be a differentiable function of a parameter vector and of the state. Given the performance measure, depending on the agent's policy parameters, these parameters are updated using a sampling-based estimate of the gradient of the average reward. While such approaches can be proved to converge under certain conditions (Baxter and Bartlett, 2001), they often lead to slow convergence, due to very high variance. A more general approach based on sensitivity analysis, which includes policy gradient methods as well as non-parametric average reward functions, has been discussed in depth in the recent manuscript by Cao (2007).

Several AC algorithms with associated convergence proofs have been proposed recently (a short review is given in Section 2.2). As far as we are aware, all the convergence results for these algorithms are based on two time scales, specifically, the actor is assumed to update its internal parameters on a much slower time scale than the one used by the critic. The intuitive reason for this time scale separation is clear, since the actor improves its policy based on the critic's estimates. It can be expected that rapid change of the policy parameters may not allow the critic to effectively evaluate the value function, which may lead to instability when used by the actor in order to re-update its parameters.

The objective of this paper is to propose an online AC algorithm and establish its convergence under conditions which do *not* require the separation into two time scales. In this context we note that recent work by Mokkadem and Pelletier (2006), based on earlier research by Polyak and colleagues, has demonstrated that combing the two-time scale approach with the averaging method of Polyak (1990), can lead to the single time scale convergence at the optimal rate. In these works the rate of convergence is defined in terms of convergence in distribution, while the present work focuses on convergence with probability 1. As far as we are aware, no rates of convergence are currently known for two time scale approaches in the latter, stronger, setting. In fact, our motivation for the current direction was based on the possible relevance of AC algorithms in a biological context (Daw et al., 2006), where it would be difficult to justify two very different time scales operating within the same anatomical structure.[1] We refer the reader to DiCastro et al. (2008) for some preliminary ideas and references related to these issues. Given the weaker conditions assumed on the time scales, our convergence result is, not surprisingly, somewhat weaker than that provided recently in Bhatnagar et al. (2008, 2009), as we are not ensured to converge to a local optimum, but only to a neighborhood of such an optimum. Nevertheless, it is shown that the neighborhood size can be algorithmically controlled. Further comparative discussion can be found in Section 2.

This paper is organized as follows. In Section 2 we briefly recapitulate current AC algorithms for which convergence proofs are available. In Section 3, we formally introduce the problem setup. We begin Section 4 by relating the TD signal to the gradient of the average reward, and then move on to motivate and derive the main AC algorithm, concluding the section with a convergence proof. A comparative discussion of the main features of our approach is presented in Section 5, followed

1. Note that the results in Mokkadem and Pelletier (2006), while providing optimal single time scale convergence, still rely on an underlying two time scale algorithm

by some simulation results in Section 6. Finally, in Section 7, we discuss the results and point out possible future work. In order to facilitate the readability of the paper, we have relegated all technical proofs to appendices.

2. Previous Work

In this section we briefly review some previous work in RL which bears direct relevance to our work. While many AC algorithms have been introduced over the years, we focus only on those for which a convergence proof is available, since the main focus of this work is on convergence issues, rather than on establishing the most practically effective algorithms. See, for example, Peters and Schaal (2008), for promising applications of AC algorithms in a robotic setting.

2.1 Direct Policy Gradient Algorithms

Direct policy gradient algorithms, employing agents which consist of an actor only, typically estimate a noisy gradient of the average reward, and are relatively close in their characteristics to AC algorithms. The main difference from the latter is that the agent does not maintain a separate value estimator for each state, but rather interacts with the environment directly, and in a sense maintains its value estimate implicitly through a mapping which signifies which path the agent should take in order to maximize its average reward per stage.

Marbach and Tsitsiklis (1998) suggested an algorithm for non-discounted environments. The gradient estimate is based on an estimate of the state values which the actor estimates while interacting with the environment. If the actor returns to a sequence of previously visited states, it re-estimates the states value, not taking into account its previous visits. This approach often results in large estimation variance.

Baxter and Bartlett (2001) proposed an online algorithm for partially observable MDPs. In this algorithm, the agent estimates the expected average reward for the non-discounted problems through an estimate of the value function of a related discounted problem. It was shown that when the discount factor approaches 1, the related discounted problem approximates the average reward per stage. Similar to the algorithms of Marbach and Tsitsiklis (1998), it suffers from relatively large estimation variance. Greensmith et al. (2004) have proposed a method for coping with the large variance by adding a baseline to the value function estimation.

2.2 Actor Critic Algorithms

As stated in Section 1, the convergence proofs of which we are aware for AC algorithms are based on two time scale stochastic approximation (Borkar, 1997), where the actor is assumed to operate on a time scale which is much slower than that used by the critic.

Konda and Borkar (1999) suggested a set of AC algorithms. In two of their algorithms (Algorithms 3 and 6), parametrized policy based actors were used while the critic was based on a lookup table. Those algorithms and their convergence proofs were specific to the Gibbs policy function in the actor.

As far as we are aware, Konda and Tsitsiklis (2003) provided the first convergence proof for an AC algorithm based on function approximation. The information passed from the critic to the actor is the critic's action-value function, and the critic's basis functions, which are explicitly used by the actor. They provided a convergence proof of their TD(λ) algorithm where λ approaches 1.

A drawback of the algorithm is that the actor and the critic must share the information regarding the actor's parameters. This detailed information sharing is a clear handicap in a biological context, which was one of the driving forces for the present work.

Finally, Bhatnagar et al. (2008, 2009) recently proposed an AC algorithm which closely resembles our proposed algorithm, and which was developed independently of ours. In this work the actor uses a parametrized policy function while the critic uses a function approximation for the state evaluation. The critic passes to the actor the TD(0) signal and based on it the actor estimates the average reward gradient. A detailed comparison will be provided in Section 5. As pointed out in Bhatnagar et al. (2008, 2009), their work is the first to provide a convergence proof for an AC algorithm incorporating bootstrapping (Sutton and Barto, 1998), where bootstrapping refers to a situation where estimates are updated based on other estimates, rather than on direct measurements (as in Monte Carlo approaches). This feature applies to our work as well. We also note that Bhatnagar et al. (2008, 2009) extend their approach to the so-called natural gradient estimator, which has been shown to lead to improved convergence in supervised learning as well as RL. The present study focuses on the standard gradient estimate, leaving the extension to natural gradients to future work.

3. The Problem Setup

In this section we describe the formal problem setup, and present a sequence of assumptions and lemmas which will be used in order to prove convergence of Algorithm 1 in Section 4. These assumptions and lemmas mainly concern the properties of the controlled Markov chain, which represents the environment, and the properties of the actor's parametrized policy function.

3.1 The Dynamics of the Environment and of the Actor

We consider an agent, composed of an actor and a critic, interacting with an environment. We model the environment as a *Markov Decision Process* (MDP) (Puterman, 1994) in discrete time with a finite state set $\mathcal{X}$ and an action set $\mathcal{U}$, which may be uncountable. We denote by $|\mathcal{X}|$ the size of the set $\mathcal{X}$. Each selected action $u \in U$ determines a stochastic matrix $P(u) = [P(y|x,u)]_{x,y\in\mathcal{X}}$ where $P(y|x,u)$ is the transition probability from a state $x \in \mathcal{X}$ to a state $y \in \mathcal{X}$ given the control u. For each state $x \in \mathcal{X}$ the agent receives a corresponding reward $r(x)$, which may be deterministic or random. In the present study we assume for simplicity that the reward is deterministic, a benign assumption which can be easily generalized.

Assumption 1 *The rewards, $\{r(x)\}_{x\in\mathcal{X}}$, are uniformly bounded by a finite constant B_r.*

The actor maintains a *parametrized policy function*. A parametrized policy function is a conditional probability function, denoted by $\mu(u|x,\theta)$, which maps an observation $x \in \mathcal{X}$ into a control $u \in \mathcal{U}$ given a parameter $\theta \in \mathbb{R}^K$. The agent's goal is to adjust the parameter θ in order to attain maximum average reward over time. For each θ, we have a Markov Chain (MC) induced by $P(y|x,u)$ and $\mu(u|x,\theta)$. The state transitions of the MC are obtained by first generating an action u according to $\mu(u|x,\theta)$, and then generating the next state according to $\{P(y|x,u)\}_{x,y\in\mathcal{X}}$. Thus, the MC has a transition matrix $P(\theta) = [P(y|x,\theta)]_{x,y\in\mathcal{X}}$ which is given by

$$P(y|x,\theta) = \int_{\mathcal{U}} P(y|x,u)d\mu(u|x,\theta). \tag{1}$$

We denote the space of these transition probabilities by $\mathcal{P} = \{P(\theta)|\theta \in \mathbb{R}^K\}$, and its closure by $\bar{\mathcal{P}}$. The following assumption is needed in the sequel in order to prove the main results (Brémaud, 1999).

Assumption 2 *Each MC, $P(\theta) \in \bar{\mathcal{P}}$, is aperiodic, recurrent, and irreducible.*

As a result of Assumption 2, we have the following lemma regarding the stationary distribution and a common recurrent state.

Lemma 1 *Under Assumption 2 we have:*

1. Each MC, $P(\theta) \in \bar{\mathcal{P}}$, has a unique stationary distribution, denoted by $\pi(\theta)$, satisfying $\pi(\theta)'P(\theta) = \pi(\theta)'$.
2. There exists a state, denoted by x^*, which is recurrent for all $P(\theta) \in \bar{\mathcal{P}}$.

Proof For the first part see Corollary 4.1 in Gallager (1995). The second part follows trivially from Assumption 2. ■

The next technical assumption states that the first and second derivatives of the parametrized policy function are bounded, and is needed to prove Lemma 3 below.

Assumption 3 *The conditional probability function $\mu(u|x,\theta)$ is twice differentiable. Moreover, there exist positive constants, B_{μ_1} and B_{μ_2}, such that for all $x \in \mathcal{X}$, $u \in \mathcal{U}$, $\theta \in \mathbb{R}^K$ and $k_1 \geq 1$, $k_2 \leq K$ we have*

$$\left|\frac{\partial \mu(u|x,\theta)}{\partial \theta_k}\right| \leq B_{\mu_1}, \quad \left|\frac{\partial^2 \mu(u|x,\theta)}{\partial \theta_{k_1} \partial \theta_{k_2}}\right| \leq B_{\mu_2}.$$

A notational comment concerning bounds *Throughout the paper we denote upper bounds on different variables by the letter B, with a subscript corresponding to the variable itself. An additional numerical subscript, 1 or 2, denotes a bound on the first or second derivative of the variable. For example, B_f, B_{f_1}, and B_{f_2} denote the bounds on the function f and its first and second derivatives respectively.*

3.2 Performance Measures

Next, we define a performance measure for an agent in an environment. The *average reward per stage* of an agent which traverses a MC starting from an initial state $x \in \mathcal{X}$ is defined by

$$J(x,\theta) \triangleq \lim_{T\to\infty} \mathrm{E}\left[\frac{1}{T}\sum_{n=0}^{T-1} r(x_n)\middle| x_0 = x, \theta\right],$$

where $\mathrm{E}[\cdot|\theta]$ denotes the expectation under the probability measure $P(\theta)$, and x_n is the state at time n. The agent's goal is to find $\theta \in \mathbb{R}^K$ which maximizes $J(x,\theta)$. The following lemma shows that under Assumption 2, the average reward per stage does not depend on the initial state; see Bertsekas (2006), vol. II, Section 4.1.

Lemma 2 *Under Assumption 2 and based on Lemma 1, the average reward per stage, $J(x,\theta)$, is independent of the starting state, is denoted by $\eta(\theta)$, and satisfies $\eta(\theta) = \pi(\theta)'r$.*

Based on Lemma 2, the agent's goal is to find a parameter vector θ, which maximizes the average reward per stage $\eta(\theta)$. In the sequel we show how this maximization can be performed by optimizing $\eta(\theta)$, using $\nabla_\theta \eta(\theta)$. A consequence of Assumption 3 and the definition of $\eta(\theta)$ is the following lemma.

Lemma 3

1. For each $x, y \in \mathcal{X}$, $1 \leq i, j \leq K$, and $\theta \in \mathbb{R}^K$, the functions $\partial P(y|x,\theta)/\partial\theta_i$ and $\partial^2 P(y|x,\theta)/\partial\theta_i\partial\theta_j$ are uniformly bounded by B_{P_1} and B_{P_2} respectively.

   (a) For each $x \in \mathcal{X}$, $1 \leq i, j \leq K$, and $\theta \in \mathbb{R}^K$, the functions $\partial\pi(x|\theta)/\partial\theta_i$ and $\partial^2\pi(x|\theta)/\partial\theta_i\partial\theta_j$ are uniformly bounded by , B_{π_1} and B_{π_2} respectively.

   (b) For all $1 \leq i, j \leq K$, and $\theta \in \mathbb{R}^K$, the functions $\eta(\theta)$, $\partial\eta(\theta)/\partial\theta_i$, $\partial^2\pi(x|\theta)/\partial\theta_i\partial\theta_j$ are uniformly bounded by , B_η, B_{η_1} and B_{η_2} respectively.

   (c) For all $x \in \mathcal{X}$ and $\theta \in \mathbb{R}^K$, there exists a constant $b_\pi > 0$ such that $\pi(x|\theta) \geq b_\pi$.

The proof is technical and is given in Appendix A.1. For later use, we define the random variable T, which denotes the first return time to the recurrent state x^*. Formally,

$$T \triangleq \min\{k > 0 | x_0 = x^*, x_k = x^*\}.$$

It is easy to show that under Assumption 2, the average reward per stage can be expressed by

$$\eta(\theta) = \lim_{T\to\infty} \mathrm{E}\left[\frac{1}{T}\sum_{n=0}^{T-1} r(x_n)\middle| x_0 = x^*, \theta\right].$$

Next, we define the *differential value function* of state $x \in \mathcal{X}$ which represents the average differential reward the agent receives upon starting from a state x and reaching the recurrent state x^* for the first time. Mathematically,

$$h(x,\theta) \triangleq \mathrm{E}\left[\sum_{n=0}^{T-1}(r(x_n) - \eta(\theta))\middle| x_0 = x, \theta\right].$$

Abusing notation slightly, we denote $h(\theta) \triangleq (h(x_1,\theta),\ldots,h(x_{|\mathcal{X}|},\theta)) \in \mathbb{R}^{|\mathcal{X}|}$. For each $\theta \in \mathbb{R}^K$ and $x \in \mathcal{X}$, $h(x,\theta)$, $r(x)$, and $\eta(\theta)$ satisfy Poisson's equation, as in Theorem 7.4.1 in Bertsekas (2006), that is,

$$h(x,\theta) = r(x) - \eta(\theta) + \sum_{y\in\mathcal{X}} P(y|x,\theta)h(y,\theta). \tag{2}$$

Based on the differential value we define the *temporal difference* (TD) between the states $x \in \mathcal{X}$ and $y \in \mathcal{X}$ (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998),

$$d(x,y,\theta) \triangleq r(x) - \eta(\theta) + h(y,\theta) - h(x,\theta). \tag{3}$$

According to common wisdom, the TD is interpreted as a prediction error. The next lemma states the boundedness of $h(x,\theta)$ and its derivatives. The proof is given in Appendix A.2.

Lemma 4

1. The differential value function, $h(x,\theta)$, is bounded and has bounded first and second derivative. Mathematically, for all $x \in \mathcal{X}$, $1 \leq i,j \leq K$, and for all $\theta \in \mathbb{R}^K$ we have

$$|h(x,\theta)| \leq B_h, \quad \left|\frac{\partial h(x,\theta)}{\partial \theta_i}\right| \leq B_{h_1}, \quad \left|\frac{\partial^2 h(x,\theta)}{\partial \theta_i \partial \theta_j}\right| \leq B_{h_2}.$$

   (a) There exists a constant B_D such that or all $\theta \in \mathbb{R}^K$ we have $|d(x,y,\theta)| \leq B_D$, where $B_D = 2(B_r + B_h)$.

3.3 The Critic's Dynamics

The critic maintains an estimate of the environmental state values. It does so by maintaining a parametrized function which approximates $h(x,\theta)$, and is denoted by $\tilde{h}(x,w)$. The function $\tilde{h}(x,w)$ is a function of the state $x \in \mathcal{X}$ and a parameter $w \in \mathbb{R}^L$. We note that $h(x,\theta)$ is a function of θ, and is induced by the actor policy $\mu(u|x,\theta)$, while $\tilde{h}(x,w)$ is a function of w. Thus, the critic's objective is to find the parameter w which yields the best approximation of $h(\theta) = (h(x_1,\theta),\ldots,h(x_{|\mathcal{X}|},\theta))$, in a sense to be defined later. We denote this optimal vector by $w^*(\theta)$. An illustration of the interplay between the actor, critic, and the environment is given in Figure 1.

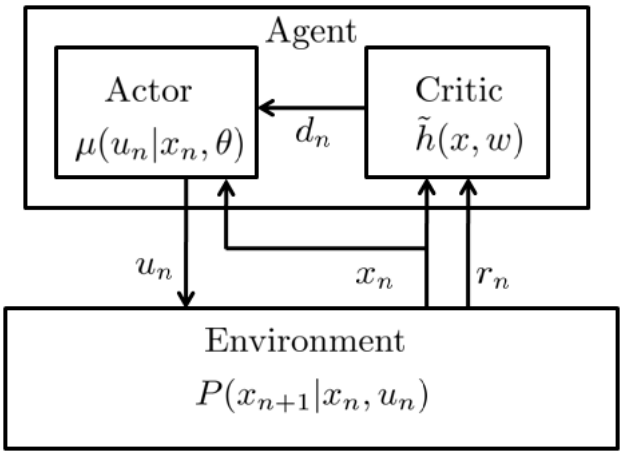


Figure 1: A schematic illustration of the dynamics between the actor, the critic, and the environment. The actor chooses an action, u_n, according to the parametrized policy $\mu(u|x,\theta)$. As a result, the environment proceeds to the next state according to the transition probability $P(x_{n+1}|x_n,u_n)$ and provides a reward. Using the TD signal, the critic improves its estimation for the environment state values while the actor improves its policy.

4. A Single Time Scale Actor Critic Algorithm with Linear Function Approximation

In this section, we present a version of an AC algorithm, along with its convergence proof. The core of the algorithm is based on (4) below, where the actor's estimate of $\nabla_\theta \eta(\theta)$ is based on the critic's estimate of the TD signal $d(x,y,\theta)$. The algorithm is composed of three iterates, one for the

actor and two for the critic. The actor maintains the iterate of the parameter vector θ corresponding to the policy $\mu(u|x,\theta)$, where its objective is to find the optimal value of θ, denoted by θ^*, which maximizes $\eta(\theta)$. The critic maintains the other two iterates. One iterate is used for estimating the average reward per stage, $\eta(\theta)$, where its estimate is denoted by $\tilde{\eta}$. The critic's second iterate maintains a parameter vector, denoted by $w \in \mathbb{R}^L$, which is used for the differential value estimate using a function approximator, denoted by $\tilde{h}(w)$. For each $\theta \in \mathbb{R}^K$, there exists a $w^*(\theta)$ which, under the policy induced by θ, is the optimal w for estimating $\tilde{\eta}(w)$. The critic's objective is to find the optimal $\tilde{\eta}$ and w.

4.1 Using the TD Signal to Estimate the Gradient of the Average Reward

We begin with a theorem which serves as the foundation for the policy gradient algorithm described in Section 4. The theorem relates the gradient of the average reward per stage, $\eta(\theta)$, to the TD signal. It was proved in Bhatnagar et al. (2008), and is similar in its structure to other theorems which connect $\eta(\theta)$ to the Q-value (Konda and Tsitsiklis, 2003), and to the differential value function (Cao, 2007; Marbach and Tsitsiklis, 1998).

We start with a definition of the *likelihood ratio derivative*

$$\psi(x,u,\theta) \triangleq \frac{\nabla_\theta \mu(u|x,\theta)}{\mu(u|x,\theta)},$$

where the gradient ∇_θ is w.r.t. θ, and $\psi(x,u,\theta) \in \mathbb{R}^K$. The following assumption states that $\psi(x,u,\theta)$ is bounded, and will be used to prove the convergence of algorithm 1.

Assumption 4 *For all $x \in \mathcal{X}$, $u \in \mathcal{U}$, and $\theta \in \mathbb{R}^K$, there exists a positive constant, B_ψ, such that*

$$\|\psi(x,u,\theta)\|_2 \leq B_\psi < \infty,$$

where $\|\cdot\|_2$ is the Euclidean L_2 norm.

Based on this, we present the following theorem which relates the gradient of $\eta(\theta)$ to the TD signal. For completeness, we supply a (straightforward) proof in Appendix B.

Theorem 5 *For any arbitrary function $f(x)$, the gradient w.r.t. θ of the average reward per stage can be expressed by*

$$\nabla_\theta \eta(\theta) = \sum_{x,y \in \mathcal{X}} P(x,u,y,\theta)\psi(x,u,\theta)d(x,y,\theta), \tag{4}$$

where $P(x,u,y,\theta)$ is the probability $\Pr(x_n = x, u_n = u, x_{n+1} = y)$ subject to the policy parameter θ.

4.2 The Updates Performed by the Critic and the Actor

We note that the following derivation regarding the critic is similar in some respects to the derivation in Section 6.3.3 in Bertsekas and Tsitsiklis (1996) and Tsitsiklis and Roy (1997). We define the following quadratic target function used to evaluate the critic's performance in assessing the differential value $h(\theta)$,

$$I(w,\theta) \triangleq \frac{1}{2}\sum_{x \in \mathcal{X}} \pi(x|\theta)\left(\tilde{h}(x,w) - h(x,\theta)\right)^2 . \tag{5}$$

The probabilities $\{\pi(x|\theta)\}_{x \in \mathcal{X}}$ are used in order to provide the proportional weight to the state estimates, according to the relative number of visits of the agent to the different states.

Limiting ourselves to the class of linear function approximations in the critic, we consider the following function for the differential value function

$$\tilde{h}(x,w) = \phi(x)'w, \tag{6}$$

where $\phi(x) \in \mathbb{R}^L$. We define $\Phi \in \mathbb{R}^{|\mathcal{X}|\times L}$ to be the matrix

$$\Phi \triangleq \begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & \phi_L(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \dots & \phi_L(x_2) \\ \vdots & \vdots & & \vdots \\ \phi_1(x_{|\mathcal{X}|}) & \phi_2(x_{|\mathcal{X}|}) & \dots & \phi_L(x_{|\mathcal{X}|}) \end{pmatrix},$$

where $\phi(\cdot)$ is a column vector. Therefore, we can express (6) in vector form as

$$\tilde{h}(w) = \Phi w,$$

where, abusing notation slightly, we set $\tilde{h}(w) = \left(\tilde{h}(x_1,w),\dots,\tilde{h}(x_{|\mathcal{X}|},w)\right)'$.

We wish to express (5), and the approximation process, in an appropriate Hilbert space. Define the matrix $\Pi(\theta)$ to be a diagonal matrix $\Pi(\theta) \triangleq \mathrm{diag}(\pi(\theta))$. Thus, (5) can be expressed as

$$I(w,\theta) = \frac{1}{2}\left\|\Pi(\theta)^{\frac{1}{2}}\left(h(\theta)-\Phi w\right)\right\|_2^2 \triangleq \frac{1}{2}\left\|h(\theta)-\Phi w\right\|_{\Pi(\theta)}^2. \tag{7}$$

In the sequel, we will need the following technical assumption.

Assumption 5

1. The columns of the matrix Φ are independent, that is, they form a basis of dimension L.

   (a) The norms of the column vectors of the matrix Φ are bounded above by 1, that is, $\|\phi_k\|_2 \leq 1$ for $1 \leq k \leq L$.

The parameter $w^*(\theta)$, which optimizes (7), can be directly computed, but involves inverting a matrix. Thus, in order to find the right estimate for $\tilde{h}(w)$, the following *gradient descent* (Bertsekas and Tsitsiklis, 1996) algorithm is suggested,

$$w_{n+1} = w_n - \gamma_n \nabla_w I(w_n,\theta), \tag{8}$$

where $\{\gamma_n\}_{n=1}^{\infty}$ is a positive series satisfying the following assumption, which will be used in proving the convergence of Algorithm 1.

Assumption 6 *The positive series* $\{\gamma_n\}_{n=1}^{\infty}$ *satisfies*

$$\sum_{n=1}^{\infty}\gamma_n = \infty, \quad \sum_{n=1}^{\infty}\gamma_n^2 < \infty.$$

Writing the term $\nabla_w I(w_n)$ explicitly yields

$$\nabla_w I(w_n) = \Phi' \Pi(\theta) \Phi w_n - \Phi' \Pi(\theta) h(\theta).$$

For each $\theta \in \mathbb{R}^K$, the value $w^*(\theta)$ is given by setting $\nabla_w I(w, \theta) = 0$, that is,

$$w^*(\theta) = \left(\Phi' \Pi(\theta) \Phi\right)^{-1} \Phi' \Pi(\theta) h(\theta).$$

Note that Bertsekas and Tsitsiklis (1996) prove that the matrix $\left(\Phi' \Pi(\theta) \Phi\right)^{-1} \Phi' \Pi(\theta)$ is a projection operator into the space spanned by Φw, with respect to the norm $\|\cdot\|_{\Pi(\theta)}$. Thus, the explicit gradient descent procedure (8) is

$$w_{n+1} = w_n - \gamma_n \Phi' \Pi(\theta) \left(\Phi w_n - h(\theta)\right). \tag{9}$$

Using the basis Φ, in order to approximates $h(\theta)$, yields an approximation error defined by

$$\varepsilon_{\text{app}}(\theta) \triangleq \inf_{w \in \mathbb{R}^L} \|h(\theta) - \Phi w\|_{\pi(\theta)} = \|h(\theta) - \Phi w^*(\theta)\|_{\pi(\theta)}.$$

We can bound this error by

$$\varepsilon_{\text{app}} \triangleq \sup_{\theta \in \mathbb{R}^K} \varepsilon_{\text{app}}(\theta). \tag{10}$$

The agent cannot access $h(x, \theta)$ directly. Instead, it can interact with the environment in order to estimate $h(x, \theta)$. We denote by $\hat{h}_n(x)$ the estimate of $h(x, \theta)$ at time step n, thus (9) becomes

$$w_{n+1} = w_n + \gamma_n \Phi' \Pi(\theta) \left(\hat{h}_n - \Phi w_n\right).$$

This procedure is termed *stochastic gradient descent* (Bertsekas and Tsitsiklis, 1996).

There exist several estimators for $\hat{h}_n$. One sound method, which performs well in practical problems (Tesauro, 1995), is the TD(λ) method; see Section 5.3.2 and 6.3.3 in Bertsekas and Tsitsiklis (1996), or Chapter 6 in Sutton and Barto (1998), where the parameter λ satisfies $0 \leq \lambda \leq 1$. This method devises an estimator which is based on previous estimates of $h(w)$, that is, w_n, and is based also on the environmental reward $r(x_n)$. This idea is a type of *a bootstrapping* algorithm, that is, using existing estimates and new information in order to build more accurate estimates; see Sutton and Barto (1998), Section 6.1.

The TD(λ) estimator for $\hat{h}_{n+1}$ is

$$\hat{h}_{n+1}(x_n) \quad = \quad (1-\lambda) \sum_{k=0}^{\infty} \lambda^k \hat{h}_{n+1}^{(k)}(x_n), \tag{11}$$

where the *k-steps predictor* is defined by

$$\hat{h}_{n+1}^{(k)}(x_n) = \left(\sum_{m=0}^{k} r(x_{n+m}) + \hat{h}_n(x_{n+k+1}) \right).$$

The idea of bootstrapping is apparent in (11): the predictor for the differential value of the state x_n at the $(n+1)$-Th time step, is based partially on the previous estimates through $\hat{h}_n(x_{n+k+1})$, and partially on new information, that is, the reward $r(x_{n+m})$. In addition, the parameter λ gives an

exponential weighting for the different k-step predictors. Thus, choosing the right λ can yield better estimators.

For the discounted setting, it was proved by Bertsekas and Tsitsiklis (1996) (p. 295) that an algorithm which implements the TD(λ) estimator (11) online and converges to the right value is the following one

$$\begin{aligned} w_{n+1} &= w_n + \gamma_n d_n e_n, \\ e_n &= \alpha\lambda e_{n-1} + \phi(x_n), \end{aligned} \tag{12}$$

where d_n is the temporal difference between the n-th and the $(n+1)$-th cycle, and e_n is the so-called *eligibility trace*; see Sections 5.3.3 and 6.3.3 in Bertsekas and Tsitsiklis (1996) or Chapter 7 in Sutton and Barto (1998), and the parameter α is the discount factor. The eligibility trace is an auxiliary variable, which is used in order to implement the idea of (11) as an online algorithm. As the name implies, the eligibility variable measures how eligible is the TD variable, d_n, in (12).

In our setting, the non-discounted case, the analogous equations for the critic, are

$$\begin{aligned} w_{n+1} &= w_n + \gamma_n \tilde{d}(x_n, x_{n+1}, w_n)\, e_n \\ \tilde{d}(x_n, x_{n+1}, w_n) &= r(x_n) - \tilde{\eta}_m + \tilde{h}(x_{n+1}, w_m) - \tilde{h}(x_n, w_m) \\ e_n &= \lambda e_{n-1} + \phi(x_n). \end{aligned}$$

The actor's iterate is motivated by Theorem 5. Similarly to the critic, the actor executes a stochastic gradient ascent step in order to find a local maximum of the average reward per stage $\eta(\theta)$. Therefore,

$$\theta_{n+1} = \theta_n + \gamma_n \psi(x_n, u_n, \theta_n) \tilde{d}_n(x_n, x_{n+1}, w_n),$$

where ψ is defined in Section 4.1. A summary of the algorithm is presented in Algorithm 1.

4.3 Convergence Proof for the AC Algorithm

In the remainder of this section, we state the main theorems related to the convergence of Algorithm 1. We present a sketch of the proof in this section, where the technical details are relegated to Appendices C and D. The proof is divided into two stages. In the first stage we relate the stochastic approximation to a set of ordinary differential equations (ODE). In the second stage, we find conditions under which the ODE system converges to a neighborhood of the optimal $\eta(\theta)$.

The ODE approach is a widely used method in the theory of stochastic approximation for investigating the asymptotic behavior of stochastic iterates, such as (13)-(15). The key idea of the technique is that the iterate can be decomposed into a mean function and a noise term, such as a martingale difference noise. As the iterates advance, the effect of the noise weakens due to repeated averaging. Moreover, since the step size of the iterate decreases (e.g., γ_n in (13)-(15)), one can show that asymptotically an interpolation of the iterates converges to a continuous solution of the ODE. Thus, the first part of the convergence proof is to find the ODE system which describes the asymptotic behavior of Algorithm 1. This ODE will be presented in Theorem 7. In the second part we use ideas from the theory of Lyapunov functions in order to characterize the relation between the constants, $|\mathcal{X}|$, Γ_η, Γ_w, etc., which ensure convergence to some neighborhood of the maximum point satisfying $\|\nabla_\theta \eta(\theta)\|_2 = 0$. Theorem 8 states conditions on this convergence.

Algorithm 1 TD AC Algorithm

Given:

- An MDP with a finite set $\mathcal{X}$ of states satisfying Assumption 2.
- An actor with a parametrized policy $\mu(u|x,\theta)$ satisfying Assumptions 3 and 4.
- A critic with a linear basis for $\tilde{h}(w)$, that is, $\{\phi\}_{i=1}^{L}$, satisfying Assumption 5.
- A set H, a constant B_w, and an operator Ψ_w according to Definition 6.
- Step parameters Γ_η and Γ_w.
- Choose a TD parameter $0 \leq \lambda < 1$.

For step $n = 0$:

- Initiate the critic and the actor variables: $\tilde{\eta}_0 = 0$,$w_0 = 0, e_0 = 0, \theta_0 = 0$.

For each step $n = 1, 2, \ldots$

Critic: Calculate the estimated TD and eligibility trace

$$\begin{aligned} \tilde{\eta}_{n+1} &= \tilde{\eta}_n + \gamma_n \Gamma_\eta \left(r(x_n) - \tilde{\eta}_n \right) \\ \tilde{h}(x, w_n) &= w_n' \phi(x), \\ \tilde{d}\left(x_n, x_{n+1}, w_n\right) &= r(x_n) - \tilde{\eta}_n + \tilde{h}(x_{n+1}, w_n) - \tilde{h}(x_n, w_n), \\ e_n &= \lambda e_{n-1} + \phi\left(x_n\right). \end{aligned} \tag{13}$$

Set,

$$w_{n+1} = w_n + \gamma_n \Gamma_w \tilde{d}\left(x_n, x_{n+1}, w_n\right) e_n \tag{14}$$

Actor:

$$\theta_{n+1} = \theta_n + \gamma_n \psi(x_n, u_n, \theta_n) \tilde{d}_n(x_n, x_{n+1}, w_n) \tag{15}$$

Project each component of w_{m+1} onto H (see Definition 6)

4.3.1 RELATE THE ALGORITHM TO AN ODE

In order to prove the convergence of this algorithm to the related ODE, we need to introduce the following assumption, which adds constraints to the iteration for w, and will be used in the sequel to prove Theorem 7. This assumption may seem restrictive at first but in practice it is not. The reason is that we usually assume the bounds of the constraints to be large enough so the iterates practically do not reach those bounds. For example, under Assumption 2 and additional mild assumptions, it is easy to show that $h(\theta)$ is uniformly bounded for all $\theta \in \mathbb{R}^K$. As a result, there exist a constant bounding $w^*(\theta)$ for all $\theta \in \mathbb{R}^K$. Choosing constraints larger than this constant will not influence the algorithm performance.

Definition 6 *Let us denote by* $\{w_i\}_{i=1}^{L}$ *the components of* w*, and choose a positive constant* B_w*. We define the set* $H \subset \mathbb{R}^K \times \mathbb{R}^L$ *to be*

$$H \triangleq \left\{(\theta, w) \,\middle|\, -\infty < \theta_i < \infty, \quad 1 \le i \le K, \quad -B_w \le w_j \le B_w, \quad 1 \le j \le L\right\},$$

and let Ψ_w *be an operator which projects* w *onto* H*, that is, for each Cramer's*$1 \le j \le L$, $\Psi_w w_j = \max(\min(w_j, B_w), -B_w)$.

The following theorem identifies the ODE system which corresponds to Algorithm 1. The detailed proof is given in Appendix C.

Theorem 7 *Define the following functions:*

$$\begin{aligned}
G(\theta) &= \Phi' \Pi(\theta) \sum_{m=0}^{\infty} \lambda^m P(\theta)^m, \\
D^{(x,u,y)}(\theta) &= \pi(x) P(u|x,\theta) P(y|x,u) \psi(x,u,\theta), \quad x,y \in \mathcal{X}, \quad u \in \mathcal{U}. \\
A(\theta) &= \Phi' \Pi(\theta) (M(\theta) - I) \Phi, \\
M(\theta) &= (1-\lambda) \sum_{m=0}^{\infty} \lambda^m P(\theta)^{m+1}, \\
b(\theta) &= \Phi' \Pi(\theta) \sum_{m=0}^{\infty} \lambda^m P(\theta)^m (r - \eta(\theta)).
\end{aligned} \tag{16}$$

Then,

1. Algorithm 1 converges to the invariant set of the following set of ODEs

$$\begin{cases}
\dot{\theta} = \nabla_\theta \eta(\theta) + \sum\limits_{x,y \in \mathcal{X} \times \mathcal{X}} D^{(x,u,y)}(\theta) \left(d(x,y,\theta) - \tilde{d}(x,y,w)\right), \\
\dot{w} = \Psi_w \left[\Gamma_w \left(A(\theta) w + b(\theta) + G(\theta)(\eta(\theta) - \tilde{\eta})\right)\right], \\
\dot{\tilde{\eta}} = \Gamma_\eta \left(\eta(\theta) - \tilde{\eta}\right),
\end{cases} \tag{17}$$

   with probability 1.

   (a) The functions in (16) are continuous with respect to θ.

4.3.2 INVESTIGATING THE ODE ASYMPTOTIC BEHAVIOR

Next, we quantify the asymptotic behavior of the system of ODEs in terms of the various algorithmic parameters. The proof of the theorem appears in Appendix D.

Theorem 8 *Consider the constants* Γ_η *and* Γ_w *as defined in Algorithm 1, and the function approximation bound* ε_{app} *as defined in* (10)*. Setting*

$$B_{\nabla\eta} \triangleq \frac{B_{\Delta td1}}{\Gamma_w} + \frac{B_{\Delta td2}}{\Gamma_\eta} + B_{\Delta td3} \varepsilon_{app},$$

where $B_{\Delta td1}, B_{\Delta td2}, B_{\Delta td3}$ *are a finite constants depending on the MDP and agent parameters. Then, the ODE system* (17) *satisfies*

$$\liminf_{t \to \infty} \|\nabla_\theta \eta(\theta_t)\| \le B_{\nabla\eta}.$$

Theorem 8 has a simple interpretation. Consider the trajectory $\eta(\theta_t)$ for large times, corresponding to the asymptotic behavior of η_n. The result implies that the trajectory visits a neighborhood of a local maximum infinitely often. Although it may leave the local vicinity of the maximum, it is guaranteed to return to it infinitely often. This occurs, since once it leaves the vicinity, the gradient of η points in a direction which has a positive projection on the gradient direction, thereby pushing the trajectory back to the vicinity of the maximum. It should be noted that in simulation (reported below) the trajectory usually remains within the vicinity of the local maximum, rarely leaving it. We also observe that by choosing appropriate values for Γ_η and Γ_w we can control the size of the ball to which the algorithm converges.

The key idea required to prove the Theorem is the following argument. If the trajectory does not satisfy $\|\nabla\eta(\theta)\|_2 \leq B_{\nabla\eta}$, we have $\dot{\eta}(\theta) > \varepsilon$ for some positive ε. As a result, we have a monotone function which increases to infinity, thereby contradicting the boundedness of $\eta(\theta)$. Thus, $\eta(\theta)$ must visit the set which satisfies $\|\nabla\eta(\theta)\|_2 \leq B_{\nabla\eta}$ infinitely often.

5. A Comparison to Other Convergence Results

In this section, we point out the main differences between Algorithm 1, the first algorithm proposed by Bhatnagar et al. (2009) and the algorithms proposed by Konda and Tsitsiklis (2003). The main dimensions along which we compare the algorithms are the time scale, the type of the TD signal, and whether the algorithm is on line or off line.

5.1 The Time Scale and Type of Convergence

As was mentioned previously, the algorithms of Bhatnagar et al. (2009) and Konda and Tsitsiklis (2003) need to operate in two time scales. More precisely, this refers to the following situation. Denote the time step of the critic's iteration by γ_n^c and the time step of the actor's iteration by γ_n^a, we have $\gamma_n^c = o(\gamma_n^a)$, that is,

$$\lim_{n\to\infty} \frac{\gamma_n^c}{\gamma_n^a} = 0.$$

The use of two time scales stems from the need of the critic to provide an accurate estimate of the state values, as in the work of Bhatnagar et al. (2009), or the state-action values, as in the work of Konda and Tsitsiklis (2003) before the actor uses them.

In the algorithm proposed here, a single time scale is used for the three iterates of Algorithm 1. We have $\gamma_n^a = \gamma_n$ for the actor iterate, $\gamma_n^{c,\eta} = \Gamma_\eta\gamma_n$ for the critic's η_n iterate, and $\gamma_n^{c,w} = \Gamma_w\gamma_n$ for the critic's w iterate. Thus,

$$\lim_{n\to\infty} \frac{\gamma_n^{c,\eta}}{\gamma_n^a} = \Gamma_\eta,$$
$$\lim_{n\to\infty} \frac{\gamma_n^{c,w}}{\gamma_n^a} = \Gamma_w.$$

with the ave

Due to the single time scale, Algorithm 1 has the potential to converge faster than algorithms based on two time scales, since both the actor and the critic may operate on the fast time scale. The drawback of Algorithm 1 is the fact that convergence to the optimal value cannot be guaranteed, as was proved by Bhatnagar et al. (2009) and by Konda and Tsitsiklis (2003). Instead, convergence to

a neighborhood in $\mathbb{R}^K$ around the optimal value is guaranteed. In order to make the neighborhood smaller, we need to choose Γ_η and Γ_w appropriately, as is stated in Theorem 8.

5.2 The TD Signal, the Information Passed Between the Actor and the Critic, and the Critic's Basis

The algorithm presented by Bhatnagar et al. (2009) is essentially a TD(0) algorithm, while the algorithm of Konda and Tsitsiklis (2003) is TD(1), Our algorithm is a TD(λ) for $0 \leq \lambda < 1$. A major difference between the approaches of Bhatnagar et al. (2009) and the present work, as compared to (Konda and Tsitsiklis, 2003), is the information passed from the critic to the actor. In the former cases, the information passed is the TD signal, while in the latter case the Q-value is passed. Additionally, in Bhatnagar et al. (2009) and in Algorithm 1 the critic's basis functions do not change through the simulation, while in Konda and Tsitsiklis (2003) the critic's basis functions are changed in each iteration according to the actor's parameter θ. Finally, we comment that Bhatnagar et al. (2009) introduced an additional algorithm, based on the so-called natural gradient, which led to improved convergence speed. In this work we limit ourselves to algorithms based on the regular gradient, and defer the incorporation of the natural gradient to future work. As stated in Section 1, our motivation in this work was the derivation of a single time scale online AC algorithm with guaranteed convergence, which may be applicable in a biological context. The more complex natural gradient approach seems more restrictive in this setting.

6. Simulations

We report empirical results applying Algorithm 1 to a set of abstract randomly constructed MDPs which are termed Average Reward Non-stationary Environment Test-bench or in short GARNET (Archibald et al., 1995). GARNET problems comprise a class of randomly constructed finite MDPs serving as a test-bench for control and RL algorithms optimizing the average reward per stage. A GARNET problem is characterized in our case by four parameters and is denoted by GARNET(X,U,B,σ). The parameter X is the number of states in the MDP, U is the number of actions, B is the branching factor of the MDP, that is, the number of non-zero entries in each line of the MDP's transition matrices, and σ is the variance of each transition reward.

We describe how a GARNET problem is generated. When constructing such a problem, we generate for each state a reward, distributed normally with zero mean and unit variance. For each state-action the reward is distributed normally with the state's reward as mean and variance σ^2. The transition matrix for each action is composed of B non-zero terms in each line which sum to one.

We note that a comparison was carried out by Bhatnagar et al. (2009) between their algorithm and the algorithm of Konda and Tsitsiklis (2003). We therefore compare our results directly to the more closely related former approach (see also Section 5.2).

We consider the same GARNET problems as those simulated by Bhatnagar et al. (2009). For completeness, we provide here the details of the simulation. For the critic's feature vector, we use a linear function approximation $\tilde{h}(x,w) = \phi(x)'w$, where $\phi(x) \in \{0,1\}^L$, and define l to be the number nonzero values in $\phi(x)$. The nonzero values are chosen uniformly at random, where any two states

have different feature vectors. The actor's feature vectors are of size $L \times |\mathcal{U}|$, and are constructed as

$$\xi(x,u) \triangleq (\overbrace{0,\ldots,0}^{L\times(u-1)}, \phi(x), \overbrace{0,\ldots,0}^{L\times(|\mathcal{U}|-u)}),$$
$$\mu(u|x,\theta) = \frac{e^{\theta'\xi(x,u)}}{\sum_{u'\in\mathcal{U}} e^{\theta'\xi(x,u')}}.$$

Bhatnagar et al. (2009) reported simulation results for two GARNET problems: GARNET$(30,4,2,0.1)$ and GARNET$(100,10,3,0.1)$. For the GARNET$(30,4,2,0.1)$ problem, Bhatnagar et al. (2009) used critic steps $\gamma_n^{c,w}$ and $\gamma_n^{c,\eta}$, and actor steps γ_n^a, where

$$\gamma_n^{c,w} = \frac{100}{1000+n^{2/3}}, \quad \gamma_n^{c,\eta} = 0.95\gamma_n^{c,w}, \quad \gamma_n^{a,\eta} = \frac{1000}{100000+n},$$

and for GARNET$(100,10,3,0.1)$ the steps were

$$\gamma_n^{c,w} = \frac{10^5}{10^6+n^{2/3}}, \quad \gamma_n^{c,\eta} = 0.95\gamma_n^{c,w}, \quad \gamma_n^{a,\eta} = \frac{10^6}{10^8+n}.$$

In our simulations we used a single time scale, γ_n, which was equal to $\gamma_n^{c,w}$ as used by Bhatnagar et al. (2009). The basis parameters for GARNET$(30,4,2,0.1)$ were $L=8$ and $l=3$, where for GARNET$(100,10,3,0.1)$ they were $L=20$ and $l=5$.

In Figures 2 we show results of applying Algorithm 1 (solid line) and algorithm 1 from Bhatnagar et al. (2009) (dashed line) on GARNET$(30,4,2,0.1)$ and GARNET$(100,10,3,0.1)$ problems. Each graph in Figure 2, represents an average of 100 independent simulations. Note that an agent with a uniform action selection policy will attain an average reward per stage of zero in these problems. Figure 3 presents similar results for GARNET$(30,15,15,0.1)$. We see from these results that in all simulations, during the initial phase, Algorithm 1 converges faster than algorithm 1 from Bhatnagar et al. (2009). The long term behavior is problem-dependent, as can be seen by comparing Figures 2 and 3; specifically, in Figure 2 the present algorithm converges to a higher value than Bhatnagar et al. (2009), while the situation is reversed in Figure 3. We refer the reader to Mokkadem and Pelletier (2006) for careful discussion of convergence rates for two time scales algorithms; a corresponding analysis of convergence rates for single time scale algorithms is currently an open problem.

The results displayed here suggest a possible avenue for combining both algorithms. More concretely, using the present approach may lead to faster initial convergence due to the single time scale setting, which allows both the actor and the critic to evolve rapidly, while switching smoothly to a two time scales approach as in Bhatnagar et al. (2009) will lead to asymptotic convergence to a point rather than to a region. This type of approach is reminiscent of the quasi-Newton algorithms in optimization, and is left for future work. As discussed in Section 5, we do not consider the natural gradient based algorithms from Bhatnagar et al. (2009) in this comparative study.

7. Discussion and Future Work

We have introduced an algorithm where the information passed from the critic to the actor is the temporal difference signal, while the critic applies a TD(λ) procedure. A policy gradient approach was used in order to update the actor's parameters, based on a critic using linear function approximation. The main contribution of this work is a convergence proof in a situation where both the actor

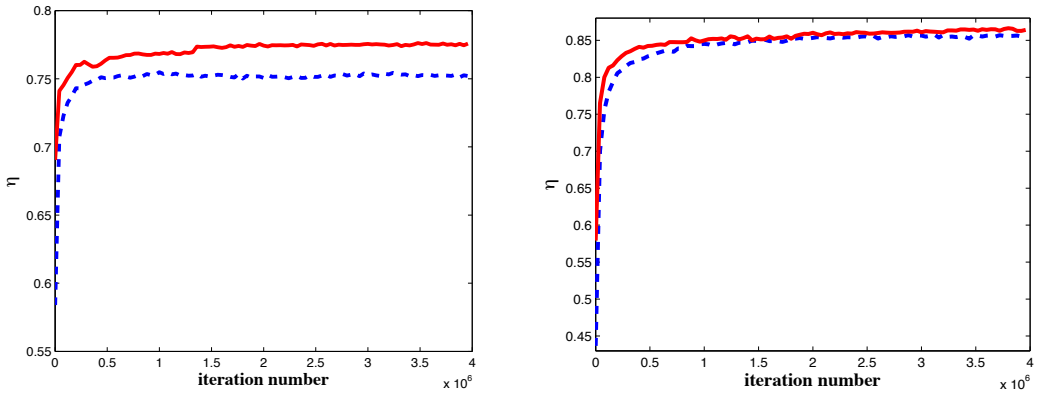


Figure 2: Simulation results applying Algorithm 1 (red solid line) and algorithm 1 from Bhatnagar et al. (2009) (blue dashed line) on a GARNET$(30,4,2,0.1)$ problem (a) and on GARNET$(100,10,3,0.1)$ problem (b). Standard errors of the mean (suppressed for visibility) are of the order of 0.04.

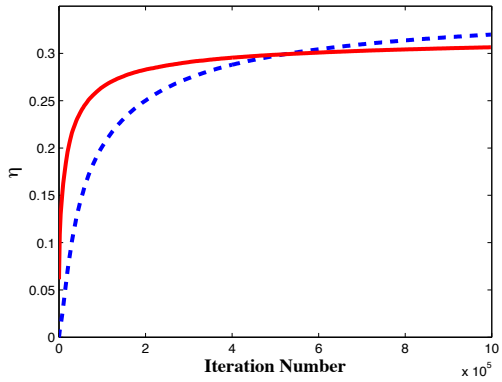


Figure 3: Simulation results applying Algorithm 1 (red solid line) and algorithm 1 from Bhatnagar et al. (2009) (blue dashed line) on a GARNET$(30,15,15,0.1)$ problem. Standard errors of the mean (suppressed for visibility) are of the order of 0.018.

and the critic operate on the same time scale. The drawback of the extra flexibility in time scales is that convergence is only guaranteed to a neighborhood of a local maximum value of the average reward per stage. However, this neighborhood depends on parameters which may be controlled to improve convergence.

This work sets the stage for much future work. First, as observed above, the size of the convergence neighborhood is inversely proportional to the step sizes Γ_w and Γ_η. In other words, in order to reduce this neighborhood we need to select larger values of Γ_w and Γ_η. This on the other hand increases the variance of the algorithm. Therefore, further investigation of methods which reduce

this variance are needed. However, the bounds used throughout are clearly rather loose, and cannot be effectively used in practical applications. Obviously, improving the bounds, and conducting careful numerical simulations in order to obtain a better practical understanding of the influence of the different algorithmic parameters, is called for. In addition, there is clearly room for combining the advantages of our approach with those of AC algorithms for which convergence to a single point is guaranteed, as discussed in Section 6,

From a biological point of view, our initial motivation to investigate TD based AC algorithms stemmed from questions related to the implementation of RL in the mammalian brain. Such a view is based on an interpretation of the transient activity of the neuromodulator dopamine as a TD signal (Schultz, 2002). Recent evidence suggested that the dorsal and ventral striatum may implement the actor and the critic, respectively Daw et al. (2006). We believe that theoretical models such as those of Bhatnagar et al. (2009) and Algorithm 1 may provide, even if partially, a firm foundation to theories at the neural level. Some initial attempts in a neural setting (using direct policy gradient rather than AC based approaches) have been made by Baras and Meir (2007) and Florian (2007). Such an approach may lead to functional insights as to how an AC paradigm may be implemented at the cellular level of the basal ganglia and cortex. An initial demonstration was given by DiCastro et al. (2008). From a theoretical perspective several issues remain open. First, strengthening Theorem 8 by replacing lim inf by lim would clearly be useful. Second, systematically combining the advantages of single time scale convergence (fast initial dynamics) and two time scale approaches (convergence to a point) would clearly be beneficial.

Acknowledgments

The authors are grateful to Mohammad Ghavamzadeh for sending them a copy of Bhatnagar et al. (2009) prior to publication, and to the anonymous reviewers for their helpful comments. The work of R. Meir was partially supported by an ISF Converging Technologies grant, and by ISF grant 665/08.

Appendix A. Proofs of Results from Section 3

We present proofs of the main results from section 3.

A.1 Proof of Lemma 3

1. Looking at (1) we see that $P(y|x,\theta)$ is a compound function of an integral and a twice differentiable function, $\mu(y|x,\theta)$, with bounded first and second derivatives according to Assumption 3. Therefore, $P(y|x,\theta)$ is a twice differentiable function with bounded first and second derivatives for all $\theta \in \mathbb{R}^K$.

2. According to Lemma 1, for each $\theta \in \mathbb{R}^K$ we have a unique solution to the following non-homogeneous linear equation system in $\{\pi(i|\theta)\}_{i=1}^{|\mathcal{X}|}$,

$$\begin{cases} \sum_{i=1}^{|\mathcal{X}|} \pi(i|\theta) P(j|i,\theta) = \pi(j|\theta), \quad j = 1,\ldots,|\mathcal{X}|-1, \\ \sum_{i=1}^{|\mathcal{X}|} \pi(i|\theta) = 1, \end{cases} \tag{18}$$

or in matrix form $M(\theta)\pi(\theta) = b$. By Assumption 2, the equation system (18) is invertible, therefore, $\det[M(\theta)] > 0$. This holds for all $P(\theta) \in \bar{P}$, thus, there exists a positive constant, b_M, which uniformly lower bounds $\det[M(\theta)]$ for all $\theta \in \mathbb{R}^K$.Thus, using Cramer's rule we have

$$\pi(i|\theta) = \frac{Q(i,\theta)}{\det[M(\theta)]},$$

where $Q(i,\theta)$ is a finite polynomial of $\{P(j|i,\theta)\}_{i,j\in\mathcal{X}}$ of at most degree $|\mathcal{X}|$ and with at most $|\mathcal{X}|!$ terms. Writing $\partial\pi(x|\theta)/\partial\theta_i$ explicitly gives

$$\begin{aligned} \left|\frac{\partial\pi(x|\theta)}{\partial\theta_i}\right| &= \left|\frac{\det[M(\theta)]\frac{\partial}{\partial\theta_i}Q(i,\theta) - Q(i,\theta)\frac{\partial}{\partial\theta_i}\det[M(\theta)]}{\det[M(\theta)]^2}\right| \\ &\leq \left|\frac{\frac{\partial}{\partial\theta_i}Q(i,\theta)}{\det[M(\theta)]}\right| + \left|\frac{Q(i,\theta)\frac{\partial}{\partial\theta_i}\det[M(\theta)]}{\det[M(\theta)]^2}\right| \\ &\leq \frac{|\mathcal{X}|\cdot|\mathcal{X}|!\cdot B_{P_1}}{b_M} + \frac{(|\mathcal{X}|\cdot|\mathcal{X}|!)\cdot B_{P_1}}{b_M^2}, \end{aligned}$$

which gives the desired bound. Following similar steps we can show the boundedness of the second derivatives.

3. The average reward per stage, $\eta(\theta)$ is a linear combination of $\{\pi(i|\theta)\}_{i=1}^{|\mathcal{X}|}$, with bounded coefficients by assumption 1. Therefore, using Section 2, $\eta(\theta)$ is twice differentiable with bounded first and second derivatives for all $\theta \in \mathbb{R}^K$.

4. Since $\pi(x|\theta)$ is the stationary distribution of a recurrent MC, according to Assumption 2 there is a positive probability to be in each state $x \in \mathcal{X}$. This applies to the closure of $\mathcal{P}$. Thus, there exist a positive constant b_π such that $\pi(x|\theta) \geq b_\pi$.

A.2 Proof of Lemma 4

1. We recall the Poisson Equation (2). We have the following system of linear equations in $\{h(x|\theta)\}_{x\in\mathcal{X}}$, namely,

$$\begin{cases} h(x|\theta) = r(x) - \eta(\theta) + \sum_{y\in\mathcal{X}} P(y|x,\theta)h(y|\theta), \quad \forall x \in \mathcal{X}, x \neq x^*, \\ h(x^*|\theta) = 0. \end{cases}$$

or in matrix form $N(\theta)h(\theta) = c$. Adding the equation $h(x^*|\theta) = 0$ yields a unique solution for the system; see Bertsekas (2006), Vol. 1, Prop. 7.4.1. Thus, using Cramer's rule we have

$h(x|\theta) = R(x,\theta)/\det[N(\theta)]$, where $R(x,\theta)$ and $\det[N(\theta)]$ are polynomial function of entries in $N(\theta)$, which are bounded and have bounded first and second derivatives according to Lemma 3. Continuing in the same steps of Lemma 3 proof, we conclude that $h(x|\theta)$ and its two first derivatives for all $x \in \mathcal{X}$ and for all $\theta \in \mathbb{R}^K$.

2. Trivially, by (3) and the previous section the result follows.

Appendix B. Proof of Theorem 5

We begin with a Lemma which was proved by Marbach and Tsitsiklis (1998). It relates the gradient of the average reward per stage to the differential value function.

Lemma 9 *The gradient of the average reward per stage can be expressed by*

$$\nabla_\theta \eta(\theta) = \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)h(y,\theta).$$

For completeness, we present a proof,which will be used in the sequel.
Proof We begin with Poisson's Equation (2) in vector form

$$h(\theta) = \bar{r} - e\eta(\theta) + P(\theta)h(\theta),$$

where e is a column vector of 1's. Taking the derivative with respect to θ and rearranging yields

$$e\nabla_\theta \eta(\theta) = -\nabla_\theta h(\theta) + \nabla_\theta P(\theta)h(\theta) + P(\theta)\nabla_\theta h(\theta).$$

Multiplying the left hand side of the last equation by the stationary distribution $\pi(\theta)'$ yields

$$\begin{aligned}\nabla_\theta \eta(\theta) &= -\pi(\theta)'\nabla_\theta h(\theta) + \pi(\theta)'\nabla_\theta P(\theta)h(\theta) + \pi(\theta)'P(\theta)\nabla_\theta h(\theta)\\ &= -\pi(\theta)'\nabla_\theta h(\theta) + \pi(\theta)'\nabla_\theta P(\theta)h(\theta) + \pi(\theta)'\nabla_\theta h(\theta)\\ &= \pi(\theta)'\nabla_\theta P(\theta)h(\theta).\end{aligned}$$

Expressing the result explicitly we obtain

$$\begin{aligned}\nabla_\theta \eta(\theta) &= \sum_{x,y\in\mathcal{X}} P(x)\nabla_\theta P(y|x,\theta)h(y,\theta)\\ &= \sum_{x,y\in\mathcal{X}} P(x)\nabla_\theta \left(\sum_u (P(y|x,u)\mu(u|x,\theta))\right) h(y,\theta)\\ &= \sum_{x,y\in\mathcal{X}} P(x)\sum_u (P(y|x,u)\nabla_\theta \mu(u|x,\theta))\, h(y,\theta)\\ &= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(y|x,u)P(x)\nabla_\theta \mu(u|x,\theta)h(y,\theta)\\ &= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(y|x,u)\mu(u|x,\theta)P(x)\frac{\nabla_\theta \mu(u|x,\theta)}{\mu(u|x,\theta)}h(y,\theta)\\ &= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)h(y,\theta).\end{aligned} \tag{19}$$

■

Based on this, we can now prove Theorem 5. We start with the result in (19).

$$\begin{aligned}
\nabla_\theta \eta(\theta) &= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)h(y,\theta). \\
&= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)\left(h(y,\theta)-h(x,\theta)+\bar{r}(x)-\eta(\theta)+f(x)\right) \\
&\quad - \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)\left(-h(x,\theta)+\bar{r}(x)-\eta(\theta)+f(x)\right) \\
&= \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)\left(d(x,y,\theta)+f(x)\right) \\
&\quad - \sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)\left(-h(x,\theta)+\bar{r}(x)-\eta(\theta)+f(x)\right)
\end{aligned}$$

In order to complete the proof, we show that the second term equals 0. We define $F(x,\theta) \triangleq -h(x|\theta)+\bar{r}(x)-\eta(\theta)+f(x)$ and obtain

$$\begin{aligned}
\sum_{x,y\in\mathcal{X},u\in\mathcal{U}} P(x,u,y,\theta)\psi(x,u,\theta)F(x,\theta) &= \sum_{x\in\mathcal{X}} \pi(x,\theta)F(x,\theta) \sum_{u\in\mathcal{U},y\in\mathcal{X}} \nabla_\theta P(y|x,u,\theta) \\
&= 0.
\end{aligned}$$

Appendix C. Proof of Theorem 7

As mentioned earlier, we use Theorem 6.1.1 from Kushner and Yin (1997). We start by describing the setup of the theorem and the main result. Then, we show that the required assumptions hold in our case.

C.1 Setup, Assumptions and Theorem 6.1.1 of Kushner and Yin (1997)

In this section we describe briefly but accurately the conditions for Theorem 6.1.1 of Kushner and Yin (1997) and state the main result. We consider the following stochastic iteration

$$y_{n+1} = \Pi_H[y_n + \gamma_n Y_n],$$

where Y_n is a vector of "observations" at time n, and Π_H is a constraint operator as defined in Definition 6. Recall that $\{x_n\}$ is a Markov chain. Based on this, define $\mathcal{F}_n$ to be the σ-algebra

$$\begin{aligned}
\mathcal{F}_n &\triangleq \sigma\{y_0, Y_{i-1}, x_i \,|\, i \le n\} \\
&= \sigma\{y_0, Y_{i-1}, x_i, y_i \,|\, i \le n\},
\end{aligned}$$

and

$$\bar{\mathcal{F}}_n \triangleq \sigma\{y_0, Y_{i-1}, y_i \,|\, i \le n\}.$$

The difference between the σ-algebras is the sequence $\{x_n\}$. Define the conditioned average iterate

$$g_n(y_n, x_n) \triangleq \mathrm{E}[Y_n \,|\, \mathcal{F}_n],$$

and the corresponding *martingale difference noise*

$$\delta M_n \triangleq Y_n - \mathrm{E}[Y_n \,|\, \mathcal{F}_n].$$

Thus, we can write the iteration as

$$y_{n+1} = y_n + \gamma_n \left(g_n \left(y_n, x_n \right) + \delta M_n + Z_n \right),$$

where Z_n is a reflection term which forces the iterate to the nearest point in the set H whenever the iterates leaves it (Kushner and Yin, 1997). Next, set

$$\bar{g}\left(y \right) \triangleq \mathrm{E}\left[g_n \left(y, x_n \right) \middle| \bar{\mathcal{F}}_n \right].$$

Later, we will see that the sum of the sequence $\{\delta M_n\}$ converges to 0, and the r.h.s of the iteration behaves approximately as a the function $\bar{g}\left(y \right)$, which yields the corresponding ODE, that is,

$$\dot{y} = \bar{g}\left(y \right).$$

The following ODE method will show that the asymptotic behavior of the iteration is equal to the asymptotic behavior of the corresponding ODE.

Define the auxiliary variable

$$t_n \triangleq \sum_{k=0}^{n-1} \gamma_{k,}$$

and the monotone piecewise constant auxiliary function

$$m\left(t \right) = \left\{ n \left| t_n \leq t < t_{n+1} \right. \right\}.$$

The following assumption, taken from Section 6.1 of Kushner and Yin (1997), is required to establish the basic Theorem. An interpretation of the assumption follows its statement.

Assumption 7 *Assume that*

1. The coefficients $\{\gamma_n\}$ satisfy $\sum_{n=1}^{\infty} \gamma_n = \infty$ and $\lim_{n\to\infty} \gamma_n = 0$.

   (a) $\sup_n \mathrm{E}\left[\left\| Y_n \right\| \right] < \infty$.

   (b) $g_n \left(y_n, x \right)$ is continuous in y_n for each x and n.

   (c) For each $\mu > 0$ and for some $T > 0$ there is a continuous function $\bar{g}\left(\cdot \right)$ such that for each y

   $$\lim_{n\to\infty} \Pr\left(\sup_{j\geq n} \max_{0\leq t\leq T} \left\| \sum_{i=m(jT)}^{m(jT+t)-1} \gamma_i \left(g_n \left(y, x_i \right) - \bar{g}\left(y \right) \right) \right\| \geq \mu \right) = 0.$$

   (d) For each $\mu > 0$ and for some $T > 0$ we have

   $$\lim_{n\to\infty} \Pr\left(\sup_{j\geq n} \max_{0\leq t\leq T} \left\| \sum_{i=m(jT)}^{m(jT+t)-1} \gamma_i \delta M_i \right\| \geq \mu \right) = 0.$$

   (e) There are measurable and non-negative functions $\rho_3\left(y \right)$ and $\rho_{n4}\left(x \right)$such that

   $$\left\| g_n \left(y_n, x \right) \right\| \leq \rho_3 \left(y \right) \rho_{n4} \left(x \right)$$

   where $\rho_3\left(y \right)$ is bounded on each bounded y-set , and for each $\mu > 0$ we have

   $$\lim_{\tau\to 0} \lim_{n\to\infty} \Pr\left(\sup_{j\geq n} \sum_{i=m(j\tau)}^{m(j\tau+\tau)-1} \gamma_i \rho_{n4} \left(x_i \right) \geq \mu \right) = 0.$$

(f) There are measurable and non-negative functions $\rho_1(y)$ and $\rho_{n2}(x)$such that $\rho_1(y)$ is bounded on each bounded y-set and

$$\|g_n(y_1,x)-g_n(y_2,x)\| \leq \rho_1(y_1-y_2)\,\rho_{n2}(x),$$

where

$$\lim_{y\to 0}\rho_1(y)=0,$$

and

$$\Pr\left(\limsup_j \sum_{i=j}^{m(t_j+\tau)} \gamma_i \rho_{i2}(x_i) < \infty\right) = 1.$$

The conditions of Assumption 7 are quite general but can be interpreted as follows. Assumptions 7.1-3 are straightforward. Assumption 7.4 is reminiscent of ergodicity, which is used to replace the state-dependent function $g_n(\cdot,\cdot)$ with the state-independent of state function $\bar{g}(\cdot)$, whereas Assumption 7.5 states that the martingale difference noise converges to 0 in probability. Assumptions 7.6 and 7.7 ensure that the function $g_n(\cdot,\cdot)$ is not unbounded and satisfies a Lipschitz condition.

The following Theorem, adapted from Kushner and Yin (1997), provides the main convergence result required. The remainder of this appendix shows that the required conditions in Assumption 7 hold.

Theorem 10 *(Adapted from Theorem 6.1.1 in Kushner and Yin 1997) Assume that algorithm 1, and Assumption 7 hold. Then y_n converges to some invariant set of the projected ODE*

$$\dot{y} = \Pi_H[\bar{g}(y)].$$

Thus, the remainder of this section is devoted to showing that Assumptions 7.1-7.7 are satisfied.

For future purposes, we express Algorithm 1 using the augmented parameter vector y_n

$$y_n \triangleq \begin{pmatrix}\theta_n' & w_n' & \tilde{\eta}_n'\end{pmatrix}', \quad \theta_n \in \mathbb{R}^K, \quad w_n \in \mathbb{R}^L, \quad \tilde{\eta}_n \in \mathbb{R}.$$

The components of Y_n are determined according to (17). The corresponding sub-vectors of $\bar{g}(y_n)$ will be denoted by

$$\bar{g}(y_n) = \begin{bmatrix}\bar{g}(\theta_n)' & \bar{g}(w_n)' & \bar{g}(\tilde{\eta}_n)'\end{bmatrix}' \in \mathbb{R}^{K+L+1},$$

and similarly

$$g_n(y_n,x_n) = \begin{bmatrix}g_n(\theta_n,x_n)' & g_n(w_n,x_n)' & g_n(\tilde{\eta}_n,x_n)'\end{bmatrix}' \in \mathbb{R}^{K+L+1}.$$

We begin by examining the components of $g_n(y_n,x_n)$ and $\bar{g}(y_n)$. The iterate $g_n(\tilde{\eta}_n,x_n)$ is

$$\begin{aligned} g_n(\tilde{\eta}_n,x_n) &= \mathrm{E}[\Gamma_\eta(r(x_n)-\tilde{\eta}_n)|\mathcal{F}_n] \\ &= \Gamma_\eta(r(x_n)-\tilde{\eta}_n), \end{aligned} \tag{20}$$

and since there is no dependence on x_n we have also

$$\bar{g}(\tilde{\eta}_n) = \Gamma_\eta(\eta(\theta)-\tilde{\eta}_n).$$

The iterate $g_n(w_n,x_n)$ is

$$\begin{aligned} g_n(w_n,x_n) &= \mathrm{E}\left[\Gamma_w \tilde{d}(x_n,x_{n+1},w_n)\, e_n \middle| \mathcal{F}_n\right] \\ &= \mathrm{E}\left[\Gamma_w \sum_{k=0}^{\infty} \lambda^k \phi(x_{n-k}) \left(r(x_n) - \tilde{\eta}_n + \phi(x_{n+1})' w_n - \phi(x_n)' w_n\right) \middle| \mathcal{F}_n\right] \\ &= \Gamma_w \sum_{k=0}^{\infty} \lambda^k \phi(x_{n-k}) \left(r(x_n) - \tilde{\eta}_n + \sum_{y\in\mathcal{X}} P(y|x_n,\theta_n)\,\phi(y)' w_n - \phi(x_n)' w_n\right), \end{aligned} \tag{21}$$

and the iterate $\bar{g}(w_n)$ is

$$\begin{aligned} \bar{g}(w_n) &= \mathrm{E}\left[g_n(w_n,x_n) \middle| \bar{\mathcal{F}}_n\right] \\ &= \mathrm{E}\left[\Gamma_w \sum_{k=0}^{\infty} \lambda^k \phi(x_{n-k}) \left(r(x_n) - \tilde{\eta}_n + \sum_{y\in\mathcal{X}} P(y|x_n,\theta_n)\,\phi(y)' w_n - \phi(x_n)' w_n\right) \middle| \bar{\mathcal{F}}\right] \\ &= \Gamma_w \sum_{k=0}^{\infty} \lambda^k \sum_{x\in\mathcal{X}} \pi(x)\phi(x) \sum_{z\in\mathcal{X}} \left[P^k\right]_{xz} \\ &\qquad \times \left(r(z) - \tilde{\eta}_n + \sum_{y\in\mathcal{X}} P(y|z,\theta_n)\,\phi(z)' w_n - \phi(y)' w_n\right), \end{aligned}$$

which, following Bertsekas and Tsitsiklis (1996) Section 6.3, can be written in matrix form

$$\bar{g}(w_n) = \Phi'\Pi(\theta_n) \left((1-\lambda)\sum_{k=0}^{\infty} \lambda^k P^{k+1} - I\right) \Phi w_n + \Phi'\Pi(\theta_n) \sum_{k=0}^{\infty} \lambda^k P^k (r - \tilde{\eta}_n).$$

With some further algebra we can express this using (16),

$$\bar{g}(w_n) = A(\theta_n)\, w_n + b(\theta_n) + G(\theta_n)\left(\eta(\theta_n) - \tilde{\eta}_n\right).$$

Finally, the iterate $g_n(\theta_n,x_n)$ is

$$\begin{aligned} g_n(\theta_n,x_n) &= \mathrm{E}\left[\tilde{d}(x_n,x_{n+1},w_n)\,\psi(x_n,u_n,\theta_n) \middle| \mathcal{F}_n\right] \\ &= \mathrm{E}\left[d(x_n,x_{n+1},\theta_n)\,\psi(x_n,u_n,\theta_n) \middle| \mathcal{F}_n\right] \\ &\quad + \mathrm{E}\left[\left(\tilde{d}(x_n,x_{n+1},w_n) - d(x_n,x_{n+1},\theta_n)\right)\psi(x_n,u_n,\theta_n) \middle| \mathcal{F}_n\right] \\ &= \mathrm{E}\left[d(x_n,x_{n+1},\theta_n)\,\psi(x_n,u_n,\theta_n) \middle| \mathcal{F}_n\right] \\ &\quad + \sum_{z\in\mathcal{X}} P(z|x_n)\,\psi(x_n,u_n,\theta_n)\left(\tilde{d}(x_n,z,w_n) - d(x_n,z,\theta_n)\right), \end{aligned} \tag{22}$$

and

$$\begin{aligned} \bar{g}(\theta_n) &= \mathrm{E}\left[\tilde{d}(x_n,x_{n+1},w_n)\,\psi(x_n,u_n,\theta_n) \middle| \bar{\mathcal{F}}_n\right] \\ &= \mathrm{E}\left[d(x_n,x_{n+1},\theta_n)\,\psi(x_n,u_n,\theta_n) \middle| \bar{\mathcal{F}}_n\right] \\ &\quad + \mathrm{E}\left[\left(\tilde{d}(x_n,x_{n+1},w_n) - d(x_n,x_{n+1},\theta_n)\right)\psi(x_n,u_n,\theta_n) \middle| \bar{\mathcal{F}}_n\right] \\ &= \nabla\eta(\theta_n) + \sum_{x,y\in\mathcal{X}} \sum_{u\in\mathcal{U}} \pi(x)\,P(u|x,\theta_n)\,P(y|x,u)\,\psi(x,u,\theta_n) \\ &\qquad \times \left(\tilde{d}(x,y,w_n) - d(x,y,\theta_n)\right). \end{aligned}$$

Next, we show that the required assumptions hold.

C.2 Satisfying Assumption 7.2

We need to show that $\sup_n \mathrm{E}\left[\|Y_n\|_2\right] < \infty$. Since later we need to show that $\sup_n \mathrm{E}\left[\|Y_n\|_2^2\right] < \infty$, and the proof of the second moment is similar to the proof of the first moment, we consider both moments here.

Lemma 11 *The sequence $\tilde{\eta}_n$ is bounded w.p. 1, $\sup_n \mathrm{E}\left[\|Y_n(\tilde{\eta}_n)\|_2\right] < \infty$, and $\sup_n \mathrm{E}\left[\|Y_n(\tilde{\eta}_n)\|_2^2\right] < \infty$*

Proof We can choose M such that $\gamma_n \Gamma_\eta < 1$ for all $n > M$. Using Assumption 3 for the boundedness of the rewards, we have

$$\begin{aligned}
\tilde{\eta}_{n+1} &= (1-\gamma_n\Gamma_\eta)\tilde{\eta}_m + \gamma_n\Gamma_\eta r(x_n) \\
&\leq (1-\gamma_n\Gamma_\eta)\tilde{\eta}_n + \gamma_n\Gamma_\eta B_r \\
&\leq \begin{cases} \tilde{\eta}_n & \text{if } \tilde{\eta}_n > B_r, \\ B_r & \text{if } \tilde{\eta}_n \leq B_r, \end{cases} \\
&\leq \max\{\tilde{\eta}_n, B_r\},
\end{aligned}$$

which means that each iterate is bounded above by the previous iterate or by a constant. We denote this bound by $B_{\tilde{\eta}}$. Using similar arguments we can prove that $\tilde{\eta}_n$ is bounded below, and the first part of the lemma is proved. Since $\tilde{\eta}_{n+1}$ is bounded the second part follows trivially. ■

Lemma 12 *We have $\sup_n E\left[\|Y_n(w_n)\|_2^2\right] < \infty$ and $\sup_n E\left[\|Y_n(w_n)\|_2\right] < \infty$*

Proof For the first part we have

$$\begin{aligned}
\mathrm{E}\left[\|Y_n(w_n)\|_2^2\right] &= \mathrm{E}\left[\left\|\Gamma_w \tilde{d}(x_n, x_{n+1}, w_n)\, e_n\right\|_2^2\right] \\
&= \Gamma_w^2 \mathrm{E}\left[\left\|\sum_{k=0}^{\infty} \lambda^k \phi(x_{n-k})\left(r(x_n) - \tilde{\eta}_n + \phi(x_{n+1})' w_n - \phi(x_n)' w_n\right)\right\|_2^2\right] \\
&\overset{(a)}{\leq} \Gamma_w^2 \mathrm{E}\left[\sum_{k=0}^{\infty} \lambda^k \left\|\phi(x_{n-k})\left(r(x_n) - \tilde{\eta}_n + \phi(x_{n+1})' w_n - \phi(x_n)' w_n\right)\right\|_2\right]^2 \\
&\leq \Gamma_w^2 \mathrm{E}\left[\sup_k \left\|\phi(x_{n-k})\left(r(x_n) - \tilde{\eta}_n + \phi(x_{n+1})' w_n - \phi(x_n)' w_n\right)\right\|_2 \sum_{k=0}^{\infty} \lambda^k\right]^2 \\
&\overset{(b)}{\leq} \frac{4\Gamma_w^2}{(1-\lambda)^2} \|\phi(x_{n-k})\|_2^2 \\
&\qquad \times \left(|r(x_n)|^2 + |\tilde{\eta}_n|^2 + \|\phi(x_{n+1})\|_2^2 \|w_n\|_2^2 + \|\phi(x_n)\|_2^2 \|w_n\|_2^2\right) \\
&\leq \frac{4\Gamma_w^2}{(1-\lambda)^2} B_\phi^2 \left(B_r^2 + B_{\tilde{\eta}}^2 + 2B_\phi^2 B_w^2\right),
\end{aligned}$$

where we used the triangle inequality in (a) and the inequality $(a+b)^2 \leq 2a^2 + 2b^2$ in (b). The bound $\sup_n \mathrm{E}\left[\|Y_n(w_n)\|_2\right] < \infty$ follows directly from the Cauchy-Schwartz inequality. ■

Lemma 13 *We have* $\sup_n E\left[\|Y_n(\theta_n)\|_2^2\right] < \infty$ *and* $\sup_n E[\|Y_n(\theta_n)\|_2] < \infty$.*The proof proceeds as in Lemma 12*.

Based on Lemmas 11, 12, and 13 we can assert Assumption 7.2

C.3 Satisfying Assumption 7.3

Assumption 7.3 requires the continuity of $g_n(y_n, x_n)$ for each n and x_n. Again, we show that this assumption holds for the three parts of the vector y_n.

Lemma 14 *The function* $g_n(\tilde{\eta}_n, x_n)$ *is a continuous function of* $\tilde{\eta}_n$ *for each n and* x_n.

Proof Since $g_n(\tilde{\eta}_n, x_n) = \Gamma_\eta(r(x_n) - \tilde{\eta}_n)$ the claim follows. ■

Lemma 15 *The function* $g_n(w_n, x_n)$ *is a continuous function of* $\tilde{\eta}_n$, w_n, *and* θ_n *for each n and* x_n.

Proof The function is

$$g_n(w_n, x_n) = \Gamma_w \sum_k^\infty \lambda^k \phi(x_{n-k}) \left(r(x_n) - \tilde{\eta}_n + \sum_{y \in \mathcal{X}} P(y|x_n, \theta_n) \phi(y)' w_n - \phi(x_n)' w_n \right).$$

The probability transition $\sum_{y \in \mathcal{X}} P(y|x_n, \theta_n)$ is a function of $\mu(u_n|x_n, \theta_n)$. Thus it is continuous in θ_n by Assumption 3, and thus $g_n(w_n, x_n)$ is continuous in $\tilde{\eta}_n$ and θ_n and the lemma follows. ■

Lemma 16 *The function* $g_n(\theta_n, x_n)$ *is a continuous function of* $\tilde{\eta}_n$, w_n, *and* θ_n *for each n and* x_n.

Proof By definition, the function $g_n(\theta_n, x_n)$ is

$$\begin{aligned} g_n(\theta_n, x_n) &= E\left[\tilde{d}(x_n, x_{n+1}, w_n) \psi(x_n, u_n, \theta_n) \middle| \mathcal{F}_n\right] \\ &= \frac{\nabla_\theta \mu(u_n|x_n, \theta_n)}{\mu(u_n|x_n, \theta_n)} \left(r(x_n) - \tilde{\eta}_n + \sum_{y \in \mathcal{X}} P(y|x_n, \theta_n) \phi(y)' w_n - \phi(x_n)' w_n \right) \end{aligned}$$

Using similar arguments to Lemma 15 the claim holds. ■

C.4 Satisfying Assumption 7.4

In this section we prove the following convergence result: for each $\mu > 0$ and for some $T > 0$ there is a continuous function $\bar{g}(\cdot)$ such that for each y

$$\lim_{n \to \infty} \Pr\left(\sup_{j \geq n} \max_{0 \leq t \leq T} \left\| \sum_{i=m(jT)}^{m(jT+t)-1} \gamma_i (g_n(y, x_i) - \bar{g}(y)) \right\| \geq \mu \right). \tag{23}$$

We start by showing that there exist independent cycles of the algorithm since the underlying Markov chain is recurrent and aperiodic. Then, we show that the cycles behave as a martingale,

thus Doob's inequality can be used. Finally we show that the sum in (23) converges to 0 w.p. 1. We start investigating the regenerative nature of the process.

Based on Lemma 2, there exists a recurrent state common to all $MC(\theta)$, denoted by x^*. We define the series of *hitting times* of the recurrent state x^* by $t_0 = 0, t_1, t_2, ...$, where t_m it the m-th time the agent hits the state x^*. Mathematically, we can define this series recursively by

$$t_{m+1} = \inf\{n | x_n = x^*, n > t_m\}, \quad t_0 = 0,$$

and $T_m \triangleq t_{m+1} - t_m$. Define the m-th cycle of the algorithm to be the set of times

$$\mathcal{T}_m \triangleq \{n | t_{m-1} \leq n < t_m\},$$

and the corresponding trajectories

$$\mathcal{C}_m \triangleq \{x_n | n \in \mathcal{T}_m\}.$$

Define a function, $\rho(k)$, which returns the cycle to which the time k belongs to, that is,

$$\rho(k) \triangleq \{m | k \in \mathcal{T}_m\}.$$

We notice that based on Lemma 1, and using the *Regenerative Cycle Theorem* (Brémaud, 1999), the cycles $\mathcal{C}_m$ are independent of each other.

Next, we examine (23), and start by defining the following events:

$$\begin{aligned}
b_n^{(1)} &\triangleq \left\{ \omega \left| \sup_{j \geq n} \max_{0 \leq t \leq T} \left\| \sum_{i=m(jT)}^{m(jT+t)-1} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right) \right\| \geq \mu \right. \right\}, \\
b_n^{(2)} &\triangleq \left\{ \omega \left| \sup_{j \geq n} \sup_{k \geq m(jT)} \left\| \sum_{i=m(jT)}^{k} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right) \right\| \geq \mu \right. \right\}, \\
b_n^{(3)} &\triangleq \left\{ \omega \left| \sup_{j \geq n} \left\| \sum_{i=n}^{\infty} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right) \right\| \geq \mu \right. \right\}.
\end{aligned}$$

It is easy to show that for each n we have $b_n^{(1)} \subset b_n^{(2)}$, thus,

$$\Pr\left(b_n^{(1)}\right) \leq \Pr\left(b_n^{(2)}\right). \tag{24}$$

It is easy to verify that the series $\left\{b_n^{(2)}\right\}$ is a subsequence of $\left\{b_n^{(3)}\right\}$. Thus, if we prove that $\lim_{n\to\infty} \Pr\left(b_n^{(3)}\right) = 0$, then $\lim_{n\to\infty} \Pr(b_n) = 0$, and using (24), Assumption 7.4 holds.

Next, we examine the sum defining the event $b_n^{(3)}$, by splitting it a sum over cycles and a sum within each cycle. We can write it as following

$$\sum_{i=n}^{\infty} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right) = \sum_{m=\rho(n)}^{\infty} \sum_{i \in \mathcal{T}_m} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right).$$

Denote $c_m \triangleq \sum_{j \in \mathcal{T}_m} \gamma_i \left(g_n(y, x_i) - \bar{g}(y) \right)$. Therefore, by the *Regenerative Cycle Theorem* (Brémaud, 1999), c_m are independent random variables. Also,

$$\mathrm{E}[c_m] = \mathrm{E}\left[\sum_{i \in \mathcal{T}_m} \gamma_i \left(g_i(y, x_i) - \bar{g}(y) \right) \right] = \mathrm{E}\left[\mathrm{E}\left[\sum_{j \in \mathcal{T}_m} \gamma_i \left(g_n(y, x_i) - \bar{g}(y) \right) \middle| \mathcal{T}_m \right] \right] = 0.$$

We argue that c_m is square integrable. To prove this we need to show that the second moments of T_m and $(g_n(y,x_i)-\bar{g}(y))$ are finite.

Lemma 17

1. The first two moments of the random times $\{T_m\}$ are bounded above by a constant B_T, for all $\theta \in \mathbb{R}^K$ and for all m, $1 \leq m < \infty$.
   (a) $\mathrm{E}\left[(g_n(y,x_i)-\bar{g}(y))^2\right] \leq B_g$
   (b) Define $\bar{\gamma}_m \triangleq \sup_{i \in \mathcal{T}_m} \gamma_i$, then $\sum_{m=0}^{\infty} \bar{\gamma}_m^2 < \infty$.
   (c) $\mathrm{E}\left[c_m^2\right] \leq (B_T B_g)^2$.

Proof ■

1. According to Assumption 2 and Lemma 1, each Markov chain in $\bar{\mathcal{P}}$ is recurrent. Thus, for each $\theta \in \mathbb{R}^K$ there exists a constant $\tilde{B}_T(\theta), 0 < \tilde{B}_T(\theta) < 1$, where for $k \leq |\mathcal{X}|$ we have

$$P(T_m = k|\theta_m) \leq \left(\tilde{B}_T(\theta_m)\right)^{\lfloor k/|\mathcal{X}| \rfloor}, \quad 1 \leq m < \infty, \quad 1 \leq k < \infty,$$

where $\lfloor a \rfloor$ is the largest integer which is not greater than a. Otherwise, if for $k > |\mathcal{X}|$ we have $\tilde{B}_T(\theta_m) = 1$ then the chain transitions equal 1 which contradicts the aperiodicity of the chains. Therefore,

$$\mathrm{E}[T_m|\theta_m] = \sum_{k=1}^{\infty} kP(T_m = k|\theta_m) \leq \sum_{k=1}^{\infty} k\left(\tilde{B}_T(\theta_m)\right)^{\lfloor k/|\mathcal{X}| \rfloor} = B_{T_1}(\theta_m) < \infty,$$

and

$$\mathrm{E}\left[T_m^2\middle|\theta_m\right] = \sum_{k=1}^{\infty} k^2 P(T_m = k|\theta_m) \leq \sum_{k=1}^{\infty} k^2\left(\tilde{B}_T(\theta_m)\right)^{\lfloor k/|\mathcal{X}| \rfloor} = B_{T_2}(\theta_m) < \infty.$$

Since the set $\bar{\mathcal{P}}$ is closed, by Assumption 2 the above holds for the closure of $\bar{\mathcal{P}}$ as well. Thus, there exists a constant B_T satisfying $B_T = \max\{\sup_\theta B_{T_1}(\theta), \sup_\theta B_{T_2}(\theta)\} < \infty$.

   (a) The proof proceeds along the same lines as the proofs of lemmas 11, 12, and 13.
   (b) The result follows trivially since the sequence $\{\bar{\gamma}_m\}$ is subsequence of the summable sequence$\{\gamma_m\}$.
   (c) By definition, for large enough m we have $\gamma_m \leq 1$. Therefore, we have

$$\begin{aligned}
\mathrm{E}\left[c_m^2\right] &= \mathrm{E}\left[\left(\sum_{j \in \mathcal{T}_m} \gamma_j (g_n(y,x_j)-\bar{g}(y))\right)^2\right] \\
&\leq \mathrm{E}\left[|\mathcal{T}_m|^2 \left(\sup_j \gamma_j\right)^2 \left(\sup_j (g_n(y,x_j)-\bar{g}(y))\right)^2\right] \\
&\leq B_T^2 B_g^2.
\end{aligned}$$

Next, we conclude by showing that Assumption 7.4 is satisfied. Define the process $d_n \triangleq \sum_{m=0}^{n} c_m$. This process is a martingale since the sequence $\{c_m\}$ is square integrable (by Lemma 17) and satisfies $\mathrm{E}[d_{m+1}|d_m] = d_m$. Using Doob's martingale inequality[2] we have

$$\begin{aligned}
&\Pr\left(\sup_{k\geq n}\sum_{m=\rho(n)}^{\rho(k)}\sum_{j\in\mathcal{T}_m}\gamma_i\left(g_n\left(y,x_i\right)-\bar{g}\left(y\right)\right)\geq\mu\right)\\
&\leq\lim_{n\to\infty}\frac{\mathrm{E}\left[\left(\sum_{m=\rho(n)}^{\infty}\sum_{j\in\mathcal{T}_m}\gamma_j\left(g_n\left(y,x_j\right)-\bar{g}\left(y\right)\right)\right)^2\right]}{\mu^2}\\
&=\lim_{n\to\infty}\frac{\sum_{m=\rho(n)}^{\infty}\mathrm{E}\left[\left(\sum_{j\in\mathcal{T}_m}\gamma_j\left(g_n\left(y,x_j\right)-\bar{g}\left(y\right)\right)\right)^2\right]}{\mu^2}\\
&\leq\lim_{n\to\infty}\sum_{m=\rho(n)}^{\infty}\bar{\gamma}_m^2 B_g B_T/\mu^2\\
&=0.
\end{aligned}$$

C.5 Satisfying Assumption 7.5

In this section we need to show that for each $\mu > 0$ and for some $T > 0$ we have

$$\lim_{n\to\infty}\Pr\left(\sup_{j\geq n}\max_{0\leq t\leq T}\left\|\sum_{i=m(jT)}^{m(jT+t)-1}\gamma_i\delta M_i\right\|\geq\mu\right)=0. \tag{25}$$

In order to follow the same lines as in Section C.4, we need to show that the second moment of the martingale difference noise, δM_i, is bounded with zero mean. By definition, $\delta M_n(\cdot)$ has zero mean.

Lemma 18 *The martingale difference noise, $\delta M_n(\cdot)$, is bounded in the second moment.*

Proof The claim is immediate from the fact that

$$\mathrm{E}\left[\left(\delta M_n\right)^2\right]=\mathrm{E}\left[\left\|Y_n-g_n\left(y_n,x_n\right)\right\|^2\right]\leq 2\mathrm{E}\left[\left\|Y_n\right\|^2+\left\|g_n\left(y_n,x_n\right)\right\|^2\right],$$

and from Lemma 11, Lemma 12, and Lemma 13. ■

Combining this fact with Lemma 18, and applying the regenerative decomposition of Section C.4, we conclude that statistically $\delta M_n(\cdot)$ behaves exactly as $(g_n(y,x_i)-\bar{g}(y))$ of Section C.4 and thus (25) holds.

C.6 Satisfying Assumption 7.6

In this section we need to prove that there are non-negative measurable functions $\rho_3(y)$ and $\rho_{n4}(x)$ such that

$$\left\|g_n\left(y_n,x\right)\right\|\leq\rho_3\left(y_n\right)\rho_{n4}\left(x\right),$$

2. If w_n is a martingale sequence then $\Pr\left(\sup_{m\geq 0}|w_n|\geq\mu\right)\leq\lim_{n\to\infty}\mathrm{E}\left[|w_n|^2\right]/\mu^2$.

where $\rho_3(y)$ is bounded on each bounded y-set, and for each $\mu > 0$ we have

$$\lim_{\tau\to 0}\lim_{n\to\infty}\Pr\left(\sup_{j\geq n}\sum_{i=m(j\tau)}^{m(j\tau+\tau)-1}\gamma_i\rho_{n4}(x_i)\geq\mu\right)=0.$$

The following lemma states a stronger condition for Assumption 7.6. In fact, we choose $\rho_3(y)$ to be a positive constant.

Lemma 19 *If* $\|g_n(y,x)\|$ is uniformly bounded for each y, x and n, then *Assumption 7.6 is satisfied*.

Proof Let us denote the upper bound by the random variable B, that is,

$$\|g_n(y,x)\|\leq B,\quad \text{w.p. 1.}$$

Thus

$$\begin{aligned}\lim_{\tau\to 0}\lim_{n\to\infty}\Pr\left(\sup_{j\geq n}\sum_{i=m(j\tau)}^{m(j\tau+\tau)-1}\gamma_i\rho_{n4}(x_i)\geq\mu\right) &\leq \lim_{\tau\to 0}\lim_{n\to\infty}\Pr\left(\sup_{j\geq n}\sum_{i=m(j\tau)}^{m(j\tau+\tau)-1}\gamma_i B\geq\mu\right)\\ &= \lim_{\tau\to 0}\lim_{n\to\infty}\Pr\left(\sup_{j\geq n}B\sum_{i=m(j\tau)}^{m(j\tau+\tau)-1}\gamma_i\geq\mu\right)\\ &\leq \lim_{\tau\to 0}\Pr(B\tau\geq\mu)\\ &= 0.\end{aligned}$$

■

Based on Lemma 19, we are left with proving that $g_n(y,x)$ is uniformly bounded. The following lemma states so.

Lemma 20 *The function* $g_n(y,x)$ is *uniformly bounded for all n*.

Proof We examine the components of $g_n(y_n,x_n)$. In (20) we showed that

$$g_n(\tilde{\eta}_n,x_n) = \Gamma_\eta\left(r(x_n)-\tilde{\eta}_n\right).$$

Since both $r(x_n)$ and $\tilde{\eta}_n$ are bounded by Assumption 1 and Lemma 11 respectively, we have a uniform bound on $g_n(\tilde{\eta}_n,x_n)$. Recalling (21) we have

$$\begin{aligned}g_n(w_n,x_n) &= \Gamma_w\sum_{k=0}^{\infty}\lambda^k\phi(x_{n-k})\left(r(x_n)-\tilde{\eta}_n+\sum_{y\in\mathcal{X}}P(y|x_n,\theta_n)\phi(y)'w_n-\phi(x_n)'w_n\right)\\ &\leq \Gamma_w\frac{1}{1-\lambda}B_\phi\left(B_r+B_{\tilde{\eta}}+2B_\phi B_w\right).\end{aligned}$$

Finally, recalling (22) we have

$$\begin{aligned}g_n(\theta_n,x_n) &= \mathrm{E}\left[\tilde{d}(x_n,x_{n+1},w_n)\psi(x_n,u_n,\theta_n)\middle|\mathcal{F}_n\right]\\ &\leq \left(B_r+B_{\tilde{\eta}}+2B_\phi B_w\right)B_\psi.\end{aligned}$$

■

C.7 Satisfying Assumption 7.7

In this section we show that there are non-negative measurable functions $\rho_1(y)$ and $\rho_{n2}(x)$ such that $\rho_1(y)$ is bounded on each bounded y-set and

$$\|g_n(y_1,x)-g_n(y_2,x)\| \leq \rho_1(y_1-y_2)\,\rho_{n2}(x) \tag{26}$$

where

$$\lim_{y\to 0}\rho_1(y)=0, \tag{27}$$

and for some $\tau>0$

$$\Pr\left(\limsup_j \sum_{i=j}^{m(t_j+\tau)} \gamma_i \rho_{i2}(x_i) < \infty\right)=1.$$

From Section C.6 we infer that we can choose $\rho_{n2}(x)$ to be a constant since $g_n(y,x)$ is uniformly bounded. Thus, we need to show the appropriate $\rho_1(\cdot)$ function. The following lemma shows it.

Lemma 21 *The following functions satisfy (26) and (27).*

1. The function $\rho_1(y)=\|\tilde{\eta}_2-\tilde{\eta}_1\|$ and $\rho_{n2}(x)=\Gamma_\eta$ for $g_n(\tilde{\eta},x)$.
   (a) The function $\rho_1(y)=\frac{1}{1-\lambda}B_\phi^2\left(\sum_{y\in\mathcal{X}}B_w\|P(y|x,\theta_1)-P(y|x,\theta_2)\|+\|w_1-w_2\|\right)$ and $\rho_{n2}(x)=\Gamma_w$ for $g_n(w,x)$.
   (b) The function $\rho_1(y)=\sum_{y\in\mathcal{X}}B_w\|P(y|x,\theta_1)-P(y|x,\theta_2)\|\cdot B_\psi$ and $\rho_{n2}(x)=1$ for $g_n(\theta,x)$.

Proof ■

1. Recalling (20) we have for $g_n(\tilde{\eta},x)$

$$\|g_n(\tilde{\eta}_1,x)-g_n(\tilde{\eta}_2,x)\| \quad\leq\quad \Gamma_\eta\|\tilde{\eta}_2-\tilde{\eta}_1\|,$$

   thus (26) and (27) are satisfied for 1.

2. Recalling (21) we have for $g_n(w,x)$

$$\begin{aligned}
&\|g_n(w_1,x)-g_n(w_2,x)\|\\
&\leq\left\|\Gamma_w\sum_k^\infty\lambda^k\phi(x_{n-k})\left(\left(\sum_{y\in\mathcal{X}}P(y|x,\theta_1)\,\phi(y)'w_1-\phi(x_n)'w_1\right)\right.\right.\\
&\quad\left.\left.-\left(\sum_{y\in\mathcal{X}}P(y|x,\theta_2)\,\phi(y)'w_2-\phi(x_n)'w_2\right)\right)\right\|\\
&\leq\frac{\Gamma_w B_\phi^2}{1-\lambda}\left(\sum_{y\in\mathcal{X}}\|P(y|x,\theta_1)\,w_1-P(y|x,\theta_2)\,w_2\|+\|w_1-w_2\|\right)\\
&\leq\frac{\Gamma_w B_\phi^2}{1-\lambda}\left(\sum_{y\in\mathcal{X}}B_w\|P(y|x,\theta_1)-P(y|x,\theta_2)\|+\|w_1-w_2\|\right)
\end{aligned}$$

(a) Trivially, with respect to w (26) and (27) are satisfied. Regarding θ, (26) and (27) are satisfied if we recall the definition of $P(y|x,\theta)$ from (1) and the continuity of $\mu(u|x,\theta)$ from Assumption 3.

(b) Recalling (22) we have for $g_n(\theta,x)$

$$\begin{aligned}\|g_n(\theta_1,x)-g_n(\theta_2,x)\| &= \left\|\mathrm{E}\left[\tilde{d}(x,y,w_1)\psi(x,u,\theta_1)\middle|\mathcal{F}_n\right]\right. \\ &\quad \left.-\mathrm{E}\left[\tilde{d}(x,y,w_2)\psi(x,u,\theta_2)\middle|\mathcal{F}_n\right]\right\| \\ &\leq \sum_{y\in\mathcal{X}} B_w \|P(y|x,\theta_1)-P(y|x,\theta_2)\| B_\psi.\end{aligned}$$

Using similar arguments to 2, (26) and (27) are satisfied for θ.

Appendix D. Proof of Theorem 8

In this section we find conditions under which Algorithm 1 converges to a neighborhood of a local maximum. More precisely, we show that $\liminf_{t\to\infty}\|\nabla\eta(\theta(t))\|_2 \leq \varepsilon_{\text{app}}+\varepsilon_{\text{dyn}}$, where the approximation error, ε_{app}, measures the error inherent in the critic's representation, and ε_{dyn} is an error related to the single time scale algorithm. We note that the approximation error depends on the basis functions chosen for the critic, and in general can be reduced only by choosing a better representation basis. The term ε_{dyn} is the dynamic error, and this error can be reduced by choosing the critic's parameters Γ_η and Γ_w appropriately.

We begin by establishing a variant of Lyapunov's theorem for asymptotic stability,[3] where instead of proving asymptotic convergence to a point, we prove convergence to a compact invariant set. Based on this result, we continue by establishing a bound on a time dependent ODE of the first order. This result is used to bound the critic's error in estimating the average reward per stage and the differential values. Finally, using these results, we establish Theorem 8.

We denote a closed ball of radius y in some normed vector space, $(\mathbb{R}^L,\|\cdot\|_2)$, by $\mathcal{B}_y$, and its surface by $\partial\mathcal{B}_y$. Also, we denote by $A\backslash B$ a set, which contains all the members of set A which are not members of B. Finally, we define the complement of $\mathcal{B}_y$ by $\mathcal{B}_y^c=\mathbb{R}^L\backslash\mathcal{B}_y$.

The following lemma is similar to Lyapunov's classic theorem for asymptotic stability; see Khalil (2002), Theorem 4.1. The main difference is that when the value of the Lyapunov function is unknown inside a ball, convergence can be established to the ball, rather than to a single point.

Lemma 22 *Consider a dynamical system, $\dot{x}=f(x)$ in a normed vector space, $(\mathbb{R}^L,\|\cdot\|)$, and a closed ball $\mathcal{B}_r\triangleq\left\{x\middle|x\in\mathbb{R}^L,\|x\|\leq r\right\}$. Suppose that there exists a continuously differentiable scalar function $V(x)$ such that $V(x)>0$ and $\dot{V}(x)<0$ for all $x\in\mathcal{B}_r^c$, and $V(x)=0$ for $x\in\partial\mathcal{B}_r$. Then,*

$$\limsup_{t\to\infty}\|x(t)\|\leq r.$$

Proof We prove two complementary cases. In the first case, we assume that $x(t)$ never enters $\mathcal{B}_r$. On the set $\mathcal{B}_r^c$, $V(x)$ is a strictly decreasing function in t, and it is bounded below, thus it converges. We denote this bound by C, and notice that $C\geq 0$ since for $x\in\mathcal{B}_r^c$, $V(x)>0$. We prove that $C=0$ by contradiction. Assume that $C>0$. Then, $x(t)$ converge to the invariant set

3. We say that the equilibrium point $x=0$ of the system $\dot{x}$=f(x) is *stable* if for each $\varepsilon>0$ there exists a $\delta>0$ such that $\|x(0)\|<\delta\Rightarrow\|x(t)\|<\varepsilon$ for all $t\geq 0$. We say that the point $x=0$ is *asymptotically stable* if it is stable and there exists a $\delta>0$ such that $\|x(0)\|<\delta$ implies $\lim_{t\to\infty}x(t)=0$ (Khalil, 2002).

$S_C \triangleq \{x|V(x) = C, x \in \mathcal{B}_r^c\}$. For each $x(t) \in \mathcal{S}_C$ we have $\dot{V}(x) < 0$. Thus, $V(x)$ continues to decrease which contradicts the boundedness from below. As a result, $V(x(t)) \to 0$.

In the second case, let us suppose that at some time, denoted by t_0, $x(t_0) \in \mathcal{B}_r$. We argue that the trajectory never leaves $\mathcal{B}_r$. Let us assume that at some time t_2, the trajectory $x(t)$ enters the set $\partial\mathcal{B}_{r+\varepsilon}$. Then on this set, we have $V(x(t_2)) > 0$. By the continuity of the trajectory $x(t)$, the trajectory must go through the set $\partial\mathcal{B}_r$. Denote the hitting time of this set by t_1. By definition we have $V(x(t_1)) = 0$. Without loss of generality, we assume that the trajectory in the times $t_1 < t \leq t_2$ is restricted to the set $\mathcal{B}_{r+\varepsilon}/\mathcal{B}_r$. Thus, since $\dot{V}(x(t)) \leq 0$ for $x \in \mathcal{B}_{r+\varepsilon}/\mathcal{B}_r$ we have

$$V(x(t_2)) = V(x(t_1)) + \int_{t_1}^{t_2} \dot{V}(x(t))dt < V(x(t_1)),$$

which contradicts the fact that $V(x(t_2)) \geq V(x(t_1))$. Since this argument holds for all $\varepsilon > 0$, the trajectory $x(t)$ never leaves $\mathcal{B}_r$. ■

The following lemma will be applied later to the linear equations (17), and more specifically, to the ODEs describing the dynamics of $\tilde{\eta}$ and w. It bounds the difference between an ODE's state variables and some time dependent functions.

Lemma 23 *Consider the following ODE in a normed space* $(\mathbb{R}^L, \|\cdot\|_2)$

$$\begin{cases} \frac{d}{dt}X(t) = \mathcal{M}(t)(X(t) - F_1(t)) + F_2(t), \\ X(0) = X_0, \end{cases} \tag{28}$$

where for sufficiently large t .

1. $\mathcal{M}(t) \in \mathbb{R}^{L\times L}$ *is a continuous matrix which satisfies* $\max_{\|x\|=1} x'\mathcal{M}(t)x \leq -\gamma < 0$ *for* $t \in \mathbb{R}$,
2. $F_1(t) \in \mathbb{R}^L$ *satisfies* $\|dF_1(t)/dt\|_2 \leq B_{F1}$,
3. $F_2(t) \in \mathbb{R}^L$ *satisfies* $\|F_2(t)\|_2 \leq B_{F2}$.

Then, the solution of the ODE satisfies $\limsup_{t\to 0} \|X(t) - F_1(t)\|_2 \leq (B_{F1} + B_{F2})/\gamma$.

Proof We express (28) as

$$\frac{d}{dt}(X(t) - F_1(t)) = \mathcal{M}(t)(X(t) - F_1(t)) - \frac{d}{dt}F_1(t) + F_2(t), \tag{29}$$

and define

$$Z(t) \triangleq (X(t) - F_1(t)), \quad G(t) \triangleq -\frac{d}{dt}F_1(t) + F_2(t).$$

Therefore, (29) can be written as

$$\dot{Z}(t) = \mathcal{M}(t)Z(t) + G(t),$$

where $\|G(t)\| \leq B_G \triangleq B_{F1} + B_{F2}$. In view of Lemma 22, we consider the function

$$V(Z) = \frac{1}{2}\left(\|Z(t)\|_2^2 - B_G^2/\gamma^2\right).$$

Let $\mathcal{B}_r$ be a ball with a radius $r = B_G/\gamma$. Thus we have $V(Z) > 0$ for $Z \in \mathcal{B}_r^c$ and $V(Z) = 0$ for $X \in \partial\mathcal{B}_r$. In order to satisfy the assumptions of Lemma 22 the condition that $\dot{V}(Z) < 0$ needs to be verified. For $\|Z(t)\|_2 > B_G/\gamma$ we have

$$\begin{aligned}
\dot{V}(Z) &= (\nabla_X V)' \dot{Z}(t) \\
&= Z(t)'\mathcal{M}(t)Z(t) + Z(t)'G(t) \\
&= \|Z(t)\|_2^2 \frac{Z(t)'}{\|Z(t)\|_2}\mathcal{M}(t)\frac{Z(t)}{\|Z(t)\|_2} + Z(t)'G(t) \\
&\leq \|Z(t)\|_2^2 \max_{\|Y(t)\|_2=1} Y(t)'\mathcal{M}(t)Y(t) + \|Z(t)\|_2 \|G(t)\|_2 \\
&= \|Z(t)\|_2 (-\gamma \|Z(t)\|_2 + B_G) \\
&< 0.
\end{aligned}$$

As a result, the assumptions of Lemma 22 are valid and the Lemma is proved. ■

The following lemma shows that the matrix $A(\theta)$, defined in (16), satisfies the conditions of Lemma 23. For the following lemmas, we define the weighted norm $\|w\|_{\Pi(\theta)}^2 \triangleq \|w'\Pi(\theta)w\|_2$.

Lemma 24 *The following inequalities hold:*

1. For any $w \in \mathbb{R}^L$and for all $\theta \in \mathbb{R}^K$, $\|P(\theta)w\|_{\Pi(\theta)} < \|w\|_{\Pi(\theta)}$..
2. The matrix $M(\theta)$ satisfies $\|M(\theta)w\|_{\Pi(\theta)} < \|w\|_{\Pi(\theta)}$ for all $\theta \in \mathbb{R}^K$ and $w \in \mathbb{R}^L$.
3. The matrix $\Pi(\theta)(M(\theta) - I)$ satisfies $x'\Pi(\theta)(M(\theta) - I)x < 0$ for all $x \in \mathbb{R}^L$ and for all $\theta \in \mathbb{R}^K$.
4. There exists a positive scalar γ such that $w'A(\theta)w < -\gamma$ for all $w'w = 1$.

Proof The following proof is similar in many aspects to the proof of Lemma 6.6 of Bertsekas and Tsitsiklis (1996). ■

1. By using Jensen's inequality for the function $f(\alpha) = \alpha^2$ we have

$$\left(\sum_{y\in\mathcal{X}} P(y|x,\theta)w(y)\right)^2 \leq \sum_{y\in\mathcal{X}} P(y|x,\theta)w(y)^2, \quad \forall x \in \mathcal{X}. \tag{30}$$

   If in Jensen's inequality we have a strictly convex fiction and non-degenerate probability measures then the inequality is strict. The function $f(\alpha)$ is strictly convex, and by Assumption 2 the matrix $P(\theta)$ is aperiodic, which implies that the matrix $P(\theta)$ is not a permutation matrix. As a result, there exists $x_0 \in \mathcal{X}$ such that the probability measure $P(y|x_0,\theta)$ is not degenerate, thus, the inequality in (30) is strict, that is,

$$\left(\sum_{y\in\mathcal{X}} P(y|x_0,\theta)w(y)\right)^2 < \sum_{y\in\mathcal{X}} P(y|x_0,\theta)w(y)^2. \tag{31}$$

Then, we have

$$\begin{aligned}
\|P(\theta)w\|_{\Pi(\theta)} &= w'P(\theta)'\Pi(\theta)P(\theta)w \\
&= \sum_{x\in\mathcal{X}}\pi(x|\theta)\left(\sum_{y\in\mathcal{X}}P(y|x,\theta)w(y)\right)^2 \\
&< \sum_{x\in\mathcal{X}}\pi(x|\theta)\sum_{y\in\mathcal{X}}P(y|x,\theta)w(y)^2 \\
&= \sum_{y\in\mathcal{X}}w(y)^2\sum_{x\in\mathcal{X}}\pi(x|\theta)P(y|x,\theta) \\
&= \sum_{y\in\mathcal{X}}w(y)^2\pi(y|\theta) \\
&= \|w\|_{\Pi(\theta)},
\end{aligned}$$

where in the inequality we have used (31).

2. Using the triangle inequality and 1 we have

$$\begin{aligned}
\|M(\theta)w\|_{\Pi(\theta)} &= \left\|(1-\lambda)\sum_{m=0}^{\infty}\lambda^m P(\theta)^{m+1}w\right\|_{\Pi(\theta)} \\
&\leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\left\|P(\theta)^{m+1}w\right\|_{\Pi(\theta)} \\
&< (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\|w\|_{\Pi(\theta)} \\
&= \|w\|_{\Pi(\theta)}.
\end{aligned}$$

3. By definition

$$\begin{aligned}
x'\Pi(\theta)M(\theta)x &= x'\Pi(\theta)^{1/2}\Pi(\theta)^{1/2}M(\theta)x \\
&\leq \left\|\Pi(\theta)^{1/2}x\right\|\cdot\left\|\Pi(\theta)^{1/2}M(\theta)x\right\| \\
&= \|x\|_{\Pi(\theta)}\|M(\theta)x\|_{\Pi(\theta)} \\
&< \|x\|_{\Pi(\theta)}\|x\|_{\Pi(\theta)}. \\
&= x'\Pi(\theta)x,
\end{aligned}$$

where in the first inequality we have used the Cauchy-Schwartz inequality, and in the second inequality we have used 1. Thus, $x'\Pi(\theta)(M(\theta)-I)x<0$ for all $x\in\mathbb{R}$, which implies that $\Pi(\theta)(M(\theta)-I)$ is a negative definite (ND) matrix.[4]

4. From 3, we know that for all $\theta\in\mathbb{R}^K$ and all $w\in\mathbb{R}^{|\mathcal{X}|}$ satisfying $w'w=1$, we have $w'\Pi(\theta)(M(\theta)-I)w<0$, and by Assumption (2), this is true also for the closure of $\left\{\Pi(\theta)(M(\theta)-I)|\theta\in\mathbb{R}^K\right\}$. Thus, there exists a positive scalar, γ', satisfying

$$w'\Pi(\theta)(M(\theta)-I)w\leq-\gamma'<0.$$

4. Usually, a ND matrix is defined for Hermitian matrices, that is, if B is an Hermitian matrix and it satisfies $x'Bx<0$ for all $x\in\mathbb{C}^K$ then B is a NSD matrix . We use here a different definition which states that a square matrix B is a ND matrix if it is real and it satisfies $x'Bx<0$ for all $x\in\mathbb{R}^k$ (Horn and Johnson, 1985).

By Assumption 5 the rank of the matrix Φ is full, thus there exists a scalar γ such that for all $w \in \mathbb{R}^L$, where $w'w = 1$, we have $w'A(\theta)w \leq -\gamma < 0$.

The following Lemma establishes the boundedness of $\dot{\theta}$.

Lemma 25 *There exists a constant* $B_{\theta 1} \triangleq B_{\eta 1} + B_\psi \left(B_D + B_r + B_{\tilde{\eta}} + 2B_\phi B_w\right)$ *such that* $\|\dot{\theta}\|_2 \leq B_{\theta 1}$.

Proof Recalling (17)

$$\begin{aligned}
\left\|\dot{\theta}\right\|_2 &= \left\|\nabla_\theta \eta(\theta) + \sum_{x,y\in\mathcal{X}\times\mathcal{X},u\in\mathcal{U}} D^{(x,u,y)}(\theta)\left(d(x,y,\theta) - \tilde{d}(x,y,w)\right)\right\|_2 \\
&\leq B_{\eta 1} + \sum_{x,y\in\mathcal{X}\times\mathcal{X},u\in\mathcal{U}} \left\|D^{(x,u,y)}(\theta)\right\|_2 \left\|d(x,y,\theta) - \tilde{d}(x,y,w)\right\|_2 \\
&\leq B_{\eta 1} + B_\psi \left(B_D + B_r + B_{\tilde{\eta}} + 2B_\phi B_w\right) \\
&\triangleq B_{\theta 1}.
\end{aligned}$$

■

Based on Lemma (25), the following Lemma shows the boundedness of $(\eta(\theta(t)) - \tilde{\eta})$.

Lemma 26 *We have*

$$\limsup_{t\to\infty} |\eta(\theta(t)) - \tilde{\eta}| \leq \frac{B_{\Delta\eta}}{\Gamma_\eta},$$

where $B_{\Delta\eta} \triangleq B_{\eta_1} B_{\theta 1}$.

Proof Using the Cauchy-Schwartz inequality we have

$$\begin{aligned}
|\dot{\eta}(\theta)| &= |\nabla\eta(\theta)'\dot{\theta}| \\
&\leq \|\nabla\eta(\theta)\|_2 \|\dot{\theta}\|_2 \\
&\leq B_{\eta_1} B_{\theta 1}.
\end{aligned} \tag{32}$$

Recalling the equation for $\tilde{\eta}$ in (17) we have

$$\dot{\tilde{\eta}} = \Gamma_\eta \left(\eta(\theta) - \tilde{\eta}\right).$$

We conclude by applying Lemma 23 and using (32) that

$$\limsup_{t\to\infty} |\eta(\theta(t)) - \tilde{\eta}| \leq \frac{B_{\eta_1} B_{\theta 1}}{\Gamma_\eta} = \frac{B_{\Delta\eta}}{\Gamma_\eta}. \tag{33}$$

■

In (33) we see that the bound on $|\eta(\theta) - \tilde{\eta}|$ is controlled by Γ_η, where larger values of Γ_η ensure smaller values of $|\eta(\theta) - \tilde{\eta}|$. Next, we bound $\|w^*(\theta) - w\|_2$. We recall the second equation of (17)

$$\begin{aligned}
\dot{w} &= \Psi_w \left[\Gamma_w \left(A(\theta)w + b(\theta) + G(\theta)(\eta(\theta) - \tilde{\eta})\right)\right], \\
A(\theta) &= \Phi'\Pi(\theta)(M - I)\Phi, \\
M(\theta) &= (1-\lambda)\sum_{m=0}^{\infty} \lambda^m P(\theta)^{m+1}, \\
b(\theta) &= \Phi'\Pi(\theta)\sum_{m=0}^{\infty} \lambda^m P(\theta)^m (r - \eta(\theta)), \\
G(\theta) &= \Phi'\Pi(\theta)\sum_{m=0}^{\infty} \lambda^m P(\theta)^m.
\end{aligned}$$

We can write the equation for $\dot{w}$ as

$$\dot{w} = \Psi_w \left[\Gamma_w \left(A(\theta) \left(w - w^*(\theta) \right) + G(\theta)(\eta(\theta) - \tilde{\eta}) \right) \right],$$

where $w^* = -A(\theta)^{-1} b(\theta)$. In order to use Lemma 23, we need to demonstrate the boundedness of $\left\| \frac{d}{dt} w^* \right\|$. The following lemma does so.

Lemma 27

1. There exists a positive constant, $B_b \triangleq \frac{1}{1-\lambda} |\mathcal{X}|^3 L B_\Phi B_r$, such that $\|b(\theta)\|_2 \leq B_b$.
   (a) There exists a positive constant, $B_G \triangleq \frac{1}{1-\lambda} |\mathcal{X}|^3 L B_\Phi$, such that $\|G(\theta)\|_2 \leq B_G$.
   (b) There exist positive constants, $\tilde{B} = B_{\pi 1} (B_r + B_\eta) B_{\theta 1} + B_{P1} (B_r + B_\eta) B_{\theta 1} + B_{\eta 1} B_{\theta 1}$ and $B_{b1} \triangleq \frac{1}{1-\lambda} |\mathcal{X}|^3 B_\Phi B_r \tilde{B}$, such that we have $\left\| \dot{b}(\theta) \right\|_2 \leq B_{b1}$.
   (c) There exist constants b_A and B_A such that
   $$0 < b_A \leq \|A(\theta)\|_2 \leq B_A.$$
   (d) There exist a constants B_{A1} such that
   $$\|A(\theta)\|_2 \leq B_{A1}.$$
   (e) We have
   $$\left\| \frac{d}{dt} \left(A(\theta)^{-1} \right) \right\|_2 \leq b_A^2 B_{A1}.$$
   (f) There exists a positive constant, B_{w1}, such that
   $$\left\| \frac{d}{dt} w^* \right\|_2 \leq B_{w1}.$$

Proof ■

1. We show that the entries of the vector $b(\theta)$ are uniformly bounded in θ, therefore, its norm is uniformly bounded in θ. Let us look at the i-th entry of the vector $b(\theta)$ (we denote by $[\cdot]_j$ the j-th row of a matrix or a vector)

$$\begin{aligned}
|[b(\theta)]_i| &= \left| \left[\Phi' \Pi(\theta) \sum_{m=0}^{\infty} \lambda^m P(\theta)^m (r - \eta(\theta)) \right]_i \right| \\
&\leq \sum_{m=0}^{\infty} \lambda^m \left| \left[\Phi' \Pi(\theta) P(\theta)^m (r - \eta(\theta)) \right]_i \right| \\
&= \sum_{m=0}^{\infty} \lambda^m \left| \sum_{l=1}^{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{X}|} \sum_{k=1}^{|\mathcal{X}|} \left[\Phi' \right]_{ik} \Pi_{kj}(\theta) \left[P(\theta)^m \right]_{jl} (r_l - \eta(\theta)) \right| \\
&\leq \frac{1}{1-\lambda} |\mathcal{X}|^3 B_\Phi B_r,
\end{aligned}$$

thus $\|b(\theta)\|_2 \leq \frac{1}{1-\lambda} |\mathcal{X}|^3 L B_\Phi B_r$ is uniformly bounded in θ.

2. The proof is accomplished by similar argument to Section 1.

3. Similarly to Section 1, we show that the entries of the vector $\dot{b}(\theta)$ are uniformly bounded in θ, therefore, its norm is uniformly bounded in θ. First, we show that the following function of $\theta(t)$ is bounded.

$$\begin{aligned}
\left|\frac{d}{dt}\left(\Pi_{kj}(\theta)\left[P(\theta)^m\right]_{jl}(r_l-\eta(\theta))\right)\right| &= \left|\nabla_\theta\left(\Pi_{kj}(\theta)\left[P(\theta)^m\right]_{jl}(r_l-\eta(\theta))\right)\dot{\theta}\right| \\
&\leq \left|\left(\nabla_\theta\Pi_{kj}(\theta)\right)\left[P(\theta)^m\right]_{jl}(r_l-\eta(\theta))\dot{\theta}\right| \\
&\quad+\left|\Pi_{kj}(\theta)\left[\nabla_\theta P(\theta)^m\right]_{jl}(r_l-\eta(\theta))\dot{\theta}\right| \\
&\quad+\left|\Pi_{kj}(\theta)\left[P(\theta)^m\right]_{jl}\nabla_\theta(r_l-\eta(\theta))\dot{\theta}\right| \\
&\leq B_{\pi 1}(B_r+B_\eta)\cdot B_{\theta 1}+B_{P1}(B_r+B_\eta)B_{\theta 1} \\
&\quad+B_{\eta 1}B_{\theta 1} \\
&= \tilde{B},
\end{aligned}$$

where we used the triangle and Cauchy-Schwartz inequalities in the first and second inequalities respectively, and Lemmas 3 and 25 in the second inequality. Thus,

$$\begin{aligned}
\left|\left[\dot{b}(\theta)\right]_i\right| &= \left|\left[\Phi'\Pi(\theta)\sum_{m=0}^{\infty}\lambda^m P(\theta)^m(r-\eta(\theta))\right]_i\right| \\
&\leq \sum_{m=0}^{\infty}\lambda^m\left|\left[\Phi'\Pi(\theta)P(\theta)^m(r-\eta(\theta))\right]_i\right| \\
&= \sum_{m=0}^{\infty}\lambda^m\left|\sum_{l=1}^{|\mathcal{X}|}\sum_{j=1}^{|\mathcal{X}|}\sum_{k=1}^{|\mathcal{X}|}\left[\Phi'\right]_{ik}\frac{d}{dt}\left(\Pi_{kj}(\theta)\left[P(\theta)^m\right]_{jl}(r_l-\eta(\theta))\right)\right| \\
&\leq \frac{1}{1-\lambda}|\mathcal{X}|^3 B_\Phi B_r\tilde{B} \\
&= B_{b1}.
\end{aligned}$$

4. Since $A(\theta)$ satisfies $y'A(\theta)y<0$ for all nonzero y, it follows that all its eigenvalues are nonzero. Therefore, the eigenvalues of $A(\theta)'A(\theta)$ are all positive and real since $A(\theta)'A(\theta)$ is a symmetric matrix. Since by Assumption 2 this holds for all $\theta\in\mathbb{R}^K$, there is a global minimum, b_A, and a global maximum, B_A, such that

$$B_A^2\geq\lambda_{\max}\left(A(\theta)'A(\theta)\right)\geq\lambda_{\min}\left(A(\theta)'A(\theta)\right)\geq b_A^2,\quad\forall\theta\in\mathbb{R}^K,$$

where we denote by $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ the minimal and maximal eigenvalues of the matrix respectively. Using Horn and Johnson (1985) section 5.6.6, we have $\lambda_{max}\left(A(\theta)'A(\theta)\right)=\|A(\theta)\|_2$, thus, we get an upper bound on the matrix norm. Let us look at the norm of

$\left\|A(\theta)^{-1}\right\|_2$,

$$\begin{aligned}\left\|A(\theta)^{-1}\right\|_2^2 &= \lambda_{\max}\left(\left(A(\theta)^{-1}\right)' A(\theta)^{-1}\right)\\ &= \lambda_{\max}\left(\left(A(\theta)'\right)^{-1} A(\theta)^{-1}\right)\\ &= \lambda_{\max}\left(\left(A(\theta)A(\theta)'\right)^{-1}\right)\\ &= 1/\lambda_{\min}\left(A(\theta)A(\theta)'\right)\\ &= 1/\lambda_{\min}\left(\left(A(\theta)'A(\theta)\right)'\right)\\ &= 1/\lambda_{\min}\left(A(\theta)'A(\theta)\right),\end{aligned}$$

thus, we the lower bound on $\left\|A(\theta)^{-1}\right\|_2$ is $\sqrt{1/\lambda_{\min}\left(A(\theta)'A(\theta)\right)}$, that is, b_A.

5. Let us look at the ij entry of the matrix $\frac{d}{dt}A(\theta)$, where using similar arguments to Section 2 we get

$$\begin{aligned}\left[\left|\frac{d}{dt}A(\theta)\right|\right]_{ij} &= \left[\left|\frac{d}{dt}\left(\Phi'\Pi(\theta)\left((1-\lambda)\sum_{m=0}^{\infty}\lambda^m P(\theta)^{m+1}-I\right)\Phi\right)\right|\right]_{ij}\\ &\leq \left[\left|\Phi'\frac{d}{dt}(\Pi(\theta))\left((1-\lambda)\sum_{m=0}^{\infty}\lambda^m P(\theta)^{m+1}-I\right)\Phi\right|\right]_{ij}\\ &\quad+\left[\left|\Phi'\Pi(\theta)\frac{d}{dt}\left((1-\lambda)\sum_{m=0}^{\infty}\lambda^m P(\theta)^{m+1}-I\right)\Phi\right|\right]_{ij}\\ &\leq B_\Phi B_{\pi 1}\frac{1}{1-\lambda}B_\Phi + B_\Phi\frac{1}{(1-\lambda)^2}B_{P1}B_\Phi.\end{aligned}$$

Since the matrix entries are uniformly bounded in θ, so is the matrix $\frac{d}{dt}A(\theta)'\frac{d}{dt}A(\theta)$, and so is the largest eigenvalue of $\frac{d}{dt}A(\theta)'\frac{d}{dt}A(\theta)$ which implies the uniform boundedness of $\left\|\frac{d}{dt}A(\theta)\right\|_2$.

6. For a general invertible square matrix, $X(t)$, we have

$$0 = \frac{d}{dt}I = \frac{d}{dt}\left(X(t)^{-1}X(t)\right) = \frac{d}{dt}\left(X(t)^{-1}\right)X(t) + X(t)^{-1}\frac{d}{dt}(X(t)).$$

Rearranging it we get

$$\frac{d}{dt}\left(X(t)^{-1}\right) = -X(t)^{-1}\frac{d}{dt}(X(t))X(t)^{-1}.$$

Using this identity yields

$$\begin{aligned}\left\|\frac{d}{dt}\left(A(\theta)^{-1}\right)\right\|_2 &= \left\|-A(\theta)^{-1}\frac{d}{dt}(A(\theta))A(\theta)^{-1}\right\|_2\\ &\leq \left\|A(\theta)^{-1}\right\|_2\cdot\left\|\frac{d}{dt}(A(\theta))\right\|_2\cdot\left\|-A(\theta)^{-1}\right\|_2\\ &= b_A^2 B_{A1}.\end{aligned}$$

7. Examining the norm of $\frac{d}{dt}w^*$ yields

$$\begin{aligned}\left\|\frac{d}{dt}w^*\right\|_2 &= \left\|\frac{d}{dt}\left(A(\theta)^{-1}b(\theta)\right)\right\|_2 \\ &= \left\|\frac{d}{dt}A(\theta)^{-1}b(\theta)+A(\theta)^{-1}\frac{d}{dt}b(\theta)\right\|_2 \\ &\leq b_A^2 B_{A1}\frac{1}{1-\lambda}|\mathcal{X}|^3 B_\Phi B_r + b_A\tilde{B} \\ &= B_{w1}.\end{aligned}$$

We wish to use Lemma 23 for (17), thus, we show that the assumptions of Lemma 23 are valid.

Lemma 28

1. We have

$$\limsup_{t\to\infty}\|w^*(\theta(t))-w(t)\|_2 \leq \frac{1}{\Gamma_w}B_{\Delta w}, \tag{34}$$

where

$$B_{\Delta w} \triangleq \frac{B_{w1}+B_G\frac{B_{\Delta\eta}}{\Gamma_\eta}}{\gamma}.$$

(a) We have

$$\limsup_{t\to\infty}\|h(\theta(t))-\tilde{h}(w(t))\|_\infty \leq \frac{B_{\Delta h1}}{\Gamma_w}+\frac{\varepsilon_{\text{app}}}{\sqrt{b_\pi}},$$

where

$$B_{\Delta h} \triangleq |\mathcal{X}|L(B_{\Delta w})^2.$$

Proof ■

1. Without loss of generality, we can eliminate the projection operator since we can choose B_w to be large enough such that $w^*(\theta)$ will be inside the bounded space. We take $\mathcal{M}(t)=A(\theta)$, $F_1(t)=w^*(\theta(t))$, and $F_2(t)=G(\theta)(\eta(\theta)-\tilde{\eta})$. By previous lemmas we can see that the Assumption 23 holds. By Lemma 27 (6), $\|\dot{w}^*(\theta)\|_2$ is bounded by B_{w_1}, by Lemma 26 we have a bound on $|(\eta(\theta)-\tilde{\eta})|$, and by Lemma 24 we have a bound on $w'A(\theta)w$. Using these bounds and applying Lemma 23 provides the desired result.

(a) Suppressing the time dependence for simplicity and expressing $\|h(\theta)-\tilde{h}(w)\|_\infty$ using ε_{app} and the previous result yields

$$\begin{aligned}\|h(\theta)-\tilde{h}(w)\|_\infty &\leq \|h(\theta)-\tilde{h}(w)\|_2 \\ &= \|h(\theta)-\tilde{h}(w^*)+\tilde{h}(w^*)-\tilde{h}(w)\|_2 \\ &\leq \|h(\theta)-\tilde{h}(w^*)\|_2+\|\tilde{h}(w^*)-\tilde{h}(w)\|_2\end{aligned} \tag{35}$$

For the first term on the r.h.s. of the final equation in (35) we have

$$\begin{aligned}\|h(\theta)-\tilde{h}(w^*)\|_2 &= \left\| \left(\Pi(\theta)^{-\frac{1}{2}}\right)\left(\Pi(\theta)^{\frac{1}{2}}\right)\left(h(\theta)-\tilde{h}(w^*)\right)\right\|_2 \\ &\leq \left\|\Pi(\theta)^{-\frac{1}{2}}\right\|_2 \left\|h(\theta)-\tilde{h}(w^*)\right\|_{\Pi(\theta)} \\ &\leq \frac{\varepsilon_{\text{app}}}{(b_\pi)^{\frac{1}{2}}}\end{aligned}$$

where we use the sub-additivity of the matrix norms in the first inequality, and Lemma 3 and the (10) in the last inequality. For the second term on the r.h.s. of the final equation in (35) we have

$$\begin{aligned}\|\tilde{h}(w^*)-\tilde{h}(w)\|_2^2 &= \|\Phi(w^*(\theta)-w)\|_2^2 \\ &= \sum_{k=1}^{|\mathcal{X}|}\left(\sum_{l=1}^{L}\phi_l(k)\left(w_l^*(\theta)-w_l\right)\right)^2 \\ &\leq \sum_{k=1}^{|\mathcal{X}|}\left(\left(\sum_{l=1}^{L}\phi_l^2(k)\right)^{\frac{1}{2}}\left(\sum_{l=1}^{L}\left(w_l^*(\theta)-w_l\right)^2\right)^{\frac{1}{2}}\right)^2 \\ &\leq \sum_{k=1}^{|\mathcal{X}|}\left(\sum_{l=1}^{L}\phi_l^2(k)\right)\left(\sum_{l=1}^{L}\left(w_l^*(\theta)-w_l\right)^2\right) \\ &\leq |\mathcal{X}|L\|w^*(\theta)-w\|_2^2 \\ &= |\mathcal{X}|L\left(B_{\Delta w}\right)^2.\end{aligned} \tag{36}$$

Combining (34)-(36) yields the desired result.

Using Lemma 28 we can provide a bound on second term of (17).

Lemma 29 *We have*

$$\limsup_{t\to\infty}\left\|\sum_{x,y\in\mathcal{X}\times\mathcal{X},u\in\mathcal{U}} D^{(x,u,y)}(\theta)\left(d(x,y,\theta)-\tilde{d}(x,y,w)\right)\right\|_2 \leq \frac{B_{\Delta td1}}{\Gamma_w}+\frac{B_{\Delta td2}}{\Gamma_\eta}+B_{\Delta td3}\varepsilon_{app}$$

where

$$B_{\Delta td1}=\frac{1}{\Gamma_w}\cdot 2B_\Psi B_{\Delta h1},\quad B_{\Delta td2}=\frac{1}{\Gamma_\eta}\cdot B_{\Delta\eta}B_\Psi,\quad B_{\Delta td3}=\frac{2B_\Psi}{\sqrt{b_\pi}}.$$

Proof Simplifying the notation by suppressing the time dependence, we bound the TD signal in the limit, that is,

$$\begin{aligned}\limsup_{t\to\infty}|d(x,y,\theta)-\tilde{d}(x,y,w)| &= \limsup_{t\to\infty}|(r(x)-\eta(\theta)+h(y,\theta)-h(x,\theta)) \\ &\quad -\left(r(x)-\tilde{\eta}+\tilde{h}(y,w)-\tilde{h}(x,w)\right)| \\ &\leq \limsup_{t\to\infty}|\eta(\theta)-\tilde{\eta}|+\limsup_{t\to\infty}2\left\|h(\theta)-\tilde{h}(w)\right\|_\infty \\ &= \frac{B_{\Delta\eta}}{\Gamma_\eta}+2\left(\frac{B_{\Delta h1}}{\Gamma_w}+\frac{\varepsilon_{\text{app}}}{\sqrt{b_\pi}}\right).\end{aligned}$$

With some more algebra we have

$$\begin{aligned}
&\limsup_{t\to\infty}\left\|\sum_{x,y\in\mathcal{X}\times\mathcal{X},u\in\mathcal{U}} D^{(x,u,y)}(\theta)\left(d(x,y,\theta)-\tilde{d}(x,y,w)\right)\right\| \\
&\leq \limsup_{t\to\infty}\sum_{x,y\in\mathcal{X}\times\mathcal{X},u\in\mathcal{U}} \pi(x)P(u|x,\theta_n)P(y|x,u)\left\|\psi(x,u,\theta_n)\right\|\cdot\left|d(x,y,\theta)-\tilde{d}(x,y,w)\right| \\
&\leq B_\Psi\left(\frac{B_{\Delta\eta}}{\Gamma_\eta}+2\left(\frac{B_{\Delta h1}}{\Gamma_w}+\frac{\varepsilon_{\text{app}}}{\sqrt{b_\pi}}\right)\right) \\
&= \frac{B_{\Delta td1}}{\Gamma_w}+\frac{B_{\Delta td2}}{\Gamma_\eta}+B_{\Delta td3}\varepsilon_{\text{app}}.
\end{aligned}$$

■

We see that the term in this bound is adjustable by choosing appropriate Γ_η and Γ_w. The concluding lemma proves the conclusion of Theorem 8.

D.1 Proof of Theorem 8

We define

$$B_{\nabla\eta} \triangleq \frac{B_{\Delta td1}}{\Gamma_w}+\frac{B_{\Delta td2}}{\Gamma_\eta}+B_{\Delta td3}\varepsilon_{\text{app}}.$$

For an arbitrary $\delta>0$, define the set

$$\mathcal{B}_\delta \triangleq \{\theta:\ \|\nabla\eta(\theta)\|\leq B_{\nabla\eta}+\delta\}.$$

We claim that the trajectory $\eta(\theta)$ visits $\mathcal{B}_\delta$ infinitely often. Assume the contrary that

$$\liminf_{t\to\infty}\|\nabla\eta(\theta)\|_2 > B_{\nabla\eta}+\delta. \tag{37}$$

Thus, on the set $\mathcal{B}_\delta^c$ for t large enough we have

$$\begin{aligned}
\dot{\eta}(\theta) &= \nabla\eta(\theta)\cdot\dot{\theta} \\
&= \nabla\eta(\theta)\cdot\left(\nabla\eta(\theta)+\sum_{x,y\in\mathcal{X}\times\mathcal{X}} D^{(x,y)}(\theta)\left(d(x,y)-\tilde{d}(x,y)\right)\right) \\
&= \|\nabla\eta(\theta)\|_2^2+\nabla\eta(\theta)\cdot\left(\sum_{x,y\in\mathcal{X}\times\mathcal{X}} D^{(x,y)}(\theta)\left(d(x,y)-\tilde{d}(x,y)\right)\right) \\
&\geq \|\nabla\eta(\theta)\|_2^2-\|\nabla\eta(\theta)\|_2\left\|\sum_{x,y\in\mathcal{X}\times\mathcal{X}} D^{(x,y)}(\theta)\left(d(x,y)-\tilde{d}(x,y)\right)\right\|_2 \\
&= \|\nabla\eta(\theta)\|_2\left(\|\nabla\eta(\theta)\|_2-B_{\nabla\eta}\right) \\
&\geq \|\nabla\eta(\theta)\|_2\left(B_{\nabla\eta}+\delta-B_{\nabla\eta}\right) \\
&> (B_{\nabla\eta}+\delta)\delta.
\end{aligned}$$

By (37), there exists a time t_0 which for all $t>t_0$ we have $\eta(\theta)\in\mathcal{B}_\delta^c$. Therefore,

$$\eta(\infty)=\eta(t_0)+\int_{t_0}^{\infty}\dot{\eta}(\theta)dt > \eta(t_0)+\int_{t_0}^{\infty}(B_D+\delta)\delta dt=\infty,$$

which contradicts the boundedness of $\eta(\theta)$. Since the claim holds for all $\delta>0$, the result follows.

References

T. Archibald, K. McKinnon, and L. Thomas. On the generation of markov decision processes. *Journal of the Operational Research Society*, 46:354–361, 1995.

D. Baras and R. Meir. Reinforcement learning, spike time dependent plasticity and the bcm rule. *Neural Comput.*, 19(8):2245–2279, Aug 2007.

J. Baxter and P.L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

D.P. Bertsekas. *Dynamic Programming and Optimal Control, Vol I & II, 3rd Ed.* Athena Scinetific, 2006.

D.P. Bertsekas and J. Tsitsiklis. *Neuro-dynamic Programming.* Athena Scinetific, 1996.

S. Bhatnagar, R.S. Sutton, M. Ghavamzadeh, and M. Lee. Incremental natural actor-critic algorithms. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 105–112, Cambridge, MA, 2008. MIT Press.

S. Bhatnagar, R.S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

V.S. Borkar. Stochastic approximation with two time scales. *Syst. Control Lett.*, 29(5):291–294, 1997.

P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues.* Springer, 1999.

X. Cao. *Stochastic Learning and Optimization: A Sensitivity-Based Approach (International Series on Discrete Event Dynamic Systems).* Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2007.

X.R. Cao and H.F. Chen. Pertubation realization, potentials, and sensitivity analysis of markov processes. *IEEE Trans. Automat. Contr*, 42:1382–1393, 1997.

N.D. Daw, Y. Niv, , and P. Dayan. *Actions, Policies, Values, and the Basal Ganglia - In: Bezard, E editor, Recent Breakthroughs in Basal Ganglia Research.* Nova Science Publishers Inc., 2006.

D. DiCastro, D. Volkinstein, and R. Meir. Temporal difference based actor critic algorithms single time scale convergence and neural implementation. In *Advances in Neural Information Processing Systems*, accepted, 2008.

R.V. Florian. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation*, 19:1468–1502, 2007.

R.G. Gallager. *Discrete Stochastic Processes.* Kluwer Academic Publishers, 1995.

E. Greensmith, P.L. Bartlett, and J. Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, 1985.

H. K. Khalil. *Nonlinear Systems, 3rd Ed.* Prentice Hall, 2002.

V.R. Konda and V.S. Borkar. Actor-critic like learning algorithms for markov decision processes. *SIAM Journal on Control and Optimization*, pages 94–123, 1999.

V.R. Konda and J. Tsitsiklis. On actor critic algorithms. *SIAM J. Control Optim.*, 42(4):1143–1166, 2003.

H.J. Kushner and G.G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer, 1997.

P. Marbach and J. Tsitsiklis. Simulation-based optimization of markov reward processes. *IEEE. Trans. Auto. Cont.*, 46(2):191–209, 1998.

A. Mokkadem and M. Pelletier. Convergence rate and averaging of nonlinear two-time-scale stochastic approximation algorithms. *Annals of Applied Probability*, 16(3):1671, 2006.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71:1180–1190, 2008.

B.T. Polyak. New method of stochastic approximation type. *Automat. Remote Control*, 51:937–946, 1990.

M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc, 1994.

W. Schultz. Getting formal with dopamine and reward. *Neuron*, 36(2):241–63, 2002.

S. Singh and P. Dayan. Analytical mean squared error curves for temporal difference learning. *Machine Learning*, 32:5–40, 1998.

R.S. Sutton and A.G. Barto. *Reinforcement Learning*. MIT Press, 1998.

G. Tesauro. Temporal difference learning and the td-gammon. *Communication of the ACM*, 38(3), March 1995.

J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, May 1997.

Journal of Machine Learning Research 11 (2010) 411-450 Submitted 11/07; Revised 11/09; Published 1/10

Dimensionality Estimation, Manifold Learning and Function Approximation using Tensor Voting

Philippos Mordohai MORDOHAI@CS.STEVENS.EDU
Castle Point on Hudson
Department of Computer Science
Stevens Institute of Technology
Hoboken, NJ 07030, USA

Gérard Medioni MEDIONI@IRIS.USC.EDU
3737 Watt Way
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, CA 90089, USA

Editor: Sanjoy Dasgupta

Abstract

We address instance-based learning from a perceptual organization standpoint and present methods for dimensionality estimation, manifold learning and function approximation. Under our approach, manifolds in high-dimensional spaces are inferred by estimating geometric relationships among the input instances. Unlike conventional manifold learning, we do not perform dimensionality reduction, but instead perform all operations in the original input space. For this purpose we employ a novel formulation of tensor voting, which allows an N-D implementation. Tensor voting is a perceptual organization framework that has mostly been applied to computer vision problems. Analyzing the estimated local structure at the inputs, we are able to obtain reliable dimensionality estimates at each instance, instead of a global estimate for the entire data set. Moreover, these local dimensionality and structure estimates enable us to measure geodesic distances and perform nonlinear interpolation for data sets with varying density, outliers, perturbation and intersections, that cannot be handled by state-of-the-art methods. Quantitative results on the estimation of local manifold structure using ground truth data are presented. In addition, we compare our approach with several leading methods for manifold learning at the task of measuring geodesic distances. Finally, we show competitive function approximation results on real data.

Keywords: dimensionality estimation, manifold learning, geodesic distance, function approximation, high-dimensional processing, tensor voting

1. Introduction

In this paper, we address a subfield of machine learning that operates in continuous domains and learns from observations that are represented as points in a Euclidean space. This type of learning is termed *instance-based* or *memory-based* learning (Mitchell, 1997). The goal of instance-based learning is to learn the underlying relationships between observations, which are points in an N-D continuous space, under the assumption that they lie in a limited part of the space, typically a manifold, the dimensionality of which is an indication of the degrees of freedom of the underlying system.

Instance-based learning has recently received renewed interest from the machine learning community, due to its many applications in the fields of pattern recognition, data mining, kinematics, function approximation and visualization, among others. This interest was sparked by a wave of new algorithms that advanced the state of the art and are capable of learning nonlinear manifolds in spaces of very high dimensionality. These include kernel PCA (Schölkopf et al., 1998), locally linear embedding (LLE) (Roweis and Saul, 2000), Isomap (Tenenbaum et al., 2000) and charting (Brand, 2003), which are reviewed in Section 2. They aim at reducing the dimensionality of the input space in a way that preserves certain geometric or statistical properties of the data. Isomap, for instance, attempts to preserve the geodesic distances between all points as the manifold is "unfolded" and mapped to a space of lower dimension.

Our research focuses on data presented as large sets of observations, possibly containing outliers, in high dimensions. We view the problem of learning an unknown function based on these observations as equivalent to learning a manifold, or manifolds, formed by a set of points. Having a good estimate of the manifold's structure, one is able to predict the positions of other points on it. The first task is to determine the intrinsic dimensionality of the data. This can provide insights on the complexity of the system that generates the data, the type of model needed to describe it, as well as the actual degrees of freedom of the system, which are not equal to the dimensionality of the input space, in general. We also estimate the orientation of a potential manifold that passes through each point. Dimensionality estimation and structure inference are accomplished simultaneously by encoding the observations as symmetric, second order, non-negative definite tensors and analyzing the outputs of tensor voting (Medioni et al., 2000). Since the process that estimates dimensionality and orientation is performed on the inputs, our approach falls under the "eager learning" category, according to Mitchell (1997). Unlike other eager approaches, however, ours is not global. This offers considerable advantages when the data become more complex, or when the number of instances is large.

We take a different path to manifold learning than Roweis and Saul (2000), Tenenbaum et al. (2000) and Brand (2003). Whereas these methods address the problem as one of dimensionality reduction, we propose an approach that does not embed the data in a lower dimensional space. Preliminary versions of this approach were published in Mordohai and Medioni (2005) and Mordohai (2005). A similar methodology was also presented by Dollár et al. (2007a). We compute local dimensionality estimates, but instead of performing dimensionality reduction, we perform all operations in the original input space, taking into account the estimated dimensionality of the data. We also estimate the orientation of the manifold locally and are able to approximate intrinsic or geodesic distances[1] and perform nonlinear interpolation. Since we perform all processing in the input space we are able to process data sets that are not manifolds globally, or ones with varying intrinsic dimensionality. The latter pose no additional difficulties, since we do not use a global estimate for the dimensionality of the data. Moreover, outliers, boundaries, intersections or disconnected components are handled naturally as in 2-D and 3-D (Medioni et al., 2000; Tang and Medioni, 1998). Quantitative results for the robustness to outliers that outnumber the inliers are presented in Sections 5 and 6. Once processing under our approach has been completed, dimensionality reduction

1. The requirements for a distance to be geodesic are stricter than those for being intrinsic. Intrinsic distances are computed on the manifold, while geodesics are based on the gradient of the distance function. For the manifolds we examine in this paper, these two almost always coincide. See Mémoli and Sapiro (2005) and references therein for more details.

can be performed using any of the approaches described in the next section to reduce the storage requirements, if appropriate and desirable.

Manifold learning serves as the basis for the last part of our work, which addresses function approximation. As suggested by Poggio and Girosi (1990), function approximation from samples and hypersurface inference are equivalent. The main assumption is that some form of smoothness exists in the data and unobserved outputs can be predicted from previously observed outputs for similar inputs. The distinction between low and high-dimensional spaces is necessary, since highly specialized methods for low-dimensional cases exist in the literature. Our approach is local, non-parametric and has a weak prior model of smoothness, which is implemented in the form of votes that communicate a point's preferred orientation to its neighbors. This generic prior and the absence of global computations allow us to address a large class of functions as well as data sets comprising very large numbers of observations. As most of the local methods reviewed in the next section, our algorithm is memory-based. This increases flexibility, since we can process data that do not conform to pre-specified models, but also increases storage requirements, since all samples are kept in memory.

All processing in our method is performed using tensor voting, which is a computational framework for perceptual organization based on the Gestalt principles of proximity and good continuation (Medioni et al., 2000). It has mainly been applied to organize generic points into coherent groups and for computer vision problems that were formulated as perceptual organization tasks. For instance, the problem of stereo vision can be formulated as the organization of potential pixel correspondences into salient surfaces, under the assumption that correct correspondences form coherent surfaces and wrong ones do not (Mordohai and Medioni, 2006). Salient structures are inferred based on the support potential correspondences receive from their neighbors in the form of votes, which are also second order tensors that are cast from each point to all other points within its neighborhood. Each vote conveys the orientation the receiver would have if the voter and receiver were in the same structure. In Section 3, we present a new implementation of tensor voting that is not limited to low-dimensional spaces as the original one of Medioni et al. (2000).

The paper is organized as follows: an overview of related work including the algorithms that are compared with ours is given in the next section; a new implementation of tensor voting applicable to N-D data is described in Section 3; results in dimensionality estimation are presented in Section 4, while results in local structure estimation are presented in Section 5; our algorithm for estimating geodesic distances and a quantitative comparison with state of the art methods are shown in Section 6; function approximation is described in Section 7; finally, Section 8 concludes the paper.

2. Related Work

In this section, we present related work in the domains of dimensionality estimation, manifold learning and multivariate function approximation.

2.1 Dimensionality Estimation

Bruske and Sommer (1998) present an approach for dimensionality estimation where an optimally topology preserving map (OTPM) is constructed for a subset of the data, which is produced after vector quantization. Principal Component Analysis (PCA) is then performed for each node of the OTPM under the assumption that the underlying structure of the data is locally linear. The average of the number of significant singular values at the nodes is the estimate of the intrinsic dimensionality.

Kégl (2003) estimates the capacity dimension of a manifold, which is equal to the topological dimension and does not depend on the distribution of the data, using an efficient approximation based on packing numbers. The algorithm takes into account dimensionality variations with scale and is based on a geometric property of the data, rather than successive projections to increasingly higher-dimensional subspaces until a certain percentage of the data is explained. Raginsky and Lazebnik (2006) present a family of dimensionality estimators based on the concept of quantization dimension. The family is parameterized by the distortion exponent and includes the method of Kégl (2003) when the distortion exponent tends to infinity. The authors show that small values of the distortion exponent yield estimators that are more robust to noise.

Costa and Hero (2004) estimate the intrinsic dimension of the manifold and the entropy of the samples using geodesic-minimal-spanning trees. The method, similarly to Isomap (Tenenbaum et al., 2000), considers global properties of the adjacency graph and thus produces a single global estimate.

Levina and Bickel (2005) compute maximum likelihood estimates of dimensionality by examining the number of neighbors included in spheres, the radii of which are selected in such a way that they contain enough points and that the density of the data contained in them can be assumed constant. These requirements cause an underestimation of the dimensionality when it is very high.

The difference between our approach and those of Bruske and Sommer (1998), Kégl (2003), Brand (2003), Weinberger and Saul (2004), Costa and Hero (2004) and Levina and Bickel (2005) is that it produces reliable dimensionality estimates at the point level. While this is not important for data sets with constant dimensionality, the ability to estimate local dimensionality reliably becomes a key factor when dealing with data generated by different unknown processes. Given reliable local estimates, the data set can be segmented in components with constant dimensionality.

2.2 Manifold Learning

Here, we briefly present recent approaches for learning low dimensional embeddings from points in high dimensional spaces. Most of them are inspired by linear techniques, such as Principal Component Analysis (PCA) (Jolliffe, 1986) and Multi-Dimensional Scaling (MDS) (Cox and Cox, 1994), based on the assumption that nonlinear manifolds can be approximated by locally linear parts.

Schölkopf et al. (1998) propose kernel PCA that extends linear PCA by implicitly mapping the inputs to a higher-dimensional space via kernels. Conceptually, applying PCA in the high-dimensional space allows the extraction of principal components that capture more information than their counterparts in the original space. The mapping to the high-dimensional space does not need to carried out explicitly, since dot product computations suffice. The choice of kernel is still an open problem. Weinberger et al. (2004) describe an approach to compute the kernel matrix by maximizing variance in feature space in the context of dimensionality reduction.

Locally Linear Embedding (LLE) was presented by Roweis and Saul (2000) and Saul and Roweis (2003). The underlying assumption is that if data lie on a locally linear, low-dimensional manifold, then each point can be reconstructed from its neighbors with appropriate weights. These weights should be the same in a low-dimensional space, the dimensionality of which is greater or equal to the intrinsic dimensionality of the manifold. The LLE algorithm computes the basis of such a low-dimensional space. The dimensionality of the embedding, however, has to be given as a parameter, since it cannot always be estimated from the data (Saul and Roweis, 2003). Moreover,

the output is an embedding of the given data, but not a mapping from the ambient to the embedding space. Global coordination of the local embeddings, and thus a mapping, can be computed according to Teh and Roweis (2003). LLE is not isometric and often fails by mapping distant points close to each other.

Tenenbaum et al. (2000) propose Isomap, which is an extension of MDS that uses geodesic instead of Euclidean distances and thus can be applied to nonlinear manifolds. The geodesic distances between points are approximated by graph distances. Then, MDS is applied on the geodesic distances to compute an embedding that preserves the property of points to be close or far away from each other. Isomap can handle points not in the original data set, and perform interpolation. C-Isomap, a variant of Isomap applicable to data with intrinsic curvature, but known distribution, and L-Isomap, a faster alternative that only uses a few landmark point for distance computations, have also been proposed by de Silva and Tenenbaum (2003). Isomap and its variants are limited to convex data sets.

The Laplacian Eigenmaps algorithm was developed by Belkin and Niyogi (2003). It computes the normalized graph Laplacian of the adjacency graph of the input data, which is an approximation of the Laplace-Beltrami operator on the manifold. It exploits locality preserving properties that were first observed in the field of clustering. The Laplacian Eigenmaps algorithm can be viewed as a generalization of LLE, since the two become identical when the weights of the graph are chosen according to the criteria of the latter. Much like LLE, the dimensionality of the manifold also has to be provided, the computed embeddings are not isometric and a mapping between the two spaces is not produced. The latter is addressed by He and Niyogi (2004) where a variation of the algorithm is proposed.

Donoho and Grimes (2003) propose Hessian LLE (HLLE), an approach similar to the above, which computes the Hessian instead of the Laplacian of the graph. The authors claim that the Hessian is better suited than the Laplacian in detecting linear patches on the manifold. The major contribution of this approach is that it proposes a global, isometric method, which, unlike Isomap, can be applied to non-convex data sets. The requirement to estimate second derivatives from possibly noisy, discrete data makes the algorithm more sensitive to noise than the others reviewed here.

Semidefinite Embedding (SDE) was proposed by Weinberger and Saul (2004, 2006) who address the problem of manifold learning by enforcing local isometry. The lengths of the sides of triangles formed by neighboring points are preserved during the embedding. These constraints can be expressed in terms of pairwise distances and the optimal embedding can be found by semidefinite programming. The method is among the most computationally demanding reviewed here, but can reliably estimate the underlying dimensionality of the inputs by locating the largest gap between the eigenvalues of the Gram matrix of the outputs. Similarly to our approach, dimensionality estimation does not require a threshold.

Other research related to ours includes the charting algorithm of Brand (2003). It computes a pseudo-invertible mapping of the data, as well as the intrinsic dimensionality of the manifold, which is estimated by examining the rate of growth of the number of points contained in hyper-spheres as a function of the radius. Linear patches, areas of curvature and noise can be distinguished using the proposed measure. At a subsequent stage a global coordinate system for the embedding is defined. This produces a mapping between the input space and the embedding space.

Wang et al. (2005) propose an adaptive version of the local tangent space alignment (LTSA) of Zhang and Zha (2004), a local dimensionality reduction method that is a variant of LLE. Wang et al. address a limitation of most of the approaches presented in this section, which is the use of

a fixed number of neighbors (k) for all points in the data. Inappropriate selection of k can cause problems at points near boundaries, or if the density of the data is not approximately constant. The authors propose a method to adapt the neighborhood size according to local criteria and demonstrate its effectiveness on data sets of varying distribution. Using an appropriate value for k at each point is important for graph-based methods, since the contributions of each neighbor are typically not weighted, making the algorithms very sensitive to the selection of k.

In a more recent paper, Sha and Saul (2005) propose Conformal Eigenmaps, an algorithm that operates on the output of LEE or Laplacian Eigenmaps to produce a conformal embedding, which preserves angles between edges in the original input space, without incurring a large increase in computational cost. A similar approach that "stiffens" the inferred manifolds employing a multi-resolution strategy was proposed by Brand (2005). Both these papers address the limitation of some of the early algorithms which preserve graph connectivity, but not local structure, during the embedding.

The most similar method to ours is that of Dollár et al. (2007a) and Dollár et al. (2007b) in which the data are not embedded in a lower dimensional space. Instead, the local structure of a manifold at a point is learned from neighboring observations and represented by a set of radial basis functions (RBFs) centered on K points discovered by K-means clustering. The manifold can then be traversed by "walking" on its tangent space between and beyond the observations. Representation by RBFs without dimensionality reduction allows the algorithm to be robust to outliers and be applicable to non-isometric manifolds. An evaluation of manifold learning using geodesic distance preservation as a metric, similar to the one of Section 6.1, is presented in Dollár et al. (2007b).

A different approach for intrinsic distance estimation that bypasses learning the structure of the manifold has been proposed by Mémoli and Sapiro (2005). It approximates intrinsic distances and geodesics by computing extrinsic Euclidean distances in a thin band that surrounds the points. The algorithm can handle manifolds in any dimension and of any co-dimension and is more robust to noise than graph-based methods, such as Isomap, since in the latter the outliers are included in the graph and perturb the approximation of geodesics.

Souvenir and Pless (2005) present an approach capable of learning multiple, potentially intersecting, manifolds of different dimensionality using an expectation maximization (EM) algorithm with a variant of MDS as the M step. Unlike our approach, however, the number and dimensionality of the manifolds have to be provided externally.

2.3 Function Approximation

Here we review previous research on function approximation focusing on methods that are applicable to large data sets in high-dimensional spaces. Neural networks are often employed as global methods for function approximation. Poggio and Girosi (1990) addressed function approximation in a regularization framework implemented as a three-layer neural network. They view the problem as hypersurface reconstruction, where the main assumption is that of smoothness. The emphasis is on the selection of the appropriate approximating functions and optimization algorithm. Other research based on neural networks includes the work of Sanger (1991) who proposed a tree-structured neural network, which does not suffer from the exponential growth with dimensionality of the number of models and parameters that plagued previous approaches. It does, however, require the selection of an appropriate set of basis functions to approximate a given function. Neural network based approaches with pre-specified types of models are also proposed by: Barron (1993) who derived

the bounds for approximation using a superposition of sigmoidal functions; Breiman (1993) who proposed a simpler and faster model based on hinging hyperplanes; and Saha et al. (1993) who used RBFs.

Xu et al. (1995) modified the training scheme for the mixture of experts architecture so that a single-loop EM algorithm is sufficient for optimization. Mitaim and Kosko (2001) approached the problem within the fuzzy systems framework. They investigated the selection of the shape of fuzzy sets for an adaptive fuzzy system and concluded that no shape emerges as the best choice. These approaches, as well as the ones based on neural networks, are global and model-based. They can achieve good performance, but they require all the inputs to be available at the same time for training and the selection of an appropriate model that matches the unknown function. If the latter is complex, the resulting model may have an impractically large number of parameters.

Support Vector Machines (SVMs), besides classification, have also been extensively applied for regression based on the work of Vapnik (1995). Collobert and Bengio (2001) address a limitation of the SVM algorithm for regression, which is its increased computational complexity as the number of samples grows, with a decomposition algorithm. It operates on a working set of the variables, while keeping fixed variables that are less likely to change.

All the above methods are deterministic and make hard decisions. On the other hand, Bayesian learning brings the advantages of probabilistic predictions and a significant decrease in the number of basis functions. Tresp (2000) introduced the Bayesian Committee Machine that is able to handle large data sets by splitting them in subsets and training an estimator for each. These estimators are combined with appropriate weights to generate the prediction. What is noteworthy about this approach is the fact that the positions of query points are taken into account in the design of the estimator and that performance improves when multiple query points are processed simultaneously. Tipping (2001) proposed a sparse Bayesian learning approach, which produces probabilistic predictions and automatically detects nuisance parameters, and the Relevance Vector Machine that can be viewed as stochastic formulation of an SVM. A Bayesian treatment of SVM-based regression can also be found in the work of Chu et al. (2004). Its advantages include reduced computational complexity over Gaussian Process Regression (GPR), reviewed below, and robustness against outliers. Inspired by Factor Analysis Regression, Ting et al. (2006) propose a Bayesian regression algorithm that is robust to ill-conditioned data, detects relevant features and identifies input and output noise.

An approach that has attracted a lot of attention is the use of Gaussian Processes (GPs) for regression. Williams and Rasmussen (1996) observed that Bayesian analysis of neural networks is difficult due to complex prior distributions over functions resulting even from simple priors over weights. Instead, if one uses Gaussian processes as priors over the functions, then Bayesian analysis can be carried out exactly. Despite the speed up due to GPs, faster implementations were still needed for practical applications. A sparse greedy GP regression algorithm was presented by Smola and Bartlett (2001) who approximate the MAP estimate by expanding in terms of a small set of kernels. Csató and Opper (2002) described an alternative sparse representation for GP regression models. It operates in an online fashion and maintains a sparse basis which is dynamically updated as more data become available.

Lawrence et al. (1996) compared a global approach using a multi-layer perceptron with a linear local approximation model. They found that the local model performed better when the density of the input data deviated a lot from being uniform. Furthermore, the local model allowed for incremental learning and cross-validation. On the other hand, it showed poorer generalization, slower performance after training and required more memory, since all input data had to be stored. The

global model performed better in higher dimensions, where data sparsity becomes a serious problem for the local alternative. Wedge et al. (2006) bring together the advantages of global and local approaches using a hybrid network architecture that combines RBFs and sigmoid neural networks. It first identifies global features of the system before adding local details via the RBFs.

Schaal and Atkeson (1998) proposed a nonparametric, local, incremental learning approach based on receptive field weighted regression. The approach is truly local since the parameters for each model and the size and shape of each receptive field are learned independently. The provided mechanisms for the addition and pruning of local models enable incremental learning as new data points become available.

Atkeson et al. (1997) survey local weighted learning methods and identify the issues that must be taken into account. These include the selection of the distance metric, the weighting function, prediction assessment and robustness to noise. The authors argue that in certain cases no values of the parameters of a global model can provide a good approximation of the true function. In these cases, a local approximation using a simpler, even linear model, is a better approach than increasing the complexity of the global model. Along these lines, Vijayakumar and Schaal (2000) proposed locally weighted projection regression, an algorithm based on successive univariate regressions along projections of the data in directions given by the gradient of the underlying function.

We opt for a local approach and address the problem as an extension of manifold learning. Note, however, that we are not limited to functions that are strictly manifolds. Using tensor voting, we are able to reliably estimate the normal and tangent space at each sample, as described in the following section. These estimates allow us to perform nonlinear interpolation and generate outputs for unobserved inputs, even under severe noise corruption.

3. Tensor Voting in High-Dimensional Spaces

The tensor voting framework, in its preliminary version (Guy and Medioni, 1996), is an implementation of two Gestalt principles, namely proximity and good continuation, for grouping generic tokens in 2-D. The 2-D domain has always been the main focus of research in perceptual organization, beginning with the research of Köhler (1920), Wertheimer (1923) and Koffka (1935). The generalization of perceptual organization to 3-D is relatively straightforward, since salient groupings can be detected by the human visual system in 3-D based on the same principles. Guy and Medioni (1997) extended tensor voting to three dimensions. The term saliency here refers to *structural saliency*, which, according to Shashua and Ullman (1988) is the property of structures to stand out due to the configuration of local elements. That is, the local elements of the structure are not salient in isolation, but instead the arrangement of the elements is what makes the structure salient. The saliency of each element is estimated by accumulating *votes* cast from its neighbors. Tensor voting is a pairwise operation in which elements cast and collect votes in local neighborhoods. Each vote is a tensor and encodes the orientation the receiver would have according to the voter, if the voter and receiver belonged in the same structure. According to the Gestalt theory, simple figures are preferable to complex alternatives, if no evidence favors the latter. The evidence in tensor voting are the position and preferred orientation, if available, of the voter and the position of the receiver. Given this information, we show examples of votes cast by an oriented voter in 2-D in Figure 1. The voter V is a point on a horizontal curve and its normal is represented by the orange arrow. (Details on the representation can be found in Section 3.1). The other points, R_1-R_4, act as receivers. The simplest curve, the one with minimum total curvature, that passes through an oriented voter and

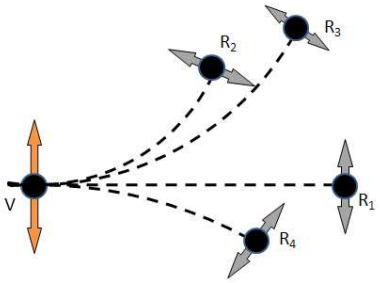
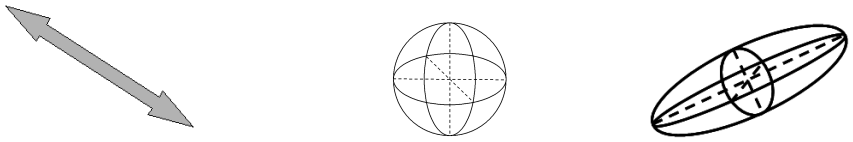


Figure 1: Illustration of tensor voting in 2-D. The voter is an oriented curve element V on a horizontal curve whose normal is represented by the orange arrow. The four receivers R_1-R_4 collect votes from V. (In practice, they would also cast votes to V and among themselves, but this is omitted here.) Each receiver is connected to V by a circular arc which is the simplest structure that can be inferred from two points, one of which is oriented. The gray votes at the receivers indicate the curve normal the receivers should have according to the voter.

a receiver is a circular arc, for which curvature is constant. Therefore, we connect the voter and receiver by a circular arc which is tangent at the voter and passes through the receiver and define the vote cast as the normal to this arc at the location of the receiver. The votes shown as gray arrows in Figure 1 represent the orientations the receivers would have according to the voter V. The magnitude of the votes decays with distance and curvature. It will be defined formally in Section 3.2. Voting from all possible types of voters, such as surface or curve elements in 3-D, can be derived from the fundamental case of a curve element voter in 2-D (Medioni et al., 2000). Tensor voting is based on strictly local computations in the neighborhoods of the inputs. The size of these neighborhoods is controlled by the only critical parameter in the framework: the scale of voting σ. The scale parameter is introduced in Eq. 3. By determining the size of the neighborhoods, scale regulates the amount of smoothness and provides a knob to the user for balancing fidelity to the data and noise reduction.

Regardless of the computational feasibility of an implementation, the same grouping principles apply to spaces with even higher dimensions. For instance, Tang et al. (2001) observed that pixel correspondences can be viewed as points in the 8-D space of free parameters of the fundamental matrix. Correct correspondences align to form a hyperplane in that space, while wrong correspondences are randomly distributed. By applying tensor voting in 8-D, Tang et al. were able to infer the dominant hyperplane and the desired parameters of the fundamental matrix. Storage and computation requirements, however, soon become prohibitively high as the dimensionality of the space increases. Even though the applicability of tensor voting as a manifold learning technique seems to have merit, the generalization of the implementation of Medioni et al. (2000) is not practical, mostly due to computational complexity and storage requirements in N dimensions. The bottleneck is the generation and storage of voting fields, the number of which is equal to the dimensionality of the space.

In this section, we describe the tensor voting framework beginning with data representation and proceeding to the voting mechanism and vote analysis. The representation and vote analysis schemes are N-D extensions of the original implementation (Medioni et al., 2000). The novelty of



(a) Oriented or *stick* tensor (b) Unoriented or *ball* tensor (c) Generic tensor

Figure 2: Examples of tensors in 3-D. The tensor on the left has only one non-zero eigenvalue and encodes a preference for an orientation parallel to the eigenvector corresponding to that eigenvalue. The eigenvalues of the tensor in the middle are all equal, and thus the tensor does not encode a preference for a particular orientation. The tensor on the right is a generic 3-D tensor.

our work is a new formulation of the voting process that is practical for spaces of dimensionality up to a few hundreds. Efficiency is considerably higher than the preliminary version of this formulation presented in Mordohai and Medioni (2005), where we focused on dimensionality estimation.

3.1 Data Representation

The representation of a token (a generic data point) is a second order, symmetric, non-negative definite tensor, which is equivalent to an $N \times N$ matrix and an ellipsoid in an N-D space. All tensors in this paper are second order, symmetric and non-negative definite, so any reference to a tensor automatically implies these properties. Three examples of tensors, in 3-D, can be seen in Figure 2. A tensor represents the structure of a manifold going through the point by encoding the normals to the manifold as eigenvectors of the tensor that correspond to non-zero eigenvalues, and the tangents as eigenvectors that correspond to zero eigenvalues. (Note that eigenvectors and vectors in general in this paper are column vectors.) For example, a point in an N-D hyperplane has one normal and $N-1$ tangents, and thus is represented by a tensor with one non-zero eigenvalue associated with an eigenvector parallel to the hyperplane's normal. The remaining $N-1$ eigenvalues are zero. A point belonging to a 2-D manifold in N-D is represented by two tangents and $N-2$ normals, and thus is represented by a tensor with two zero eigenvalues associated with the eigenvectors that span the tangent space of the manifold. The tensor also has $N-2$ non-zero, equal eigenvalues whose corresponding eigenvectors span the manifold's normal space. Two special cases of tensors are: the *stick* tensor that has only one non-zero eigenvalue and represents perfect certainty for a hyperplane normal to the eigenvector that corresponds to the non-zero eigenvalue; and the *ball* tensor that has all eigenvalues equal and non-zero which represents perfect uncertainty in orientation, or, in other words, just the presence of an unoriented point.

The tensors can be formed by the summation of the direct products ($\vec{n}\vec{n}^T$) of the eigenvectors that span the normal space of the manifold. The tensor at a point on a manifold of dimensionality d, with $\vec{n}_i$ being the unit vectors that span the normal space, can be computed as follows:

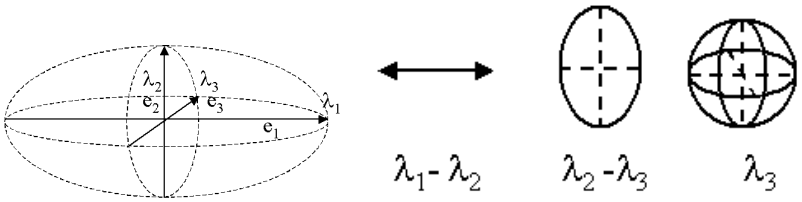
$$T = \sum_{i=1}^{d} \vec{n}_i \vec{n}_i^T . \tag{1}$$

An unoriented point can be represented by a *ball* tensor which contains all possible normals and is encoded as the $N \times N$ identity matrix. Any point on a manifold of known dimensionality and orientation can be encoded in this representation by appropriately constructed tensors, according to Eq. 1.

On the other hand, given an N-D second order, symmetric, non-negative definite tensor, the type of structure encoded in it can be inferred by examining its eigensystem. Any such tensor can be decomposed as in the following equation:

$$\begin{aligned}\mathbf{T} &= \sum_{d=1}^{N} \lambda_d \hat{e}_d \hat{e}_d^T = \\ &= (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) + ... + \lambda_N(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + ... + \hat{e}_N\hat{e}_N^T) \\ &= \sum_{d=1}^{N-1} [(\lambda_d - \lambda_{d+1}) \sum_{k=1}^{d} \hat{e}_d\hat{e}_d^T] + \lambda_N(\hat{e}_1\hat{e}_1^T + ... + \hat{e}_N\hat{e}_N^T)\end{aligned} \tag{2}$$

where λ_d are the eigenvalues in descending order of magnitude and $\hat{e}_d$ are the corresponding eigenvectors. The tensor simultaneously encodes *all* possible types of structure. The confidence, or *saliency* in perceptual organization terms, of the type that has d normals is encoded in the difference $\lambda_d - \lambda_{d+1}$, or λ_N for the ball tensor. If only one of these eigenvalue differences is not zero, then the tensor encodes a single type of structure. Otherwise, more than one type can be present at the location of the tensor, each having a saliency value given by the appropriate difference between consecutive eigenvalues of λ_N. An illustration of tensor decomposition in 3-D can be seen in Figure 3.



(a) Generic 3-D tensor (b) Elementary 3-D tensors

Figure 3: Tensor decomposition in 3-D. A generic tensor can be decomposed into the stick, plate and ball components that have a normal subspace of rank one, two and three respectively.

3.2 The Voting Process

After the inputs have been encoded with tensors, the information they contain is propagated to their neighbors via a voting operation. Given a tensor at A and a tensor at B, the vote the token at A (the *voter*) casts to B (the *receiver*) has the orientation the receiver would have, if both the voter and receiver belong to the same structure. The magnitude of the vote is a function of the confidence we have that the voter and receiver indeed belong to the same structure.

3.2.1 STICK VOTING

We first examine the case of a voter associated with a *stick* tensor, that is the normal space is a single vector in N-D. We claim that, in the absence of other information, the arc of the *osculating circle* (the circle that shares the same normal as a curve at the given point) at A that goes through B is the most likely smooth path between A and B, since it minimizes total curvature. The center of the circle is denoted by C in Figure 4(a). (For visualization purposes, the illustrations are for the 2-D and 3-D cases.) In case of straight continuation from A to B, the osculating circle degenerates to a straight line. Similar use of circular arcs can also be found in Parent and Zucker (1989), Saund (1992), Sarkar and Boyer (1994) and Yen and Finkel (1998). The vote is also a stick tensor and is generated as described in Section 3 according to the following equation:

$$\begin{aligned}
\mathbf{S}_{vote}(s,\theta,\kappa) &= e^{-(\frac{s^2+c\kappa^2}{\sigma^2})} \begin{bmatrix} -\sin(2\theta) \\ \cos(2\theta) \end{bmatrix} [-\sin(2\theta) \;\; \cos(2\theta)], \qquad (3)\\
\theta &= \arcsin(\frac{\vec{v}^T \hat{e}_1}{\|\vec{v}\|}),\\
s &= \frac{\theta\|\vec{v}\|}{\sin(\theta)},\\
\kappa &= \frac{2\sin(\theta)}{\|\vec{v}\|}.
\end{aligned}$$

In the above equation, s is the length of the arc between the voter and receiver, and κ is its curvature (which can be computed from the radius AC of the osculating circle in Figure 4(a)), σ is the scale of voting, and c is a constant, which controls the degree of decay with curvature. The constant c is a function of the scale and is optimized to make the extension of two orthogonal line segments to from a right angle equally likely to the completion of the contour with a rounded corner (Guy and Medioni, 1996). Its value is given by: $c = \frac{-16log(0.1)\times(\sigma-1)}{\pi^2}$. The scale σ essentially controls the range within which tokens can influence other tokens. It can also be viewed as a measure of smoothness that regulates the inevitable trade-off between over-smoothing and over-fitting. Small values preserve details better, but are more vulnerable to noise and over-fitting. Large values produce smoother approximations that are more robust to noise. As shown in Section 7, the results are very stable with respect to the scale. Note that σ is the only free parameter in the framework.

The vote as defined above is on the plane defined by A, B and the normal at A. Regardless of the dimensionality of the space, stick vote generation always takes place in a 2-D subspace defined by the position of the voter and the receiver and the orientation of the voter. (This explains why Eq. 3 is defined in a 2-D space.) Stick vote computation is identical in any space between 2 and N dimensions. After the vote has been computed, it has to be transformed to the N-D space and aligned to the voter by a rotation and translation. For simplicity, we also use the notation:

$$\mathbf{S}_{vote}(A,B,\vec{n}) = \mathbf{S}(s(A,B,\vec{n}),\theta(A,B,\vec{n}),\kappa(A,B,\vec{n})) \qquad (4)$$

to denote the stick vote from A to B with $\vec{n}$ being the normal at A. $s(A,B,\vec{n})$, $\theta(A,B,\vec{n})$ and $\kappa(A,B,\vec{n})$ are the resulting values of the parameters of Eq. 3 given A,B and $\vec{n}$.

According to the Gestalt principles we wish to enforce, the magnitude of the vote should be a function of proximity and smooth continuation. Thus the influence from a point to another attenuates with distance, to minimize interference from unrelated points; and curvature, to favor straight

continuation over curved alternatives. Moreover, no votes are cast if the receiver is at an angle larger than 45° with respect to the tangent of the osculating circle at the voter. Similar restrictions on regions of influence also appear in Heitger and von der Heydt (1993), Yen and Finkel (1998) and Li (1998) to prevent high-curvature connections without support from the data. Votes corresponding to such connections would have been very weak regardless of the restriction since their magnitude is attenuated due to curvature. The saliency decay function (Gaussian) of Eq. 3 has infinite support, but for practical purposes the field is truncated so that negligible votes do not have to be computed. For all experiments shown here, we limited voting neighborhoods to the extent in which the magnitude of a vote is more than 3% of the magnitude of the voter. Both truncation beyond 45° and truncation beyond a certain distance are not critical choices, but are made to eliminate the computation of insignificant votes.

3.2.2 N-D FORMULATION OF TENSOR VOTING

We have shown that stick vote computation is identical up to a simple transformation from 2-D to N-D. Now we turn our attention to votes generated by voters that are not stick tensors. In the original formulation (Medioni et al., 2000) these votes can be computed by integrating the votes cast by a rotating stick tensor that spans the normal space of the voting tensor. Since the resulting integral has no closed form solution, the integration is approximated numerically by taking sample positions of the rotating stick tensor and adding the votes it generates at each point within the voting neighborhood. As a result, votes that cover the voting neighborhood are pre-computed and stored in voting fields. The advantage of this scheme is that all votes are generated based on the stick voting field. Its computational complexity, however, makes its application in high-dimensional spaces prohibitive. Voting fields are used as look-up tables to retrieve votes via interpolation between the pre-computed samples. For instance, a voting field in 10-D with k samples per axis, requires storage for k^{10} 10×10 tensors, which need to be computed via numerical integration over 10 variables. Thus, the use of pre-computed voting fields becomes impractical as dimensionality grows. At the same time, the probability of using a pre-computed vote decreases.

Here, we present a simplified vote generation scheme that allows the direct computation of votes from arbitrary tensors in arbitrary dimensions. Storage requirements are limited to storing the tensors at each sample, since explicit voting fields are not used any more. The advantage of the novel vote generation scheme is that it does not require integration. As in the original formulation, the eigenstructure of the vote represents the normal and tangent spaces that the receiver would have, if the voter and receiver belong in the same smooth structure.

3.2.3 BALL VOTING

For the generation of ball votes, we propose the following direct computation. It is based on the observation that the vote generated by a ball voter propagates the voter's preference for a straight line that connects it to the receiver (Figure 4(b)). The straight line is the simplest and smoothest continuation from a point to another point in the absence of other information. Thus, the vote generated by a ball voter is a tensor that spans the $(N-1)$-D normal space of the line and has one zero eigenvalue associated with the eigenvector that is parallel to the line. Its magnitude is a function of only the distance between the two points, since curvature is zero. Taking these observations into account, the ball vote can be constructed by subtracting the direct product of the unit vector in the direction from the voter to the receiver from a full rank tensor with equal eigenvalues (i.e., the

identity matrix). The resulting tensor is attenuated by the same Gaussian weight according to the distance between the voter and the receiver.

$$\mathbf{B}_{vote}(s) = e^{-\left(\frac{s^2}{\sigma^2}\right)} \left(I - \frac{\vec{v}\vec{v}^T}{\|\vec{v}^T\vec{v}\|} \right) \tag{5}$$

where $\vec{v}$ is a unit vector parallel to the line connecting the voter and the receiver and I is the N-D identity matrix. In this case, $s = |\vec{v}|$ and we omit θ and κ since they do not affect the computation.

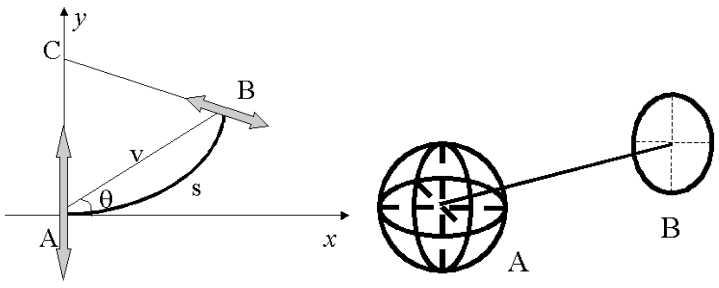
Along the lines of Equation 4, we define a simpler notation:

$$\mathbf{B}_{vote}(A,B) = \mathbf{B}_{vote}(s(A,B)) \tag{6}$$

where $s(A,B) = |\vec{v}|$.

3.2.4 VOTING BY ELEMENTARY TENSORS

To complete the description of vote generation, we need to describe the case of a tensor that has d equal eigenvalues, where d is not equal to 1 or N. (An example of such a tensor would be a curve element in 3-D, which has a rank-two normal subspace and a rank-one tangent subspace.) The description in this section also applies to ball and stick tensors, but we use the above direct computations, which are faster. Let $\vec{v}$ be the vector connecting the voting and receiving points. It can be decomposed into $\vec{v}_t$ and $\vec{v}_n$ in the tangent and normal spaces of the voter respectively. The new vote generation process is based on the observation that curvature in Eq. 3 is not a factor when θ is zero, or, in other words, if the voting stick is orthogonal to $\vec{v}_n$. We can exploit this by defining a new basis for the normal space of the voter that includes $\vec{v}_n$. The new basis is computed using the Gramm-Schmidt procedure. The vote is then constructed as the tensor addition of the votes cast by stick tensors parallel to the new basis vectors. Among those votes, only the one generated by the stick tensor parallel to $\vec{v}_n$ is not parallel to the normal space of the voter and curvature has to be considered. All other votes are a function of the length of $\vec{v}_t$ only. See Figure 5 for an illustration in 3-D. Analytically, the vote is computed as the summation of d stick votes cast by the new basis of the normal space. Let N_S denote the normal space of the voter and let $\vec{b}_i, i \in [1,d]$ be a basis for it with $\vec{b}_1$ being parallel to $\vec{v}_n$. If $S_{vote}(A,B,\vec{b})$ is the function that generates the stick vote from a



(a) Stick voting (b) Ball voting

Figure 4: Vote generation for a stick and a ball voter. The votes are functions of the position of the voter A and receiver B and the tensor of the voter.

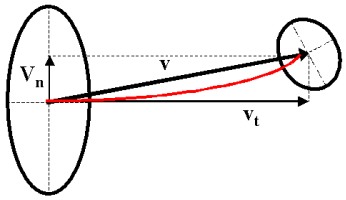


Figure 5: Vote generation for generic tensors. The voter here is a tensor with a 2-D normal subspace in 3-D. The vector connecting the voter and receiver is decomposed into $\vec{v}_n$ and $\vec{v}_t$ that lie in the normal and tangent space of the voter. A new basis that includes $\vec{v}_n$ is defined for the normal space and each basis component casts a stick vote. Only the vote generated by the orientation parallel to $\vec{v}_n$ is not parallel to the normal space. Tensor addition of the stick votes produces the combined vote.

unit stick tensor at A parallel to $\vec{b}$ to the receiver B, then the vote from a generic tensor with normal space N is given by:

$$\mathbf{V}_{vote}(A,B,\mathbf{T}_{e,d}) = S_{vote}(A,B,\vec{b}_1) + \sum_{i\in[2,d]} S_{vote}(A,B,\vec{b}_i). \tag{7}$$

In the above equation, $\mathbf{T}_{e,d}$ denotes the elementary voting tensor with d equal non-zero eigenvalues. On the right-hand side, all the terms are pure stick tensors parallel to the voters, except the first one which is affected by the curvature of the path connecting the voter and receiver. Therefore, the computation of the last $d-1$ terms is equivalent to applying the Gaussian weight to the voting sticks and adding them at the position of the receiver. Only one vote requires a full computation of orientation and magnitude. This makes the proposed scheme computationally inexpensive.

3.2.5 THE VOTING PROCESS

During the voting process (Algorithm 1), each point casts a vote to all its neighbors within the voting neighborhood. If the voters are not pure elementary tensors, that is if more than one saliency value is non-zero, they are decomposed before voting according to Eq. 2. Then, each component votes separately and the vote is weighted by $\lambda_d - \lambda_{d+1}$, except the ball component whose vote is weighted by λ_N. Besides the voting tensor $\mathbf{T}$, points also have a receiving tensor $\mathbf{R}$ that acts as vote accumulator. Votes are accumulated at each point by tensor addition, which is equivalent to matrix addition.

3.3 Vote Analysis

Vote analysis takes place after voting to determine the most likely type of structure and the orientation at each point. There are $N+1$ types of structure in an N-D space ranging from 0-D points to N-D hypervolumes.

The eigensystem of the receiving tensor $\mathbf{R}$ is computed once at the end of the voting process and the tensor is decomposed as in Eq. 2. The estimate of local intrinsic dimensionality is given by the maximum gap in the eigenvalues. Quantitative results on dimensionality estimation are presented in Section 4. In general, if the maximum eigenvalue gap is $\lambda_d - \lambda_{d+1}$, the estimated local intrinsic

Algorithm 1 The Voting Process

1. Initialization
Read M input points P_i and initial conditions, if available.
for all $i \in [1,M]$ **do**
  Initialize $\mathbf{T}_i$ according to initial conditions or set equal to the identity I.
  Compute $\mathbf{T}_i$'s eigensystem $(\lambda_d^{(i)}, \hat{e}_d^{(i)})$.
  Set vote accumulator $\mathbf{R}_i \leftarrow 0$
end for
Initialize Approximate Nearest Neighbor (ANN) k-d tree (Arya et al., 1998) for fast neighbor retrieval.

2. Voting
for all $i \in [1,M]$ **do**
  for all P_j in P_i's neighborhood **do**
    if $\lambda_1 - \lambda_2 > 0$ **then**
      Compute stick vote $\mathbf{S}_{vote}(P_i, P_j, \hat{e}_1^{(i)})$ from P_i to P_j according to Eq. 4.
    end if
    if $\lambda_N > 0$ **then**
      Compute ball vote $\mathbf{B}_{vote}(P_i, P_j)$ from P_i to P_j according to Eq. 6.
    end if
    for $d = 2$ to $N-1$ **do**
      if $\lambda_d - \lambda_{d+1} > 0$ **then**
        Compute vote $\mathbf{V}_{vote}(P_i, P_j, \mathbf{T}_{e,d}^{(i)})$ according to Eq. 7.
      end if
    end for
    Add votes to P_j's accumulator

$$\mathbf{R}_j \leftarrow \mathbf{R}_j + (\lambda_1 - \lambda_2)\mathbf{S}_{vote}(P_i, P_j, \hat{e}_1) + (\lambda_N)\mathbf{B}_{vote}(P_i, P_j) + \sum_{d \in [2, N-1]} (\lambda_d - \lambda_{d+1})\mathbf{V}_{vote}(P_i, P_j, \mathbf{T}_{e,d}^{(i)})$$

  end for
end for

3. Vote Analysis
for all $i \in [1,M]$ **do**
  Compute eigensystem of $\mathbf{R}_i$ (Eq. 2) to determine dimensionality and orientation.
  $\mathbf{T}_i \leftarrow \mathbf{R}_i$
end for

dimensionality is $N-d$, and the manifold has d normals and $N-d$ tangents. Moreover, the first d eigenvectors that correspond to the largest eigenvalues are the normals to the manifold, and the remaining eigenvectors are the tangents. Outliers can be detected since all their eigenvalues are

small and no preferred structure type emerges. This happens because they are more isolated than inliers, thus they do not receive votes that consistently support any salient structure. Our past and current research has demonstrated that tensor voting is very robust against outliers.

This vote accumulation and analysis method does not optimize any explicit objective function, especially not a global one. Dimensionality emerges from the accumulation of votes, but it is not a equal to the average, nor the median, nor the majority of the dimensionalities of the voters. For instance, the accumulation of votes from elements of two or more intersecting curves in 2-D results in a rank-two normal space at the junction. If one restricts the analysis to the estimates of orientation, tensor voting can be viewed as a method for maximizing an objective at each point. The weighted (tensor) sum of all votes received is up to a constant equivalent to the weighted mean in the space of symmetric, non-negative definite, second-order tensors. This can be thought of as the tensor that maximizes the consensus among the incoming votes. In that sense, assuming dimensionality is provided by some other process, the estimated orientation at each point is the maximum likelihood estimate given the incoming votes. It should be pointed out here, that the sum is used for all subsequent computations, since the magnitude of the eigenvalues and the of the gaps between them are measures of saliency.

In all subsequent sections, the eigensystem of the accumulator tensor is used as the voter during subsequent processing steps described in the following sections.

Figure 6: 20,000 points sampled from the "Swiss Roll" data set in 3-D.

4. Dimensionality Estimation

In this section, we present experimental results in dimensionality estimation. According to Section 3.3, the intrinsic dimensionality at each point can be found as the maximum gap in the eigenvalues of the tensor after votes from its neighboring points have been collected. All inputs consist of unoriented points since no orientation information is provided and are encoded as ball tensors.

4.1 Swiss Roll

The first experiment is on the "Swiss Roll" data set, which is frequently used in the literature and is available at `http://isomap.stanford.edu/`. It contains $20,000$ points on a 2-D manifold in 3-D (Figure 6). We sample two random variables from independent uniform distributions to generate Swiss rolls with as many as $20,000$ points, as in Tenenbaum et al. (2000), and as few as $1,250$ points. We present quantitative results for 10 different instances of $1,250$, $5,000$ and $20,000$ points in Table 1 as a function of σ. The first column for each set of results shows the percentage of points with correct dimensionality estimates, that is for which $\lambda_1 - \lambda_2$ is the maximal eigenvalue gap. The

Points	1250			5000			20000		
σ	DE	NN	Time	DE	NN	Time	DE	NN	Time
1	0.5%	1.6	0.2	4.3%	3.6	0.4	22.3%	11.2	3.8
2	4.2%	3.7	0.2	22.3%	11.1	0.8	67.9%	40.8	5.4
4	21.5%	11.0	0.2	70.0%	40.7	0.8	98.0%	156.3	12.3
8	64.5%	38.3	0.2	96.9%	148.9	2.6	99.9%	587.2	36.9
12	87.4%	83.4	0.4	99.3%	324.7	5.0	99.9%	1285.4	77.4
16	94.3%	182.4	0.7	99.5%	715.7	10.1	99.8%	2838.0	164.8
20	94.9%	302.6	1.0	99.2%	1179.4	16.5	99.5%	4700.9	268.7
30	91.0%	703.6	2.3	97.0%	2750.8	37.4	97.6%	10966.5	615.2
40	83.6%	1071.7	3.6	86.7%	4271.5	57.7	87.7%	16998.2	947.3

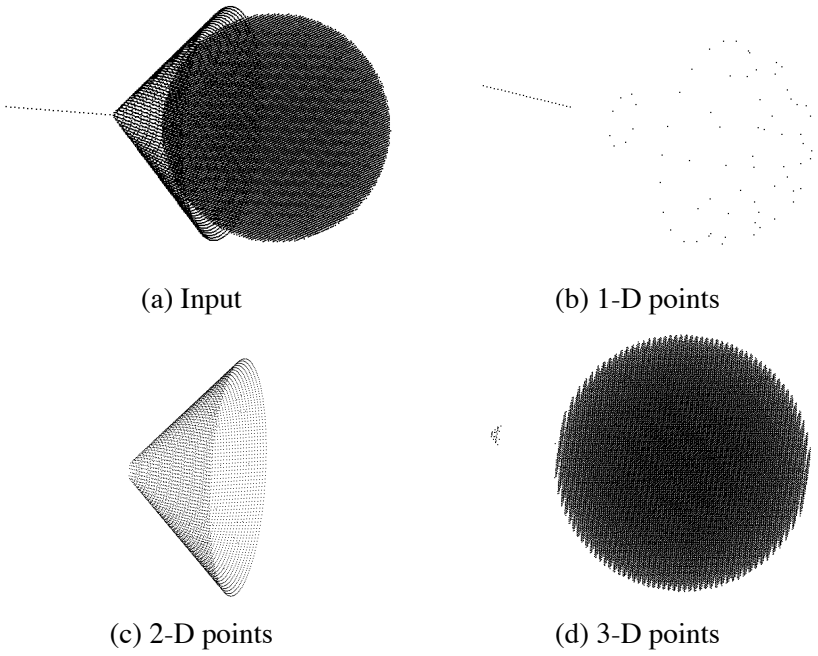
Table 1: Rate of correct dimensionality estimation (DE), average number of neighbors per point and execution times (in seconds) as functions of σ and the number of samples for the "Swiss Roll" data set. All experiments have been repeated 10 times on random samplings of the Swiss Roll function. Note that the range of scales includes extreme values as evidenced by the very high and very low numbers of neighbors in several cases.

second column shows the average number of nearest neighbors included in the voting neighborhood of each point. The third column shows processing times of a single-threaded C++ implementation running on an Intell Pentium 4 processor at $2.50GHz$. We have also repeated the experiment on 10 instances of 500 points from the Swiss Roll using the same values of the scale. Meaningful results are obtained for $\sigma > 8$. The peak of correct dimensionality estimation is at 80.3% for $\sigma = 20$.

A conclusion that can be drawn from Table 1 is that the accuracy is high and stable for a large range of values of σ, as long as a few neighbors are included in each neighborhood. The majority of neighborhoods being empty is an indication of inappropriate scale selection. Performance degrades as scale increases and the neighborhoods become too large to capture the curvature of the manifold. This robustness to large variations in parameter selection are due to the weighting of the votes according to Eqs. 3, 5 and 7 and alleviates the need for extensive parameter tuning.

4.2 Structures with Varying Dimensionality

The second data set is in 4-D and contains points sampled from three structures: a line, a 2-D cone and a 3-D hyper-sphere. The hyper-sphere is a structure with three degrees of freedom. It cannot be unfolded unless we remove a small part from it. Figure 7(a) shows the first three dimensions of the data. The data set contains a total $135,864$ points, which are encoded as ball tensors. Tensor voting is performed with $\sigma = 200$. Figures 7(b-d) show the points classified according to their dimensionality. Methods based on dimensionality reduction or methods that estimate a single dimensionality estimate for the data set would fail for this data set because of the presence of structures with different dimensionalities and because the hyper-sphere cannot be unfolded.



(a) Input (b) 1-D points

(c) 2-D points (d) 3-D points

Figure 7: Data of varying dimensionality in 4-D. (The first three dimensions of the input and the classified points are shown.) Note that the hyper-sphere is empty in 4-D, but appears as a full sphere when visualized in 3-D.

4.3 Data in High Dimensions

The data sets for this experiment were generated by sampling a few thousand points from a low-dimensional space (3- or 4-D) and mapping them to a medium dimensional space (14- to 16-D) using linear and quadratic functions. The generated points were then rotated and embedded in a 50- to 150-D space. Outliers drawn from a uniform distribution inside the bounding box of the data were added to the data set. The percentage of correct point-wise dimensionality estimates after tensor voting can be seen in Table 2.

Intrinsic Dimensionality	Linear Mappings	Quadratic Mappings	Space Dimensions	Dimensionality Estimation (%)
4	10	6	50	93.6
3	8	6	100	97.4
4	10	6	100	93.9
3	8	6	150	97.3

Table 2: Rate of correct dimensionality estimation for high dimensional data.

5. Manifold Learning

In this section we show quantitative results on estimating manifold orientation for various data sets.

Points	1250	5000	20000
σ	Orientation Error	Orientation Error	Orientation Error
1	47.2	28.1	3.1
2	28.7	3.5	**0.4**
4	4.9	**0.9**	0.5
8	**2.0**	1.3	1.1
12	2.5	2.0	1.9
16	3.5	3.1	3.0
20	5.4	4.8	4.7
30	16.9	15.0	14.9
40	28.3	26.2	25.9

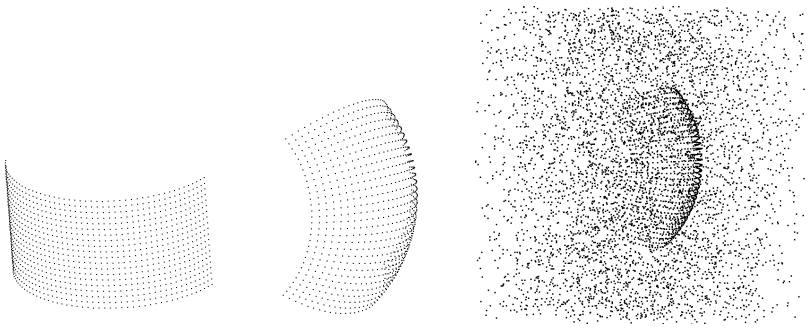
Table 3: Error (in degrees) in surface normal orientation estimation as a function of σ and the number of samples for the "Swiss Roll" data set. The error reported is the (unsigned) angle between the eigenvector corresponding to the largest eigenvalue of the estimated tensor at each point and the ground truth surface normal. See also Table 1 for processing times and the average number of points in each neighborhood.

5.1 Swiss Roll

We begin this section by completing the presentation of our experiments on the Swiss Roll data sets described in the previous section. Here we show the accuracy of normal estimation, regardless of whether the dimensionality was estimated correctly, for the experiments of Table 1. Table 3 is a complement to Table 1, which contains information on the average number of points in the voting neighborhoods and processing time that is not repeated here. The error reported is the (unsigned) angle between the eigenvector corresponding to the largest eigenvalue of the estimated tensor at each point and the ground truth surface normal. These results are over 10 different samplings for each number of points reported in the table.

For comparison, we also estimated the orientation at each point of the Swiss Roll using local PCA computed on the point's k nearest neighbors. We performed an exhaustive search over k, but only report the best results here. As for tensor voting, orientation accuracy was measured on 10 instances of each data set. The lowest errors for $1,250$, $5,000$ and $20,000$ points are 2.51°, 1.16° and 0.56° for values of k equal to 9, 10 and 13 respectively. These errors are approximately 20% larger than the lowest errors achieved by tensor voting, which are shown in bold in Table 3. It should be noted, that, unlike tensor voting, local PCA cannot be used to refine these estimates or take advantage of existing orientation estimates that may be available at the inputs.

We observe that accuracy using tensor voting is very high for a large range of scales and improves, as expected, with higher data density. Random values (around 45°) result when the neighborhood does not contain enough points for normal estimation. See Mitra et al. (2004) and Lalonde et al. (2005) for an analysis of the 3-D case based on the *Gershgorin Circle Theorem* that provides



(a) Cylinder (b) Sphere (c) Noisy sphere

Figure 8: Data sets used in Sections 5 and 6.

bounds for the eigenvalues of square matrices. The authors show how noise and curvature affect the estimation of curve and surface normals under some assumptions about the distribution of the points. The conclusions are that for large neighborhood sizes, errors caused by curvature dominate, while for small neighborhood sizes, errors due to noise dominate. (There is no noise in the data for this experiment.)

5.2 Spherical and Cylindrical Sections

Here, we present quantitative results on simple data sets in 3-D for which ground truth can be analytically computed. In Section 6, we process the same data with state of the art manifold learning algorithms and compare their results against ours. The two data sets are a section of a cylinder and a section of a sphere shown in Figure 8. The cylindrical section spans 150° and consists of 1000 points. The spherical section spans $90^\circ \times 90^\circ$ and consists of 900 points. Both are approximately uniformly sampled. The points are represented by ball tensors, assuming no information about their orientation. In the first part of the experiment, we compute local dimensionality and normal orientation as a function of scale. The results are presented in Tables 4 and 5. The results show that if the scale is not too small, dimensionality estimation is very reliable. For all scales the orientation errors are below 0.4^o.

The same experiments were performed for the spherical section in the presence of outliers. Quantitative results are shown in the following tables for a number of outliers that ranges from 900 (equal to the inliers) to 5000. The latter data set is shown in Figure 8(c). The outliers are drawn from a uniform distribution inside an extended bounding box of the data. Note that performance was evaluated only on the points that belong to the sphere and not the outliers. Larger values of the scale prove to be more robust to noise, as expected. The smallest values of the scale result in voting neighborhoods that include less than 10 points, which are insufficient. Taking this into account, performance is still good even with wrong parameter selection. Also note that one could reject the outliers by thresholding, since they have smaller eigenvalues than the inliers, and perform tensor voting again to obtain even better estimates of structure and dimensionality. Even a single pass of tensor voting, however, turns out to be very effective, especially considering that no other method can handle such a large number of outliers. Foregoing the low-dimensional embedding is a main reason that allows our method to perform well in the presence of noise, since embedding random outliers in a low-dimensional space would make their influence more detrimental. This is

σ	Average Neighbors	Orientation Error (°)	Dimensionality Estimation (%)
10	5	0.06	4
20	9	0.07	90
30	9	0.08	90
40	12	0.09	90
50	20	0.10	100
60	20	0.11	100
70	23	0.12	100
80	25	0.12	100
90	30	0.13	100
100	34	0.14	100

Table 4: Results on the cylinder data set. Shown in the first column is σ, in the second is the average number of neighbors that cast votes to each point, in the third the average error in degrees of the estimated normals, and in the fourth the accuracy of dimensionality estimation.

σ	Average Neighbors	Orientation Error (°)	Dimensionality Estimation (%)
10	5	0.20	44
20	9	0.23	65
30	11	0.24	93
40	20	0.26	94
50	21	0.27	94
60	23	0.29	94
70	26	0.31	94
80	32	0.34	94
90	36	0.36	94
100	39	0.38	97

Table 5: Results on the sphere data set. The columns are the same as in Table 4.

due to the structure imposed to them by the mapping, which makes the outliers less random, and due to the increase in their density in the low-dimensional space compared to that in the original high-dimensional space.

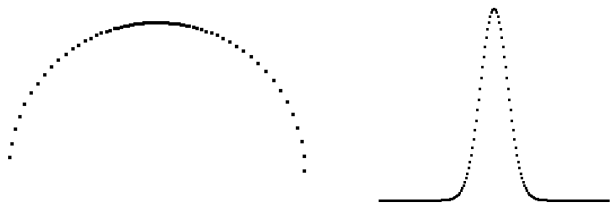
5.3 Data with Non-uniform Density

We also conducted two experiments on functions proposed in Wang et al. (2005). The key difficulty with these functions is the non-uniform density of the data. In the first example we attempt to estimate the tangent at the samples of:

$$x_i = [\cos(t_i) \;\; \sin(t_i)]^T \qquad t_i \in [0,\pi],\; t_{i+1} - t_i = 0.1(0.001 + |\cos(t_i)|) \tag{8}$$

Outliers	900		3000		5000	
σ	OE	DE	OE	DE	OE	DE
10	1.15	44	3.68	41	6.04	39
20	0.93	65	2.95	52	4.73	59
30	0.88	92	2.63	88	4.15	85
40	0.88	93	2.49	90	3.85	88
50	0.90	93	2.41	92	3.63	91
60	0.93	94	2.38	93	3.50	93
70	0.97	94	2.38	93	3.43	93
80	1.00	94	2.38	94	3.38	94
90	1.04	95	2.38	95	3.34	94
100	1.07	97	2.39	95	3.31	95

Table 6: Results on the sphere data set contaminated by noise. OE: error in normal estimation in degrees, DE: percentage of correct dimensionality estimation.



(a) Samples from Eq. 8 (b) Samples from Eq. 9

Figure 9: Input data for the two experiments proposed by Wang et al. (2005).

where the distance between consecutive samples is far from uniform. See Figure 9(a) for the inputs and the second column of Table 7 for quantitative results on tangent estimation for 152 points as a function of scale.

In the second example, which is also taken from Wang et al. (2005), points are uniformly sampled on the t-axis from the [-6, 6] interval. The output is produced by the following function:

$$x_i = [t_i \;\; 10e^{-t_i^2}]. \tag{9}$$

The points, as can be seen in Figure 9(b), are not uniformly spaced. The quantitative results on tangent estimation accuracy for 180 and 360 samples from the same interval are reported in the last two columns of Table 7. Naturally, as the sampling becomes denser, the quality of the approximation improves. What should be emphasized here is the stability of the results as a function of σ. Even with as few as 5 or 6 neighbors included in the voting neighborhood, the tangent at each point is estimated quite accurately.

σ	Eq. 8 152 points	Eq. 9 180 points	Eq. 9 360 points
10	0.60	4.52	2.45
20	0.32	3.37	1.89
30	0.36	2.92	1.61
40	0.40	2.68	1.43
50	0.44	2.48	1.22
60	0.48	2.48	1.08
70	0.51	2.18	0.95
80	0.54	2.18	0.83
90	0.58	2.02	0.68
100	0.61	2.03	0.57

Table 7: Error in degrees for tangent estimation for the functions of Eq. 8 and Eq. 9.

6. Geodesic Distances and Nonlinear Interpolation

In this section, we present an algorithm that can interpolate, and thus produce new points, on the manifold and is also able to evaluate geodesic distances between points. Both of these capabilities are useful tools for many applications. The key concept is that the intrinsic distance between any two points on a manifold can be approximated by taking small steps on the manifold, collecting votes, estimating the local tangent space and advancing on it until the destination is reached. Such processes have been reported in Mordohai (2005), Dollár et al. (2007a) and Dollár et al. (2007b).

Processing begins by learning the manifold structure, as in the previous section, usually starting from unoriented points that are represented by ball tensors. Then, we select a starting point that has to be on the manifold and a target point or a desired direction from the starting point. At each step, we can project the desired direction on the tangent space of the current point and create a new point at a small distance. The tangent space of the new point is computed by collecting votes from the neighboring points, as in regular tensor voting. Note that the tensors used here are no longer balls, but the ones resulting from the previous pass of tensor voting, according to Algorithm 1, step 3. The desired direction is then projected on the tangent space of the new point and so forth until the destination is reached. The process is illustrated in Figure 10, where we start from point A and wish to reach B. We project $\vec{t}$, the vector from A to B, on the estimated tangent space of A and obtain its projection $\vec{p}$. Then, we take a small step along $\vec{p}$ to point A_1, on which we collect votes to obtain an estimate of its tangent space. The desired direction is then projected on the tangent space of each new point until the destination is reached within ε. The geodesic distance between A and B is approximated by measuring the length of the path. In the process, we have also generated a number of new points on the manifold, which may be a desirable by-product for some applications.

There are certain configurations that cannot be handled by the algorithm described above without additional precautions. One such configuration is when the source or destination point or both are in deep concavities which attract the desired direction if the step from A to A_1 is not large enough to move the path outside the concave region. A multi-scale implementation of the above scheme can overcome this problem. A few intermediate points can be marked using a large value for the step and then used as intermediate destinations with a finer step size. Under this scheme, the path

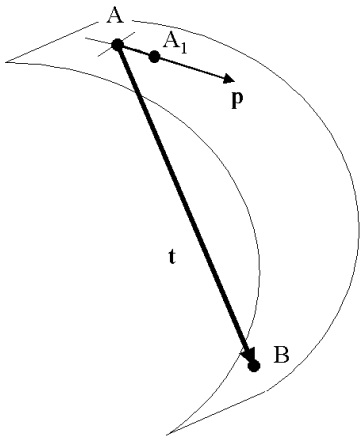


Figure 10: Nonlinear interpolation on the tangent space of a manifold.

converges to the destination and the geodesic distance is approximated accurately using a small step size. A second failure mode of the simple algorithm is for cases where the desired direction may vanish. This may occur in a manifold such as the "Swiss Roll" (Figure 6) if the destination lies on the normal space of the current point. Adding memory or inertia to the system when the desired direction vanishes, effectively addresses this situation. It should be noted that our algorithm does not handle holes and boundaries properly at its current stage of development.

6.1 Comparison with State-of-the-Art algorithms

The first experiment on manifold distance estimation is a quantitative evaluation against some of the most widely used algorithms of the literature. For the results reported in Table 8, we learn the local structure of the cylinder and sphere manifolds of the previous section using tensor voting. We also compute embeddings using LLE (Roweis and Saul, 2000), Isomap (Tenenbaum et al., 2000), Laplacian eigenmaps (Belkin and Niyogi, 2003), HLLE (Donoho and Grimes, 2003) and SDE (Weinberger and Saul, 2004). Matlab implementations for these methods can be downloaded from the following internet locations.

- LLE from `http://www.cs.toronto.edu/~roweis/lle/code.html`
- Isomap from `http://isomap.stanford.edu/`
- Laplacian Eigenmaps from `http://people.cs.uchicago.edu/~misha/ManifoldLearning/index.html`
- HLLE from `http://basis.stanford.edu/HLLE` and
- SDE from `http://www.seas.upenn.edu/~kilianw/sde/download.htm`.

We are grateful to the authors for making the code for their methods available to the community. We have also made our software publicly available at:
`http://iris.usc.edu/~medioni/download/download.htm`.

The experiment is performed as follows. We randomly select 5000 pairs of points on each manifold and attempt to measure the geodesic distance between the points of each pair in the input

space using tensor voting and in the embedding space using the other five methods. The estimated distances are compared to the ground truth: $r\Delta\theta$ for the sphere and $\sqrt{(r\Delta\theta)^2+(\Delta z)^2}$ for the cylinder. Among the above approaches, only Isomap and SDE produce isometric embeddings, and only Isomap preserves the absolute distances between the input and the embedding space. To make the evaluation fair, we compute a uniform scale that minimizes the error between the computed distances and the ground truth for all methods, except Isomap for which it is not necessary. Thus, perfect distance ratios would be awarded a perfect rating in the evaluation, even if the absolute magnitudes of the distances are meaningless in the embedding space. For all the algorithms, we tried a wide range for the number of neighbors, K. In some cases, we were not able to produce good embeddings of the data for any value of K. This occurred more frequently for the cylinder, probably due to its data density not being perfectly uniform. Errors above 20% indicate very poor performance, which is also confirmed by visual inspection of the embeddings.

Even though among the other approaches only Isomap and SDE produce isometric embeddings, while the rest produce embeddings that only preserve local structure, we think that the evaluation of the quality of manifold learning based on the computation of pairwise distances is a fair measure for the performance of all algorithms, since high quality manifold learning should minimize distortions. The distances on which we evaluate the different algorithms are both large and small, with the latter measuring the presence of local distortions. Quantitative results, in the form of the average absolute difference between the estimated and the ground truth distances as a percentage of the latter, are presented in Tables 8-10, along with the parameter that achieves the best performance for each method. In the case of tensor voting, the same scale was used for both learning the manifold and computing distances.

We also apply our method in the presence of 900, 3000 and 5000 outliers, while the inliers for the sphere and the cylinder data sets are 900 and 1000 respectively. The outliers are generated according to a uniform distribution. The error rates using tensor voting for the sphere are 0.39%, 0.47% and 0.53% respectively. The rates for the cylinder are 0.77%, 1.17% and 1.22%. Compared with the noise free case, these results demonstrate that our approach degrades slowly in the presence of outliers. The best performance achieved by any other method is 3.54% on the sphere data set with 900 outliers by Isomap. Complete results are shown in Table 9. In many cases, we were unable to achieve useful embeddings for data sets with outliers. We were not able to perform this experiment

Data Set	Sphere		Cylinder	
	K	Err(%)	K	Err(%)
LLE	18	5.08	6	26.52
Isomap	6	1.98	30	0.35
Laplacian Eigenmaps	16	11.03	10	29.36
HLLE	12	3.89	40	26.81
SDE	2	5.14	6	25.57
TV (σ)	60	0.34	50	0.62

Table 8: Error rates in distance measurement between pairs of points on the manifolds. The best result of each method is reported along with the number of neighbors used for the embedding (K), or the scale σ in the case of tensor voting (TV).

Data Set	Sphere 900	outliers	Cylinder 900	outliers
	K	Err(%)	K	Err(%)
LLE	40	60.74	6	15.40
Isomap	18	3.54	14	11.41
Laplacian Eigenmaps	6	13.97	14	27.98
HLLE	30	8.73	30	23.67
SDE		N/A		N/A
TV (σ)	70	0.39	100	0.77

Table 9: Error rates in distance measurement between pairs of points on the manifolds under outlier corruption. The best result of each method is reported along with the number of neighbors used for the embedding (K), or the scale σ in the case of tensor voting (TV). Note that HLLE fails to compute an embedding for small values of K, while SDE fails at both examples for all choices of K.

Data Set	σ	Error rate
Sphere (3000 outliers)	80	0.47
Sphere (5000 outliers)	100	0.53
Cylinder (3000 outliers)	100	1.17
Cylinder (5000 outliers)	100	1.22

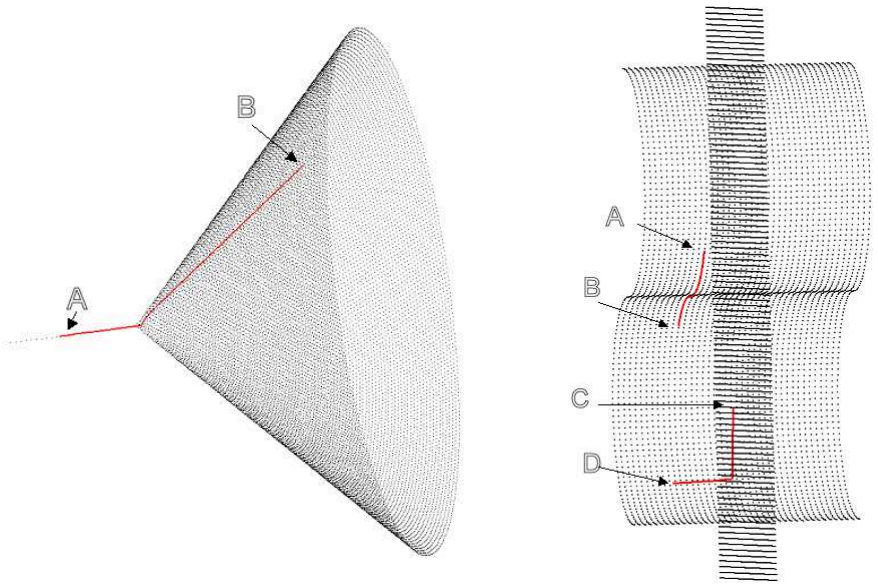
Table 10: Error rates for our approach for the experiment of Section 6.1 in the presence of 3000 and 5000 outliers.

in the presence of more than 3000 outliers with any graph-based method, probably because the graph structure is severely corrupted by the outliers.

6.2 Data Sets with Varying Dimensionality and Intersections

For the final experiment of this section, we create synthetic data in 3-D that were embedded in higher dimensions. The first data set consists of a line and a cone. The points are embedded in 50-D by three orthonormal 50-D vectors and initialized as ball tensors. Tensor voting is performed in the 50-D space and a path from point A on the line to point B on the cone is interpolated as in the previous experiment, making sure that it belongs to the local tangent space, which changes dimensionality from one to two. The data is re-projected back to 3-D for visualization in Figure 11(a).

In the second part of the experiment, we generate an intersecting S-shaped surface and a plane (a total of 11,000 points) and 30,000 outliers from a uniform distribution, and embed them in a 30-D space. Without explicitly removing the noise, we interpolate between two points on the S (A and B) and a point on the S and a point on the plane (C and D) and create the paths shown in Figure 11(b) re-projected in 3-D. The first path is curved, while the second jumps from manifold to manifold without deviating from the optimal path. (The outliers are not shown for clarity.) Processing time for 41,000



(a) Line and cone (b) S and plane

Figure 11: Nonlinear interpolation in 50-D with varying dimensionality (a) and 30-D with intersecting manifolds under noise corruption (b).

points in 30-D is 2 min. and 40 sec. on a Pentium 4 at $2.8GHz$ using voting neighborhoods that included an average of 44 points.

7. Generation of Unobserved Samples and Nonparametric Function Approximation

In this section, we build upon the results of the previous section to address function approximation. A common practice is to treat functions with multiple outputs as multiple single-output functions. We adopt this scheme here, even though nothing prohibits us from directly approximating multiple-input multiple-output functions. We assume that observations in N-D that include values for the input and output variables are available for training. The difference with the examples of the previous sections is that the queries are given as input vectors with unknown output values, and thus are of lower dimensionality than the voting space. The required module to convert this problem to that of Section 6 is one that can find a point on the manifold that corresponds to an input similar to the query. Then, in order to predict the output y of the function for an unknown input $\vec{x}$, under the assumption of local smoothness, we move on the manifold formed by the training samples until we reach the point corresponding to the given input coordinates. To ensure that we always remain on the manifold, we need to start from a point on it and proceed as in the previous section.

One way to find a suitable starting point is to find the nearest neighbor of $\vec{x}$ in the input space, which has fewer dimensions than the joint input-output (voting) space. Then, we can compute the desired direction in the low dimensional space and project it to the input-output space. If many outputs are possible for a given input (if the data have not been generated by a function in the strict sense), we have to either find neighbors at each branch of the function and produce multiple

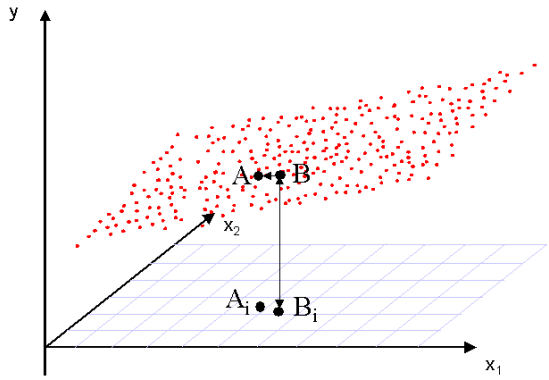


Figure 12: Interpolation to obtain output value for unknown input point A_i. B_i is the nearest neighbor in the input space and corresponds to B in the joint input-output space. We can march from B on the manifold to arrive at the desired solution A that projects on A_i in the input space.

outputs, or use other information, such as the previous state of the system, to pursue only one of the alternatives. One could find multiple nearest neighbors, run the proposed algorithm starting from each of them and produce a multi-valued answer with a probability associated with each potential output value.

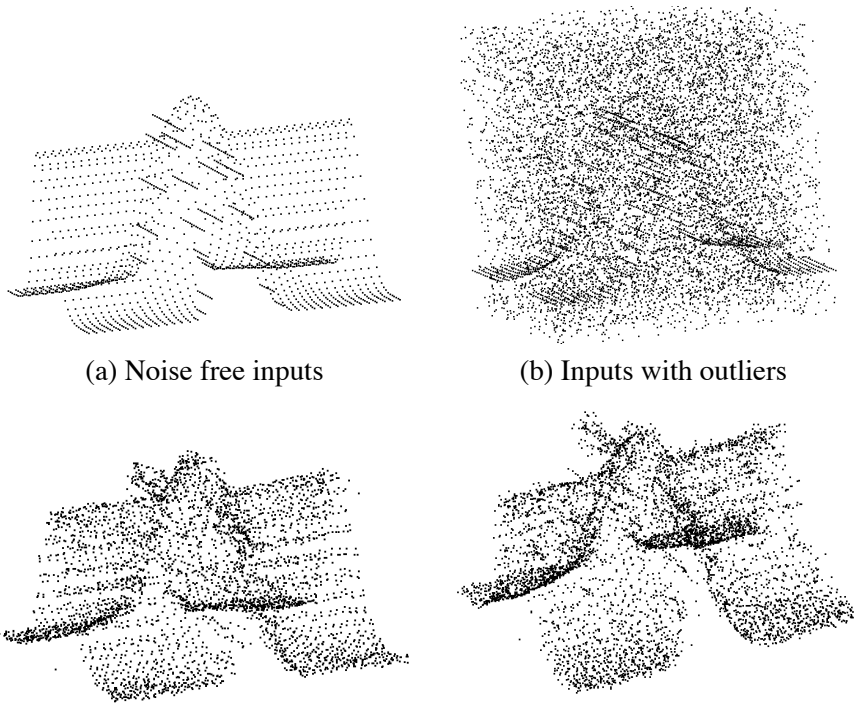
Figure 12 provides a simple illustration. We begin with a point A_i in the input space. We proceed by finding its nearest neighbor among the projections of the training data on the input space B_i. (Even if B_i is not the nearest neighbor the scheme still works but possibly requires more steps.) The sample B in the input-output space that corresponds to B_i is the starting point on the manifold. The desired direction is the projection of the A_iB_i vector on the tangent space of B. Now, we are in the case described in Section 6, where the starting point and the desired direction are known. Processing stops when the input coordinates of the point on the path from B are within ε of A_i. The corresponding point A in the input-output space is the desired interpolated sample.

As in all the experiments presented in this paper, the input points are encoded as ball tensors, since we assume that we have no knowledge of their orientation. We first attempt to approximate the following function, proposed by Schaal and Atkeson (1998):

$$y = max\{e^{-10x_1^2} \;\; e^{-50x_2^2} \;\; 1.25e^{-5(x_1^2+x_2^2)}\}. \tag{10}$$

1681 samples of y are generated by uniformly sampling the $[-1,1]\times[-1,1]$ square. We perform four experiments with increasing degree of difficulty. In all cases, after voting on the given inputs, we generate new samples by interpolating between the input points. The four configurations and noise conditions were:

- In the first experiment, we performed all operations with noise free data in 3-D.
- For the second experiment, we added 8405 outliers (five times more than the inliers) drawn from a uniform distribution in a $2\times2\times2$ cube containing the data.
- For the third experiment, we added Gaussian noise with variance 0.01 to the coordinates of all points while adding the same number of outliers as above.



(a) Noise free inputs

(b) Inputs with outliers

(c) Interpolated points with outliers and perturbation

(d) Interpolated points with outliers and perturbation in 60-D

Figure 13: Inputs and interpolated points for Eq. 10. The top row shows the noise-free inputs and the noisy input set where only 20% of the points are inliers. The bottom row shows the points generated in 3-D and 60-D respectively. In both cases the inputs were contaminated with outliers and Gaussian noise.

- Finally, we embedded the perturbed data (and the outliers) in a 60-D space, before voting and nonlinear interpolation.

The noise-free and noisy input, as well as the generated points can be seen in Figure 13. We computed the mean square error between the outputs generated by our method and Eq. 10 normalized by the variance of the noise-free data. The NMSE for all cases is reported in Table 11. Robustness against outliers is due to the fact that the inliers form a consistent surface and thus receive votes that support the correct local structure from other inliers. Outliers, on the other hand, are random and do not form any structure. They cast and receive inconsistent votes and therefore neither develop a preference for a certain manifold nor significantly disrupt the structure estimates at the inliers. They can be removed by simple thresholding since all their eigenvalues are small and almost equal, but this is not done here. Note that performance in 60-D is actually better since the interference by outliers is reduced as the dimensionality of the space increases. Tensor voting is also robust against perturbation of the coordinates as long as it is not biased to favor a certain direction. If the perturbation is zero-mean, its effects on individual votes are essentially canceled out, because they only contribute to the ball component of the accumulated tensor at each point, causing small errors in orientation estimation.

Experiment	NMSE
Noise-free	0.0041
Outliers	0.0170
Outliers & N(0, 0.01)	0.0349
Outliers & N(0, 0.01) in 60-D	0.0241

Table 11: Normalized MSE for the interpolated points of Eq. 10 under different noise conditions.

7.1 Results on Real Data

The final experiments are on real data taken from the University of California at Irvine Machine Learning Repository (Newman et al., 1998) available online at `http://www.ics.uci.edu/~mlearn/MLRepository.html` and the University of Toronto DELVE archive (`http://www.cs.toronto.edu/~delve`)(Rasmussen et al., 1996). We used the "abalone", "Boston housing" and "Computer activity" data sets. These data sets were selected because they contain data from a single class in spaces of 9, 14 and 22 dimensions respectively. In each case one variable is treated as the output and all others as inputs. Most of the input variables are continuous, but vary widely in magnitude. Since our method is based on distance relationships between the samples, we re-scaled the data so that the ratio of maximum to minimum standard deviation of the variables was approximately $10:1$, instead of the original, which for several cases exceeded $1000:1$. We split the data in training and test sets and perform tensor voting on the training data to learn the structure of the manifold. For each test sample, we begin by finding the average position of a few nearest neighbors in the $(N-1)$-D input space and then follow a path on the estimated manifold in N-D until the desired input coordinates are reached. The value of the output variable when the input variables are equal to the query is the estimate returned by our method.

In the case of the "abalone" data, we follow common practice and map the first variable from M for male, F for female and I for infant to $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ respectively. The resulting space is 12-D, with the last variable being the one we attempt to estimate. Again, we scale the data so that the variances of different variables are comparable. Following most authors, we divide the 4177 samples in training and test sets containing 3000 and 1177 samples respectively. Our results after 10 runs of the experiment with randomly selected training and test sets and a comparison with a number of other published results on the same data can be seen in Table 12. We also applied our algorithm to the "Boston housing" data set, which has been extensively used as a benchmark. It contains 506 samples of house prices as a function of 13 variables. We use training and test sets containing 481 and 25 points respectively. Due to the small size of the test set, we repeated the experiment 20 times always using as queries points that had not been included in the test set before, thus using virtually all points for queries. Error rates can be seen in Table 12. Finally, we used the "computer activity" data set from the DELVE archive (Rasmussen et al., 1996). It contains 8192 observations in a 22-D space. We used 2000 samples for training and the rest for testing.

In summary, our algorithm achieves what we think is satisfactory performance despite the fact that the data sets contain insufficient samples to describe complex manifolds. The sparseness of the data under-uses the capability of our approach to handle complex nonlinear manifolds. We observed that for certain query points there are very few similar samples in the training set. In the absence of enough samples, our algorithm may not improve the initial solution given by the nearest neighbors

of the point. These errors cause the ranking of our results in terms of RMS to be worse than in terms of mean absolute error (MAE) in Table 12.

	Abalone		Housing		Computer	
	MAE	RMS	MAE	RMS	MAE	RMS
BCM (Tresp, 2000)	-	-	-	3.100	-	-
GPR1 (Tresp, 2000)	-	-	-	3.013	-	-
RVM (Tipping, 2001)	-	-	-	8.04	-	-
SVM (Tipping, 2001)	-	-	-	7.46	-	-
Sparse GPR (Smola and Bartlett, 2001)	1.785	-	-	-	-	-
GPR2 (Smola and Bartlett, 2001)	1.782	-	-	-	-	-
Online GPR (Schwaighofer and Tresp, 2003)	-	2.111	-	-	-	-
BCM2 (Schwaighofer and Tresp, 2003)	-	2.111	-	-	-	-
Inductive SRM (Schwaighofer and Tresp, 2003)	-	2.109	-	-	-	-
Transductive SRM (Schwaighofer and Tresp, 2003)	-	2.109	-	-	-	-
SVR (Chu et al., 2004)	1.421	2.141	2.13	3.205	2.28	3.715
BSVR (Chu et al., 2004)	1.464	2.134	2.19	3.513	2.33	4.194
GPR-ARD (Chu et al., 2004)	1.493	2.134	2.01	2.884	1.686	2.362
BSVR-ARD (Chu et al., 2004)	1.454	2.119	1.86	2.645	1.687	2.408
Tensor voting	1.630	2.500	1.272	1.860	1.970	2.815

Table 12: Mean Absolute Error (MAE) and Root Mean squared Error (RMS) for benchmark data sets. Unless otherwise noted, training and testing is performed with 3000 and 1177 samples for "abalone", 481 and 25 for "Boston housing" and 2000 and 6192 for "computer activity", respectively. Results by other methods were not generated by us. BCM is the Bayesian committee machine of Tresp (2000) and GPR1 is Gaussian process regression implemented by Tresp. 400 samples are used for training for "Boston housing" for BCM and GPR1. RVM is the relevance vector machine of Tipping (2001) and SVM is a support vector machine implemented by Tipping. Sparse GPR is the algorithm of Smola and Bartlett (2001) and GPR2 is their implementation of Gaussian process regression. 4000 samples are used for training for the "abalone" data set for Sparse GPR and GPR2. Online GPR is the algorithm of Csató and Opper (2002), BCM2 is the Bayesian committee machine implemented by Schwaighofer and Tresp (2003), while inductive and transductive SRM are algorithms from the same paper. The number of training and testing samples is not provided by the authors for Online GPR, BCM2, inductive and transductive SRM. SVR is the support vector regression of Vapnik (1995), BSVR is Bayesian support vector regression of Chu et al. (2004), GPR-ARD is Gaussian process regression using the ARD Gaussian covariance function and BSVR-ARD is BSVR using the ARD function (Chu et al., 2004).

8. Discussion

We have presented an approach for dimensionality estimation, manifold learning and function approximation that offers certain advantages over the state of the art. Tensor voting may on the surface look similar to other local, instance-based learning algorithms that propagate information from point to point, but the fact that the votes are tensors and not scalars allows them to convey considerably more information. The properties of the tensor representation, which can handle the simultaneous presence of multiple orientations and structure types, allow the reliable inference of the normal and tangent space at each point. In addition, tensor voting is very robust against outliers, as demonstrated for the 2-D and 3-D case in numerous publications including Tang and Medioni (1998) and Medioni et al. (2000). This property holds in higher dimensions, where random noise is even more scattered. See for instance the results presented in Table 11.

It should also be noted that the votes attenuate with distance and curvature. This is a more intuitive formulation than using the k nearest neighbors with equal weights, since some of them may be too far, or belong to a different part of the manifold. For both tensor voting and the methods presented in Section 2, however, the distance metric in the input space has to be meaningful. Our method is less sensitive to a somewhat incorrect selection of the distance metric since all neighbors do not contribute equally. After this choice has been made, the only free parameter in our approach is σ, the scale of voting. Small values tend to preserve details better, while large values are more robust to noise. The scale can be selected automatically by randomly sampling a few points before voting and making sure that enough points are included in their voting neighborhoods. Our results show that sensitivity with respect to scale is small, as shown in Tables 1, 3, 4-6 and 7. The number of points that can be considered sufficient is a function of the dimensionality of the space, the intrinsic dimensionality of the data, as well as noise and curvature. The two latter factors have been analyzed by Mitra et al. (2004) and Lalonde et al. (2005), using the Gershgorin Circle Theorem, for the case of curves in 2-D and surfaces in 3-D under mild restrictions on data distribution. To the best of our knowledge no similar analysis has been done for manifolds with co-dimension other than one or in high-dimensional spaces. A thorough investigation of these issues is among the objectives of our future research.

Our algorithms fail when the available observations do not suffice to represent the manifold, as for instance in the face with varying pose and illumination data set of Tenenbaum et al. (2000), where 698 instances represent a manifold in 4096-D. Global methods may be more successful in such situations. The number of sufficient samples for tensor voting cannot be easily predicted from the dimensionality of the space, since it also depends on the complexity (curvature) of the underlying manifolds. (See also the discussion above.) For instance, 486 samples in 14-D turn out to be sufficient for the "Boston housing" function approximation experiment. Nevertheless, in many practical cases the challenges are the over-abundance of data and the need for efficient processing of large data sets. Tensor voting is a well suited framework for such cases, since it can efficiently process hundreds of thousands of points in spaces of up to a few hundred dimensions.

Another important advantage of tensor voting is the absence of global computations, which makes time complexity $O(NMlogM)$, where N is the dimensionality of the space and M is the number of points. This property enables us to process data sets with very large number of points. Computation time does not become impractical as the number of points grows, assuming that more points are added to the data set in such a way that the density remains constant. In this case, the number of votes cast per point remains constant and time requirements grow linearly. Com-

plexity is adversely affected by the dimensionality of the space N, since eigen-decomposition of $N \times N$ tensors has to be performed resulting in a complexity that is cubic with respect to N due to the eigensystem computations that are $O(N^3)$. For most practical purposes, however, the number of points has to be considerably larger[2] than the dimensionality of the space ($M \gg N$) to allow structure inference. The complexity for a nearest neighbor query using the ANN k-d tree (Arya et al., 1998) is $O(NlogM)$ and one query is required for each voter. Thus the total complexity is $O(NMlogM + MN^3) \approx O(NMlogM)$. Computational complexity, therefore, is reasonable with respect to the largest parameter, which for our methods to work has to be M. Table 1 shows the effect of data set and neighborhood size on processing time. Notice that time is linear with respect to the average number of points in each neighborhood, as expected. Space requirements for M $N \times N$ tensors are $O(MN^2)$.

In terms of dimensionality estimation, we are able to obtain accurate estimates at the point level. Moreover, since the dimensionality is found as the maximum gap in the eigenvalues of the tensor at each point, no thresholds are needed. Under most other approaches, the dimensionality has to be provided, or, at best, an average intrinsic dimensionality is estimated for the entire data set, as in Bruske and Sommer (1998), Brand (2003), Kégl (2003), Weinberger and Saul (2004) and Costa and Hero (2004).

The novelty of our approach regarding manifold learning is that it is not based on dimensionality reduction. Instead, we perform tasks such as geodesic distance measurement and nonlinear interpolation in the input space. Experimental results show that we can perform these tasks in the presence of outlier noise at high accuracy, even without explicitly removing the outliers from the data. This choice also broadens the range of data sets we can process. While isometric embeddings can be achieved for a certain class of manifolds, we are able to process non-flat manifolds and even non-manifolds. The last experiment of Section 6 demonstrates our ability to work with data sets of varying dimensionality or with intersecting manifolds. To the best of our knowledge, this is impossible with any other method. If dimensionality reduction is desired due to its considerable reduction in storage requirements, a dimensionality reduction method, such as Roweis and Saul (2000), Tenenbaum et al. (2000), Belkin and Niyogi (2003), Brand (2003), Donoho and Grimes (2003) and Weinberger and Saul (2006), can be used after tensor voting. The benefits of this process are in the form of noise robustness and smooth component identification, with respect to both dimensionality and orientation, via tensor voting followed by memory savings via dimensionality reduction.

We have also presented, in Section 7, a local nonparametric approach for function approximation that combines the advantages of local methods with the efficient representation and information propagation of tensor voting. Local function approximation methods are more flexible in the type of functions they can approximate, since the properties of the function are allowed to vary throughout the space. Our approach, in particular, has no parameters that have to be selected, such as the number and type of local models to be used, besides the scale of voting. Its drawback, in line with other local methods, is higher memory requirements. We have shown that we can process challenging examples from the literature under very adverse noise conditions. As shown in the example of Eq. 10, even when more than 80% of the samples are outliers and the inliers are corrupted by noise in the form of perturbation, we are still able to correctly predict unobserved outputs. We have also shown in Table 12 promising results on publicly available data sets, despite the fact that they are not

2. While in principal three points are sufficient to define a surface, 10 points are sufficient to define a 9-D manifold and so forth, one or two orders of magnitude more points are required for practical applications in our experience.

well suited for our approach, since the number of observations they contain is hardly sufficient to define the manifold.

As mentioned above, an issue we do not fully address here is that of the selection of an appropriate distance metric. We assume that the Euclidean distance in the input coordinate system is a meaningful distance metric. This is not the case if the coordinates are not of the same type. On the other hand, a metric such as the Mahalanobis distance is not necessarily appropriate in all cases, since the data typically lie in a limited part of the input space and scaling all dimensions, including the redundant ones, to achieve equal variance would be detrimental. For the experiments shown in Section 7, we apply heuristic scaling of the coordinates when necessary. We intend to develop a systematic way based on cross-validation that automatically scales the coordinates by maximizing prediction performance for the observations that have been left out.

Our future research will focus on addressing the limitations of our current algorithm and extending its capabilities. An interpolation mechanism that takes into account holes and boundaries during geodesic distance approximation should be implemented. Additionally, in the area of function approximation, the issue of approximating functions with multiple branches for the same input value, which often appear in practical applications, has to be handled more rigorously. We also intend to develop an online, incremental version of our approach, possibly including a forgetting and an updating module, that will be able to process data as they are collected, instead of requiring the entire data set to proceed. Potential applications of our work include challenging real problems, such as the study of direct and inverse kinematics where there are typically large numbers of samples in spaces of up to a few hundred dimensions. Function approximation for complex functions, for which global models would become very complicated, is another area where our methods could be effective. Finally, one can view the proposed approach as learning data from a single class, which can serve as the groundwork for an approach for pattern recognition, data mining, supervised and unsupervised classification.

Acknowledgments

This research has been supported by the National Science Foundation grant IIS 03 29247. The authors would like to thank Adit Sahasrabudhe for his help with some of the experiments.

References

S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45:891–923, 1998.

C.G. Atkeson, A.W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11(1-5):11–73, 1997.

A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

M. Brand. Nonrigid embeddings for dimensionality reduction. In *European Conference on Machine Learning*, pages 47–59, 2005.

M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15*, pages 961–968. MIT Press, Cambridge, MA, 2003.

L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):572–575, 1998.

W. Chu, S.S. Keerthi, and C.J. Ong. Bayesian support vector regression using a unified loss function. *IEEE Transactions on Neural Networks*, 15(1):29–44, 2004.

R. Collobert and S. Bengio. Svmtorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1(2):143–160, 2001.

J. Costa and A.O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Transactions on Signal Process*, 52(8):2210–2221, 2004.

T. Cox and M. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.

L. Csató and M. Opper. Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, 2003.

P. Dollár, V. Rabaud, and S. Belongie. Learning to traverse image manifolds. In *Advances in Neural Information Processing Systems 19*, pages 361–368. MIT Press, Cambridge, MA, 2007a.

P. Dollár, V. Rabaud, and S. Belongie. Non-isometric manifold learning: Analysis and an algorithm. In *International Conference on Machine Learning*, 2007b.

D. Donoho and C. Grimes. Hessian eigenmaps: new tools for nonlinear dimensionality reduction. In *Proceedings of National Academy of Science*, pages 5591–5596, 2003.

G. Guy and G. Medioni. Inferring global perceptual contours from local features. *International Journal of Computer Vision*, 20(1/2):113–133, 1996.

G. Guy and G. Medioni. Inference of surfaces, 3D curves, and junctions from sparse, noisy, 3D data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1265–1277, 1997.

X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

F. Heitger and R. von der Heydt. A computational model of neural contour processing: Figure-ground segregation and illusory contours. In *International Conference on Computer Vision*, pages 32–40, 1993.

I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

B. Kégl. Intrinsic dimension estimation using packing numbers. In *Advances in Neural Information Processing Systems 15*, pages 681–688. MIT Press, Cambridge, MA, 2003.

K. Koffka. *Principles of Gestalt Psychology*. Harcourt, Brace, New York, 1935.

W. Köhler. Physical gestalten. *W.D. Ellis (ed), A source book of Gestalt psychology (1950)*, pages 17–54, 1920.

J.-F. Lalonde, R. Unnikrishnan, N. Vandapel, and M. Hebert. Scale selection for classification of point-sampled 3-d surfaces. In *International Conference on 3-D Digital Imaging and Modeling*, 2005.

S. Lawrence, A. C. Tsoi, and A. D. Back. Function approximation with neural networks and local methods: Bias, variance and smoothness. In *Australian Conference on Neural Networks*, pages 16–21, 1996.

E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 17*, pages 777–784. MIT Press, Cambridge, MA, 2005.

Z. Li. A neural model of contour integration in the primary visual cortex. *Neural Computation*, 10: 903–940, 1998.

G. Medioni, M.S. Lee, and C.K. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier, New York, NY, 2000.

Facundo Mémoli and Guillermo Sapiro. Distance functions and geodesics on submanifolds of R^d and point clouds. *SIAM Journal of Applied Mathematics*, 65(4):1227–1260, 2005.

S. Mitaim and B. Kosko. The shape of fuzzy sets in adaptive function approximation. *IEEE Transactions on Fuzzy Systems*, 9(4):637–656, 2001.

T.M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

N.J. Mitra, A. Nguyen, and L. Guibas. Estimating surface normals in noisy point cloud data. *International Journal of Computational Geometry and Applications*, 14(4–5):261–276, 2004.

P. Mordohai. *A Perceptual Organization Approach for Figure Completion, Binocular and Multiple-View Stereo and Machine Learning using Tensor Voting*. Ph.D. Thesis, University of Southern California, 2005.

P. Mordohai and G. Medioni. Unsupervised dimensionality estimation and manifold learning in high-dimensional spaces by tensor voting. *International Joint Conference on Artificial Intelligence*, 2005.

P. Mordohai and G. Medioni. Stereo using monocular cues within the tensor voting framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):968–982, 2006.

D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

P. Parent and S.W. Zucker. Trace inference, curvature consistency, and curve detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):823–839, 1989.

T. Poggio and F. Girosi. Networks for approximation and learning. *Proc. of IEEE*, 78(9):1481–1497, 1990.

M. Raginsky and S. Lazebnik. Estimation of intrinsic dimensionality using high-rate vector quantization. In *Advances in Neural Information Processing Systems 18*, pages 1105–1112. MIT Press, Cambridge, MA, 2006.

C.E. Rasmussen, R.M. Neal, G. Hinton, D. van Camp, M. Revow, Z. Ghahramani, R. Kustra, and R. Tibshirani. The DELVE archive, 1996. URL `http://www.cs.toronto.edu/~delve`.

S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

A. Saha, C.L. Wu, and D.S. Tang. Approximation, dimension reduction, and nonconvex optimization using linear superpositions of gaussians. *IEEE Transactions on Computers*, 42(10): 1222–1233, 1993.

T. D. Sanger. A tree-structured algorithm for reducing computation in networks with separable basis functions. *Neural Computation*, 3(1):67–78, 1991.

S. Sarkar and K.L. Boyer. A computational structure for preattentive perceptual organization: Graphical enumeration and voting methods. *IEEE Transactions on Systems, Man and Cybernetics*, 24:246–267, 1994.

L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

E. Saund. Labeling of curvilinear structure across scales by token grouping. In *International Conference on Computer Vision and Pattern Recognition*, pages 257–263, 1992.

S. Schaal and C.G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

A. Schwaighofer and V. Tresp. Transductive and inductive methods for approximate gaussian process regression. In *Advances in Neural Information Processing Systems 15*, pages 953–960. MIT Press, Cambridge, MA, 2003.

F. Sha and L. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *International Conference on Machine Learning*, pages 784–791, 2005.

A. Shashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *International Conference on Computer Vision*, pages 321–327, 1988.

A.J. Smola and P.L. Bartlett. Sparse greedy gaussian process regression. In *Advances in Neural Information Processing Systems 13*, pages 619–625. MIT Press, Cambridge, MA, 2001.

R. Souvenir and R. Pless. Manifold clustering. In *International Conf on Computer Vision*, pages I: 648–653, 2005.

C.K. Tang and G. Medioni. Inference of integrated surface, curve, and junction descriptions from sparse 3d data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1206–1223, 1998.

C.K. Tang, G. Medioni, and M.S. Lee. N-dimensional tensor voting and application to epipolar geometry estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8): 829–844, 2001.

Y.W. Teh and S. Roweis. Automatic alignment of local representations. In *Advances in Neural Information Processing Systems 15*, pages 841–848. MIT Press, Cambridge, MA, 2003.

J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

J. Ting, A D'Souza, and S. Schaal. bayesian regression with input noise for high dimensional data. In *International Conference on Machine Learning*, 2006.

M.E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1(3):211–244, 2001.

V. Tresp. A bayesian committee machine. *Neural Computation*, 12(11):2719–2741, 2000.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, Berlin Heidelberg New York, 1995.

S. Vijayakumar and S. Schaal. Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space. In *International Conference on Machine Learning*, pages I:288–293, 2000.

J. Wang, Z. Zhang, and H. Zha. Adaptive manifold learning. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

D. Wedge, D. Ingram, D. McLean, C. Mingham, and Z. Bandar. On global-local artificial neural networks for function approximation. *IEEE Transactions on Neural Networks*, 17(4):942–952, 2006.

K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70(1):77–90, 2006.

K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *International Conference on Machine Learning*, pages 839–846, 2004.

K.Q. Weinberger and L.K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *International Conference on Computer Vision and Pattern Recognition*, pages II: 988–995, 2004.

M. Wertheimer. Laws of organization in perceptual forms. *Psycologische Forschung, Translation by W. Ellis, A source book of Gestalt psychology (1938)*, 4:301–350, 1923.

C.K.I. Williams and C.E. Rasmussen. Gaussian processes for regression. In *Advances in Neural Information Processing Systems 8*, pages 514–520. MIT Press, Cambridge, MA, 1996.

L. Xu, M. I. Jordan, and G. E. Hinton. An alternative model for mixtures of experts. In *Advances in Neural Information Processing Systems 7*, pages 633–640. MIT Press, Cambridge, MA, 1995.

S.C. Yen and L.H. Finkel. Extraction of perceptually salient contours by striate cortical networks. *Vision Research*, 38(5):719–741, 1998.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26(1):313–338, 2004.

Journal of Machine Learning Research 11 (2010) 451-490 Submitted 4/09; Revised 12/09; Published 2/10

Information Retrieval Perspective to Nonlinear Dimensionality Reduction for Data Visualization

Jarkko Venna JARKKO.VENNA@NUMOS.FI
Jaakko Peltonen JAAKKO.PELTONEN@TKK.FI
Kristian Nybo KRISTIAN.NYBO@TKK.FI
Helena Aidos HELENA.AIDOS@TKK.FI
Samuel Kaski SAMUEL.KASKI@TKK.FI
Aalto University School of Science and Technology
Department of Information and Computer Science
P.O. Box 15400, FI-00076 Aalto, Finland

Editor: Yoshua Bengio

Abstract

Nonlinear dimensionality reduction methods are often used to visualize high-dimensional data, although the existing methods have been designed for other related tasks such as manifold learning. It has been difficult to assess the quality of visualizations since the task has not been well-defined. We give a rigorous definition for a specific visualization task, resulting in quantifiable goodness measures and new visualization methods. The task is *information retrieval* given the visualization: to find similar data based on the similarities shown on the display. The fundamental tradeoff between precision and recall of information retrieval can then be quantified in visualizations as well. The user needs to give the relative cost of missing similar points vs. retrieving dissimilar points, after which the total cost can be measured. We then introduce a new method NeRV (*neighbor retrieval visualizer*) which produces an optimal visualization by minimizing the cost. We further derive a variant for supervised visualization; class information is taken rigorously into account when computing the similarity relationships. We show empirically that the unsupervised version outperforms existing unsupervised dimensionality reduction methods in the visualization task, and the supervised version outperforms existing supervised methods.

Keywords: information retrieval, manifold learning, multidimensional scaling, nonlinear dimensionality reduction, visualization

1. Introduction

Visualization of high-dimensional data sets is one of the traditional applications of nonlinear dimensionality reduction methods. In high-dimensional data, such as experimental data where each dimension corresponds to a different measured variable, dependencies between different dimensions often restrict the data points to a manifold whose dimensionality is much lower than the dimensionality of the data space. Many methods are designed for *manifold learning*, that is, to find and unfold the lower-dimensional manifold. There has been a research boom in manifold learning since 2000, and there now exist many methods that are known to unfold at least certain kinds of manifolds successfully. Some of the successful methods include isomap (Tenenbaum et al., 2000), locally linear embedding (LLE; Roweis and Saul, 2000), Laplacian eigenmap (LE; Belkin and Niyogi, 2002a), and maximum variance unfolding (MVU; Weinberger and Saul, 2006).

It has turned out that the manifold learning methods are not necessarily good for information visualization. Several methods had severe difficulties when the output dimensionality was fixed to two for visualization purposes (Venna and Kaski, 2007a). This is natural since they have been designed to find a manifold, not to compress it into a lower dimensionality.

In this paper we discuss the specific visualization task of projecting the data to points on a two-dimensional display. Note that this task is different from manifold learning, in case the inherent dimensionality of the manifold is higher than two and the manifold cannot be represented perfectly in two dimensions. As the representation is necessarily imperfect, defining and using a *measure of goodness* of the representation is crucial. However, in spite of the large amount of research into methods for extracting manifolds, there has been very little discussion on what a good two-dimensional representation should be like and how the goodness should be measured. In a recent survey of 69 papers on dimensionality reduction from years 2000–2006 (Venna, 2007) it was found that 28 ($\approx 40\%$) of the papers only presented visualizations of toy or real data sets as a proof of quality. Most of the more quantitative approaches were based on one of two strategies. The first is to measure preservation of all pairwise distances or the order of all pairwise distances. Examples of this approach include the multidimensional scaling (MDS)-type cost functions like Sammon's cost and Stress, methods that relate the distances in the input space to the output space, and various correlation measures that assess the preservation of all pairwise distances. The other common quality assurance strategy is to classify the data in the low-dimensional space and report the classification performance.

The problem with using the above approaches to measure visualization performance is that their connection to visualization is unclear and indirect at best. Unless the purpose of the visualization is to help with a classification task, it is not obvious what the classification accuracy of a projection reveals about its goodness as a visualization. Preservation of pairwise distances, the other widely adopted principle, is a well-defined goal; it is a reasonable goal if the analyst wishes to use the visualization to assess distances between selected pairs of data points, but we argue that this is not the typical way how an analyst would use a visualization, at least in the early stages of analysis when no hypothesis about the data has yet been formed. Most approaches including ours are based on pairwise distances at heart, but we take into account the context of each pairwise distance, yielding a more natural way of evaluating visualization performance; the resulting method has a natural and rigorous interpretation which we discuss below and in the following sections.

In this paper we make rigorous the specific information visualization task of projecting a high-dimensional data set onto a two-dimensional plane for visualizing similarity relationships. This task has a very natural mapping into an information retrieval task as will be discussed in Section 2. The conceptualization as information retrieval explicitly reveals the necessary tradeoff between precision and recall, of making true similarities visible and avoiding false similarities. The tradeoff can be quantified exactly once costs have been assigned to each of the two error types, and once the total cost has been defined, it can be optimized as will be discussed in Section 3. We then show that the resulting method, called NeRV for *neighbor retrieval visualizer*, can be further extended to supervised visualization, and that both the unsupervised and supervised methods empirically outperform their alternatives. NeRV includes the previous method called stochastic neighbor embedding (SNE; Hinton and Roweis, 2002) as a special case where the tradeoff is set so that only recall is maximized; thus we give a new information retrieval interpretation to SNE.

This paper extends our earlier conference paper (Venna and Kaski, 2007b) which introduced the ideas in a preliminary form with preliminary experiments. The current paper gives the full justification and comprehensive experiments, and also introduces the supervised version of NeRV.

2. Visualization as Information Retrieval

In this section we define formally the specific visualization task; this is a novel formalization of visualization as an *information retrieval task*. We first give the definition for a simplified setup in Section 2.1, and then generalize it in Section 2.2.

2.1 Similarity Visualization with Binary Neighborhood Relationships

In the following we first define the specific visualization task and a cost function for it; we then show that the cost function is related to the traditional information retrieval measures *precision* and *recall*.

2.1.1 TASK DEFINITION: SIMILARITY VISUALIZATION

Let $\{\mathbf{x}_i\}_{i=1}^N$ be a set of input data samples, and let each sample i have an *input neighborhood* P_i, consisting of samples that are close to i. Typically, P_i might consist of all input samples (other than i itself) that fall within some radius of i, or alternatively P_i might consist of a fixed number of input samples most similar to i. In either case, let r_i be the size of the set P_i.

The *goal of similarity visualization* is to produce low-dimensional output coordinates $\{\mathbf{y}_i\}_{i=1}^N$ for the input data, usable in visual information retrieval. Given any sample i as a query, in visual information retrieval samples are retrieved based on the visualization; the retrieved result is a set Q_i of samples that are close to $\mathbf{y}_i$ in the visualization; we call Q_i the *output neighborhood*. The Q_i typically consists of all input samples j (other than i itself) whose visualization coordinates $\mathbf{y}_j$ are within some radius of $\mathbf{y}_i$ in the visualization, or alternatively Q_i might consist of a fixed number of input samples whose output coordinates are nearest to $\mathbf{y}_i$. In either case, let k_i be the number of points in the set Q_i. The number of points in Q_i may be different from the number of points in P_i; for example, if many points have been placed close to $\mathbf{y}_i$ in the visualization, then retrieving all points within a certain radius of $\mathbf{y}_i$ might yield too many retrieved points, compared to how many are neighbors in the input space. Figure 1 illustrates the setup.

The remaining question is what is a good visualization, that is, what is the cost function. Denote the number of samples that are in both Q_i and P_i by $N_{\mathrm{TP},i}$ (true positives), samples that are in Q_i but not in P_i by $N_{\mathrm{FP},i}$ (false positives), and samples that are in P_i but not Q_i by $N_{\mathrm{MISS},i}$ (misses). Assume the user has assigned a cost C_{FP} for each false positive and C_{MISS} for each miss. The total cost E_i for query i, summed over all data points, then is

$$E_i = N_{\mathrm{FP},i} C_{\mathrm{FP}} + N_{\mathrm{MISS},i} C_{\mathrm{MISS}} \,. \tag{1}$$

2.1.2 RELATIONSHIP TO PRECISION AND RECALL

The cost function of similarity visualization (1) bears a close relationship to the traditional measures of information retrieval, precision and recall. If we allow C_{MISS} to be a function of the total number of relevant points r, more specifically $C_{\mathrm{MISS}}(r_i) = C'_{\mathrm{MISS}}/r_i$, and take the cost per retrieved point by

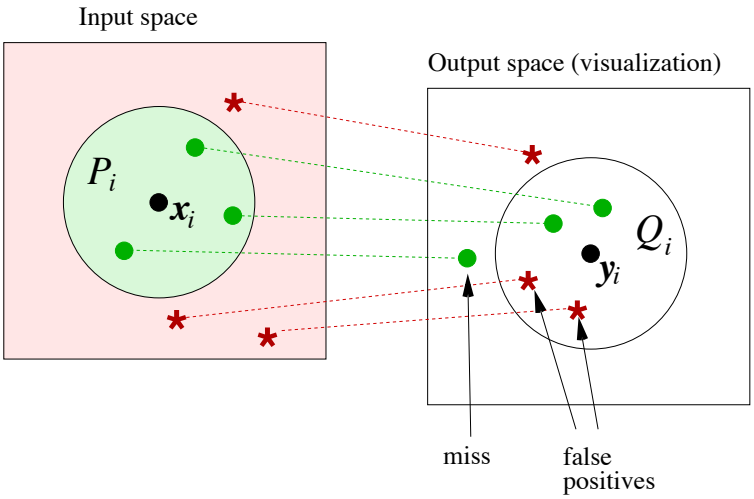


Figure 1: Diagram of the types of errors in visualization.

dividing by k_i, the total cost becomes

$$\begin{aligned} E(k_i, r_i) &= \frac{1}{k_i} E(r_i) = \frac{1}{k_i} \left(N_{\mathrm{FP},i} C_{\mathrm{FP}} + N_{\mathrm{MISS},i} C_{\mathrm{MISS}}(r_i) \right) \\ &= C_{\mathrm{FP}} \frac{N_{\mathrm{FP},i}}{k_i} + \frac{C'_{MISS}}{k_i} \frac{N_{\mathrm{MISS},i}}{r_i} \\ &= C_{\mathrm{FP}} (1 - \mathrm{precision}(i)) + \frac{C'_{MISS}}{k_i} (1 - \mathrm{recall}(i)) \, . \end{aligned}$$

The traditional definition of precision for a single query is

$$\mathrm{precision}(i) = \frac{N_{\mathrm{TP},i}}{k_i} = 1 - \frac{N_{\mathrm{FP},i}}{k_i} \, ,$$

and recall is

$$\mathrm{recall}(i) = \frac{N_{\mathrm{TP},i}}{r_i} = 1 - \frac{N_{\mathrm{MISS},i}}{r_i} \, .$$

Hence, fixing the costs C_{FP} and C_{MISS} and minimizing (1) corresponds to maximizing a specific weighted combination of precision and recall.

Finally, to assess performance of the full visualization the cost needs to be averaged over all samples (queries) which yields mean precision and recall of the visualization.

2.1.3 DISCUSSION

Given a high-dimensional data set, it is generally not possible to show all the similarity relationships within the data on a low-dimensional display; therefore, all linear or nonlinear dimensionality reduction methods need to make a tradeoff about which kinds of similarity relationships they aim to show on the display. Equation (1) fixes the tradeoff given the costs of the two kinds of errors. Figure 2 illustrates this tradeoff (computed with methods introduced in Section 3) with a toy example where a three-dimensional sphere surface is visualized in two dimensions. If we take some

query point in the visualization and retrieve a set of points close-by in the visualization, in display **A** such retrieval yields few false positives but many misses, whereas in display **B** the retrieval yields few misses but many false positives. The tradeoff can also be seen in the (mean) precision-recall curves for the two visualizations, where the number of retrieved points is varied to yield the curve. Visualization **A** reaches higher values of precision, but the precision drops much before high recall is reached. Visualization **B** has lower precision at the left end of the curve, but precision does not drop as much even when high recall is reached.

Note that in order to quantify the tradeoff, both precision and recall need to be used. This requires a rich enough retrieval model, in the sense that the number of retrieved points can be different from the number of relevant points, so that precision and recall get different values. It is well-known in information retrieval that if the numbers of relevant and retrieved items (here points) are equal, precision and recall become equal. The recent "local continuity" criterion (Equation 9 in Chen and Buja, 2009) is simply precision/recall under this constraint; we thus give a novel information retrieval interpretation of it as a side result. Such a criterion is useful but it gives only a limited view of the quality of visualizations, because it corresponds to a limited retrieval model and cannot fully quantify the precision-recall tradeoff. In this paper we will use fixed-radius neighborhoods (defined more precisely in Section 2.2) in the visualizations, which naturally yields differing numbers of retrieved and relevant points.

The simple visualization setup presented in this section is a novel formulation of visualization and useful as a clearly defined starting point. However, for practical use it has a shortcoming: the overly simple binary fixed-size neighborhoods do not take into account *grades of relevance*. The cost function does not penalize violating the original similarity ordering of neighbor samples; and the cost function penalizes all neighborhood violations with the same cost. Next we will introduce a more practical visualization setup.

2.2 Similarity Visualization with Continuous Neighborhood Relationships

We generalize the simple binary neighborhood case by defining probabilistic neighborhoods both in the (i) input and (ii) output spaces, and (iii) replacing the binary precision and recall measures with probabilistic ones. It will finally be shown that for binary neighborhoods, interpreted as a constant high probability of being a neighbor within the neighborhood set and a constant low probability elsewhere, the measures reduce to the standard precision and recall.

2.2.1 PROBABILISTIC MODEL OF RETRIEVAL

We start by defining the neighborhood in the output space, and do that by defining a probability distribution over the neighbor points. Such a distribution is interpretable as a model about how the user does the retrieval given the visualization display.

Given the location of the query point on the display, $\mathbf{y}_i$, suppose that the user selects one point at a time for inspection. Denote by $q_{j|i}$ the probability that the user chooses $\mathbf{y}_j$. If we can define such probabilities, they will define a *probabilistic model of retrieval* for the neighbors of $\mathbf{y}_i$.

The form of $q_{j|i}$ can be defined by a few axiomatic choices and a few arbitrary ones. Since the $q_{j|i}$ are a probability distribution over j for each i, they must be nonnegative and sum to one over j; therefore we can represent them as $q_{j|i} = \exp(-f_{i,j})/\sum_{k\neq i}\exp(-f_{i,k})$ where $f_{i,j} \in \mathbb{R}$. The $f_{i,j}$ should be an increasing function of distance (dissimilarity) between $\mathbf{y}_i$ and $\mathbf{y}_j$; we further assume that $f_{i,j}$ depends only on $\mathbf{y}_i$ and $\mathbf{y}_j$ and not on the other points $\mathbf{y}_k$. It remains to choose the form of

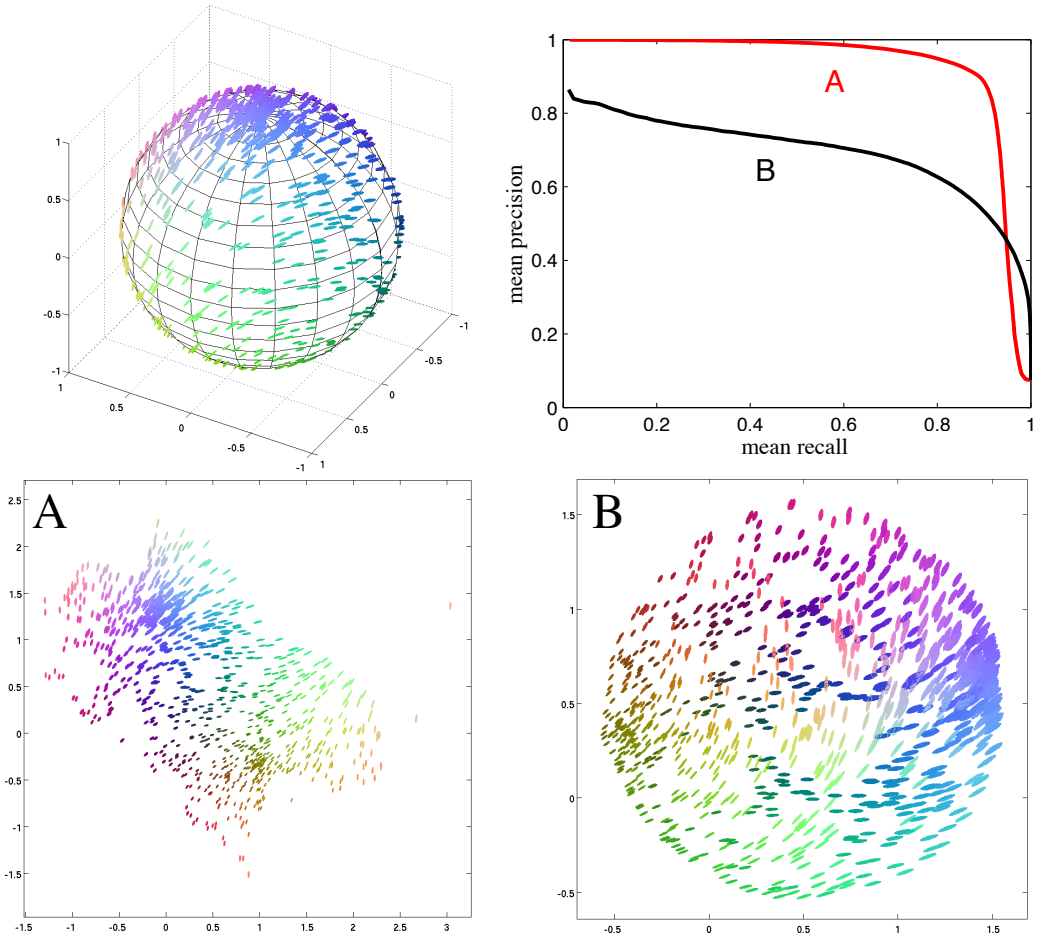


Figure 2: Demonstration of the tradeoff between false positives and misses. **Top left:** A three-dimensional data set sampled from the surface of a sphere; only the front hemisphere is shown for clarity. The glyph shapes (size, elongation, and angle) show the three-dimensional coordinates of each point; the colors in the online version show the same information. **Bottom:** Two embeddings of the data set. In the embedding **A**, the sphere has been cut open and folded out. This embedding eliminates *false positives*, but there are some *misses* because points on different sides of the tear end up far away from each other. In contrast, the embedding **B** minimizes the number of misses by simply squashing the sphere flat; this results in a large number of false positives because points on opposite sides of the sphere are mapped close to each other. **Top right:** mean precision-mean recall curves with input neighborhood size $r = 75$, as a function of the output neighborhood size k, for the two projections. The embedding **A** has better precision (yielding higher values at the left end of the curve) whereas the embedding **B** has better recall (yielding higher values at the right end of the curve).

$f_{i,j}$. In general there should not be any reason to favor any particular neighbor point, and hence the form should not depend on j. It could depend on i, however; we assume it has a simple quadratic form $f_{i,j} = ||\mathbf{y}_i - \mathbf{y}_j||^2/\sigma_i^2$ where $||\mathbf{y}_i - \mathbf{y}_j||$ is the Euclidean distance and the positive multiplier $1/\sigma_i^2$ allows the function to grow at an individual rate for each i. This yields the definition

$$q_{j|i} = \frac{\exp(-\frac{\|\mathbf{y}_i - \mathbf{y}_j\|^2}{\sigma_i^2})}{\sum_{k \neq i} \exp(-\frac{\|\mathbf{y}_i - \mathbf{y}_k\|^2}{\sigma_i^2})} \,. \tag{2}$$

2.2.2 PROBABILISTIC MODEL OF RELEVANCE

We extend the simple binary neighborhoods of input data samples to probabilistic neighborhoods as follows. Suppose that if the user was choosing the neighbors of a query point i in the original data space, she would choose point j with probability $p_{j|i}$. The $p_{j|i}$ define a *probabilistic model of relevance* for the original data, and are equivalent to a neighborhood around i: the higher the chance of choosing this neighbor, the larger its relevance to i.

We define the probability $p_{j|i}$ analogously to $q_{j|i}$, as

$$p_{j|i} = \frac{\exp(-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{\sigma_i^2})}{\sum_{k \neq i} \exp(-\frac{d(\mathbf{x}_i, \mathbf{x}_k)^2}{\sigma_i^2})} \,, \tag{3}$$

where $d(\cdot,\cdot)$ is a suitable difference measure in the original data, and $\mathbf{x}_i$ refers to the point in the original data that is represented by $\mathbf{y}_i$ in the visualization. Some data sets may provide the values of $d(\cdot,\cdot)$ directly; otherwise the analyst can choose a difference measure suitable for the data feature vectors. Later in this paper we will use both the simple Euclidean distance and a more complicated distance measure that incorporates additional information about the data.

Given known values of $d(\cdot,\cdot)$, the above definition of the neighborhood $p_{j|i}$ can be motivated by the same arguments as $q_{j|i}$. That is, the given form of $p_{j|i}$ is a good choice if no other information about the original neighborhoods is available. Other choices are possible too; in particular, if the data directly includes neighbor probabilities, they can simply be used as the $p_{j|i}$. Likewise, if more accurate models of user behavior are available, they can be plugged in place of $q_{j|i}$. The forms of $p_{j|i}$ and $q_{j|i}$ need not be the same.

For each point i, the scaling parameter σ_i controls how quickly the probabilities $p_{j|i}$ fall off with distance. These parameters could be fixed by prior knowledge, but without such knowledge it is reasonable to set the σ_i by specifying how much flexibility there should be about the choice of neighbors. That is, we set σ_i to a value that makes the entropy of the $p_{\cdot|i}$ distribution equal to $\log k$, where k is a rough upper limit for the number of relevant neighbors, set by the user. We use the same relative scale σ_i both in the input and output spaces (Equations 2 and 3).

2.2.3 COST FUNCTIONS

The remaining task is to measure how well the retrieval done in the output space, given the visualization, matches the true relevances defined in the input space. Both were above defined in terms of distributions, and a natural candidate for the measure is the *Kullback-Leibler divergence*, defined as

$$D(p_i, q_i) = \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

where p_i and q_i are the neighbor distributions for a particular point i, in the input space and in the visualization respectively. For the particular probability distributions defined above the Kullback-Leibler divergence turns out to be intimately related to precision and recall. Specifically, for any query i, the Kullback-Leibler divergence $D(p_i, q_i)$ is a generalization of recall, and $D(q_i, p_i)$ is a generalization of precision; for simple "binary" neighborhood definitions, the Kullback-Leibler divergences and the precision-recall measures become equivalent. The proof is in Appendix A.

We call $D(q_i, p_i)$ *smoothed precision* and $D(p_i, q_i)$ *smoothed recall*. To evaluate a complete visualization rather than a single query, we define aggregate measures in the standard fashion: mean smoothed precision is defined as $\mathbb{E}_i[D(q_i, p_i)]$ and mean smoothed recall as $\mathbb{E}_i[D(p_i, q_i)]$, where $\mathbb{E}$ denotes expectation and the means are taken over queries (data points i).

Mean smoothed precision and recall are analogous to mean precision and recall in that we cannot in general reach the optimum of both simultaneously. We return to Figure 2 which illustrates the tradeoff for nonlinear projections of a three-dimensional sphere surface. The subfigure **A** was created by maximizing mean smoothed precision; the sphere has been cut open and folded out, which minimizes the number of false positives but also incurs some misses because some points located on opposite edges of the point cloud were originally close to each other on the sphere. The subfigure **B** was created by maximizing mean smoothed recall; the sphere is squashed flat, which minimizes the number of misses, as all the points that were close to each other in the original data are close to each other in the visualization. However, there are then a large number of false positives because opposite sides of the sphere have been mapped on top of each other, so that many points that appear close to each other in the visualization are actually originally far away from each other.

2.2.4 EASIER-TO-INTERPRET ALTERNATIVE GOODNESS MEASURES

Mean smoothed precision and recall are rigorous and well-motivated measures of visualization performance, but they have one practical shortcoming for human analysts: the errors have no upper bound, and the scale will tend to depend on the data set. The measures are very useful for comparing several visualizations of the same data, and will turn out to be useful as optimization criteria, but we would additionally like to have measures where the plain numbers are easily interpretable. We address this by introducing *mean rank-based smoothed precision and recall*: simply replace the distances in the definitions of $p_{j|i}$ and $q_{j|i}$ with ranks, so that the probability for the nearest neighbor uses a distance of 1, the probability for the second nearest neighbor a distance of 2, and so on. This imposes an upper bound on the error because the worst case scenario is that the ranks in the data set are reversed in the visualization. Dividing the errors by their upper bounds gives us measures that lie in the interval $[0, 1]$ regardless of the data and are thus much easier to interpret. The downside is that substituting ranks for distances makes the measures disregard much of the neighborhood structure in the data, so we suggest using mean rank-based smoothed precision and recall as easier-to-interpret, but less discriminating complements to, rather than replacements of, mean smoothed precision and recall.

3. Neighborhood Retrieval Visualizer (NeRV)

In Section 2 we defined similarity visualization as an information retrieval task. The quality of a visualization can be measured by the two loss functions, mean smoothed precision and recall. These measures generalize the straightforward precision and recall measures to non-binary neighborhoods. They have the further advantage of being continuous and differentiable functions of the

output visualization coordinates. It is then easy to use the measures as *optimization criteria* for a visualization method. We now introduce a visualization algorithm that optimizes visual information retrieval performance. We call the algorithm the *neighborhood retrieval visualizer* (NeRV).

As demonstrated in Figure 2, precision and recall cannot in general be minimized simultaneously, and the user has to choose which loss function (average smoothed precision or recall) is more important, by assigning a cost for misses and a cost for false positives. Once these costs have been assigned, the visualization task is simply to minimize the total cost. In practice the relative cost of false positives to misses is given as a parameter λ. The NeRV cost function then becomes

$$E_{\mathrm{NeRV}} = \lambda \mathbb{E}_i[D(p_i, q_i)] + (1-\lambda)\mathbb{E}_i[D(q_i, p_i)] \propto \lambda \sum_i \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} + (1-\lambda) \sum_i \sum_{j \neq i} q_{j|i} \log \frac{q_{j|i}}{p_{j|i}} \quad (4)$$

where, for example, setting λ to 0.1 indicates that the user considers an error in precision $(1-0.1)/0.1 = 9$ times as expensive as a similar error in recall.

To optimize the cost function (4) with respect to the output coordinates $\mathbf{y}_i$ of each data point, we use a standard conjugate gradient algorithm. The computational complexity of each iteration is $O(dn^2)$, where n is the number of data points and d the dimension of the projection. (In our earlier conference paper a coarse approximate algorithm was required for speed; this turned out to be unnecessary, and the $O(dn^2)$ complexity does not require any approximation.) Note that if a pairwise distance matrix in the input space is not directly provided as data, it can as usual be computed from input features; this is a one-time computation done at the start of the algorithm and takes $O(Dn^2)$ time, where D is the input dimensionality.

In general, NeRV optimizes a user-defined cost which forms a tradeoff between mean smoothed precision and mean smoothed recall. If we set $\lambda = 1$ in Equation (4), we obtain the cost function of stochastic neighbor embedding (SNE; see Hinton and Roweis, 2002). Hence we get as a side result a new interpretation of SNE as a method that maximizes mean smoothed recall.

3.0.5 PRACTICAL ADVICE ON OPTIMIZATION

After computing the distance matrix from the input data, we scale the input distances so that the average distance is equal to 1. We use a random projection onto the unit square as a starting point for the algorithm. Even this simple choice has turned out to give better results than alternatives; a more intelligent initialization, such as projecting the data using principal component analysis, can of course also be used.

To speed up convergence and avoid local minima, we apply a further initialization step: we run ten rounds of conjugate gradient (two conjugate gradient steps per round), and after each round decrease the neighborhood scaling parameters σ_i used in Equations (2) and (3). Initially, we set the σ_i to half the diameter of the input data. We decrease them linearly so that the final value makes the entropy of the $p_{j|i}$ distribution equal to an effective number of neighbors k, which is the choice recommended in Section 2.2. This initialization step has the same complexity $O(dn^2)$ per iteration as the rest of the algorithm. After this initialization phase we perform twenty standard conjugate gradient steps.

4. Using NeRV for Unsupervised Visualization

It is easy to apply NeRV for unsupervised dimensionality reduction. As in any unsupervised analysis, the analyst first chooses a suitable unsupervised similarity or distance measure for the input data; for vector-valued input data this can be the standard Euclidean distance (which we will use here), or it can be some other measure suggested by domain knowledge. Once the analyst has specified the relative importance of precision and recall by choosing a value for λ, the NeRV algorithm computes the embedding based on the distances it is given.

In this section we will make extensive experiments comparing the performance of NeRV with other dimensionality reduction methods on unsupervised visualization of several data sets, including both benchmark data sets and real-life bioinformatics data sets. In the following subsections, we describe the comparison methods and data sets, briefly discuss the experimental methodology, and present the results.

4.1 Comparison Methods for Unsupervised Visualization

For the task of unsupervised visualization we compare the performance of NeRV with the following unsupervised nonlinear dimensionality reduction methods: principal component analysis (PCA; Hotelling, 1933), metric multidimensional scaling (MDS; see Borg and Groenen, 1997), locally linear embedding (LLE; Roweis and Saul, 2000), Laplacian eigenmap (LE; Belkin and Niyogi, 2002a), Hessian-based locally linear embedding (HLLE; Donoho and Grimes, 2003), isomap (Tenenbaum et al., 2000), curvilinear component analysis (CCA; Demartines and Hérault, 1997), curvilinear distance analysis (CDA; Lee et al., 2004), maximum variance unfolding (MVU; Weinberger and Saul, 2006), landmark maximum variance unfolding (LMVU; Weinberger et al., 2005), and our previous method local MDS (LMDS; Venna and Kaski, 2006).

Principal component analysis (PCA; Hotelling, 1933) finds linear projections that maximally preserve the variance in the data. More technically, the projection directions can be found by solving for the eigenvalues and eigenvectors of the covariance matrix $\mathbf{C}_x$ of the input data points. The eigenvectors corresponding to the two or three largest eigenvalues are collected into a matrix $\mathbf{A}$, and the data points $\mathbf{x}_i$ can then be visualized by projecting them with $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i$, where $\mathbf{y}_i$ is the obtained low-dimensional representation of $\mathbf{x}_i$. PCA is very closely related to linear multidimensional scaling (linear MDS, also called classical scaling; Torgerson, 1952; Gower, 1966), which tries to find low-dimensional coordinates preserving squared distances. It can be shown (Gower, 1966) that when the dimensionality of the sought solutions is the same and the distance measure is Euclidean, the projection of the original data to the PCA subspace equals the configuration of points found by linear MDS. This implies that PCA tries to preserve the squared distances between data points, and that linear MDS finds a solution that is a linear projection of the original data.

Traditional *multidimensional scaling* (MDS; see Borg and Groenen, 1997) exists in several different variants, but they all have a common goal: to find a configuration of output coordinates that preserves the pairwise distance matrix of the input data. For the comparison experiments we chose *metric MDS* which is the simplest nonlinear MDS method; its cost function (Kruskal, 1964), called the raw stress, is

$$E = \sum_{i,j} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2, \tag{5}$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the distance of points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the input space and $d(\mathbf{y}_i, \mathbf{y}_j)$ is the distance of their corresponding representations (locations) $\mathbf{y}_i$ and $\mathbf{y}_j$ in the output space. This cost function is minimized with respect to the representations $\mathbf{y}_i$.

Isomap (Tenenbaum et al., 2000) is an interesting variant of MDS, which again finds a configuration of output coordinates matching a given distance matrix. The difference is that Isomap does not compute pairwise input-space distances as simple Euclidean distances but as *geodesic distances* along the manifold of the data (technically, along a graph formed by connecting all k-nearest neighbors). Given these geodesic distances the output coordinates are found by standard linear MDS. When output coordinates are found for such input distances, the manifold structure in the original data becomes unfolded; it has been shown (Bernstein et al., 2000) that this algorithm is asymptotically able to recover certain types of manifolds. We used the isomap implementation available at `http://isomap.stanford.edu` in the experiments.

Curvilinear component analysis (CCA; Demartines and Hérault, 1997) is a variant of MDS that tries to preserve only distances between points that are near each other in the visualization. This is achieved by weighting each term in the MDS cost function (5) by a coefficient that depends on the corresponding pairwise distance in the visualization. In the implementation we use, the coefficient is simply a step function that equals 1 if the distance is below a predetermined threshold and 0 if it is larger.

Curvilinear distance analysis (CDA; Lee et al., 2000, 2004) is an extension of CCA. The idea is to replace the Euclidean distances in the original space with geodesic distances in the same manner as in the isomap algorithm. Otherwise the algorithm stays the same.

Local MDS (LMDS; Venna and Kaski, 2006) is our earlier method, an extension of CCA that focuses on local proximities with a tunable cost function tradeoff. It can be seen as a first step in the development of the ideas of NeRV.

The *locally linear embedding* (LLE; Roweis and Saul, 2000) algorithm is based on the assumption that the data manifold is smooth enough and is sampled densely enough, such that each data point lies close to a locally linear subspace on the manifold. LLE makes a locally linear approximation of the whole data manifold: LLE first estimates a local coordinate system for each data point, by calculating linear coefficients that reconstruct the data point as well as possible from its k nearest neighbors. To unfold the manifold, LLE finds low-dimensional coordinates that preserve the previously estimated local coordinate systems as well as possible. Technically, LLE first minimizes the reconstruction error $E(\mathbf{W}) = \sum_i \|\mathbf{x}_i - \sum_j W_{i,j}\mathbf{x}_j\|^2$ with respect to the coefficients $W_{i,j}$, under the constraints that $W_{i,j} = 0$ if i and j are not neighbors, and $\sum_j W_{i,j} = 1$. Given the weights, the low-dimensional configuration of points is next found by minimizing $E(\mathbf{Y}) = \sum_i \|\mathbf{y}_i - \sum_j W_{i,j}\mathbf{y}_j\|^2$ with respect to the low-dimensional representation $\mathbf{y}_i$ of each data point.

The *Laplacian eigenmap* (LE; see Belkin and Niyogi, 2002a) uses a graph embedding approach. An undirected k-nearest-neighbor graph is formed, where each data point is a vertex. Points i and j are connected by an edge with weight $W_{i,j} = 1$ if j is among the k nearest neighbors of i, otherwise the edge weight is set to zero; this simple weighting method has been found to work well in practice (Belkin and Niyogi, 2002b). To find a low-dimensional embedding of the graph, the algorithm tries to put points that are connected in the graph as close to each other as possible and does not care what happens to the other points. Technically, it minimizes $\frac{1}{2}\sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{i,j} = \mathbf{y}^T\mathbf{L}\mathbf{y}$ with respect to the low-dimensional point locations $\mathbf{y}_i$, where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian and $\mathbf{D}$ is a diagonal matrix with elements $\mathrm{D}_{ii} = \sum_j W_{i,j}$. However, this cost function has an undesirable trivial solution: putting all points in the same position would minimize the cost. This can be avoided by adding suit-

able constraints. In practice the low-dimensional configuration is found by solving the generalized eigenvalue problem $\mathbf{Ly} = \lambda\mathbf{Dy}$ (Belkin and Niyogi, 2002a). The smallest eigenvalue corresponds to the trivial solution, but the eigenvectors corresponding to the next smallest eigenvalues give the Laplacian eigenmap solution.

The Laplacian eigenmap algorithm reduces to solving a generalized eigenvalue problem because the cost function that is minimized is a quadratic form involving the Laplacian matrix $\mathbf{L}$. The *Hessian-based locally linear embedding* (HLLE; Donoho and Grimes, 2003) algorithm is similar, but the Laplacian $\mathbf{L}$ is replaced by the Hessian $\mathbf{H}$.

The *maximum variance unfolding* algorithm (MVU; Weinberger and Saul, 2006) expresses dimensionality reduction as a semidefinite programming problem. One way of unfolding a folded flag is to pull its four corners apart, but not so hard as to tear the flag. MVU applies this idea to projecting a manifold: the projection maximizes variance (pulling apart) while preserving distances between neighbors (no tears). The constraint of local distance preservation can be expressed in terms of the Gram matrix $\mathbf{K}$ of the mapping. Maximizing the variance of the mapping is equivalent to maximizing the trace of $\mathbf{K}$ under a set of constraints, which, it turns out, can be done using semidefinite programming.

A notable disadvantage of MVU is the time required to solve a semidefinite program for $n \times n$ matrices when the number of data points n is large. *Landmark MVU* (LMVU; Weinberger et al., 2005) addresses this issue by significantly reducing the size of the semidefinite programming problem. Like LLE, LMVU assumes that the data manifold is sufficiently smooth and densely sampled that it is locally approximately linear. Instead of embedding all the data points directly as MVU does, LMVU randomly chooses $m \ll n$ inputs as so-called landmarks. Because of the local linearity assumption, the other data points can be approximately reconstructed from the landmarks using a linear transformation. It follows that the Gram matrix $\mathbf{K}$ can be approximated using the $m \times m$ submatrix of inner products between landmarks. Hence we only need to optimize over $m \times m$ matrices, a much smaller semidefinite program. Other recent approaches for speeding up MVU include matrix factorization based on a graph Laplacian (Weinberger et al., 2007).

In addition to the above comparison methods, other recent work on dimensionality reduction includes *minimum volume embedding* (MVE; Shaw and Jebara, 2007), which is similar to MVU, but where MVU maximizes the whole trace of the Gram matrix (the sum of all eigenvalues), MVE maximizes the sum of the first few eigenvalues and minimizes the sum of the rest, in order to preserve the largest amount of eigenspectrum energy in the few dimensions that remain after dimensionality reduction. In practice, a variational upper bound of the resulting criterion is optimized.

Very recently, a number of unsupervised methods have been compared by van der Maaten et al. (2009) in terms of classification accuracy and our old criteria trustworthiness-continuity.

4.2 Data Sets for Unsupervised Visualization

We used two synthetic benchmark data sets and four real-life data sets for our experiments.

The *plain s-curve* data set is an artificial set sampled from an S-shaped two-dimensional surface embedded in three-dimensional space. An almost perfect two-dimensional representation should be possible for a non-linear dimensionality reduction method, so this data set works as a sanity check.

The *noisy s-curve* data set is otherwise identical to the plain s-curve data set, but significant spherical normally distributed noise has been added to each data point. The result is a cloud of points where the original S-shape is difficult to discern by visual inspection.

The *faces* data set consists of ten different face images of 40 different people, for a total of 400 images. For a given subject, the images vary in terms of lighting and facial expressions. The size of each image is 64×64 pixels, with 256 grey levels per pixel. The data set is available for download at `http://www.cs.toronto.edu/~roweis/data.html`.

The *mouse gene expression* data set is a collection of gene expression profiles from different mouse tissues (Su et al., 2002). Expression of over 13,000 mouse genes had been measured in 45 tissues. We used an extremely simple filtering method, similar to that originally used by Su et al. (2002), to select the genes for visualization. Of the mouse genes clearly expressed (average difference in Affymetrix chips, AD $>$ 200) in at least one of the 45 tissues (dimensions), a random sample of 1600 genes (points) was selected. After this the variance in each tissue was normalized to unity.

The *gene expression compendium* data set is a large collection of human gene expression arrays (`http://dags.stanford.edu/cancer`; Segal et al., 2004). Since the current implementations of all methods do not tolerate missing data we removed samples with missing values altogether. First we removed genes that were missing from more than 300 arrays. Then we removed the arrays for which values were still missing. This resulted in a data set containing 1278 points and 1339 dimensions.

The *sea-water temperature time series* data set (Liitiäinen and Lendasse, 2007) is a time series of weekly temperature measurements of sea water over several years. Each data point is a time window of 52 weeks, which is shifted one week forward for the next data point. Altogether there are 823 data points and 52 dimensions.

4.3 Methodology for the Unsupervised Experiments

The performance of NeRV was compared with 11 unsupervised dimensionality reduction methods described in Section 4.1, namely principal component analysis (PCA), metric multidimensional scaling (here simply denoted MDS), locally linear embedding (LLE), Laplacian eigenmap (LE), Hessian-based locally linear embedding (HLLE), isomap, curvilinear component analysis (CCA), curvilinear distance analysis (CDA), maximum variance unfolding (MVU), landmark maximum variance unfolding (LMVU), and local MDS (LMDS). LLE, LE, HLLE, MVU, LMVU and isomap were computed with code from their developers; MDS, CCA and CDA used our code.

4.3.1 GOODNESS MEASURES

We used four pairs of performance measures to compare the methods. The first pair is *mean smoothed precision-mean smoothed recall*, that is, our new measures of visualization quality. The scale of input neighborhoods was fixed to 20 relevant neighbors (see Section 2.2).

Although we feel, as explained in Section 2, that smoothed precision and smoothed recall are more sophisticated measures of visualization performance than precision and recall, we have also plotted standard *mean precision-mean recall* curves. The curves were plotted by fixing the 20 nearest neighbors of a point in the original data as the set of relevant items, and then varying the number of neighbors retrieved from the visualization between 1 and 100, plotting mean precision and recall for each number.

Our third pair of measures are the rank-based variants of our new measures, *mean rank-based smoothed precision-mean rank-based smoothed recall*. Recall that we introduced the rank-based

variants as easier-to-interpret, but less discriminating, alternatives to mean smoothed precision and mean smoothed recall. The scale of input neighborhoods was again fixed to 20 relevant neighbors.

Our fourth pair of measures is *trustworthiness-continuity* (Kaski et al., 2003). The intuitive motivation behind these measures was the same trade-off between precision and recall as in this paper, but the measures were defined in a more ad hoc way. At the time we did not have the clear connection to information retrieval which makes NeRV particularly attractive, and we did not optimize the measures. Trustworthiness and continuity can, however, now be used as partly independent measures of visualization quality. To compute the trustworthiness and continuity, we used neighborhoods of each point containing the 20 nearest neighbors.

As a fifth measure, when data classes are available, we use *classification error* given the display, with a standard k-nearest neighbor classifier where we set $k = 5$.

4.3.2 CHOICE OF PARAMETERS

Whenever we needed to choose a parameter for any method, we used the same criterion, namely the F-measure computed from the new rank-based measures. That is, we chose the parameter yielding the largest value of $2(P \cdot R)/(P+R)$ where P and R are the mean rank-based smoothed precision and recall.

Many of the methods have a parameter k denoting the number of nearest neighbors for constructing a neighborhood graph; for each method and each data set we tested values of k ranging from 4 to 20, and chose the value that produced the best F-measure. (For MVU and LMVU we used a smaller parameter range to save computational time. For MVU k ranged from 4 to 6; for LMVU k ranged from 3 to 9.) The exceptions are local MDS (LMDS), one of our own earlier methods, and NeRV, for which we simply set k to 20 without optimizing it.

Methods that may have local optima were run five times with different random initializations and the best run (again, in terms of the F-measure) was selected.

4.4 Results of Unsupervised Visualization

We will next show visualizations for a few sets, and measure quantitatively the results of several. We begin by showing an example of a NeRV visualization for the plain S-curve data set in Figure 3. Later in this section we will show a NeRV visualization of a synthetic face data set (Figure 8), and in Section 4.6 of the *faces* data set of real face images (Figure 11). The quantitative results are spread across four figures (Figures 4–7), each of which contains results for one pair of measures and all six data sets.

We first show the curves of *mean smoothed precision-mean smoothed recall*, that is, the loss functions associated with our formalization of visualization as information retrieval. The results are shown in Figure 4. NeRV and local MDS (LMDS) form curves parameterized by λ, which ranges from 0 to 1.0 for NeRV and from 0 to 0.9 for LMDS. NeRV was clearly the best-performing method on all six data sets, which is of course to be expected since NeRV directly optimizes a linear combination of these measures. LMDS has a relatively good mean smoothed precision, but does not perform as well in terms of mean smoothed recall. Simple metric MDS also stands out as a consistently reasonably good method.

Because we formulated visualization as an information retrieval task, it is natural to also try existing measures of information retrieval performance, that is, mean precision and mean recall, even though they do not take into account grades of relevance as discussed in Section 2.1. Standard

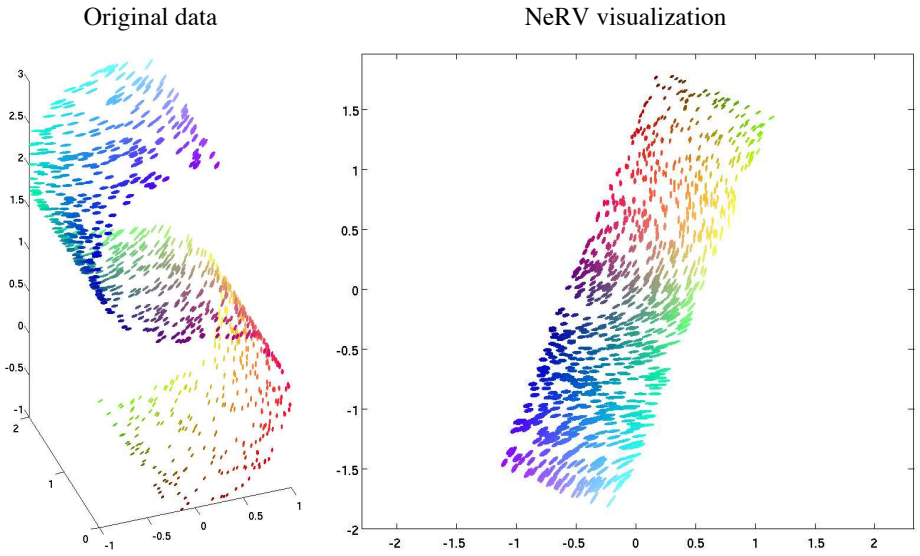


Figure 3: Left: Plain S-curve data set. The glyph shapes (size, elongation, and angle) show the three-dimensional coordinates of each point; the colors in the online version show the same information. Right: NeRV visualization (here $\lambda = 0.8$).

mean precision-mean recall curves are shown in Figure 5; for NeRV and LMDS, we show the curve for a single λ value picked by the F-measure as described in Section 4.3. Even with these coarse measures, NeRV shows excellent performance: NeRV is best on four data sets in terms of the area under the curve, CDA and CCA are each best on one data set.

Next, we plot our easier-to-interpret but less discriminating alternative measures of visualization performance. The curves of *mean rank-based smoothed precision-mean rank-based smoothed recall* are shown in Figure 6. These measures lie between 0 and 1, and may hence be easier to compare between data sets. With these measures, NeRV again performs best on all data sets; LMDS also performs well, especially on the seawater temperature data.

Finally, we plot the curves of *trustworthiness-continuity*, shown in Figure 7. The results are fairly similar to the new rank-based measures: once again NeRV performs best on all data sets and LMDS also performs well, especially on the seawater temperature data.

4.4.1 EXPERIMENT WITH A KNOWN UNDERLYING MANIFOLD

To further test how well the methods are able to recover the neighborhood structure inherent in the data we studied a synthetic face data set where a known underlying manifold defines the relevant items (neighbors) of each point. The SculptFaces data contains 698 synthetic images of a face (sized 64×64 pixels each). The pose and direction of lighting have been changed in a systematic way to create a manifold in the image space (`http://web.mit.edu/cocosci/isomap/datasets.html`; Tenenbaum et al., 2000). We used the raw pixel data as input features.

The pose and lighting parameters used to generate the images are available. These parameters define the manifold of the faces embedded in the very high-dimensional image space. For any face

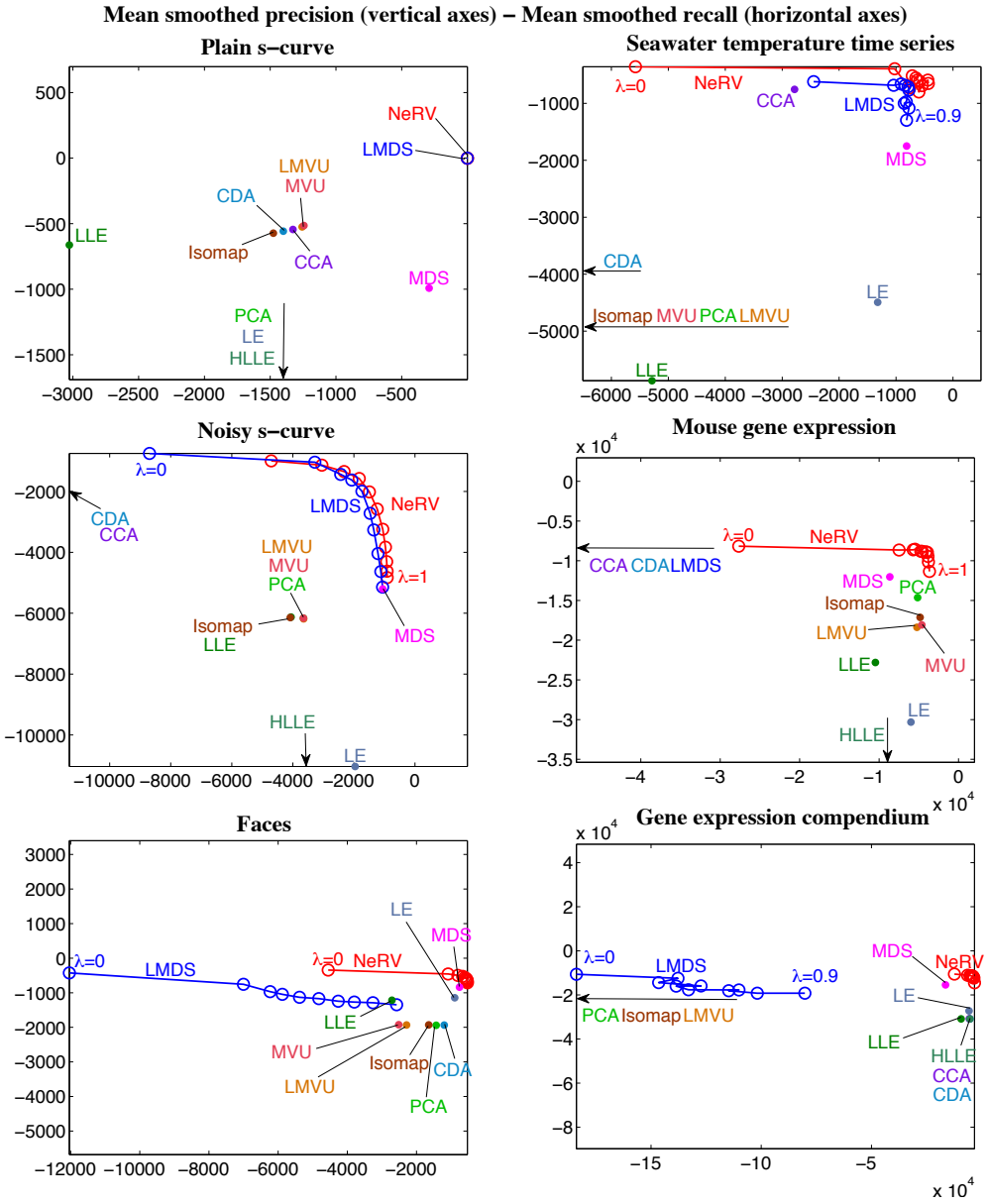


Figure 4: *Mean smoothed precision-mean smoothed recall* plotted for all six data sets. For clarity, only a few of the best-performing methods are shown for each data set. We have actually plotted $-1\cdot$(mean smoothed precision) and $-1\cdot$(mean smoothed recall) to maintain visual consistency with the plots for other measures: in each plot, the best performing methods appear in the top right corner.

image, the relevant other faces are the ones that are neighbors with respect to the pose and lighting parameters; we defined the ground truth neighborhoods using Euclidean distances in the pose and lighting space, and we fixed the scale of the ground truth neighborhoods to 20 relevant neighbors (see Section 2.2).

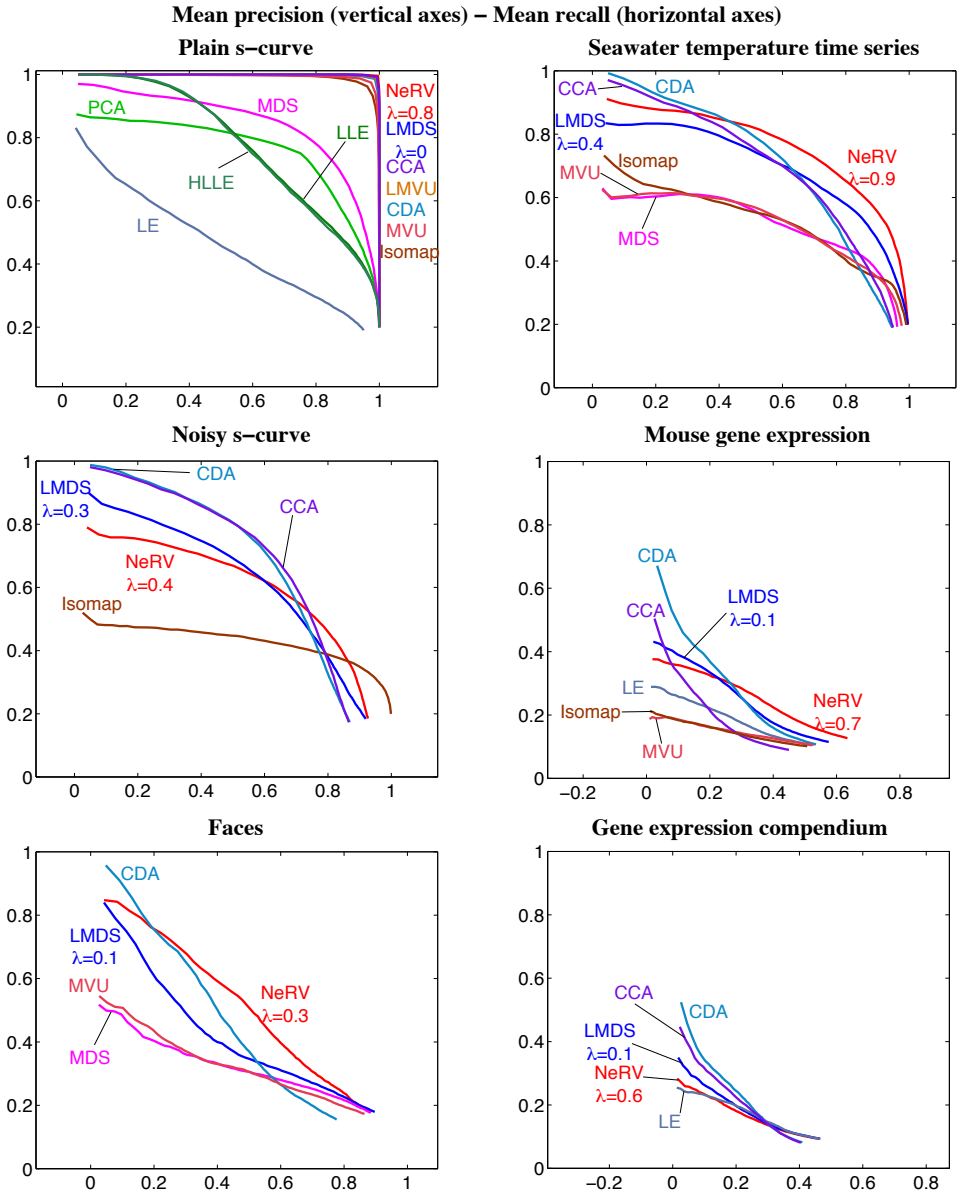


Figure 5: *Mean precision-mean recall* curves plotted for all six data sets. For clarity, only the best methods (with largest area under curve) are shown for each data set. In each plot, the best performance is in the top right corner. For NeRV and LMDS, a single λ value picked with the F-measure is shown.

We ran all methods for this data as in all experiments in this section, and then calculated four performance curves (*mean smoothed precision-mean smoothed recall*, *mean precision-mean recall*, *mean rank-based smoothed precision-mean rank-based smoothed recall*, and *trustworthiness-continuity*) using the neighborhoods in pose and lighting space as the ground truth. The results are shown in Figure 8. In spite of the very high dimensionality of the input space and the reduction of

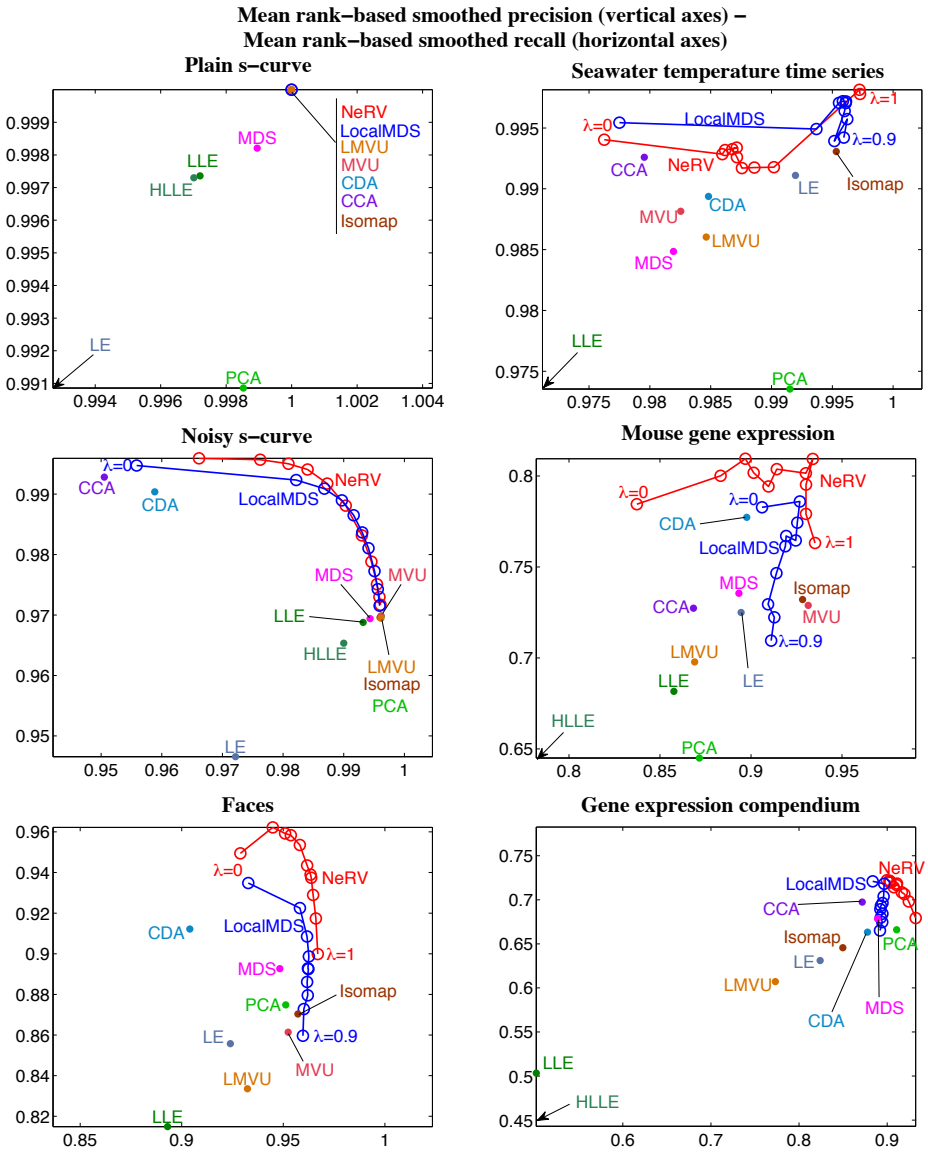


Figure 6: *Mean rank-based smoothed precision-mean rank-based smoothed recall* plotted for all six data sets. For clarity, only a few of the best performing methods are shown for each data set. We have actually plotted 1−(mean rank-based smoothed precision) and 1−(mean rank-based smoothed recall) to maintain visual consistency with the plots for other measures: in each plot, the best performance is in the top right corner.

the manifold dimension from three to two, NeRV was able to recover the structure well. NeRV is the best according to both of our proposed measures of visualization performance, mean smoothed precision and recall; MDS and local MDS also perform well. In terms of the simple mean precision and mean recall NeRV is the second best with CDA being slightly better. In terms of the rank-based measures, NeRV is the best in terms of precision; LE and MDS attain the best mean rank-based

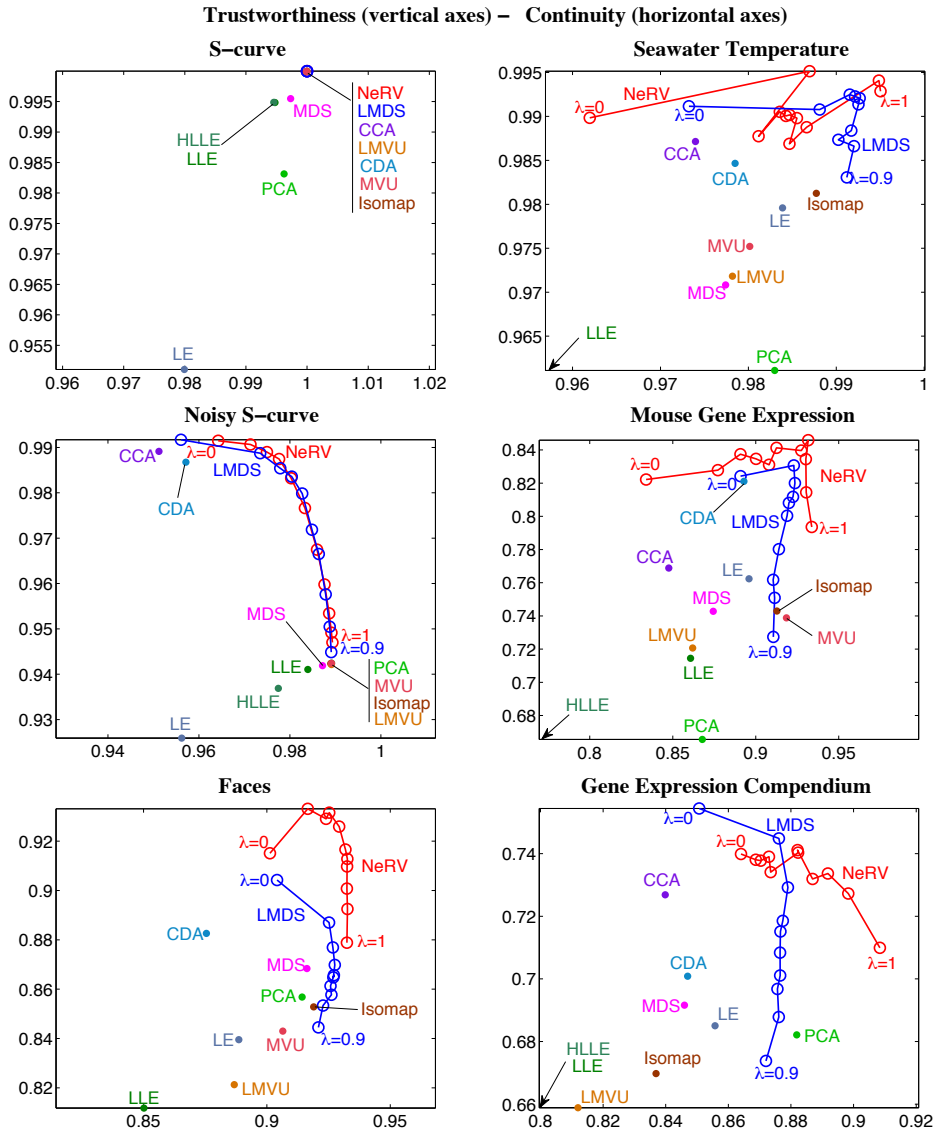


Figure 7: *Trustworthiness-continuity* plotted for all six data sets. For clarity, only a few of the best performing methods are shown for each data set. In each plot, the best performance is in the top right corner.

smoothed recall; and local MDS and CDA also perform well. When performance was measured with trustworthiness and continuity, NeRV was the best in terms of trustworthiness while MVU and Isomap attained the highest continuity.

Overall, NeRV was the best in these unsupervised visualization tasks, although it was not the best in all, and in some tasks it had tough competition.

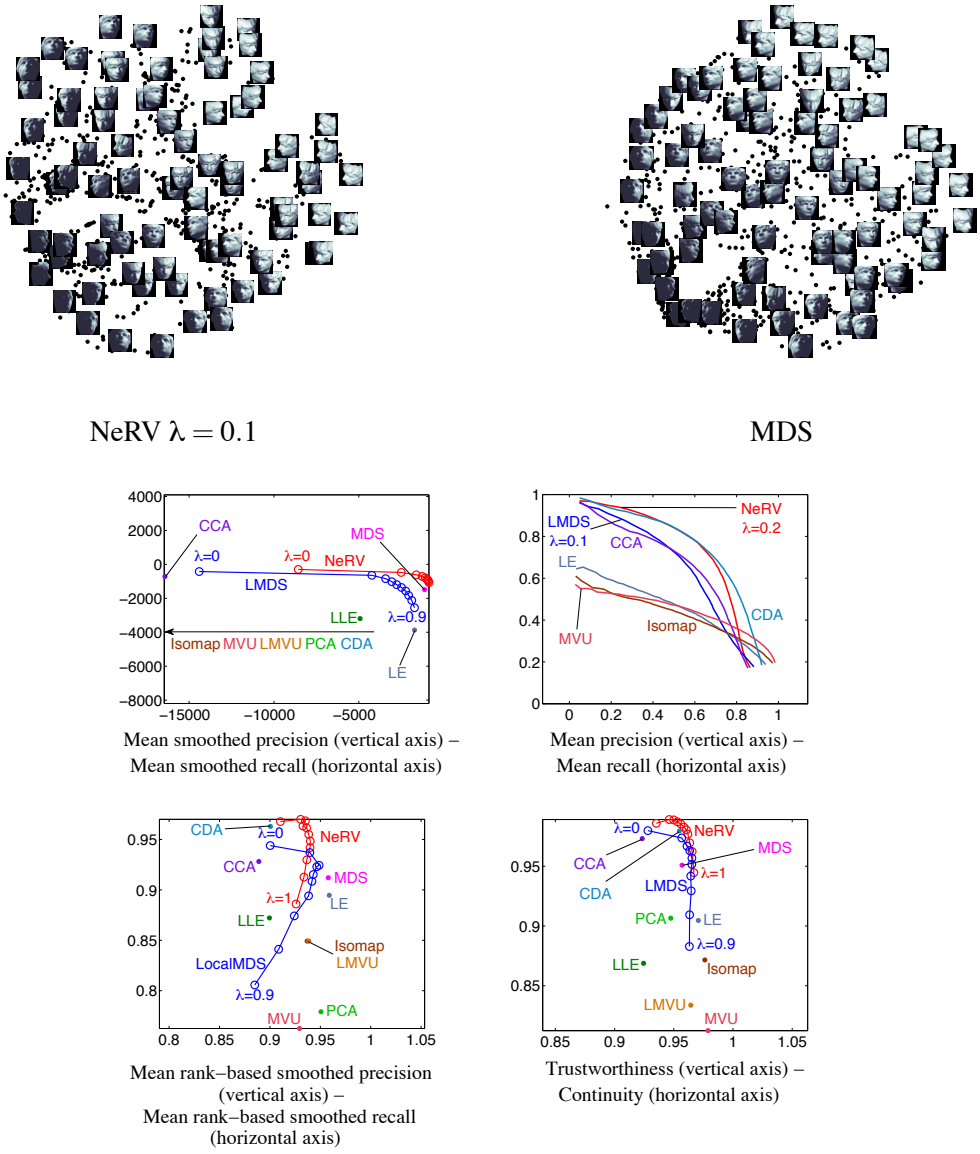


Figure 8: **Top:** Sample projections of the SculptFaces data set (NeRV vs. the best alternative). **Bottom:** How well were the ground truth neighbors in pose-lighting space retrieved from the image data, evaluated by four pairs of measures. The measures were computed the same way as before, as described in Section 4.3, but here taking the known pose and lighting information as the input data. Only the best performing methods are shown for clarity.

4.5 Comparison by Unsupervised Classification

For the data sets where a classification of samples is available, we additionally follow a traditional way to evaluate visualizations: we measure how well samples can be classified based on the visualization.

Data set	Dimensions	Classes
Letter	16	26
Phoneme	20	13
Landsat	36	6
TIMIT	12	41

Table 1: The data sets used in the unsupervised classification experiments.

Here all methods are unsupervised, that is, class labels of samples are not used in computing the visualization. The parameters of methods are again chosen as described in Section 4.3.2. Methods are evaluated by k-nearest neighbor classification accuracy (with $k = 5$), that is, each sample in the visualization is classified by majority vote of its k nearest neighbors in the visualization, and the classification is compared to the ground truth label.

We use four benchmark data sets, all of which include class labels, to compare the performances of the methods. The data sets are summarized in Table 1. For all data sets we used a randomly chosen subset of 1500 samples in the experiments, to save computation time.

The *letter recognition* data set (denoted Letter) is from the UCI Machine Learning Repository (Blake and Merz, 1998); it is a 16-dimensional data set with 26 classes, which are 4×4 images of the 26 capital letters of the alphabet. These letters are based on 20 different fonts which have been distorted to produce the final images.

The *phoneme* data set (denoted Phoneme) is taken from LVQ-PAK (Kohonen et al., 1996) and consists of phoneme samples represented by a 20-dimensional vector of features plus a class label indicating which phoneme is actually represented. There are a total of 13 classes.

The *landsat satellite* data set (denoted Landsat) is from UCI Machine Learning Repository (Blake and Merz, 1998). Each data point is a 36-dimensional vector, corresponding to a 3×3 satellite image measured in four spectral bands; the class label of the point indicates the terrain type in the image (6 possibilitities, for example red soil).

The *TIMIT* data set is taken from the DARPA TIMIT speech database (TIMIT). It is similar to the phoneme data from LVQ-PAK but the feature vectors are 12-dimensional and there are 41 classes in total.

The resulting classification error rates are shown in Table 2. NeRV is best on two out of four data sets and second best on a third set (there our old method LocalMDS is best). CDA is best on one.

4.6 NeRV, Joint Probabilities, and t-Distributions

Very recently, based on stochastic neighbor embedding (SNE), van der Maaten and Hinton (2008) have proposed a modified method called t-SNE, which has performed well in unsupervised experiments. The t-SNE makes two changes compared to SNE; in this section we describe the changes and show that the same changes can be made to NeRV, yielding a variant that we call t-NeRV. We then provide a new information retrieval interpretation for t-NeRV and t-SNE.

We start by analyzing the differences between t-SNE and the original stochastic neighbor embedding. The original SNE minimizes the sum of Kullback-Leibler divergences

$$\sum_i \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

	Letter	Phoneme	Landsat	TIMIT
Eigenmap	0.914	0.121	0.168	0.674
LLE	n/a	0.118	0.212	0.722
Isomap	0.847	0.134	0.156	0.721
MVU	0.763	0.155	0.153	0.699
LMVU	0.819	0.208	0.151	0.787
MDS	0.823	0.189	0.151	0.705
CDA	**0.336**	0.118	0.141	0.643
CCA	0.422	0.098	0.143	0.633
NeRV	0.532	**0.079**	0.139	**0.626**
LocalMDS	0.499	0.118	**0.128**	0.637

Table 2: Error rates of k-nearest neighbor classification based on the visualization, for unsupervised visualization methods. The best results for each data set are in bold; n/a denotes that LLE did not yield a result for the Letter data. NeRV attains the lowest error rate for two data sets and second lowest error rate for one data set.

where $p_{j|i}$ and $q_{j|i}$ are defined by Equations (3) and (2). We showed in Section 2.2 that this cost function has an information retrieval interpretation: it corresponds to mean smoothed recall of retrieving neighbors of query points. The t-SNE method makes two changes which we discuss below.

4.6.1 COST FUNCTION BASED ON JOINT PROBABILITIES

The first change in t-SNE is to the cost function: t-SNE minimizes a "symmetric version" of the cost function, defined as

$$\sum_i \sum_{j \neq i} p_{i,j} \log \frac{p_{i,j}}{q_{i,j}}$$

where the $p_{i,j}$ and $q_{i,j}$ are now joint probabilities over both i and j, so that $\sum_{i,j} p_{i,j} = 1$ and similarly for $q_{i,j}$. The term "symmetric" comes from the fact that the joint probabilities are defined in a specific way for which $p_{i,j} = p_{j,i}$ and $q_{i,j} = q_{j,i}$; note that this need not be the case for all definitions of the joint probabilities.

4.6.2 DEFINITIONS OF THE JOINT PROBABILITIES

The second change in t-SNE is that the joint probabilities are defined in a manner which does not yield quite the same conditional probabilities as in Equations (3) and (2). The joint probabilities are defined as

$$p_{i,j} = \frac{1}{2n}(p_{i|j} + p_{j|i}) \tag{6}$$

where $p_{i|j}$ and $p_{j|i}$ are computed by Equation (3) and n is the total number of data points in the data set, and

$$q_{i,j} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}} \,. \tag{7}$$

The former equation is intended to ensure that, in the input space, even outlier points will have some other points as neighbors. The latter equation means that, in the visualization, the joint probability

falls off according to a (normalized) t-distribution with one degree of freedom, which is intended to help with a *crowding problem*: because the volume of a small-dimensional neighborhood grows slower than the volume of a high-dimensional one, the neighborhood ends up stretched in the visualization so that moderately distant point pairs are placed too far apart. This tends to cause clumping of the data in the center of the visualization. Since the t-distribution has heavier tails than a Gaussian, using such a distribution for the $q_{i,j}$ makes the visualization less affected by the placement of the moderately distant point pairs, and hence better able to focus on other features of the data.

4.6.3 NEW METHOD: T-NERV

We can easily apply the above-described changes to the cost function in NeRV; we call the resulting variant t-NeRV. We define the cost function as

$$E_{\text{t-NeRV}} = \lambda \sum_i \sum_{j \neq i} p_{i,j} \log \frac{p_{i,j}}{q_{i,j}} + (1-\lambda) \sum_i \sum_{j \neq i} q_{i,j} \log \frac{q_{i,j}}{p_{i,j}} = \lambda D(p,q) + (1-\lambda) D(q,p) \qquad (8)$$

where p and q are the joint distributions over i and j defined by the $p_{i,j}$ and the $q_{i,j}$, and the individual joint probabilities are given by Equations (6) and (7).

It can be shown that this changed cost function again has a natural information retrieval interpretation: it corresponds to the tradeoff between *smoothed precision* and *smoothed recall* of a *two-step information retrieval task*, where an analyst looks at a visualization and (step 1) picks a query point and then (step 2) picks a neighbor for the query point. The probability of picking a query point i depends on how many other points are close to it (that is, it depends on $\sum_j q_{i,j}$), and the probability of picking a neighbor depends as usual on the relative closenesses of the neighbors to the query. Both choices are done based on the visualization, and the choices are compared by smoothed precision and smoothed recall to the relevant pairs of queries and neighbors that are defined based on the input space. The parameter λ again controls the tradeoff between precision and recall.

The connection between the $D(p,q)$ and the recall of the two-step retrieval task can be shown by a similar proof as in Appendix A, the main difference being that conditional distributions $p_{j|i}$ and $q_{j|i}$ are replaced by joint distributions $p_{i,j}$ and $q_{i,j}$, and the sums then go over both i and j. The connection between $D(q,p)$ and precision can be shown analogously.

As a special case, setting $\lambda = 1$ in the above cost function, that is, optimizing only smoothed recall of the two-step retrieval task, yields the cost function of t-SNE. We therefore provide a novel information retrieval interpretation of t-SNE as a method that maximizes recall of query points and their neighbors.

The main conceptual difference between NeRV and t-NeRV is that in t-NeRV the probability of picking a query point in the visualization and in the input space depends on the densities in the visualization and input space respectively; in NeRV all potential query points are treated equally. Which treatment of query points should be used depends on the task of the analyst. Additionally, NeRV and t-NeRV have differences in the technical forms of the probabilities, that is, whether t-distributions or Gaussians are used etc.

The t-NeRV method can be optimized with respect to visualization coordinates $\mathbf{y}_i$ of points, by conjugate gradient optimization as in NeRV; the computational complexity is also the same.

4.6.4 COMPARISON

We briefly compare t-NeRV and NeRV on the Faces data set. The setup is the same as in the previous comparison experiments (Figures 4–7). For t-NeRV we use the effective number of neighbors $k = 40$

to compute the joint probabilities $p_{i,j}$; this corresponds to the perplexity value used by the authors of t-SNE (van der Maaten and Hinton, 2008).

Figure 9 shows the results for the four unsupervised evaluation criteria. According to the mean smoothed precision and mean smoothed recall measures, t-NeRV does worse in terms of recall. The rank-based measures indicate a similar result; however, there t-NeRV does fairly well in terms of mean rank-based smoothed precision. The trustworthiness-continuity curves are similar to the rank-based measures. The curves of mean precision versus mean recall show that t-NeRV does achieve better precision for small values of recall (i.e., for small retrieved neighborhoods), while NeRV does slightly better for larger retrieved neighborhoods. These measures correspond to the information retrieval interpretation of NeRV which is slightly different from that of t-NeRV, as discussed above. Figure 9 **E** shows mean smoothed precision/recall in the t-NeRV sense, where t-NeRV naturally performs relatively better.

Lastly, we computed k-nearest neighbor classification error rate (using $k = 5$) with respect to the identity of the persons in the images. NeRV (with $\lambda = 0.3$) attained an error rate of 0.394 and t-NeRV (with $\lambda = 0.8$) an error rate of 0.226. Here t-NeRV is better; this may be because it avoids the problem of crowding samples near the center of the visualization.

Figures 10-12 show example visualizations of the faces data set. First we show a well-performing comparison method (CDA; Figure 10); it has arranged the faces well in terms of keeping images of the same person in a single area; however, the areas of each person are diffuse and close to other persons, hence there is no strong separation between persons on the display. NeRV, here optimized to maximize precision, makes clearly tighter clusters of each person (Figure 11), which yields better retrieval of neighbor face images. However, NeRV has here placed several persons close to each other in the center of the visualization. The t-NeRV visualization, again optimized to maximize precision (Figure 12) has lessened this behavior, placing the clusters of faces more evenly.

Overall, t-NeRV is a useful alternative formulation of NeRV, and may be useful for data sets especially where crowding near the center of the visualization is an issue.

5. Using NeRV for Supervised Visualization

In this section we show how to use NeRV for supervised visualization. The key idea is simple: NeRV can be computed based on any input-space distances $d(\mathbf{x}_i, \mathbf{x}_j)$, not only the standard Euclidean distances. All that is required for supervised visualization is to compute the input-space distances in a supervised manner. The distances are then plugged into the NeRV algorithm and the visualization proceeds as usual. Note that doing the visualization modularly in two steps is an advantage, since it will be later possible to easily change the algorithm used in either step if desired.

Conveniently, rigorous methods exist for learning a supervised metric from labeled data samples. Learning of supervised metrics has recently been extensively studied for classification purposes and for some semi-supervised tasks, with both simple linear approaches and complicated nonlinear ones; see, for instance, works by Xing et al. (2003), Chang and Yeung (2004), Globerson and Roweis (2006) and Weinberger et al. (2006). Any such metric can in principle be used to compute distances for NeRV. Here we use an early one, which is flexible and can be directly plugged in the NeRV, namely the *learning metric* (Kaski et al., 2001; Kaski and Sinkkonen, 2004; Peltonen et al., 2004) which was originally proposed for data exploration tasks.

We will call NeRV computed with the supervised distances "supervised NeRV" (SNeRV). The information retrieval interpretation of NeRV carries over to SNeRV. Under certain parameter set-

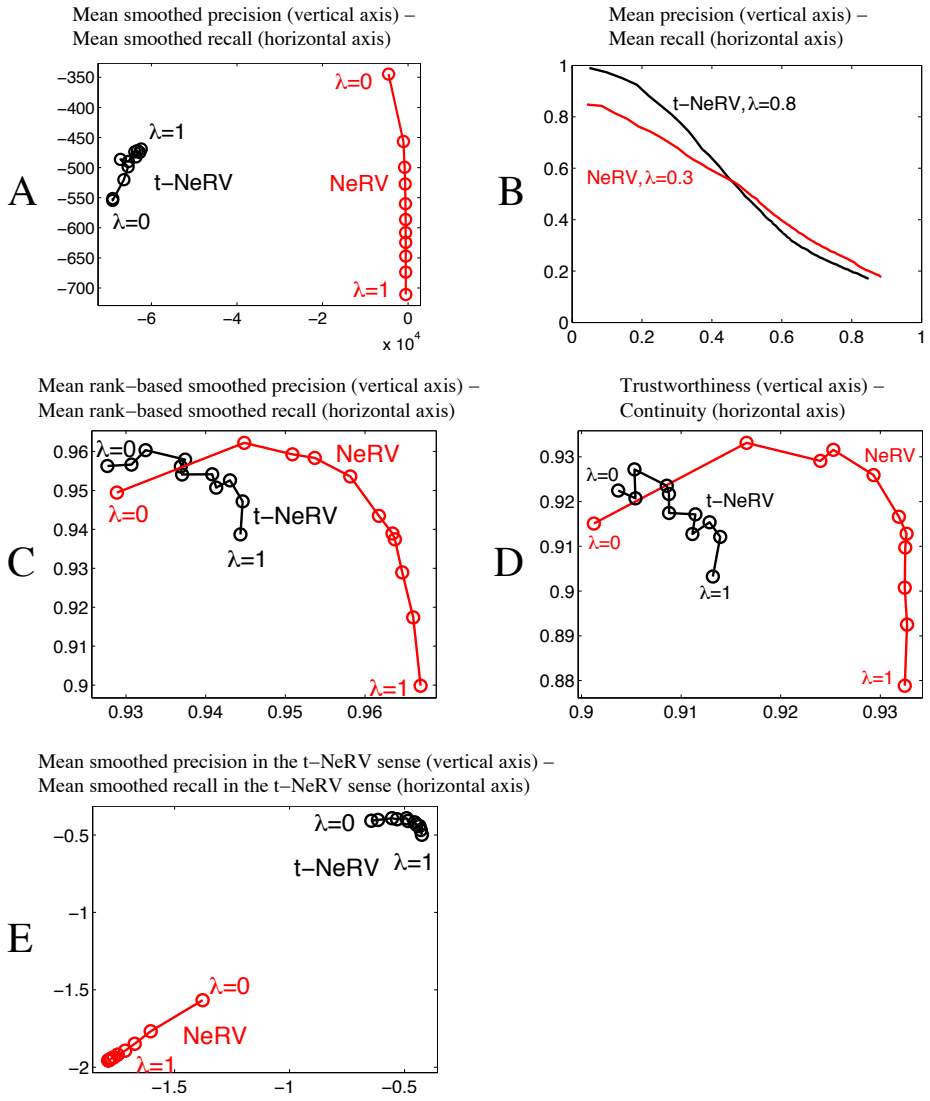


Figure 9: Comparison of NeRV and t-NeRV on the Faces data set according to the four goodness measures described in Section 4.3 (**A**-**D**), and for mean smoothed precision/recall corresponding to the information retrieval interpretation of t-NeRV (**E**; first and second terms of Eqn. 8).

tings SNeRV can be seen as a new, supervised version of stochastic neighbor embedding, but more generally it manages a flexible tradeoff between precision and recall of the information retrieval just like the unsupervised NeRV does.

SNeRV has the useful property that it can directly compute embeddings for unlabeled training points as well as labeled ones. By contrast, some supervised nonlinear dimensionality reduction methods (Geng et al., 2005; Liu et al., 2005; Song et al., 2008) only give embeddings for labeled points; for unlabeled points, the mapping is approximated for instance by interpolation or by training

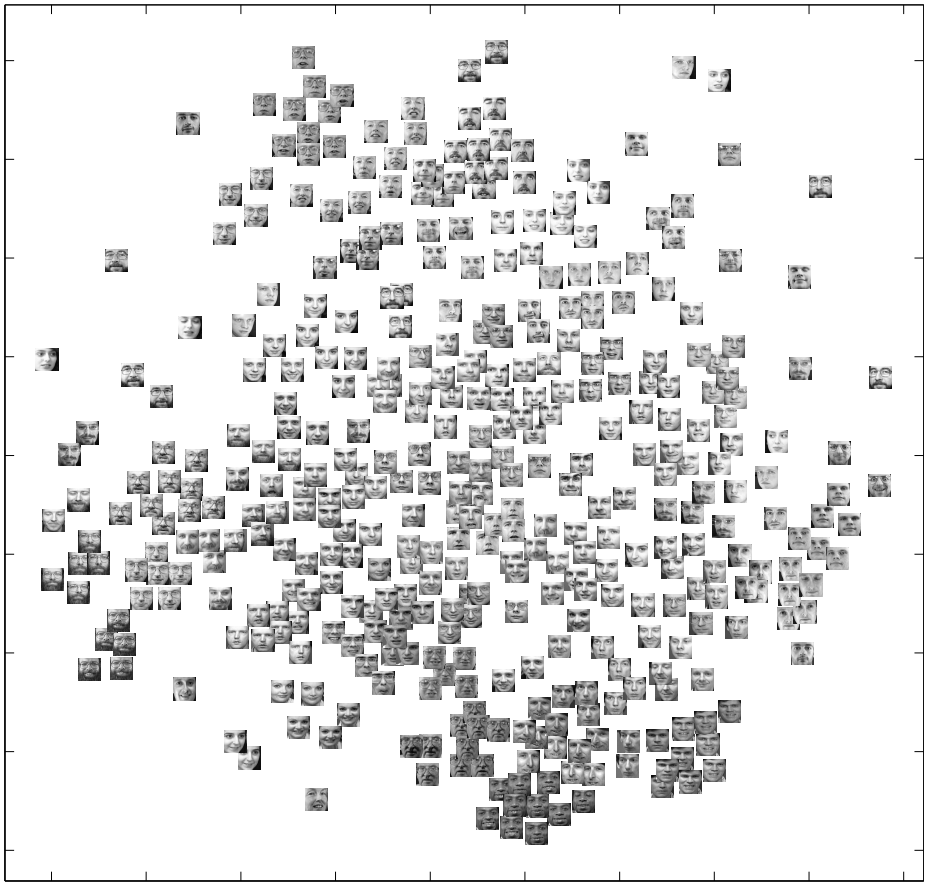


Figure 10: Example visualization of the Faces data set with CDA.

a neural network. Such approximation is not needed for SNeRV. (On the other hand, a trained neural network can embed not only unlabeled training points, but also previously unseen new points; if such generalization is desired, the same kinds of approximate mappings can be learned for SNeRV.)

In the next subsections we present the details of the distance computation, and then describe experimental comparisons showing that SNeRV outperforms several existing supervised methods.

5.1 Supervised Distances for NeRV

The input-space distances for SNeRV are computed using *learning metrics* (Kaski et al., 2001; Kaski and Sinkkonen, 2004; Peltonen et al., 2004). It is a formalism particularly suited for so-called "supervised unsupervised learning" where the final goal is still to make discoveries as in unsupervised learning, but the metric helps to focus the analysis by emphasizing useful features and, moreover, does that locally, differently for different samples. Learning metrics have previously been applied to clustering and visualization.

In brief, the learning metric is a Riemannian topology-preserving metric that measures distances in terms of changes in the class distribution. The class distribution is estimated through

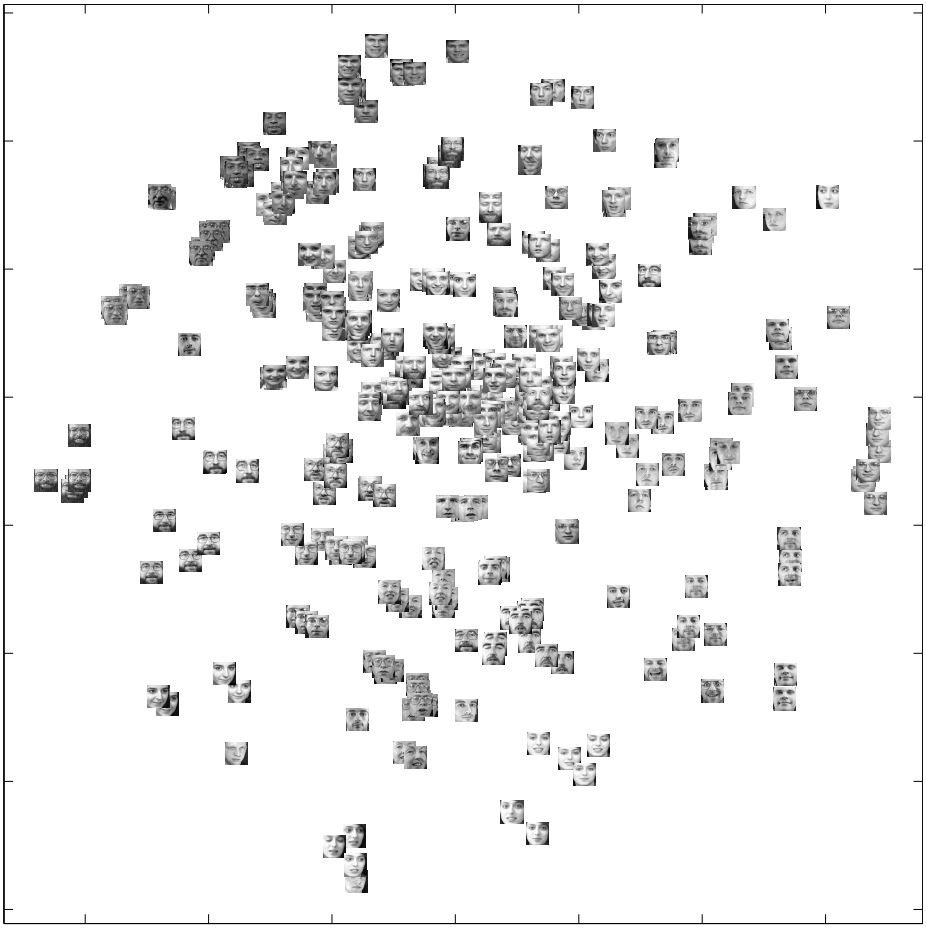


Figure 11: Example visualization of the Faces data set with NeRV, here maximizing precision (tradeoff parameter $\lambda = 0$).

conditional density estimation from labeled samples. Topology preservation helps in generalizing to new points, since class information cannot override the input space topology. In this metric, we can compute input-space distances between any two data points, and hence visualize the points with NeRV, whether they have known labels or not.

5.1.1 DEFINITION

The learning metric is a so-called Riemannian metric. Such a metric is defined in a local manner; between two (infinitesimally) close-by points it has a simple form, and this simple form is extended through path integrals to global distances.

In the learning metric, the squared distance between two close-by points $\mathbf{x}_1$ and $\mathbf{x}_2$ is given by the quadratic form

$$d_L(\mathbf{x}_1, \mathbf{x}_2)^2 = (\mathbf{x}_1 - \mathbf{x}_2)^T \mathbf{J}(\mathbf{x}_1)(\mathbf{x}_1 - \mathbf{x}_2). \tag{9}$$

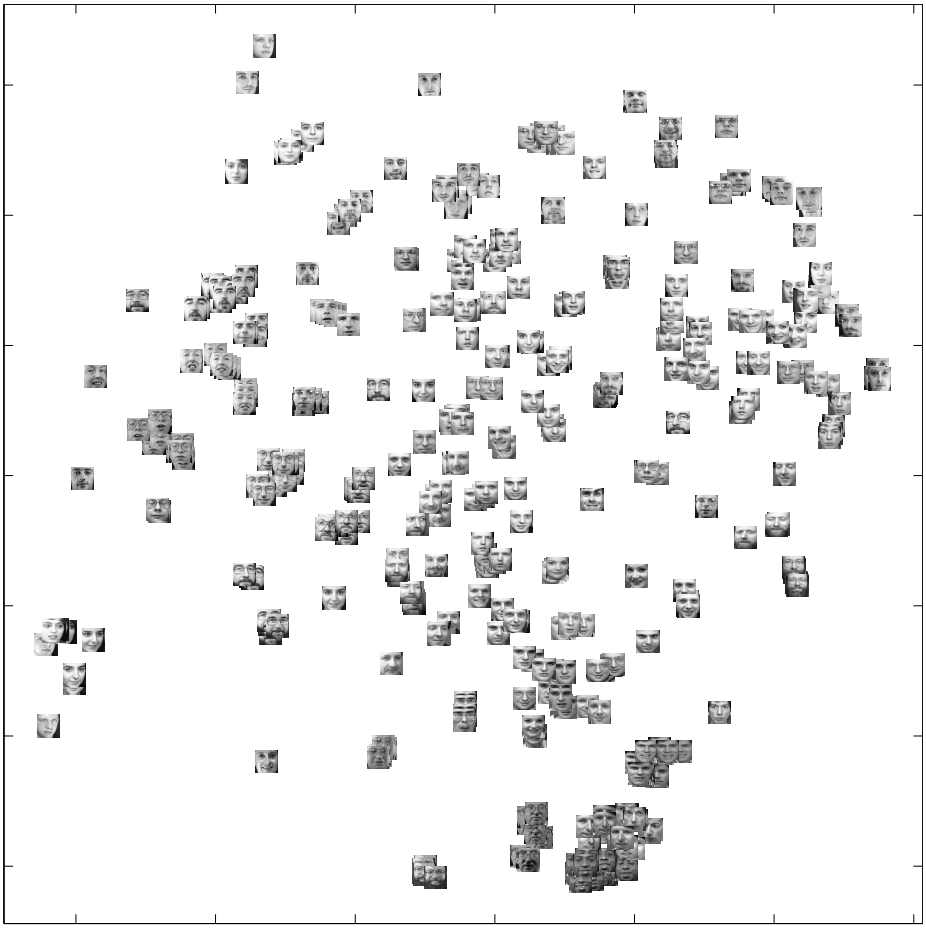


Figure 12: Example visualization of the Faces data set with t-NeRV, here maximizing precision (tradeoff parameter $\lambda = 0$).

Here $\mathbf{J}(\mathbf{x})$ is the Fisher information matrix which describes the local dependency of the conditional class distribution on the input features, that is,

$$\mathbf{J}(\mathbf{x}) = \sum_c p(c|\mathbf{x}) \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right)^T .$$

Here the c are the classes and the $p(c|\mathbf{x})$ are the conditional class probabilities at point $\mathbf{x}$. The idea is that the local distances grow the most along directions where the conditional class distribution $p(c|\mathbf{x})$ changes the most. It can be shown that the quadratic form (9) is, for close-by points, equivalent to the Kullback-Leibler divergence $D(p(c|\mathbf{x}_1), p(c|\mathbf{x}_2))$.

The general distance $d_L(\mathbf{x}_1, \mathbf{x}_2)$ between two far-away points $\mathbf{x}_1$ and $\mathbf{x}_2$ is defined in the standard fashion of Riemannian metrics: the distance is the *minimal path integral* over local distances, where the minimum is taken over all possible paths connecting $\mathbf{x}_1$ and $\mathbf{x}_2$. Notice that in a Riemannian metric, the straight path may not yield the minimum distance.

Learning metrics defined in the above manner satisfy the three criteria required of any metric: the distances d_L are nonnegative, symmetric, and satisfy the triangle inequality. Because the learning metric distances are defined as minimal path integrals they preserve the topology of the input space; roughly speaking, if the distance between two points is small, then there must be a path between them where distances are small along the entire path.

5.1.2 PRACTICAL COMPUTATION

In order to compute local distances using the Fisher information matrices $\mathbf{J}(\mathbf{x})$, we need an estimate for the conditional probability distributions $p(c|\mathbf{x})$. We learn the distributions by optimizing a discriminative mixture of labeled Gaussian densities for the data (Peltonen et al., 2004). The conditional density estimate is of the form

$$\hat{p}(c|\mathbf{x}) = \frac{\sum_{k=1}^{K} \beta_{ck} \exp(-||\mathbf{x}-\mathbf{m}_k||^2/2\sigma^2)}{\sum_{k=1}^{K} \exp(-||\mathbf{x}-\mathbf{m}_k||^2/2\sigma^2)} \tag{10}$$

where the number of Gaussians K, the centroids $\mathbf{m}_k$, the class probabilities β_{ck} and the Gaussian width σ (standard deviation) are parameters of the estimate; we require that the β_{ck} are nonnegative and that $\sum_c \beta_{ck} = 1$ for all k. The $\mathbf{m}_k$ and β_{ck} are optimized by a conjugate gradient algorithm to maximize the conditional class likelihood, and K and σ are chosen by internal cross-validation (see Section 5.3).

Given the Fisher matrices, we next need to compute the global distances between all point pairs. In most cases the minimal path integrals in the global distance definition cannot be computed analytically, and we use a graph-based approximation. We first form a fully connected graph between all known data points, where the path between each pair of points is approximated by a straight line. For these straight paths, the path integral can be computed by piecewise approximation (see Peltonen et al., 2004, for details; we use $T = 10$ pieces in all experiments). We could then use graph search (Floyd's algorithm) to find the shortest paths in the graph and use the shortest path distances as the learning metric distances. This graph approximation would take $O(n^3)$ time where n is the number of data points; note that this would not be excessive since a similar graph computation is needed in methods like isomap. However, in our experiments the straight line paths yielded about equally good results, so we simply use them, which takes only $O(n^2)$ time. Therefore SNeRV as a whole took only $O(n^2)$ time just like NeRV.

5.2 Comparison Methods for Supervised Visualization

For each data set to be visualized, the choice of supervised vs. unsupervised visualization is up to the analyst; in general, supervised embedding will preserve differences between classes better but at the expense of within-class details. In the experiments of this section we concentrate on comparing performances of supervised methods; we will compare SNeRV to three recent supervised nonlinear embedding methods.

Multiple relational embedding (MRE; Memisevic and Hinton, 2005) was proposed as an extension of stochastic neighbor embedding (Hinton and Roweis, 2002). MRE minimizes a sum of mismatches, measured by Kullback-Leibler divergence, between neighborhoods in the embedding and several different input neighborhoods: typically one of the input neighborhoods is derived from the input-space coordinates and the others are derived from auxiliary information such as labels.

MRE is able to use unlabeled data; for unlabeled points, divergences that involve neighborhoods based on labels are simply left out of the cost function.

Colored maximum variance unfolding (Song et al., 2008) is an extension of the unsupervised maximum variance unfolding. It maximizes the dependency between the embedding coordinates and the labels according to the Hilbert-Schmidt independence criterion, which is based on a cross-covariance operator. This leads to constrained optimization of the output kernel. Because of these details the method is also called maximum unfolding via Hilbert-Schmidt independence criterion (MUHSIC); we use this abbreviation.

Supervised isomap (S-Isomap; Geng et al., 2005) is an extension of the unsupervised isomap. The only difference to unsupervised isomap is a new definition of the input-space distances: roughly speaking, distances between points in different classes will grow faster than distances between same-class points. The actual embedding is done in the same way as in unsupervised isomap (described in Section 4.1). Other supervised extensions of isomap have been introduced by Li and Guo (2006) and Gu and Xu (2007).

Parametric embedding (PE; Iwata et al., 2007) represents the embedded data with a Gaussian mixture model with all Gaussians having the same covariances in the embedding space, and attempts to preserve the topology of the original data by minimizing a sum of Kullback-Leibler divergences.

Neighbourhood component analysis (NCA; Goldberger et al., 2005; see also Kaski and Peltonen, 2003, Peltonen and Kaski, 2005) is a linear and non-parametric dimensionality reduction method which learns a Mahalanobis distance measure such that, in the transformed space, k-nearest neighbor classification achieves the maximum accuracy.

5.3 Methodology for the Supervised Experiments

We used the four benchmark data sets having class information (Letter, Phoneme, Landsat, and TIMIT described in Section 4.5) to compare supervised NeRV and the five supervised visualization methods described in Section 5.2, namely multiple relational embedding (MRE), colored maximum variance unfolding (MUHSIC), supervised isomap (S-Isomap), parametric embedding (PE), and neighbourhood component analysis (NCA). We used a standard 10-fold cross-validation setup: in each fold we reserve one of the subsets for testing and use the rest of the data for training. For each data set, we use SNeRV and the comparison methods to find 2-dimensional visualizations.

In principle we could evaluate the results as in Section 4.3 for the unsupervised experiments, that is, by mean smoothed precision and recall; the only difference would be to use the supervised learning metric for the evaluation. However, unlike SNeRV, the other methods have not been formulated using the same supervised metrics. To make an unbiased comparison of the methods, we resort to a simple indirect evaluation: we evaluate the performance of the four methods by class prediction accuracy of the resulting visualizations. Although it is an indirect measure, the accuracy is a reasonable choice for unbiased comparison and has been used in several supervised dimensionality reduction papers. In more detail, we provide test point locations during training but not their labels; after the methods have computed their visualization results, we classify the test points by running a k-nearest neighbor classifier ($k = 5$) on the embedded data, and evaluate the classification error rates of the methods.

We use a standard internal 10-fold validation strategy to choose all parameters which are not optimized by their respective algorithms: each training set is subdivided into 10 folds where 9/10 of data is used for learning and 1/10 for validation; we learn visualizations with the different parameter

values; the values that yielded the best classification accuracy for the embedded validation points are then chosen and used to compute the final visualization for the whole training data.

We ran two versions of SNeRV using $\lambda = 0.1$ and $\lambda = 0.3$. The scaling parameters σ_i were set by fixing the entropy of the input neighborhoods as described in Section 2.2. Here we specified the rough upper limit for the number of relevant neighbors as $0.5 \cdot n/K$ where n is the number of data points and K is the number of mixture components used to estimate the metric; this choice roughly means that for well-separated mixture components, each data point will on average consider half of the data points from the same mixture component as relevant neighbors. A simplified validation sufficed for the number K and width σ of Gaussians: we did not need to run the embedding step but picked the values that gave best conditional class likelihood for validation points in the input space. For S-Isomap we chose its parameter α and its number of nearest neighbors using the validation sets, and trained a generalized radial basis function network to project new points, as suggested by Geng et al. (2005). For MUHSIC, the parameters are the regularization parameter ν, number of nearest neighbors, and number of eigenvectors in the graph Laplacian, and we used linear interpolation to project new points as suggested by the MUHSIC authors. For MRE the only free parameter is its neighborhood smoothness parameter σ_{MRE}. For the PE one needs to provide a conditional density estimate: we used the same one that SNeRV uses (see Equation 10) to obtain a comparison as unbiased as possible. Neighbourhood component analysis is a non-parametric method, therefore we did not need to choose any parameters for it.

5.4 Results of Supervised Visualization

Figure 13 shows the average error rate over the 10 folds as well as the standard deviation. The best two methods are SNeRV and PE, which give good results in all data sets. On two of the data sets (Letter and TIMIT) SNeRV is clearly the best; on the other two data sets (Phoneme and Landsat) SNeRV is about as good as the best of the remaining methods (S-Isomap and parametric embedding, respectively). MRE is clearly worse than the other methods, whereas MUHSIC and NCA results depend on the data set: on Letter they are the second and third worst methods after MRE, while in the other data sets they are not far from the best methods.

The value of the tradeoff parameter λ did not affect the performance of SNeRV much; both $\lambda = 0.1$ and $\lambda = 0.3$ produced good projections.

To evaluate whether the best method on each data set is statistically significantly better than the next best one, we performed a paired t-test of the performances across the 10 cross-validation folds (Table 3). The two best methods compared are always SNeRV and parametric embedding, except for the Phoneme data set for which the two best methods are SNeRV and S-Isomap. For the Letter and the TIMIT data sets SNeRV is significantly better than the next best method, while for the other two data sets the difference is not significant. In summary, all the significant differences are in favor of SNeRV.

Figure 14 presents sample visualizations of the letter recognition data set; projection results are shown for one of the 10 cross-validation folds, including both training and test points. Although there is some overlap, in general SNeRV shows distinct clusters of classes—for example, the letter "M" is a well separated cluster in the top of the figure.

Parametric embedding also manages to separate some letters, such as "A" and "I", but there is a severe overlap of classes in the center of the figure. In S-Isomap we see that there are a few very well separated clusters of classes, like the letters "W" and "N", but there is a large area with

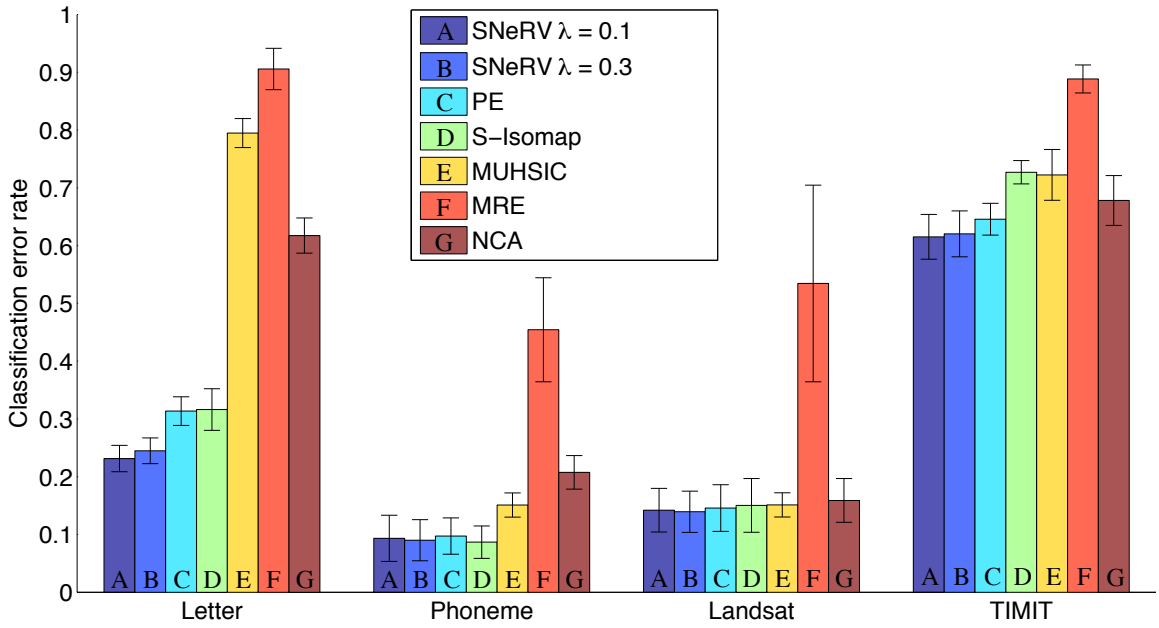


Figure 13: Performance of the supervised nonlinear embedding methods in each benchmark data set. The results are average classification error rates over 10 cross-validation folds (smaller is better), and the standard deviations are shown with error bars.

Data set	Best method	Second best	p-value
Letter	**SNeRV ($\lambda = 0.1$)**	PE	$1.8 \cdot 10^{-6}$
Phoneme	S-Isomap	SNeRV ($\lambda = 0.3$)	0.54
Landsat	SNeRV ($\lambda = 0.3$)	PE	0.28
TIMIT	**SNeRV ($\lambda = 0.1$)**	PE	$3.4 \cdot 10^{-3}$

Table 3: Statistical significance of the difference between the two best methods. The p-values are from a paired t-test of the 10-fold cross-validation results; statistically significant winners have been boldfaced.

overlapping classes near the center of the right edge of the figure. This overlap is worse than in SNeRV but still roughly comparable; by contrast, MUHSIC, MRE and NCA performed poorly on this data set, leaving most classes severely overlapped.

6. Conclusions and Discussion

By formulating the task of nonlinear projection for information visualization as an information retrieval task, we have derived a rigorously motivated pair of measures for visualization performance, *mean smoothed precision* and *mean smoothed recall*. We showed that these new measures are extensions of two traditional information retrieval measures: mean smoothed precision can be interpreted as a more sophisticated extension of mean precision, the proportion of false positives in the neigh-

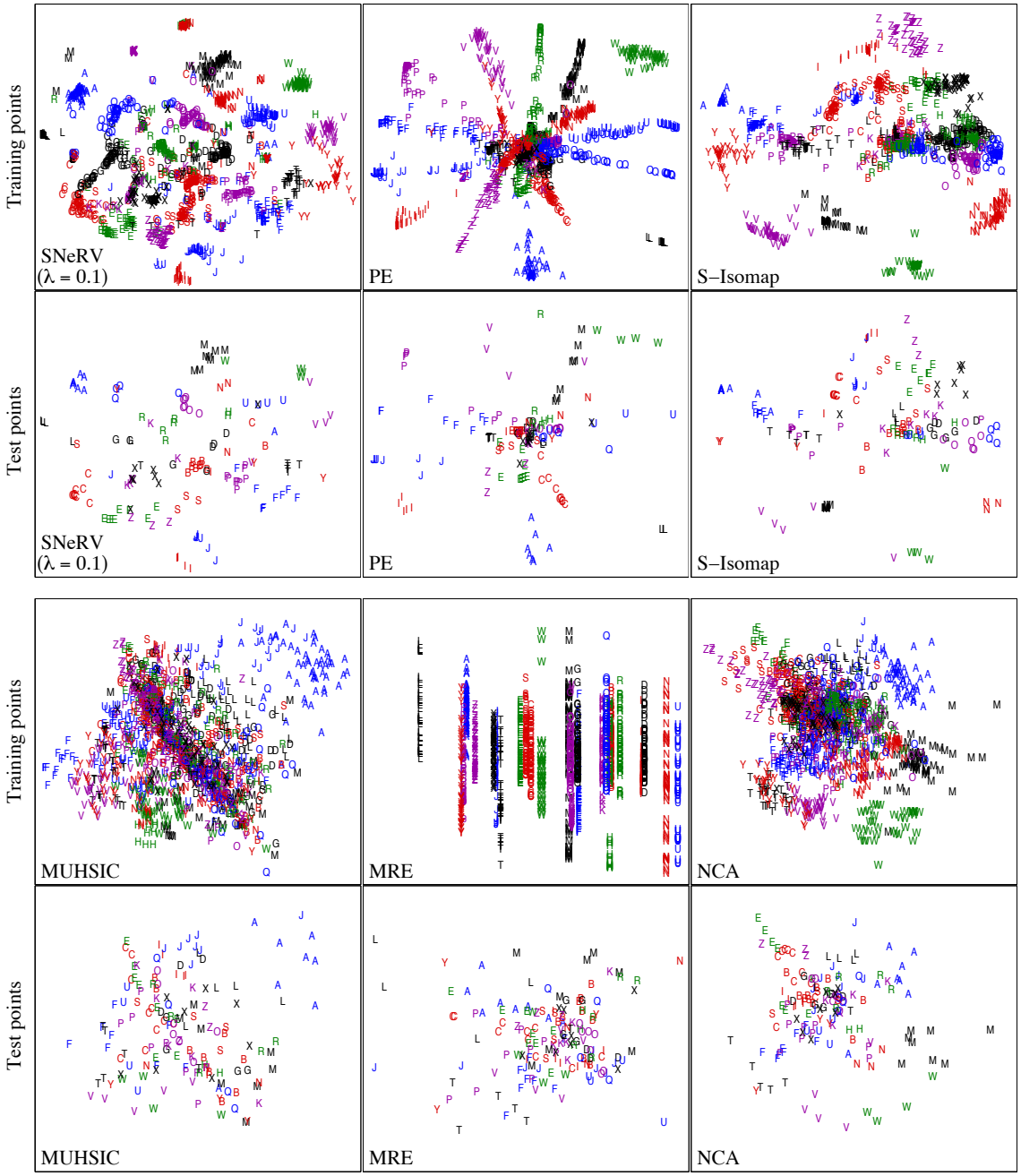


Figure 14: Visualizations of the letter recognition data set by all supervised methods.

borhood retrieved from the visualization. Analogously, mean smoothed recall is an extension of mean recall, the proportion of misses incurred by the retrieved neighborhood.

We introduced an algorithm called *neighbor retrieval visualizer* (NeRV) that optimizes the total cost, interpretable as a tradeoff between mean smoothed precision and mean smoothed recall. The tradeoff is governed by a parameter λ set by the user according to the desired cost of a miss relative

to the desired cost of a false positive. The earlier method stochastic neighbor embedding is obtained as a special case when $\lambda = 1$, optimizing mean smoothed recall.

We showed that NeRV can be used for both unsupervised and supervised visualization. For unsupervised visualization, we simply use fixed input distances; for supervised visualization we learn a supervised distance metric for the input space and plug the resulting input distances to the NeRV algorithm. In the latter case the key idea is to use supervision (labeled data) in a way that does not override the input feature space; we use a topology-preserving class-discriminative metric called the *learning metric* for the input space.

In unsupervised visualization, NeRV outperformed alternatives for most of the six data sets we tried, for four different pairs of measures, and was overall the best method. NeRV also performed well in a comparison by unsupervised classification. Many of the best manifold extraction methods perform surprisingly poorly, most likely because they have not been designed to reduce the dimensionality below the intrinsic dimensionality of the data manifold. In visualization, however, we generally have no choice but to reduce the dimensionality of the data to two or three, even if its intrinsic dimensionality is higher. NeRV is designed to find a mapping that is, in a well-defined sense, optimal for a certain type of visualization regardless of the intrinsic dimensionality of the data.

In supervised visualization, the supervised version of NeRV performed as well as or better than the best alternative method Parametric embedding; this shows that the plug-in learning metrics work well in incorporating supervision.

6.1 Discussion

NeRV models relevance using probability distributions, which makes sense if the total "amount" of relevance for any query is normalized to a fixed sum. Such normalization is desirable for any relevance measure, because for any query (point of interest) the relevance of a retrieved neighbor point should depend on its proximity relative to the proximities of other points, rather than on its absolute distance from the query point. (Our previous method, local MDS, can be thought of as an attempt to approximate NeRV without the normalization.)

The Kullback-Leibler divergences in NeRV are natural choices for measuring the difference between two probability distributions, but in principle other divergence measures could be used as well. The notions of neighbor retrieval and a probabilistic relevance model are the crucial parts of NeRV, not the specific divergence measure.

Our notion of plug-in supervised metrics could in principle be used with other methods too; other unsupervised embedding algorithms that work based on a distance matrix can also be turned into supervised versions, by plugging in learning metric distances into the distance matrix. We performed an initial experiment with Sammon's mapping (Peltonen et al., 2004); a similar idea for isomap appeared later (Weng et al., 2005). However, we believe that NeRV is an especially attractive choice for the embedding step since it has the information retrieval interpretation and it performed well empirically.

An implementation of the NeRV and local MDS algorithms as well as the *mean smoothed precision-mean smoothed recall* measures is available at `http://www.cis.hut.fi/projects/mi/software/dredviz`

Acknowledgments

The authors belong to the Adaptive Informatics Research Centre, a national CoE of the Academy of Finland, and to Helsinki Institute for Information Technology HIIT. Jarkko Venna is currently at Numos Ltd. JP was supported by the Academy of Finland, decision number 123983, and HA by the Portuguese Foundation for Science and Technology, scholarship number SFRH/BD/39642/2007. This work was also supported in part by the PASCAL2 Network of Excellence, ICT 216886.

Appendix A. Proof of the Connection between the Probabilistic Cost Functions and Precision and Recall

In Section 2.2 we introduced Kullback-Leibler divergences as cost functions for visual neighbor retrieval, based on probability distributions q_i and p_i which generalize the relevance model implicit in precision and recall. We will next show that in the simple case of "binary neighborhoods" the cost functions reduce to precision and recall. By "binary neighborhoods" we mean that, in both the input space and the visualization, (i) the point of interest has some number of relevant neighbors and all the other points are completely irrelevant, and (ii) the points that are relevant are all equally relevant.

In the probabilistic model the binary neighborhoods can be interpreted as follows. Let i be the point of interest, and let P_i be the set of relevant neighbors for point i in the input space. P_i can be the set of all points (other than i itself) falling inside some fixed radius from point i in the input space, or it can be the set containing some fixed number of points nearest to i in the input space. In either case, let r_i be the size of P_i.

We define that the relevant neighbors of the point of interest i have an equal non-zero probability of being chosen, and all the other points have a near-zero probability of being chosen. In other words, we define

$$p^*_{j|i} = \begin{cases} a_i \equiv \frac{1-\delta}{r_i}\,, & \text{if point } j \text{ is in } P_i \\ b_i \equiv \frac{\delta}{N-r_i-1}\,, & \text{otherwise.} \end{cases}$$

Here N is the total number of data points, and $0 < \delta \ll 0.5$ gives the irrelevant points a very small probability.

Similarly, let Q_i be the set of neighbors for point i in the visualization. Again, Q_i can be the set of all points (other than i itself) falling inside some fixed radius from point i in the visualization, or it can be the set containing some fixed number of points nearest to i in the visualization. In either case, let k_i be the size of Q_i. Note that the sizes of Q_i and P_i can be different, that is, k_i can be different from r_i.

We define the probability of choosing a neighbor from the visualization as

$$q^*_{j|i} = \begin{cases} c_i \equiv \frac{1-\delta}{k_i}\,, & \text{if point } j \text{ is in } Q_i \\ d_i \equiv \frac{\delta}{N-k_i-1}\,, & \text{otherwise.} \end{cases}$$

Consider the Kullback-Leibler divergence $D(p^*_i, q^*_i)$ for any fixed i. We now show that minimizing this divergence is equivalent to maximizing recall where point i is the query. The divergence is a sum over elements $p^*_{j|i} \log \frac{p^*_{j|i}}{q^*_{j|i}}$, thus the sum can be divided into four parts depending on which

value $p^*_{j|i}$ takes (two possibilities) and which value $q^*_{j|i}$ takes (two possibilities). We get

$$\begin{aligned}
D(p^*_i, q^*_i) = &\sum_{j\neq i, p^*_{j|i}=a_i, q^*_{j|i}=c_i} \left(a_i \log \frac{a_i}{c_i}\right) + \sum_{j\neq i, p^*_{j|i}=a_i, q^*_{j|i}=d_i} \left(a_i \log \frac{a_i}{d_i}\right) \\
&+ \sum_{j\neq i, p^*_{j|i}=b_i, q^*_{j|i}=c_i} \left(b_i \log \frac{b_i}{c_i}\right) + \sum_{j\neq i, p^*_{j|i}=b_i, q^*_{j|i}=d_i} \left(b_i \log \frac{b_i}{d_i}\right) \\
&= \left(a_i \log \frac{a_i}{c_i}\right) N_{\mathrm{TP},i} + \left(a_i \log \frac{a_i}{d_i}\right) N_{\mathrm{MISS},i} + \left(b_i \log \frac{b_i}{c_i}\right) N_{\mathrm{FP},i} + \left(b_i \log \frac{b_i}{d_i}\right) N_{\mathrm{TN},i}
\end{aligned}$$

where on the last line the terms inside parentheses are simply constant coefficients. Here $N_{\mathrm{TP},i}$ is the number of true positives for this query, that is, points for which the probability is high in both the data and the visualization. The number of misses, that is, the number of points that have a low probability in the visualization although the probability in the data is high, is $N_{\mathrm{MISS},i}$. The number of false positives (high probability in the visualization, low in the data) is $N_{\mathrm{FP},i}$. Finally the number of true negatives (low probability in both the visualization and the data) is $N_{\mathrm{TN},i}$.

It is straightforward to check that if δ is very small, then the coefficients for the misses and false positives dominate the divergence. This yields

$$\begin{aligned}
D(p^*_i, q^*_i) &\approx N_{\mathrm{MISS},i} \left(a_i \log \frac{a_i}{d_i}\right) + N_{\mathrm{FP},i} \left(b_i \log \frac{b_i}{c_i}\right) \\
&= N_{\mathrm{MISS},i} \frac{1-\delta}{r_i} \left(\log \frac{(N-k_i-1)}{\delta} + \log \frac{(1-\delta)}{r_i}\right) \\
&+ N_{\mathrm{FP},i} \frac{\delta}{N-r_i-1} \left(\log \frac{\delta}{N-r_i-1} - \log \frac{(1-\delta)}{k_i}\right) \\
&= N_{\mathrm{MISS},i} \frac{1-\delta}{r_i} \left(\log \frac{(N-k_i-1)}{r_i} + \log \frac{(1-\delta)}{\delta}\right) \\
&\qquad + N_{\mathrm{FP},i} \frac{\delta}{N-r_i-1} \left(\log \frac{k_i}{N-r_i-1} - \log \frac{(1-\delta)}{\delta}\right) . \quad (11)
\end{aligned}$$

Because the terms $\log[(1-\delta)/\delta]$ dominate the other logarithmic terms, (11) further simplifies to

$$\begin{aligned}
D(p^*_i, q^*_i) &\approx \left(N_{\mathrm{MISS},i} \frac{1-\delta}{r_i} - N_{\mathrm{FP},i} \frac{\delta}{N-r_i-1}\right) \log \frac{(1-\delta)}{\delta} \\
&\approx N_{\mathrm{MISS},i} \frac{1-\delta}{r_i} \log \frac{(1-\delta)}{\delta} = \frac{N_{\mathrm{MISS},i}}{r_i} C
\end{aligned}$$

where C is a constant that only depends on δ and not on i. Hence if we minimized this cost function, we would be maximizing the recall of the query, which is defined as

$$\mathrm{recall}(i) = \frac{N_{\mathrm{TP},i}}{r_i} = 1 - \frac{N_{\mathrm{MISS},i}}{r_i} .$$

We can analogously show that for any fixed i, minimizing $D(q^*_i, p^*_i)$ is equivalent to maximizing precision of the corresponding query.

Because $D(q^*_i, p^*_i)$ and $D(p^*_i, q^*_i)$ are equivalent to precision and recall, and p_i and q_i can be seen as more sophisticated generalizations of p^*_i and q^*_i, we interpret $D(q_i, p_i)$ and $D(p_i, q_i)$ as more sophisticated generalizations of precision and recall.

References

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, Cambridge, MA, 2002a.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. Technical Report TR-2002-01, Department of Computer Science,The University of Chicago, 2002b.

Mira Bernstein, Vin de Silva, John C. Langford, and Joshua B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Department of Psychology, Stanford University, 2000.

Catherine L. Blake and C. J. Merz. UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 1998.

Ingwer Borg and Patrick Groenen. *Modern Multidimensional Scaling*. Springer, New York, 1997.

Hong Chang and Dit-Yan Yeung. Locally linear metric adaptation for semi-supervised clustering. In *Proceedings of the Twenty-first International Conference on Machine Learning 2004*, pages 153–160. Omnipress, Madison, WI, 2004.

Lisha Chen and Andreas Buja. Local multidimensional scaling for nonlinear dimension reduction, graph drawing and proximity analysis. *Journal of the American Statistical Association*, 104(485): 209–219, 2009.

Pierre Demartines and Jeanny Hérault. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 8(1):148–154, 1997.

David L. Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100:5591–5596, 2003.

Xin Geng, De-Chuan Zhan, and Zhi-Hua Zhou. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, 35:1098–1107, 2005.

Amir Globerson and Sam Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing 18*, pages 451–458. MIT Press, Cambridge, MA, 2006.

Jacob Goldberger, Sam Roweis, Geoffrey Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, Cambridge, MA, 2005.

John C. Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–338, 1966.

Rui-jun Gu and Wen-bo Xu. Weighted kernel Isomap for data visualization and pattern classification. In Y. Wang, Y.-m. Cheung, and H. Liu, editors, *Computational Intelligence and Security (CIS 2006)*, pages 1050–1057. Springer-Verlag, Berlin Heidelberg, 2007.

Geoffrey Hinton and Sam T. Roweis. Stochastic neighbor embedding. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural information processings systems 14*, pages 833–840. MIT Press, Cambridge, MA, 2002.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441,498–520, 1933.

Tomoharu Iwata, Kazumi Saito, Naonori Ueda, Sean Stromsten, Thomas L. Griffiths, and Joshua B. Tenenbaum. Parametric embedding for class visualization. *Neural Computation*, 19:2536–2556, 2007.

Samuel Kaski and Jaakko Peltonen. Informative discriminant analysis. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pages 329–336. AAAI Press, Menlo Park, CA, 2003.

Samuel Kaski and Janne Sinkkonen. Principle of learning metrics for exploratory data analysis. *Journal of VLSI Signal Processing, special issue on Machine Learning for Signal Processing*, 37: 177–188, 2004.

Samuel Kaski, Janne Sinkkonen, and Jaakko Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12:936–947, 2001.

Samuel Kaski, Janne Nikkilä, Merja Oja, Jarkko Venna, Petri Törönen, and Eero Castrén. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, 4:48, 2003.

Teuvo Kohonen, Jussi Hynninen, Jari Kangas, Jorma Laaksonen, and Kari Torkkola. LVQ_PAK: The learning vector quantization program package. Technical Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science, FIN-02150 Espoo, Finland, 1996.

Joseph B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrica*, 29(1):1–27, 1964.

John Aldo Lee, Amaury Lendasse, Nicolas Donckers, and Michel Verleysen. A robust nonlinear projection method. In M. Verleysen, editor, *ESANN'2000, Eighth European Symposium on Artificial Neural Networks*, pages 13–20. D-Facto Publications, Bruges, Belgium, 2000.

John Aldo Lee, Amaury Lendasse, and Michel Verleysen. Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. *Neurocomputing*, 57:49–76, 2004.

Chun-Guang Li and Jun Guo. Supervised Isomap with explicit mapping. In J.-S. Pan, P. Shi, and Y. Zhao, editors, *Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06)*, volume 3, pages 345–348. IEEE, 2006.

E. Liitiäinen and A. Lendasse. Variable scaling for time series prediction: Application to the ESTSP'07 and the NN3 forecasting competitions. In *IJCNN 2007, International Joint Conference on Neural Networks*, pages 2812–2816. IEEE, Piscataway, NJ, 2007.

Ning Liu, Fengshan Bai, Jun Yan, Benyu Zhang, Zheng Chen, and Wei-Ying Ma. Supervised semi-definite embedding for email data cleaning and visualization. In Y. Zhang, K. Tanaka, J. Xu Yu, S. Wang, and M. Li, editors, *Web Technologies Research and Development–APWeb 2005*, pages 972–982. Springer-Verlag, Berlin Heidelberg, 2005.

Roland Memisevic and Geoffrey Hinton. Multiple relational embedding. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 913–920. MIT Press, Cambridge, MA, 2005.

Jaakko Peltonen and Samuel Kaski. Discriminative components of data. *IEEE Transactions on Neural Networks*, 16(1):68–83, 2005.

Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved learning of Riemannian metrics for exploratory analysis. *Neural Networks*, 17:1087–1100, 2004.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

Eran Segal, Nir Friedman, Daphne Koller, and Aviv Regev. A module map showing conditional activity of expression modules in cancer. *Nature Genetics*, 36:1090–1098, 2004.

Blake Shaw and Tony Jebara. Minimum volume embedding. In M. Meila and X. Shen, editors, *Proceedings of AISTATS*07, the 11th International Conference on Artificial Intelligence and Statistics (JMLR Workshop and Conference Proceedings Volume 2)*, pages 460-467, 2007.

Le Song, Alex Smola, Kersten Borgwardt, and Arthur Gretton. Colored maximum variance unfolding. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1385–1392. MIT Press, Cambridge, MA, 2008.

Andrew I. Su, Michael P. Cooke, Keith A. Ching, Yaron Hakak, John R. Walker, Tim Wiltshire, Anthony P. Orth, Raquel G. Vega, Lisa M. Sapinoso, Aziz Moqrich, Ardem Patapoutian, Garret M. Hampton, Peter G. Schultz, and John B. Hogenesch. Large-scale analysis of the human and mouse transcriptomes. *Proceedings of the National Academy of Sciences*, 99:4465–4470, 2002.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, December 2000.

TIMIT. CD-ROM prototype version of the DARPA TIMIT acoustic-phonetic speech database, 1998.

Warren S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17:401–419, 1952.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

Laurens van der Maaten, Eric Postma, and Jaap van der Herik. Dimensionality reduction: A comparative review. Technical report 2009–005, Tilburg centre for Creative Computing, Tilburg University, Tilburg, The Netherlands, 2009.

Jarkko Venna. *Dimensionality Reduction for Visual Exploration of Similarity Structures*. PhD thesis, Helsinki University of Technology, Espoo, Finland, 2007.

Jarkko Venna and Samuel Kaski. Local multidimensional scaling. *Neural Networks*, 19:889–99, 2006.

Jarkko Venna and Samuel Kaski. Comparison of visualization methods for an atlas of gene expression data sets. *Information Visualization*, 6:139–154, 2007a.

Jarkko Venna and Samuel Kaski. Nonlinear dimensionality reduction as information retrieval. In M. Meila and X. Shen, editors, *Proceedings of AISTATS*07, the 11th International Conference on Artificial Intelligence and Statistics (JMLR Workshop and Conference Proceedings Volume 2)*, pages 572–579, 2007b.

Kilian Weinberger, Benjamin Packer, and Lawrence Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, pages 381–388. Society for Artificial Intelligence and Statistics, 2005. (Available electronically at `http://www.gatsby.ucl.ac.uk/aistats/`).

Kilian Weinberger, John Blitzer, and Lawrence Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1473–1480. MIT Press, Cambridge, MA, 2006.

Kilian Q. Weinberger and Lawrence K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70:77–90, 2006.

Kilian Q. Weinberger, Fei Sha, Qihui Zhu, and Lawrence K. Saul. Graph Laplacian regularization for large-scale semidefinite programming. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1489–1496. MIT Press, Cambridge, MA, 2007.

Shifeng Weng, Changshui Zhang, and Zhonglin Lin. Exploring the structure of supervised data by Discriminant Isometric Mapping. *Pattern Recognition*, 38:599–601, 2005.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*. MIT Press, Cambridge, MA, 2003.

Journal of Machine Learning Research 11 (2010) 491-516 Submitted 3/09; Revised 8/09; Published 2/10

Classification Using Geometric Level Sets

Kush R. Varshney KRV@MIT.EDU
Alan S. Willsky WILLSKY@MIT.EDU
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139-4309, USA

Editor: Ulrike von Luxburg

Abstract

A variational level set method is developed for the supervised classification problem. Nonlinear classifier decision boundaries are obtained by minimizing an energy functional that is composed of an empirical risk term with a margin-based loss and a geometric regularization term new to machine learning: the surface area of the decision boundary. This geometric level set classifier is analyzed in terms of consistency and complexity through the calculation of its ε-entropy. For multicategory classification, an efficient scheme is developed using a logarithmic number of decision functions in the number of classes rather than the typical linear number of decision functions. Geometric level set classification yields performance results on benchmark data sets that are competitive with well-established methods.

Keywords: level set methods, nonlinear classification, geometric regularization, consistency, complexity

1. Introduction

Variational level set methods, pioneered by Osher and Sethian (1988), have found application in fluid mechanics, computational geometry, image processing and computer vision, computer graphics, materials science, and numerous other fields, but have heretofore found little application in machine learning. The goal of this paper is to introduce a level set approach to the archetypal machine learning problem of supervised classification. We propose an implicit level set representation for classifier decision boundaries, a margin-based objective regularized by a surface area penalty, and an Euler-Lagrange descent optimization algorithm for training.

Several well-developed techniques for supervised discriminative learning exist in the literature, including the perceptron algorithm (Rosenblatt, 1958), logistic regression (Efron, 1975), and support vector machines (SVMs) (Vapnik, 1995). All of these approaches, in their basic form, produce linear decision boundaries. Nonlinear boundaries in the input space can be obtained by mapping the input space to a feature space of higher (possibly infinite) dimension by taking nonlinear functions of the input variables. Learning algorithms are then applied to the new higher-dimensional feature space by treating each dimension linearly. They retain the efficiency of the input lower-dimensional space for particular sets of nonlinear functions that admit the kernel trick (Schölkopf and Smola, 2002).

As an alternative to kernel methods for generalizing linear methods, we propose finding nonlinear decision boundaries directly in the input space. We propose an energy functional for clas-

sification that is composed of an empirical risk term that uses a margin-based loss function and a complexity term that is the length of the decision boundary for a two-dimensional input space and the surface area of the decision boundary more generally. The empirical risk term is standard in many classification methods. What is new in this work is the measurement of decision boundary complexity by surface area, an inherently geometric quantity, and the idea of using variational level set methods for optimization in discriminative learning.

We use the term *contour* to refer to a one-dimensional curve in a two-dimensional space, a two-dimensional surface in a three-dimensional space, and generally a $D-1$ dimensional hypersurface in a D-dimensional space. Classifier decision boundaries partition the input space into regions corresponding to the different class labels. If the region corresponding to one class label is composed of several unconnected pieces, then the corresponding decision boundary is composed of several unconnected pieces; we refer to this entire collection of decision boundaries as the contour. The level set representation is a flexible, implicit representation for contours that does not require knowing the number of disjoint pieces in advance. The contour is represented by a smooth, Lipschitz continuous, scalar-valued function, known as the *level set function*, whose domain is the input space. The contour is implicitly specified as the zero level set of the level set function.

Level set methods entail not only the representation, but also the minimization of an energy functional whose argument is the contour.[1] For example, in foreground-background image segmentation, a popular energy functional is mean squared error of image intensity with two different 'true' image intensities inside the contour and outside the contour. Minimizing this energy produces a good segmentation when the two regions differ in image intensity. In order to perform the minimization, a gradient descent approach is used. The first variation of the functional is found using the calculus of variations; starting from an initial contour, a gradient flow is followed iteratively to converge to a minimum. This procedure is known as *curve evolution* or *contour evolution*.

The connection between level set methods (particularly for image segmentation) and classification has been noticed before, but to the best of our knowledge, there has been little prior work in this area. Boczko et al. (2006) only hint at the idea of using level set methods for classification. Tomczyk and Szczepaniak (2005) do not consider fully general input spaces. Specifically, examples in the training and test sets must be pixels in an image with the data vector containing the spatial index of the pixel along with other variables. Cai and Sowmya (2007) do consider general feature spaces, but have a very different energy functional than our margin-based loss functional. Theirs is based on counts of training examples in grid cells and is similar to the mean squared error functional for image segmentation described earlier. Their learning is also based on one-class classification rather than standard discriminative classification, which is the framework we follow. Yip et al. (2006) use variational level set methods for density-based clustering in general feature spaces, rather than for learning classifiers.

Cremers et al. (2007) dichotomize image segmentation approaches into those that use spatially continuous representations and those that use spatially discrete representations, with level set methods being the main spatially continuous approaches. There have been methods using discrete representations that bear some ties to our methods. An example of a spatially discrete approach uses normalized graph cuts (Shi and Malik, 2000), a technique that has been used extensively in unsupervised learning for general features unrelated to images as well. Normalized decision boundary surface area is implicitly penalized in this discrete setting. Geometric notions of complexity in su-

1. In the image segmentation literature, variational energy minimization approaches often go by the name *active contours*, whether implemented using level set methods or not.

pervised classification tied to decision boundary surface area have been suggested by Ho and Basu (2002), but also defined in a discrete way related to graph cuts. In contrast, the continuous formulation we employ using level sets involves very different mathematical foundations, including explicit minimization of a criterion involving surface area. Moreover, the continuous framework—and in particular the natural way in which level set functions enter into the criterion—lead to new gradient descent algorithms to determine optimal decision boundaries. By embedding our criterion in a continuous setting, the surface area complexity term is defined intrinsically rather than being defined in terms of the graph of available training examples.

There are some other methods in the literature for finding nonlinear decision boundaries directly in the input space related to image segmentation, but these methods use neither contour evolution for optimization, nor the surface area of the decision boundary as a complexity term, as in the level set classification method proposed in this paper. A connection is drawn between classification and level set image segmentation in Scott and Nowak (2006) and Willett and Nowak (2007), but the formulation is through decision trees, not contour evolution. Tomczyk (2005), Tomczyk and Szczepaniak (2006), and Tomczyk et al. (2007) present a simulated annealing formulation given the name adaptive potential active hypercontours for finding nonlinear decision boundaries in both the classification and clustering problems; their work considers the use of radial basis functions in representing the decision boundary. Pölzlbauer et al. (2008) construct nonlinear decision boundaries in the input space from connected linear segments. In some ways, their approach is similar to active contours methods in image segmentation such as snakes that do not use the level set representation: changes in topology of the decision boundary in the optimization are difficult to handle. (The implicit level set representation takes care of topology changes naturally.)

The theory of classification with Lipschitz functions was discussed by von Luxburg and Bousquet (2004). As mentioned previously, level set functions are Lipschitz functions and the specific level set function that we use, the *signed distance function*, has a unit Lipschitz constant. Von Luxburg and Bousquet minimize the Lipschitz constant, whereas in our formulation, the Lipschitz constant is fixed. The von Luxburg and Bousquet formulation requires the specification of a subspace of Lipschitz functions over which to optimize in order to prevent overfitting, but does not resolve the question of how to select this subspace. The surface area penalty that we propose provides a natural specification for subspaces of signed distance functions.

The maximum allowable surface area parameterizes nested subspaces. We calculate the ε-entropy (Kolmogorov and Tihomirov, 1961) of these signed distance function subspaces and use the result to characterize geometric level set classification theoretically. In particular, we look at the consistency and convergence of level set classifiers as the size of the training set grows. We also look at the Rademacher complexity (Koltchinskii, 2001; Bartlett and Mendelson, 2002) of level set classifiers.

For the multicategory classification problem with $M > 2$ classes, typically binary classification methods are extended using the *one-against-all* construction (Hsu and Lin, 2002). The one-against-all scheme represents the classifier with M decision functions. We propose a more parsimonious representation of the multicategory level set classifier that uses $\log_2 M$ decision functions.[2] A collection of $\log_2 M$ level set functions can implicitly specify M regions using a binary encoding like a Venn diagram (Vese and Chan, 2002). This proposed logarithmic multicategory classification is

2. It is certainly possible to use one-against-all with the proposed level set classification methodology. In fact, there are M-category level set methods that use M level set functions (Samson et al., 2000; Paragios and Deriche, 2002), but they are less parsimonious than the approach we follow.

new, as there is no logarithmic formulation for M-category classification in the machine learning literature. The energy functional that is minimized has a multicategory empirical risk term and surface area penalties on $\log_2 M$ contours.

The level set representation of classifier decision boundaries, the surface area regularization term, the logarithmic multicategory classification scheme, and other contributions of this paper are not only interesting academically, but also practically. We compare the classification performance of geometric level set classification on several binary and multicategory data sets from the UCI Repository and find the results to be competitive with many classifiers used in practice.

Level set methods are usually implemented on a discretized grid, that is the values of the level set function are maintained and updated on a grid. In physics and image processing applications, it nearly always suffices to work in two- or three-dimensional spaces. In classification problems, however, the input data space can be high-dimensional. Implementation of level set methods for large input space dimension becomes cumbersome due to the need to store and update a grid of that large dimension. One way to address this practical limitation is to represent the level set function by a superposition of radial basis functions (RBFs) instead of on a grid (Cecil et al., 2004; Slabaugh et al., 2007; Gelas et al., 2007). We follow this implementation strategy in obtaining classification results.

In Section 2, we detail geometric level set classification in the binary case, giving the objective to be minimized and the contour evolution to perform the minimization. In Section 3, we provide theoretical analysis of the binary level set classifier given in Section 2. The main result is the calculation of the ε-entropy of the space of level set classifiers as a function of the maximum allowable decision boundary surface area; this result is then applied to characterize consistency and complexity. Section 4 goes over multicategory level set classification. In Section 5, we describe the RBF level set implementation and use that implementation to compare the classification test performance of geometric level set classification to the performance of several other classifiers. Section 6 concludes and provides a summary of the work.

2. Binary Geometric Level Set Classification

In the standard binary classification problem, we are given training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with data vectors $\mathbf{x}_i \in \Omega \subset \mathbb{R}^D$ and class labels $y_i \in \{+1, -1\}$ drawn according to some unknown probability density function $p_{\mathbf{X},Y}(\mathbf{x}, y)$ and we would like to learn a classifier $\hat{y} : \Omega \to \{+1, -1\}$ that classifies previously unseen data vectors well. A popular approach specifies the classifier as $\hat{y}(\mathbf{x}) = \text{sign}(\varphi(\mathbf{x}))$, where φ is a scalar-valued function. This function is obtained by minimizing the empirical risk over a model class $\mathcal{F}$:

$$\min_{\varphi \in \mathcal{F}} \sum_{i=1}^{n} \mathrm{L}(y_i \varphi(\mathbf{x}_i)). \tag{1}$$

A wide variety of margin-based loss functions L are employed in different classification methods, including the logistic loss in logistic regression, the hinge loss in the SVM, and the exponential loss in boosting: $\mathrm{L}_{\text{logistic}}(z) = \log(1 + e^{-z})$, $\mathrm{L}_{\text{hinge}}(z) = \max\{0, 1 - z\}$, and $\mathrm{L}_{\text{exponential}}(z) = e^{-z}$ (Bartlett et al., 2006; Lin, 2004). The loss function on which classifiers are typically evaluated is the misclassification zero-one loss: $\mathrm{L}_{\text{zero-one}}(z) = \text{step}(-z)$, where step is the Heaviside unit step function.

Limiting the model class is a way to control the complexity of the learned classifier in order to increase its generalization ability. This is the central idea of the structural risk minimization

principle (Vapnik, 1995). A model subclass can be specified directly through a constraint in the optimization by taking $\mathcal{F}$ to be the subclass in (1). Alternatively, it may be specified through a regularization term J with weight λ:

$$\min_{\varphi\in\mathcal{F}} \sum_{i=1}^{n} \mathrm{L}(y_i\varphi(\mathbf{x}_i)) + \lambda \mathrm{J}(\varphi), \tag{2}$$

where $\mathcal{F}$ indicates a broader class within which the subclass is delineated by $\mathrm{J}(\varphi)$.

We propose a geometric regularization term novel to machine learning, the surface area of the decision boundary:

$$\mathrm{J}(\varphi) = \oint_{\varphi=0} d\mathbf{s}, \tag{3}$$

where $d\mathbf{s}$ is an infinitesimal surface area element on the decision boundary. Decision boundaries that shatter more points are more tortuous than decision boundaries that shatter fewer points. The regularization functional (3) promotes smooth, less tortuous decision boundaries. It is experimentally shown in Varshney and Willsky (2008) that with this regularization term, there is an inverse relationship between the regularization parameter λ and the Vapnik-Chervonenkis (VC) dimension of the classifier. An analytical discussion of complexity is provided in Section 3.3. The empirical risk term and regularization term must be properly balanced as the regularization term by itself drives the decision boundary to be a set of infinitesimal hyperspheres.

We now describe how to find a classifier that minimizes (2) with the new regularization term (3) using the level set methodology. As mentioned in Section 1, the level set approach implicitly represents a $(D-1)$-dimensional contour C in a D-dimensional space Ω by a scalar-valued, Lipschitz continuous function φ known as the level set function (Osher and Fedkiw, 2003). The contour is the zero level set of φ. Contour C partitions Ω into the regions $\mathcal{R}$ and $\mathcal{R}^c$, which can be simply connected, multiply connected, or composed of several components. The level set function $\varphi(\mathbf{x})$ satisfies the properties: $\varphi(\mathbf{x}) < 0$ for $\mathbf{x} \in \mathcal{R}$, $\varphi(\mathbf{x}) > 0$ for $\mathbf{x} \in \mathcal{R}^c$, and $\varphi(\mathbf{x}) = 0$ for $\mathbf{x}$ on the contour C.

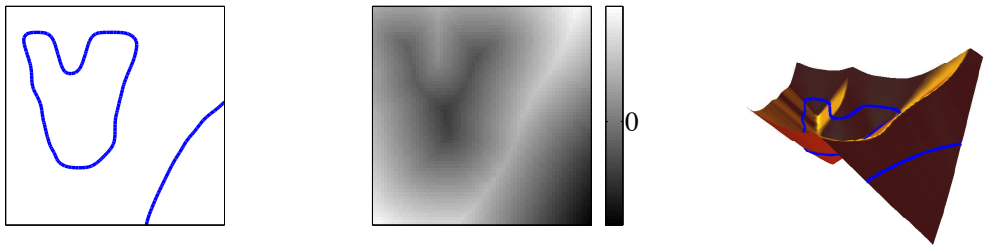
The level set function is often specialized to be the signed distance function, including in our work. The magnitude of the signed distance function at a point equals its distance to C, and its sign indicates whether it is in $\mathcal{R}$ or $\mathcal{R}^c$. The signed distance function satisfies the additional constraint that $\|\nabla\varphi(\mathbf{x})\| = 1$ and has Lipschitz constant equal to one. Illustrating the representation, a contour in a $D = 2$-dimensional space and its corresponding signed distance function are shown in Figure 1. For classification, we take $\mathcal{F}$ to be the set of all signed distance functions on the domain Ω in (2).

The general form of objective functionals in variational level set methods is

$$\mathrm{E}(C) = \int_{\mathcal{R}} g_{\mathrm{r}}(\mathbf{x}) d\mathbf{x} + \oint_{C} g_{\mathrm{b}}(C(\mathbf{s})) d\mathbf{s}, \tag{4}$$

where g_{r} is a region-based function and g_{b} is a boundary-based function. Note that the integral in the region-based functional is over $\mathcal{R}$, which is determined by C. Region-based functionals may also be integrals over $\mathcal{R}^c$, in place of or in addition to integrals over $\mathcal{R}$. In contour evolution, starting from some initial contour, the minimum of (4) is approached iteratively via a flow in the negative gradient direction. If we parameterize the iterations of the flow with a time parameter t, then it may be shown using the calculus of variations that the flow of C that implements the gradient descent is

$$\frac{\partial C}{\partial t} = -g_{\mathrm{r}}\mathbf{n} - \left(g_{\mathrm{b}}\kappa - \langle \nabla g_{\mathrm{b}}, \mathbf{n}\rangle\right)\mathbf{n}, \tag{5}$$



(a) (b) (c)

Figure 1: An illustration of the signed distance function representation of a contour with $D = 2$. The contour is shown in (a), its signed distance function is shown by shading in (b), and as a surface plot marked with the zero level set in (c).

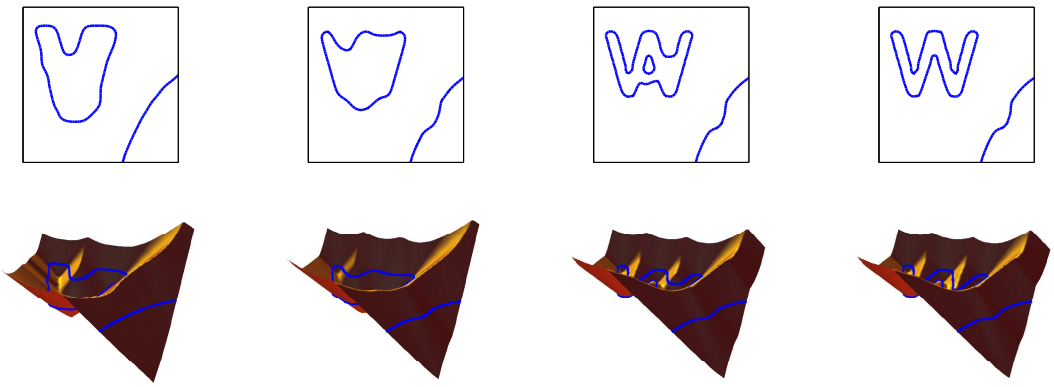


Figure 2: Iterations of an illustrative curve evolution proceeding from left to right. The top row shows the curve and the bottom row shows the corresponding signed distance function.

where $\mathbf{n}$ is the outward unit normal to C, and κ is its mean curvature (Caselles et al., 1997; Osher and Fedkiw, 2003). The mean curvature of a surface is an extrinsic measure of curvature from differential geometry that is the average of the principal curvatures. If the region-based function is integrated over $\mathcal{R}^c$, then the sign of the first term in (5) is reversed.

The flow of the contour corresponds to a flow of the signed distance function. The unit normal to the contour is $\nabla\varphi$ in terms of the signed distance function and the mean curvature is $\nabla^2\varphi$. The level set flow corresponding to (5) is

$$\frac{\partial\varphi(\mathbf{x})}{\partial t} = -g_{\mathrm{r}}(\mathbf{x})\nabla\varphi(\mathbf{x}) - \left(g_{\mathrm{b}}(\mathbf{x})\nabla^2\varphi(\mathbf{x}) - \langle\nabla g_{\mathrm{b}}(\mathbf{x}), \nabla\varphi(\mathbf{x})\rangle\right)\nabla\varphi(\mathbf{x}). \tag{6}$$

Figure 2 illustrates iterations of contour evolution.

For the classification problem, we have the following energy functional to be minimized:

$$\mathrm{E}(\varphi) = \sum_{i=1}^{n} \mathrm{L}(y_i \varphi(\mathbf{x}_i)) + \lambda \oint_C d\mathbf{s}. \tag{7}$$

The surface area regularization is a boundary-based functional with $g_\mathrm{b} = 1$ and the margin-based loss can be expressed as a region-based functional with g_r incorporating $\mathrm{L}(y_i \varphi(\mathbf{x}_i))$. Applying (4)–(6) to this energy functional yields the gradient descent flow

$$\left.\frac{\partial \varphi(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_i} = \begin{cases} \mathrm{L}(y_i \varphi(\mathbf{x}_i)) \nabla\varphi(\mathbf{x}_i) - \lambda \nabla^2 \varphi(\mathbf{x}_i) \nabla\varphi(\mathbf{x}_i), & \varphi(\mathbf{x}_i) < 0 \\ -\mathrm{L}(y_i \varphi(\mathbf{x}_i)) \nabla\varphi(\mathbf{x}_i) - \lambda \nabla^2 \varphi(\mathbf{x}_i) \nabla\varphi(\mathbf{x}_i), & \varphi(\mathbf{x}_i) > 0 \end{cases}. \tag{8}$$

In doing the contour evolution, note that we never compute the surface area of the decision boundary, which is oftentimes intractable, but just its gradient descent flow.

The derivative (8) does not take the constraint $\|\nabla\varphi\| = 1$ into account: the result of updating a signed distance function using (8) is not a signed distance function. Frequent reinitialization of the level set function as a signed distance function is important because (7) depends on the magnitude of φ, not just its sign. This reinitialization is done iteratively using (Sussman et al., 1994):

$$\frac{\partial \varphi(\mathbf{x})}{\partial t} = \mathrm{sign}(\varphi(\mathbf{x}))(1 - \|\nabla\varphi(\mathbf{x})\|).$$

With linear margin-based classifiers, including the original primal formulation of the SVM, the concept of margin is equivalent to Euclidean distance from the decision boundary in the input space. With kernel methods, however, this equivalence is lost; the quantity referred to as the margin, $y\varphi(\mathbf{x})$, is not the same as distance from $\mathbf{x}$ to the decision boundary in the input space. As discussed by Akaho (2004), oftentimes it is of interest to maximize the minimum distance to the decision boundary in the input space among all of the training examples. With the signed distance function representation, the margin $y\varphi(\mathbf{x})$ is equivalent to Euclidean distance from the decision boundary and hence is a satisfying nonlinear generalization to linear margin-based methods.

We now present two synthetic examples to illustrate this approach and its behavior. In both examples, there are $n = 1000$ points in the training set with $D = 2$. The first example has 502 points with label $y_i = -1$ and 498 points with label $y_i = +1$ and is separable by an elliptical decision boundary. The second example has 400 points with label $y_i = -1$ and 600 points with label $y_i = +1$ and is not separable by a simple shape, but has the -1 labeled points in a strip.

In these two examples, in the other examples in the rest of the paper, and in the performance results of Section 5.2, we use the logistic loss function for L in the objective (7). In these two examples, the surface area penalty has weight $\lambda = 0.5$; the value $\lambda = 0.5$ is a default parameter value that gives good performance with a variety of data sets regardless of their dimensionality D and can be used if one does not wish to optimize λ using cross-validation. Contour evolution minimization requires an initial decision boundary. In the portion of Ω where there are no training examples, we set the initial decision boundary to be a uniform grid of small components; this small seed initialization is common in level set methods. In the part of Ω where there are training examples, we use the locations and labels of the training examples to set the initial decision boundary. We assign a positive value to the initial signed distance function in locations of positively labeled examples and a negative value in locations of negatively labeled examples. The initial decision boundaries for the two examples are shown in the top left panels of Figure 3 and Figure 4.

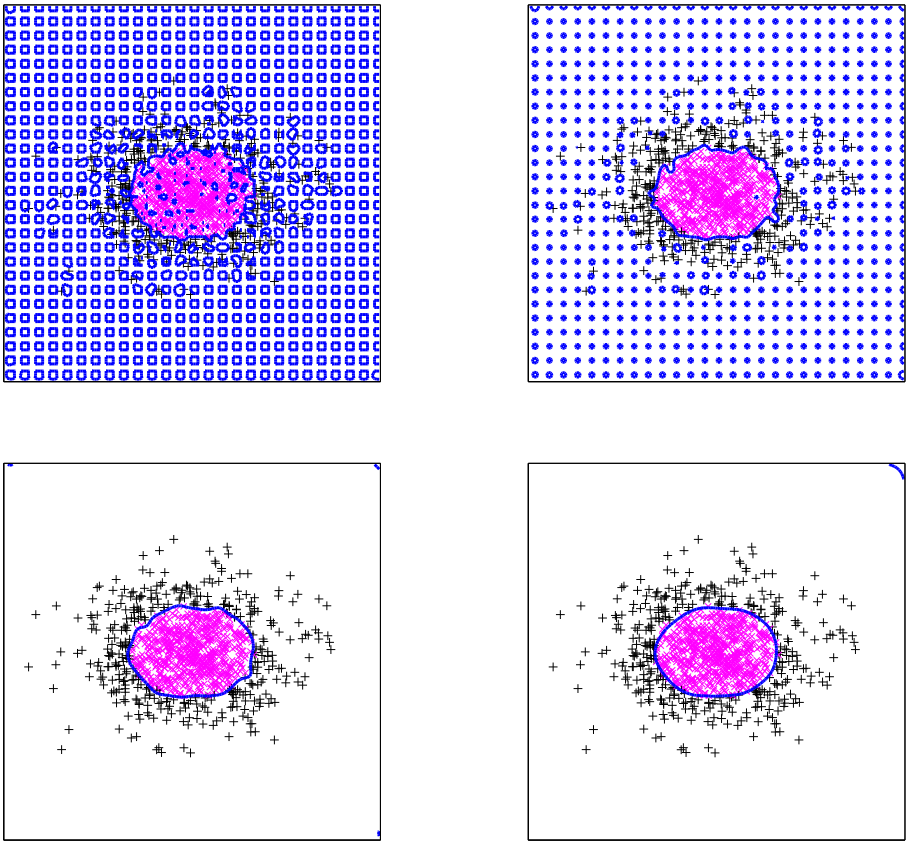


Figure 3: Curve evolution iterations with $\lambda = 0.5$ for an example training set proceeding in raster scan order starting from the top left. The magenta $\times$ markers indicate class label -1 and the black $+$ markers indicate class label $+1$. The blue line is the decision boundary.

Two intermediate iterations and the final decision boundary are also shown in Figure 3 and Figure 4. Solutions are as expected: an elliptical decision boundary and a strip-like decision boundary have been recovered. In the final decision boundaries of both examples, there is a small curved piece of the decision boundary in the top right corner of Ω where there are no training examples. This piece is an artifact of the initialization and the regularization term, and does not affect classifier performance. (The corner piece of the decision boundary is a minimal surface, a surface of zero mean curvature, which is a critical point of the surface area regularization functional (3), but not the global minimum. It is not important, assuming we have a representative training set.)

For a visual comparison of the effect of the surface area penalty weight, we show the solution decision boundaries of the geometric level set classifier for two other values of λ, 0.005 and 0.05, with the data set used in the example of Figure 4. As can be seen in comparing this figure with the bottom right panel of Figure 4, the smaller the value of λ, the longer and more tortuous the decision boundary. Small values of λ lead to overfitting.

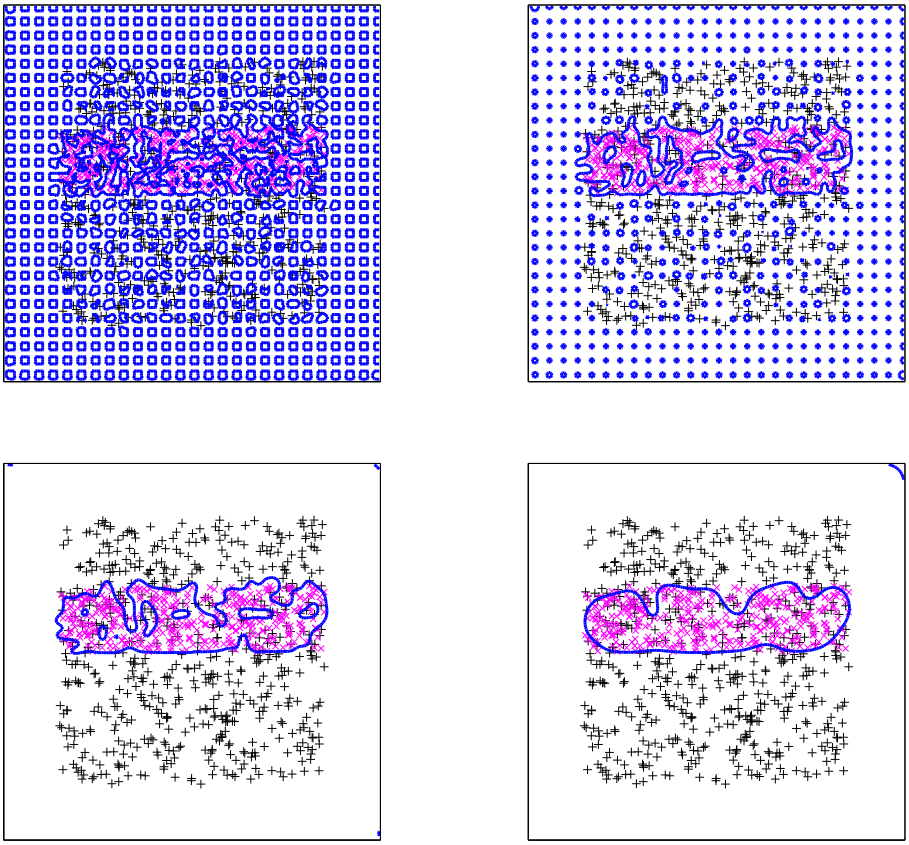
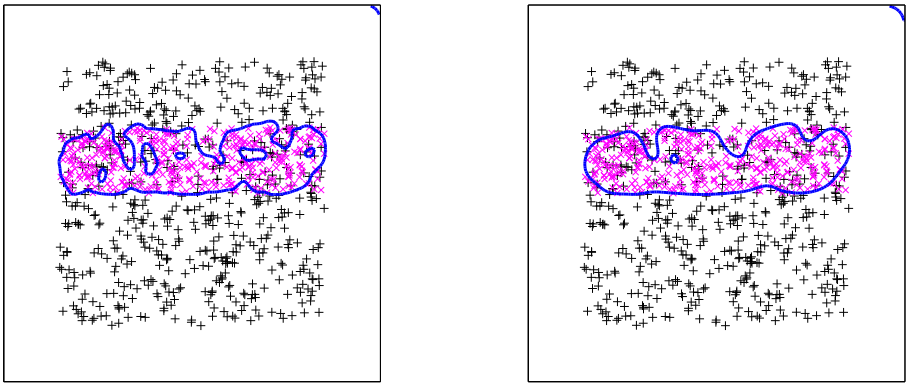


Figure 4: Curve evolution iterations with $\lambda = 0.5$ for an example training set proceeding in raster scan order starting from the top left. The magenta $\times$ markers indicate class label -1 and the black $+$ markers indicate class label $+1$. The blue line is the decision boundary.

In this section, we have described the basic method for nonlinear margin-based binary classification based on level set methods and illustrated its operation on two synthetic data sets. The next two sections build upon this core binary level set classification in two directions: theoretical analysis, and multicategory classification.

3. Consistency and Complexity Analysis

In this section, we provide analytical characterizations of the consistency and complexity of the level set classifier with surface area regularization described in Section 2. The main tool used in these characterizations is ε-entropy. Once we have an expression for the ε-entropy of the set of geometric level set classifiers, we can then apply consistency and complexity results from learning theory that are based on it. The beginning of this section is devoted to finding the ε-entropy of the space of signed distance functions with a surface area constraint with respect to the uniform or L_∞ metric on



(a) (b)

Figure 5: Solution decision boundaries with (a) $\lambda = 0.005$ and (b) $\lambda = 0.05$ for an example training set. The magenta $\times$ markers indicate class label -1 and the black $+$ markers indicate class label $+1$. The blue line is the decision boundary.

functions. The end of the section gives results on classifier consistency and complexity. The main findings are that level set classifiers are consistent, and that complexity is monotonically related to the surface area constraint, and thus the regularization term can be used to prevent underfitting and overfitting.

3.1 ε-Entropy

The ε-covering number of a metric space is the minimal number of sets with radius not exceeding ε required to cover that space. The ε-entropy is the base-two logarithm of the ε-covering number. These quantities are useful values in characterizing learning (Kulkarni, 1989; Williamson et al., 2001; Lin, 2004; von Luxburg and Bousquet, 2004; Steinwart, 2005; Bartlett et al., 2006). Kolmogorov and Tihomirov (1961), Dudley (1974, 1979), and others provide ε-entropy calculations for various classes of functions and various classes of sets, but the particular class we are considering, signed distance functions with a constraint on the surface area of the zero level set, does not appear in the literature. The second and third examples in Section 2 of Kolmogorov and Tihomirov (1961) are related, and the general approach we take for obtaining the ε-entropy of level set classifiers is similar to those two examples.

In classification, it is always possible to scale and shift the data and this is often done in practice. Without losing much generality and dispensing with some bothersome bookkeeping, we consider signed distance functions defined on the unit hypercube, that is $\Omega = [0,1]^D$, and we employ the uniform or L_∞ metric, $\rho_\infty(\varphi_1,\varphi_2) = \sup_{\mathbf{x}\in\Omega} |\varphi_1(\mathbf{x}) - \varphi_2(\mathbf{x})|$. We denote the set of all signed distance functions whose zero level set has surface area less than s by $\mathcal{F}_s$, its ε-covering number with respect to the uniform metric as $N_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$, and its ε-entropy as $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$. We begin with the $D = 1$ case and then come to general D.

Figure 6a shows a signed distance function over the unit interval. Due to the $\|\nabla\varphi\| = 1$ con-

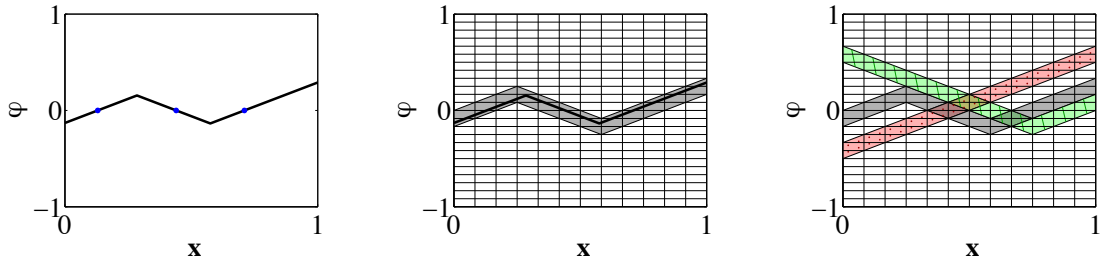


Figure 6: A $D = 1$-dimensional signed distance function in $\Omega = [0,1]$ is shown in (a), marked with its zero level set. The ε-corridor with $\varepsilon = \frac{1}{12}$ that contains the signed distance function is shown in (b), shaded in gray. The ε-corridor of (b), whose center line has three zero crossings is shown in (c), again shaded in gray, along with an ε-corridor whose center line has two zero crossings, shaded in green with stripes, and an ε-corridor whose center line has one zero crossing, shaded in red with dots.

straint, its slope is either $+1$ or -1 almost everywhere. The slope changes sign exactly once between two consecutive points in the zero level set. The signed distance function takes values in the range between positive and negative one.[3] In the $D = 1$ context, by surface area we mean the number of points in the zero level set, for example three in Figure 6a.

In finding $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$, we will use sets known as ε-*corridors*, which are particular balls of radius ε measured using ρ_∞ in the space of signed distance functions. We use the terminology of Kolmogorov and Tihomirov (or translator Hewitt), but our definition is slightly different than theirs. An ε-corridor is a strip of height 2ε for all $\mathbf{x}$. Let us define $\nu = \lceil \varepsilon^{-1} \rceil$. At $\mathbf{x} = 0$, the bottom and top of a corridor are at $2j\varepsilon$ and $2(j+1)\varepsilon$ respectively for some integer j, where $-\nu \leq 2j < \nu$. The slope of the corridor is either $+1$ or -1 for all $\mathbf{x}$ and the slope can only change at values of $\mathbf{x}$ that are multiples of ε. Additionally, the center line of the ε-corridor is a signed distance function, changing slope halfway between consecutive points in its zero level set and only there. The ε-corridor in which the signed distance function of Figure 6a falls is indicated in Figure 6b. Other ε-corridors are shown in Figure 6c.

By construction, each signed distance function is a member of exactly one ε-corridor. This is because since at $\mathbf{x} = 0$ the bottom and top of ε-corridors are at consecutive integer multiples of 2ε and since the center line of the corridor is a signed distance function, each signed distance function starts in one ε-corridor at $\mathbf{x} = 0$ and does not escape from it in the interval $(0,1]$. Also, an ε-corridor whose center line has s points in its zero level set contains only signed distance functions with at least s points in their zero level sets.

3. There are several ways to define the signed distance function in the degenerate cases ($\mathcal{R} = \Omega$, $\mathcal{R}^c = \emptyset$) and ($\mathcal{R} = \emptyset$, $\mathcal{R}^c = \Omega$), including the assignments $-\infty$ and $+\infty$, or -1 and $+1$ (Delfour and Zolésio, 2001). For our purposes, it suffices to say that we have chosen a unique function for the $\mathcal{R} = \Omega$ case and a unique function for the $\mathcal{R}^c = \Omega$ case.

Theorem 1 *The ε-entropy of the set of signed distance functions defined over $\Omega = [0,1]$ with zero level set having less than s points is:*

$$H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) = \log_2 \left(\sum_{k=1}^{s} \binom{\lceil \varepsilon^{-1} \rceil - 1}{k-1} \right) + 1.$$

Proof Since ε-corridors only change slope at multiples of ε, we can divide the abscissa into ν pieces. (Each piece has width ε except the last one if ε^{-1} is not an integer.) In each of the ν subintervals, the center line of a corridor is either wholly positive or wholly negative. Enumerating the full set of ε-corridors is equivalent to enumerating binary strings of length ν. Thus, without a constraint s, there are 2^ν ε-corridors. Since, by construction, ε-corridors tile the space of signed distance functions, $N_{\rho_\infty,\varepsilon}(\mathcal{F}) = 2^\nu$.

With the s constraint on ε-corridors, the enumeration is equivalent to twice the number of compositions of the positive integer ν by a sum of s or less positive integers. Twice because for every composition, there is one version in which the first subinterval of the corridor center is positive and one version in which it is negative. As an example, the red corridor in Figure 6c can be composed with two positive integers $(5+7)$, the green corridor by three $(7+4+1)$, and the gray corridor by four $(1+4+4+3)$. The number of compositions of ν by k positive integers is $\binom{\nu-1}{k-1}$. Note that the zero-crossings are unordered for this enumeration and that the set $\mathcal{F}_s$ includes all of the signed distance functions with surface area smaller than s as well. Therefore:

$$N_{\rho_\infty,\varepsilon}(\mathcal{F}_s) = 2 \sum_{k=1}^{s} \binom{\nu-1}{k-1}.$$

The result then follows because $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) = \log_2 N_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$. ■

The combinatorial formula in Theorem 1 is difficult to work with, so we give a highly accurate approximation.

Theorem 2 *The ε-entropy of the set of signed distance functions defined over $\Omega = [0,1]$ with zero level set having less than s points is:*

$$H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \approx \lceil \varepsilon^{-1} \rceil + \log_2 \Phi \left(\frac{2s - \lceil \varepsilon^{-1} \rceil}{\sqrt{\lceil \varepsilon^{-1} \rceil - 1}} \right),$$

where Φ is the standard Gaussian cumulative distribution function (cdf).

Proof Note that for a binomial random variable Z with $(\nu-1)$ Bernoulli trials having success probability $\frac{1}{2}$:

$$\Pr[Z < z] = 2^{-\nu} \cdot 2 \sum_{k=1}^{z} \binom{\nu-1}{k-1},$$

and that $N_{\rho_\infty,\varepsilon}(\mathcal{F}_s) = 2^\nu \Pr[Z < s]$. The result follows from the de Moivre-Laplace theorem and continuity correction, which are used to approximate the binomial distribution with the Gaussian distribution. ■

The central limit theorem tells us that the approximation works well when the Bernoulli success probability is one half, which it is in our case, and when the number of trials is large, which corresponds to small ε. The continuous approximation is better in the middle of the domain, when $s \approx \nu/2$, than in the tails. However, in the tails, the calculation of the exact expression in Theorem 1 is tractable. Since Φ is a cdf taking values in the range zero to one, $\log_2 \Phi$ is nonpositive. The surface area constraint only serves to reduce the ε-entropy.

The ε-entropy calculation in Theorem 1 and Theorem 2 is for the $D = 1$ case. We now discuss the case with general D. Recall that $\Omega = [0,1]^D$. Once again, we construct ε-corridors that tile the space of signed distance functions. In the one-dimensional case, the ultimate object of interest for enumeration is a string of length ν with binary labels. In the two-dimensional case, the corresponding object is a ν-by-ν grid of ε-by-ε squares with binary labels, and in general a D-dimensional Cartesian grid of hypercubes of volume ε^D, ν on each side. The surface area of the zero level set is the number of interior faces in the Cartesian grid whose adjoining ε^D hypercubes have different binary labels.

Theorem 3 *The ε-entropy of the set of signed distance functions defined over $\Omega = [0,1]^D$ with zero level set having surface area less than s is:*

$$H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \approx \left\lceil \varepsilon^{-1} \right\rceil^D + \log_2 \Phi \left(\frac{2s - D\left(\left\lceil \varepsilon^{-1} \right\rceil - 1\right) \left\lceil \varepsilon^{-1} \right\rceil^{D-1} - 1}{\sqrt{D\left(\left\lceil \varepsilon^{-1} \right\rceil - 1\right) \left\lceil \varepsilon^{-1} \right\rceil^{D-1}}} \right),$$

where Φ is the standard Gaussian cdf.

Proof In the one-dimensional case, it is easy to see that the number of segments is ν and the number of interior faces is $\nu - 1$. For a general D-dimensional Cartesian grid with ν hypercubes on each side, the number of hypercubes is ν^D and the number of interior faces is $D(\nu-1)\nu^{D-1}$. The result follows by substituting ν^D for ν and $D(\nu-1)\nu^{D-1}$ for $\nu - 1$ in Theorem 2. ■

Theorem 2 is a special case of Theorem 3 with $D = 1$. It is common to find the dimension of the space D in the exponent of ε^{-1} in ε-entropy calculations as we do here.

The ε-entropy calculation for level set classifiers given here enables us to analytically characterize their consistency properties as the size of the training set goes to infinity in Section 3.2 through ε-entropy-based classifier consistency results. The calculation also enables us to characterize the Rademacher complexity of level set classifiers in Section 3.3 through ε-entropy-based complexity results.

3.2 Consistency

In the binary classification problem, with training set of size n drawn from $p_{\mathbf{X},Y}(\mathbf{x}, y)$, a consistent classifier is one whose probability of error converges in the limit as n goes to infinity to the probability of error of the Bayes optimal decision rule. The optimal decision rule to minimize the probability of error is $\hat{y}^*(\mathbf{x}) = \text{sign}(p_{Y|\mathbf{X}}(Y = 1|\mathbf{X} = \mathbf{x}) - \frac{1}{2})$. Introducing notation, let the probability of error achieved by this decision rule be R^*. Also denote the probability of error of a level set classifier $\text{sign}(\varphi^{(n)})$ learned from a training set of size n as $R(\text{sign}(\varphi^{(n)}))$. For consistency, it is required that $R(\text{sign}(\varphi^{(n)})) - R^*$ converge in probability to zero.

The learned classifier $\text{sign}(\varphi^{(n)})$ minimizes the energy functional (7), and consequently the properties of $R(\text{sign}(\varphi^{(n)}))$ are affected by both the margin-based loss function L and by the regularization term. Lin (2004), Steinwart (2005), and Bartlett et al. (2006) have given conditions on the loss function necessary for a margin-based classifier to be consistent. Common margin-based loss functions including the logistic loss and exponential loss meet the conditions. Lin calls a loss function that meets the necessary conditions *Fisher-consistent*. Fisher consistency of the loss function is not enough, however, to imply consistency of the classifier overall. The regularization term must also be analyzed; since the regularization term based on surface area we introduce is new, so is the following analysis.

Concentrating on the surface area regularization, we adapt Theorem 4.1 of Lin, which is based on ε-entropy. The analysis is based on the method of sieves, where sieves $\mathcal{F}_n$ are an increasing sequence of subspaces of a function space $\mathcal{F}$. In our case, $\mathcal{F}$ is the set of signed distance functions on Ω and the sieves, $\mathcal{F}_{s(n)}$, are subsets of signed distance functions whose zero level sets have surface area less than $s(n)$, that is $\oint_{\varphi=0} d\mathbf{s} < s(n)$. Such a constraint is related to the regularization expression $\text{E}(\varphi)$ given in (7) through the method of Lagrange multipliers, with λ inversely related to $s(n)$. In the following, the function $s(n)$ is increasing in n and thus the conclusions of the theorem provide asymptotic results on consistency as the strength of the regularization term decreases as more training samples are made available. The sieve estimate is:

$$\varphi^{(n)} = \arg \min_{\varphi \in \mathcal{F}_{s(n)}} \sum_{i=1}^{n} \text{L}(y_i \varphi(\mathbf{x}_i)). \tag{9}$$

Having found $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$ in Section 3.1, we can apply Theorem 4.1 of Lin (2004), yielding the following theorem.

Theorem 4 Let L be a Fisher-consistent loss function in (9); let $\tilde{\varphi} = \arg\min_{\varphi \in \mathcal{F}} E[\text{L}(Y\varphi(\mathbf{X}))]$, where $\mathcal{F}$ is the space of signed distance functions on $[0,1]^D$; and let $\mathcal{F}_{s(n)}$ be a sequence of sieves. Then for sieve estimate $\varphi^{(n)}$, we have[4]

$$R(\text{sign}(\varphi^{(n)})) - R^* = O_P\left(\max\left\{n^{-\tau}, \inf_{\varphi \in \mathcal{F}_{s(n)}} \int (\varphi(\mathbf{x}) - \tilde{\varphi}(\mathbf{x}))^2 p_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}\right\}\right),$$

where

$$\tau = \begin{cases} \frac{1}{3}, & D = 1 \\ \frac{1}{4} - \frac{\log\log n}{2\log n}, & D = 2 \\ \frac{1}{2D}, & D \geq 3 \end{cases}.$$

Proof The result is a direct application of Theorem 4.1 of Lin (2004), which is in turn an application of Theorem 1 of Shen and Wong (1994). In order to apply this theorem, we need to note two things. First, that signed distance functions on $[0,1]^D$ are bounded (by a value of 1) in the L_∞ norm. Second, that there exists an A such that $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \leq A\varepsilon^{-D}$. Based on Theorem 3, we see that $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \leq \nu^D$ because the logarithm of the cdf is nonpositive. Since $\nu = \lceil \varepsilon^{-1} \rceil$, if ε^{-1} is an integer, then $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \leq \varepsilon^{-D}$ and otherwise there exists an A such that $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s) \leq A\varepsilon^{-D}$. ■

4. The notation $Z_n = O_P(\zeta_n)$ means that the random variable Z_n is bounded in probability at the rate ζ_n (van der Vaart, 1998).

Clearly $n^{-\tau}$ goes to zero as n goes to infinity. Also, $\inf_{\varphi \in \mathcal{F}_{s(n)}} \int (\varphi(\mathbf{x}) - \tilde{\varphi}(\mathbf{x}))^2 p_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}$ goes to zero when $s(n)$ is large enough so that the surface area constraint is no longer applicable.[5] Thus, level set classifiers are consistent.

3.3 Rademacher Complexity

The principal idea of the structural risk minimization principle is that the generalization error is the sum of an empirical risk term and a capacity term (Vapnik, 1995). The two terms should be sensibly balanced in order to achieve low generalization error. Here, we use the ε-entropy of signed distance functions constrained in decision boundary surface area to characterize the capacity term. In particular, we look at the Rademacher complexity of $\mathcal{F}_s$ as a function of s (Koltchinskii, 2001; Bartlett and Mendelson, 2002).

The Rademacher average of a class $\mathcal{F}$, denoted $\hat{R}_n(\mathcal{F})$, satisfies (von Luxburg and Bousquet, 2004):

$$\hat{R}_n(\mathcal{F}) \leq 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}} \int_{\frac{\varepsilon}{4}}^{\infty} \sqrt{H_{\rho_{2,n},\varepsilon'}(\mathcal{F})} d\varepsilon',$$

where $\rho_{2,n}(\varphi_1, \varphi_2) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\varphi_1(\mathbf{x}_i) - \varphi_2(\mathbf{x}_i))^2}$ is the empirical ℓ_2 metric. We found $H_{\rho_\infty,\varepsilon}(\mathcal{F}_s)$ for signed distance functions with surface area less than s in Section 3.1, and $H_{\rho_{2,n},\varepsilon}(\mathcal{F}) \leq H_{\rho_\infty,\varepsilon}(\mathcal{F})$. Thus, we can characterize the complexity of level set classifiers via the Rademacher capacity term (von Luxburg and Bousquet, 2004):

$$C_{\mathrm{Rad}}(\mathcal{F}_s, n) = 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}} \int_{\frac{\varepsilon}{4}}^{\infty} \sqrt{H_{\rho_\infty,\varepsilon'}(\mathcal{F})} d\varepsilon'. \tag{10}$$

With $\Omega = [0,1]^D$, the upper limit of the integral in (10) is one rather than infinity because ε cannot be greater than one.

In Figure 7, we plot C_{Rad} as a function of s for three values of D, and fixed ε and n. Having a fixed ε models the discretized grid implementation of level set methods. As the value of s increases, decision boundaries with more area are available. Decision boundaries with large surface area are more complex than smoother decision boundaries with small surface area. Hence the complexity term increases as a function of s. We have also empirically found the same relationship between the VC dimension and the surface area penalty (Varshney and Willsky, 2008). Consequently, the surface area penalty can be used to control the complexity of the classifier, and prevent underfitting and overfitting. The Rademacher capacity term may be used in setting the regularization parameter λ.

4. Multicategory Geometric Level Set Classification

Thus far, we have considered binary classification. In this section, we extend level set classification to the multicategory case with $M > 2$ classes labeled $y \in \{1, \ldots, M\}$. We represent the decision boundaries using $m = \lceil \log_2 M \rceil$ signed distance functions $\{\varphi_1(\mathbf{x}), \ldots, \varphi_m(\mathbf{x})\}$. Using such a set of level set functions we can represent 2^m regions $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_{2^m}\}$ through a binary encoding

5. For a given ε, there is a maximum possible surface area; the constraint is no longer applicable when the constraint is larger than this maximum possible surface area. Also note that s and λ are inversely related.

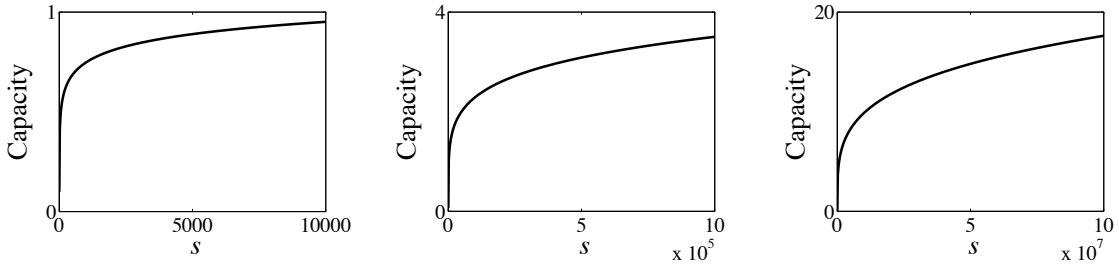


Figure 7: The Rademacher capacity term (10) as a function of s for signed distance functions on $\Omega = [0,1]^D$ with surface area less than s with (a) $D = 2$, (b) $D = 3$, and (c) $D = 4$. The values of ε and n are fixed at 0.01 and 1000 respectively.

(Vese and Chan, 2002). Thus, for $\mathbf{x} \in \mathcal{R}_1$, $(\varphi_1(\mathbf{x}) < 0) \wedge \cdots \wedge (\varphi_m(\mathbf{x}) < 0)$; for $\mathbf{x} \in \mathcal{R}_2$, $(\varphi_1(\mathbf{x}) < 0) \wedge \cdots \wedge (\varphi_{m-1}(\mathbf{x}) < 0) \wedge (\varphi_m > 0)$; and for $\mathbf{x} \in \mathcal{R}_{2^m}$, $(\varphi_1(\mathbf{x}) > 0) \wedge \cdots \wedge (\varphi_m(\mathbf{x}) > 0)$.

This binary encoding specifies the regions, but in order to apply margin-based loss functions, we also need a value for margin. In binary classification, the special encoding $y \in \{-1,+1\}$ allows $y\varphi(\mathbf{x})$ to be the argument to the loss function. For multicategory classification, the argument to the loss function is through functions $\psi_y(\mathbf{x})$, which are also specified through a binary encoding: $\psi_1(\mathbf{x}) = \max\{+\varphi_1(\mathbf{x}), \ldots, +\varphi_m(\mathbf{x})\}$, $\psi_2(\mathbf{x}) = \max\{+\varphi_1(\mathbf{x}), \ldots, +\varphi_{m-1}(\mathbf{x}), -\varphi_m(\mathbf{x})\}$, and $\psi_{2^m}(\mathbf{x}) = \max\{-\varphi_1(\mathbf{x}), \ldots, -\varphi_m(\mathbf{x})\}$. Then, the M-ary level set classification energy functional we propose is

$$\mathrm{E}(\varphi_1, \ldots, \varphi_m) = \sum_{i=1}^{n} \mathrm{L}(\psi_{y_i}(\mathbf{x}_i)) + \frac{\lambda}{m} \sum_{j=1}^{m} \oint_{\varphi_j = 0} d\mathbf{s}. \tag{11}$$

The same margin-based loss functions used in the binary case, such as the hinge and logistic loss functions, may be used in the multicategory case (Zou et al., 2006, 2008). The regularization term included in (11) is the sum of the surface areas of the zero level sets of the m signed distance functions.

The gradient descent flows for the m signed distance functions are

$$\left.\frac{\partial \varphi_1(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_i} = \begin{cases} \mathrm{L}(\psi_{y_i}(\mathbf{x}_i))\nabla\varphi_1(\mathbf{x}_i) - \frac{\lambda}{m}\nabla^2\varphi_1(\mathbf{x}_i)\nabla\varphi_1(\mathbf{x}_i), & \varphi_1(\mathbf{x}_i) < 0 \\ -\mathrm{L}(\psi_{y_i}(\mathbf{x}_i))\nabla\varphi_1(\mathbf{x}_i) - \frac{\lambda}{m}\nabla^2\varphi_1(\mathbf{x}_i)\nabla\varphi_1(\mathbf{x}_i), & \varphi_1(\mathbf{x}_i) > 0 \end{cases}$$

$$\vdots$$

$$\left.\frac{\partial \varphi_m(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_i} = \begin{cases} \mathrm{L}(\psi_{y_i}(\mathbf{x}_i))\nabla\varphi_m(\mathbf{x}_i) - \frac{\lambda}{m}\nabla^2\varphi_m(\mathbf{x}_i)\nabla\varphi_m(\mathbf{x}_i), & \varphi_m(\mathbf{x}_i) < 0 \\ -\mathrm{L}(\psi_{y_i}(\mathbf{x}_i))\nabla\varphi_m(\mathbf{x}_i) - \frac{\lambda}{m}\nabla^2\varphi_m(\mathbf{x}_i)\nabla\varphi_m(\mathbf{x}_i), & \varphi_m(\mathbf{x}_i) > 0 \end{cases}.$$

In the case $M = 2$ and $m = 1$, the energy functional and gradient flow revert back to binary level set classification described in Section 2.

The proposed multicategory classifier is different from the commonly used technique known as one-against-all (Hsu and Lin, 2002), which constructs an M-ary classifier from M binary classifiers,

both because it treats all M classes simultaneously in the objective and because the decision regions are represented by a logarithmic rather than linear number of decision functions. Zou et al. (2006) also treat all M classes simultaneously in the objective, but their multicategory kernel machines use M decision functions. In fact, to the best of our knowledge, there is no M-ary classifier representation in the literature using as few as $\lceil \log_2 M \rceil$ decision functions. Methods that combine binary classifier outputs using error-correcting codes make use of a logarithmic number of binary classifiers with a larger multiplicative constant, such as $\lceil 10 \log M \rceil$ or $\lceil 15 \log M \rceil$ (Rifkin and Klautau, 2004; Allwein et al., 2000).

We give an example showing multicategory level set classification with $M = 4$ and $D = 2$. The data set has 250 points for each of the four class labels $y_i = 1$, $y_i = 2$, $y_i = 3$, and $y_i = 4$. The classes are not perfectly separable by simple boundaries. With four classes, we use $m = 2$ signed distance functions. Figure 8 shows the evolution of the two contours, the magenta and cyan curves. The final decision region for class $y = 1$ is the portion inside both the magenta and cyan curves, and coincides with the training examples with class label 1. The final decision region for class 2 is the region inside the magenta curve but outside the cyan curve, the final decision region for class 3 is the region inside the cyan curve, but outside the magenta curve, and the final decision region for class 4 is outside both curves. The final decision boundaries are fairly smooth and partition the space with small training error.

5. Implementation and Classification Results

In this section, we describe how to implement geometric level set classification practically using RBFs and give classification performance results when applied to several real binary and multicategory data sets.

5.1 Radial Basis Function Level Set Method

There have been many developments in level set methods since the original work of Osher and Sethian (1988). One development in particular is to represent the level set function by a superposition of RBFs instead of on a grid (Cecil et al., 2004; Slabaugh et al., 2007; Gelas et al., 2007). Grid-based representation of the level set function is not amenable to classification in high-dimensional input spaces because the memory and computational requirements are exponential in the dimension of the input space. A nonparametric RBF representation, however, is tractable for classification. The RBF level set method we use to minimize the energy functionals (7) and (11) for binary and multicategory margin-based classification is most similar to that described by Gelas et al. (2007) for image processing.

The starting point of the RBF level set approach is describing the level set function $\varphi(\mathbf{x})$ via a strictly positive definite[6] RBF $\mathrm{K}(\cdot)$ as follows:

$$\varphi(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \mathrm{K}\left(\|\mathbf{x} - \mathbf{x}_i\|\right). \tag{12}$$

The zero level set of φ defined in this way is the contour C. For the classification problem, we take the centers $\mathbf{x}_i$ to be the data vectors of the training set. Then, constructing an $n \times n$ matrix $\mathbf{H}$ with

6. A more complete discussion including conditionally positive definite RBFs would add a polynomial term to (12), to span the null space of the RBF (Wendland, 2005).

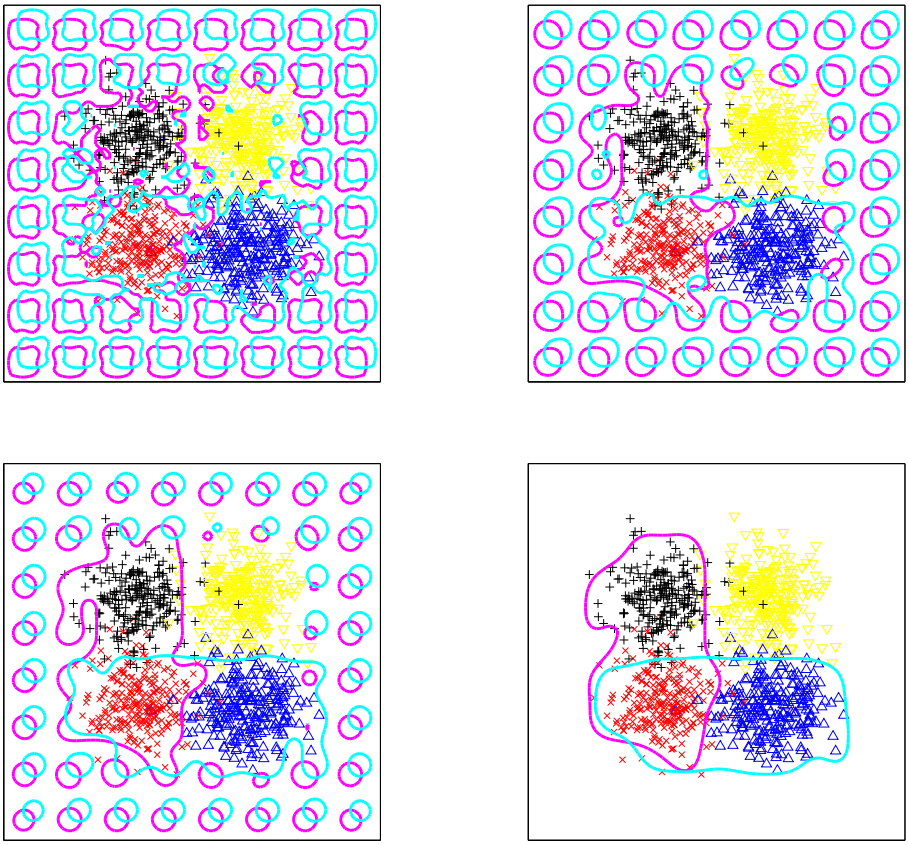


Figure 8: Curve evolution iterations with $\lambda = 0.5$ for multicategory classification proceeding in raster scan order. The red $\times$ markers indicate class label 1, the black $+$ markers indicate class label 2, the blue $\triangle$ markers indicate class label 3, and the yellow $\triangledown$ markers indicate class label 4. The magenta and cyan lines are the zero level sets of the $m = 2$ signed distance functions and together make up the decision boundary.

elements $\{\mathbf{H}\}_{ij} = \mathrm{K}(\|\mathbf{x}_i - \mathbf{x}_j\|)$, and letting $\boldsymbol{\alpha}$ be the vector of coefficients in (12), we have:

$$\begin{bmatrix} \varphi(\mathbf{x}_1) \\ \vdots \\ \varphi(\mathbf{x}_n) \end{bmatrix} = \mathbf{H}\boldsymbol{\alpha}.$$

To minimize an energy functional of C, the level set optimization is over the coefficients $\boldsymbol{\alpha}$ with $\mathbf{H}$ fixed. In order to perform contour evolution with the RBF representation, a time parameter t is introduced like in Section 2, giving:

$$\mathbf{H}\frac{d\boldsymbol{\alpha}}{dt} = \begin{bmatrix} \left.\frac{\partial\varphi(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_1} \\ \vdots \\ \left.\frac{\partial\varphi(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_n} \end{bmatrix}. \tag{13}$$

For the binary margin-based classification problem with surface area regularization that we are interested in solving, we substitute the gradient flow (8) into the right side of (13). For the multicategory classification problem, we have m level set functions as discussed in Section 4 and each one has a gradient flow to be substituted into an expression like (13).

The iteration for the contour evolution is then:

$$\boldsymbol{\alpha}^{(k+1)} = \boldsymbol{\alpha}^{(k)} - \tau\mathbf{H}^{-1}\begin{bmatrix} \left.\frac{\partial\varphi^{(k)}(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_1} \\ \vdots \\ \left.\frac{\partial\varphi^{(k)}(\mathbf{x})}{\partial t}\right|_{\mathbf{x}=\mathbf{x}_n} \end{bmatrix}, \tag{14}$$

where τ is a small step size and $\varphi^{(k)}$ comes from $\boldsymbol{\alpha}^{(k)}$. We normalize $\boldsymbol{\alpha}$ according to the ℓ_1-norm after every iteration.

The RBF-represented level set function is not a signed distance function. However, as discussed by Gelas et al. (2007), normalizing the coefficient vector $\boldsymbol{\alpha}$ with respect to the ℓ_1-norm after every iteration of (14) has a similar effect as reinitializing the level set function as a signed distance function. The Lipschitz constant of the level set function is constrained by this normalization. The analysis of Section 3 applies with minor modification for level set functions with a given Lipschitz constant and surface area constraint. The RBF level set approach is similar to kernel machines with the RBF kernel in the sense that the decision function is represented by a linear combination of RBFs. However, kernel methods in the literature minimize a reproducing kernel Hilbert space squared norm for regularization, whereas the geometric level set classifier minimizes decision boundary surface area for regularization. The regularization term and consequently inductive bias of the geometric level set classifier is new and different compared to existing kernel methods. The solution decision boundary is the zero level set of a function of the form given in (12). Of course this representation does not capture all possible functions, but, given that we use a number of RBFs equal to the number of training examples, the granularity of this representation is well-matched to the data. This is similar to the situation found in other contexts such as kernel machines using RBFs.

We initialize the decision boundary with $\boldsymbol{\alpha} = n(\mathbf{H}^{-1}\mathbf{y})/\|\mathbf{H}^{-1}\mathbf{y}\|_1$, where $\mathbf{y}$ is a vector of the n class labels in the training set. Figure 9 shows this initialization and following RBF-implemented

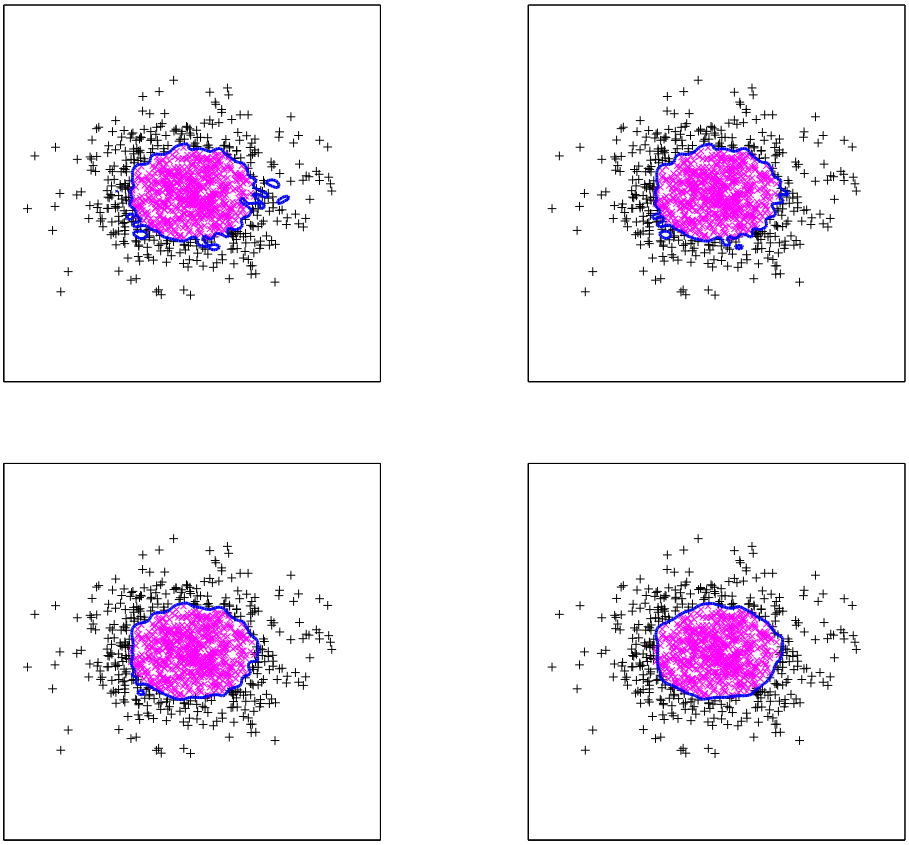


Figure 9: Curve evolution iterations with RBF implementation and $\lambda = 0.5$ for example training set proceeding in raster scan order. The magenta $\times$ markers indicate class label -1 and the black $+$ markers indicate class label $+1$. The blue line is the decision boundary.

contour evolution on the elliptically-separable data set presented in Section 2. The initial decision boundary is tortuous. It is smoothed out by the surface area penalty during the course of the contour evolution, thereby improving the generalization of the learned classifier as desired. To initialize the m vectors $\boldsymbol{\alpha}$ in M category classification, we use m length n vectors of positive and negative ones constructed from the binary encoding instead of $\mathbf{y}$.

5.2 Classification Results

We give classifier performance results on benchmark data sets from the UCI Machine Learning Repository (Asuncion and Newman, 2007) for geometric level set classification and compare them to the performance of several other classifiers, concluding that level set classification is a competitive technique. We present the tenfold cross-validation classification error performance with RBF level set implementation on four binary data sets: Pima Indians Diabetes ($n = 768$, $D = 8$), Wis-

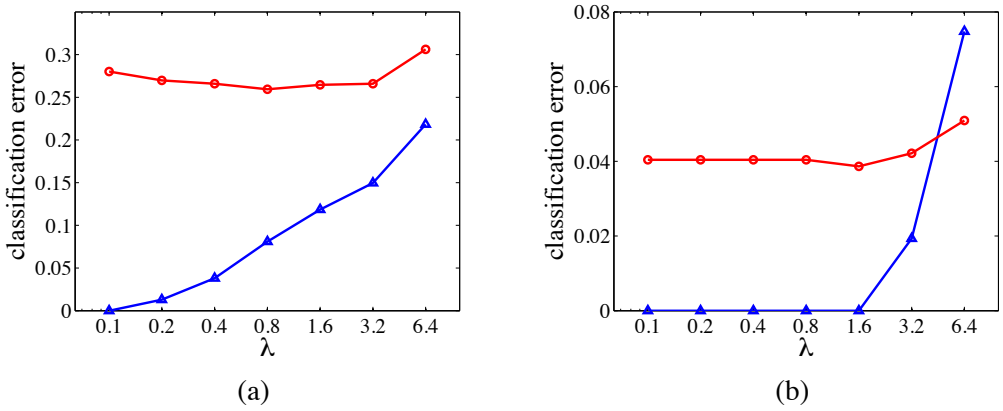


Figure 10: Tenfold cross-validation training error (blue line with triangle markers) and test error (red line with circle markers) for the (a) Pima, and (b) WDBC data sets as a function of the regularization parameter λ on a logarithmic scale.

consin Diagnostic Breast Cancer ($n = 569$, $D = 30$), BUPA Liver Disorders ($n = 345$, $D = 6$) and Johns Hopkins University Ionosphere ($n = 351$, $D = 34$), and four multicategory data sets: Wine Recognition ($n = 178$, $M = 3$, $D = 13$), Iris ($n = 150$, $M = 3$, $D = 4$), Glass Identification ($n = 214$, $M = 6$, $D = 9$), and Image Segmentation ($n = 2310$, $M = 7$, $D = 19$). For the binary data sets, there is $m = 1$ level set function, for the wine and iris data sets $m = 2$ level set functions, and for the glass and segmentation data sets $m = 3$ level set functions.

We shift and scale the data so that each of the input dimensions has zero mean and unit variance, use the RBF $K(\|\mathbf{x} - \mathbf{x}_i\|) = e^{-\|\mathbf{x}-\mathbf{x}_i\|^2}$, the logistic loss function, $\tau = 1/m$, and the initialization $\boldsymbol{\alpha} = n(\mathbf{H}^{-1}\mathbf{y})/\|\mathbf{H}^{-1}\mathbf{y}\|_1$. First, we look at classification error as a function of λ. Figure 10 shows the tenfold cross-validation training and test errors for the Pima and WDBC data sets; other data sets yield similar plots. The plots show evidence of the structural risk minimization principle and complexity analysis given in Section 3.3. For small λ (corresponding to large surface area constraint s), the model class is too complex and we see that although the training error is zero, the test error is not minimal due to overfitting. For large λ, the model class is not complex enough; the training error is large and the test error is not minimal due to underfitting. There is an intermediate value of λ that achieves the minimal test error. However, we notice that the test error is fairly insensitive to the value of λ. The test error does not change much over the plotted range.

In Table 1, we report the tenfold cross-validation test error (as a percentage) on the eight data sets and compare the performance to nine other classifiers.[7] On each of the ten folds, we set λ using cross-validation. Specifically, we perform fivefold cross-validation on the nine tenths of the full data set that is the training data for that fold. We select the λ from the set of values $\{0.2, 0.4, 0.8, 1.6, 3.2\}$ that minimizes the fivefold cross-validation test error. The performance results of the nine other

7. For lower-dimensional data sets (up to about $D = 12$), it is possible to use optimal dyadic decision trees (Scott and Nowak, 2006; Blanchard et al., 2007). We found that the results using such trees are not significantly better than those obtained using the C4.4 and C4.5 decision trees (which could be applied to all of the data sets without concern for dimensionality).

Data Set (M,D)	NB	BN	kNN	C4.4	C4.5	NBT	SVM	RBN	LLS	GLS
Pima $(2,8)$	23.69	25.64	27.86	27.33	26.17	25.64	22.66	24.60	29.94	25.94
WDBC $(2,30)$	7.02	4.92	3.68	7.20	6.85	7.21	2.28	5.79	6.50	4.04
Liver $(2,6)$	44.61	43.75	41.75	31.01	31.29	33.87	41.72	35.65	37.39	37.61
Ionos. $(2,34)$	17.38	10.54	17.38	8.54	8.54	10.27	11.40	7.38	13.11	13.67
Wine $(3,13)$	3.37	1.11	5.00	6.14	6.14	3.37	1.67	1.70	5.03	3.92
Iris $(3,4)$	4.00	7.33	4.67	4.00	4.00	6.00	4.00	4.67	3.33	6.00
Glass $(6,9)$	50.52	25.24	29.89	33.68	34.13	24.78	42.49	34.50	38.77	36.95
Segm. $(7,19)$	18.93	9.60	5.20	4.27	4.27	5.67	8.07	13.07	14.40	4.03

Table 1: Tenfold cross-validation error percentage of geometric level set classifier (GLS) with RBF level set implementation on several data sets compared to error percentages of various other classifiers reported in Cai and Sowmya (2007). The other classifiers are: naïve Bayes classifier (NB), Bayes net classifier (BN), k-nearest neighbor with inverse distance weighting (kNN), C4.4 decision tree (C4.4), C4.5 decision tree (C4.5), naïve Bayes tree classifier (NBT), SVM with polynomial kernel (SVM), radial basis function network (RBN), and learning level set classifier (LLS) of Cai and Sowmya (2007).

classifiers are as given by Cai and Sowmya (2007), who report the same tenfold cross-validation test error that we do for the geometric level set classifier. Details about parameter settings for the other nine classifiers may be found in Cai and Sowmya (2007).

The geometric level set classifier outperforms each of the other classifiers at least once among the four binary data sets, and is generally competitive overall. Level set classification is also competitive on the multicategory data sets. In fact, it gives the smallest error among all of the classifiers on the segmentation data set. The proposed classifier is competitive for data sets of both small and large dimensionality D; there is no apparent relationship between D and the performance of the geometric level set classifier in comparison to other methods.

6. Conclusion

Level set methods are powerful computational techniques that have not yet been widely adopted in machine learning. Our main goal with this contribution is to open a conduit between the application area of learning and the computational technique of level set methods. Towards that end, we have developed a nonlinear, nonparametric classifier based on level set methods that minimizes margin-based empirical risk in both the binary and multicategory cases, and is regularized by a geometric complexity penalty novel to classification. This approach is an alternative to kernel machines for learning nonlinear decision boundaries in the input space and is in some ways a more natural generalization of linear methods.

The variational level set formulation is flexible in allowing the inclusion of various geometric priors defined in the input space. The surface area regularization term is one such example, but others may also be included. Another example is an energy functional that measures feature relevance using the partial derivative of the signed distance function (Domeniconi et al., 2005), and can be used for ℓ_1-regularized feature subset selection as discussed in Varshney and Willsky (2008).

We have provided an analysis of the classifier by characterizing its ε-entropy. This characterization leads to results on consistency and complexity. We have described a multicategory level set classification procedure with a logarithmic number of decision functions, rather than the linear number that is typical in classification and decision making, through a binary encoding made possible by the level set representation.

It is a known fact that with finite training data, no one classification method is best for all data sets. Performance of classifiers may vary quite a bit depending on the data characteristics because of differing inductive biases. The classifier presented in this paper provides a new option when choosing a classifier. The results on standard data sets indicate that the level set classifier is competitive with other state-of-the-art classifiers. It would be interesting to systematically find domains in the space of data set characteristics for which the geometric level set classifier outperforms other classifiers (Ho and Basu, 2002).

Acknowledgments

This work was supported in part by an NSF Graduate Research Fellowship, by Shell International Exploration and Production, Inc., and by a MURI funded through ARO Grant W911NF-06-1-0076. The authors thank Justin H. G. Dauwels, John W. Fisher, III and Sujay R. Sanghavi for discussions. The level set methods toolbox by Barış Sümengen was used in producing figures in Section 2 and Section 4.

References

Shotaro Akaho. SVM that maximizes the margin in the input space. *Systems and Computers in Japan*, 35(14):78–86, December 2004.

Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach to margin classifiers. *Journal of Machine Learning Research*, 1:113–141, December 2000.

Arthur Asuncion and David J. Newman. UCI machine learning repository. Available at http://archive.ics.uci.edu/ml/, 2007.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, November 2002.

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, March 2006.

Gilles Blanchard, Christin Schäfer, Yves Rozenholc, and Klaus-Robert Müller. Optimal dyadic decision trees. *Machine Learning*, 66(2–3):209–241, March 2007.

Erik M. Boczko, Todd R. Young, Minhui Xie, and Di Wu. Comparison of binary classification based on signed distance functions with support vector machines. In *Proceedings of the Ohio Collaborative Conference on Bioinformatics*, Athens, Ohio, June 2006.

Xiongcai Cai and Arcot Sowmya. Level learning set: A novel classifier based on active contour models. In Joost N. Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenič, and Andrzej Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning*, pages 79–90, Warsaw, Poland, 2007.

Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, February 1997.

Thomas Cecil, Jianliang Qian, and Stanley Osher. Numerical methods for high dimensional Hamilton–Jacobi equations using radial basis functions. *Journal of Computational Physics*, 196 (1):327–347, May 2004.

Daniel Cremers, Mikael Rousson, and Rachid Deriche. A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *International Journal of Computer Vision*, 72(2):195–215, April 2007.

Michel C. Delfour and Jean-Paul Zolésio. *Shapes and Geometries: Analysis, Differential Calculus, and Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 2001.

Carlotta Domeniconi, Dimitrios Gunopulos, and Jing Peng. Large margin nearest neighbor classifiers. *IEEE Transactions on Neural Networks*, 16(4):899–909, July 2005.

Richard M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *Journal of Approximation Theory*, 10(3):227–236, March 1974.

Richard M. Dudley. Correction to "metric entropy of some classes of sets with differentiable boundaries". *Journal of Approximation Theory*, 26(2):192–193, June 1979.

Bradley Efron. The efficiency of logistic regression compared to normal discriminant analysis. *Journal of the American Statistical Association*, 70(352):892–898, December 1975.

Arnaud Gelas, Olivier Bernard, Denis Friboulet, and Rémy Prost. Compactly supported radial basis functions based collocation method for level-set evolution in image segmentation. *IEEE Transactions on Image Processing*, 16(7):1873–1887, July 2007.

Tin Kam Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, March 2002.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, March 2002.

Andrey N. Kolmogorov and Vladimir M. Tihomirov. ε-entropy and ε-capacity of sets in functional spaces. *American Mathematical Society Translations Series 2*, 17:277–364, 1961.

Vladimir Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, July 2001.

Sanjeev R. Kulkarni. On metric entropy, Vapnik–Chervonenkis dimension, and learnability for a class of distributions. Technical Report P-1910, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, September 1989.

Yi Lin. A note on margin-based loss functions in classification. *Statistics & Probability Letters*, 68 (1):73–82, June 2004.

Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–695, June 2004.

Stanley Osher and Ronald Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, New York, 2003.

Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, November 1988.

Nikos Paragios and Rachid Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3):223–247, February 2002.

Georg Pölzlbauer, Thomas Lidy, and Andreas Rauber. Decision manifolds—a supervised learning algorithm based on self-organization. *IEEE Transactions on Neural Networks*, 19(9):1518–1530, September 2008.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, January 2004.

Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

Christophe Samson, Laure Blanc-Féraud, Gilles Aubert, and Josiane Zerubia. A level set model for image classification. *International Journal of Computer Vision*, 40(3):187–197, December 2000.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, Massachusetts, 2002.

Clayton Scott and Robert D. Nowak. Minimax-optimal classification with dyadic decision trees. *IEEE Transactions on Information Theory*, 52(4):1335–1353, April 2006.

Xiaotong Shen and Wing Hung Wong. Convergence rate of sieve estimates. *The Annals of Statistics*, 22(2):580–615, June 1994.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.

Gregory G. Slabaugh, H. Quynh Dinh, and Gözde B. Unal. A variational approach to the evolution of radial basis functions for image segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, June 2007.

Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, January 2005.

Mark Sussman, Peter Smereka, and Stanley Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114(1):146–159, September 1994.

Arkadiusz Tomczyk. Active hypercontours and contextual classification. In Halina Kwasnicka and Marcin Paprzycki, editors, *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, pages 256–261, Wroclaw, Poland, September 2005.

Arkadiusz Tomczyk and Piotr S. Szczepaniak. On the relationship between active contours and contextual classification. In Marek Kurzyński, Edward Puchała, Michał Woźniak, and Andrzej Żołnierek, editors, *Proceedings of the 4th International Conference on Computer Recognition Systems*, pages 303–310, Rydzyna, Poland, 2005.

Arkadiusz Tomczyk and Piotr S. Szczepaniak. Adaptive potential active hypercontours. In Leszek Rutkowski, Ryszard Tadeusiewicz, Lotfi A. Zadeh, and Jacek Zurada, editors, *Proceedings of the 8th International Conference on Artificial Intelligence and Soft Computing*, pages 692–701, Zakopane, Poland, June 2006.

Arkadiusz Tomczyk, Piotr S. Szczepaniak, and Michal Pryczek. Active contours as knowledge discovery methods. In Vincent Corruble, Masayuki Takeda, and Einoshin Suzuki, editors, *Proceedings of the 10th International Conference on Discovery Science*, pages 209–218, Sendai, Japan, 2007.

Aad W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, Cambridge, England, 1998.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

Kush R. Varshney and Alan S. Willsky. Supervised learning of classifiers via level set segmentation. In *Proceedings of the IEEE Workshop on Machine Learning for Signal Processing*, pages 115–120, Cancún, Mexico, October 2008.

Luminita A. Vese and Tony F. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *International Journal of Computer Vision*, 50(3):271–293, December 2002.

Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, England, 2005.

Rebecca Willett and Robert D. Nowak. Minimax optimal level-set estimation. *IEEE Transactions on Image Processing*, 16(12):2965–2979, December 2007.

Robert C. Williamson, Alexander J. Smola, and Bernhard Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. *IEEE Transactions on Information Theory*, 47(6):2516–2532, September 2001.

Andy M. Yip, Chris Ding, and Tony F. Chan. Dynamic cluster formation using level set methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):877–889, June 2006.

Hui Zou, Ji Zhu, and Trevor Hastie. The margin vector, admissible loss and multi-class margin-based classifiers. Technical report, University of Minnesota, 2006.

Hui Zou, Ji Zhu, and Trevor Hastie. New multicategory boosting algorithms based on multicategory Fisher-consistent losses. *Annals of Applied Statistics*, 2(4):1290–1306, December 2008.

Journal of Machine Learning Research 11 (2010) 517-553 Submitted 11/08; Revised 7/09; Published 2/10

Generalized Power Method for Sparse Principal Component Analysis

Michel Journée M.JOURNEE@ULG.AC.BE
Dept. of Electrical Engineering and Computer Science
University of Liège
B-4000 Liège, Belgium

Yurii Nesterov YURII.NESTEROV@UCLOUVAIN.BE
Peter Richtárik PETER.RICHTARIK@UCLOUVAIN.BE
Center for Operations Research and Econometrics
Catholic University of Louvain
Voie du Roman Pays 34, B-1348 Louvain-la-Neuve, Belgium

Rodolphe Sepulchre R.SEPULCHRE@ULG.AC.BE
Dept. of Electrical Engineering and Computer Science
University of Liège
B-4000 Liège, Belgium

Editor: Aapo Hyvarinen

Abstract

In this paper we develop a new approach to sparse principal component analysis (sparse PCA). We propose two single-unit and two block optimization formulations of the sparse PCA problem, aimed at extracting a single sparse dominant principal component of a data matrix, or more components at once, respectively. While the initial formulations involve nonconvex functions, and are therefore computationally intractable, we rewrite them into the form of an optimization program involving maximization of a convex function on a compact set. The dimension of the search space is decreased enormously if the data matrix has many more columns (variables) than rows. We then propose and analyze a simple gradient method suited for the task. It appears that our algorithm has best convergence properties in the case when either the objective function or the feasible set are strongly convex, which is the case with our single-unit formulations and can be enforced in the block case. Finally, we demonstrate numerically on a set of random and gene expression test problems that our approach outperforms existing algorithms both in quality of the obtained solution and in computational speed.

Keywords: sparse PCA, power method, gradient ascent, strongly convex sets, block algorithms

1. Introduction

Principal component analysis (PCA) is a well established tool for making sense of high dimensional data by reducing it to a smaller dimension. It has applications virtually in all areas of science—machine learning, image processing, engineering, genetics, neurocomputing, chemistry, meteorology, control theory, computer networks—to name just a few—where large data sets are encountered. It is important that having reduced dimension, the essential characteristics of the data are retained. If $A \in \mathbf{R}^{p\times n}$ is a matrix encoding p samples of n variables, with n being large, PCA aims at finding a few linear combinations of these variables, called *principal components*, which point in orthogonal directions explaining as much of the variance in the data as possible. If the variables contained in

the columns of A are centered, then the classical PCA can be written in terms of the scaled *sample covariance matrix* $\Sigma = A^T A$ as follows:

$$\text{Find} \quad z^* = \arg\max_{z^T z \leq 1} z^T \Sigma z. \tag{1}$$

Extracting one component amounts to computing the dominant eigenvector of Σ (or, equivalently, dominant right singular vector of A). Full PCA involves the computation of the singular value decomposition (SVD) of A. Principal components are, in general, combinations of all the input variables, that is, the *loading vector* z^* is not expected to have many zero coefficients. In most applications, however, the original variables have concrete physical meaning and PCA then appears especially interpretable if the extracted components are composed only from a small number of the original variables. In the case of gene expression data, for instance, each variable represents the expression level of a particular gene. A good analysis tool for biological interpretation should be capable to highlight "simple" structures in the genome—structures expected to involve a few genes only—that explain a significant amount of the specific biological processes encoded in the data. Components that are linear combinations of a small number of variables are, quite naturally, usually easier to interpret. It is clear, however, that with this additional goal, some of the explained variance has to be sacrificed. The objective of *sparse principal component analysis* (sparse PCA) is to find a reasonable *trade-off* between these conflicting goals. One would like to explain *as much* variability in the data as possible, using components constructed from *as few* variables as possible. This is the classical trade-off between *statistical fidelity* and *interpretability*.

For about a decade, sparse PCA has been a topic of active research. Historically, the first suggested approaches were based on ad-hoc methods involving post-processing of the components obtained from classical PCA. For example, Jolliffe (1995) consider using various rotation techniques to find sparse loading vectors in the subspace identified by PCA. Cadima and Jolliffe (1995) proposed to simply set to zero the PCA loadings which are in absolute value smaller than some threshold constant.

In recent years, more involved approaches have been put forward—approaches that consider the conflicting goals of explaining variability and achieving representation sparsity simultaneously. These methods usually cast the sparse PCA problem in the form of an optimization program, aiming at maximizing explained variance penalized for the number of non-zero loadings. For instance, the SCoTLASS algorithm proposed by Jolliffe et al. (2003) aims at maximizing the Rayleigh quotient of the covariance matrix of the data under the ℓ_1-norm based Lasso penalty (Tibshirani, 1996). Zou et al. (2006) formulate sparse PCA as a regression-type optimization problem and impose the Lasso penalty on the regression coefficients. d'Aspremont et al. (2007) in their DSPCA algorithm exploit convex optimization tools to solve a convex relaxation of the sparse PCA problem. Shen and Huang (2008) adapt the singular value decomposition (SVD) to compute low-rank matrix approximations of the data matrix under various sparsity-inducing penalties. Greedy methods, which are typical for combinatorial problems, have been investigated by Moghaddam et al. (2006). Finally, d'Aspremont et al. (2008) proposed a greedy heuristic accompanied with a certificate of optimality.

In many applications, several components need to be identified. The traditional approach consists of incorporating an existing single-unit algorithm in a deflation scheme, and computing the desired number of components sequentially (see, e.g., d'Aspremont et al. 2007). In the case of Rayleigh quotient maximization it is well-known that computing several components at once instead of computing them one-by-one by deflation with the classical power method might present

better convergence whenever the largest eigenvalues of the underlying matrix are close to each other (see, e.g., Parlett 1980). Therefore, block approaches for sparse PCA are expected to be more efficient on ill-posed problems.

In this paper we consider two single-unit (Section 2.1 and 2.2) and two block formulations (Section 2.3 and 2.4) of sparse PCA, aimed at extracting m sparse principal components, with $m = 1$ in the former case and $p \geq m > 1$ in the latter. Each of these two groups comes in two variants, depending on the type of penalty we use to enforce sparsity—either ℓ_1 or ℓ_0 (cardinality).[1] Although we assume a direct access to the data matrix A, these formulations also hold when only the covariance matrix Σ is available, provided that a factorization of the form $\Sigma = A^T A$ is identified (e.g., by eigenvalue decomposition or by Cholesky decomposition).

While our basic formulations involve maximization of a *nonconvex* function on a space of dimension involving n, we construct *reformulations* that cast the problem into the form of maximization of a *convex* function on the unit Euclidean sphere in $\mathbf{R}^p$ (in the $m = 1$ case) or the *Stiefel manifold*[2] in $\mathbf{R}^{p\times m}$ (in the $m > 1$ case). The advantage of the reformulation becomes apparent when trying to solve problems with many variables ($n \gg p$), since we manage to avoid searching a space of large dimension.[3] At the same time, due to the convexity of the new cost function we are able to propose and *analyze* the iteration-complexity of a simple gradient-type scheme, which appears to be well suited for problems of this form. In particular, we study (Section 3) a first-order method for solving an optimization problem of the form

$$f^* = \max_{x\in Q} f(x), \tag{P}$$

where Q is a compact subset of a finite-dimensional vector space and f is convex. It appears that our method has best theoretical convergence properties when either f or Q are strongly convex, which is the case in the single unit case (unit ball is strongly convex) and can be enforced in the block case by adding a strongly convex regularizing term to the objective function, constant on the feasible set. We do not, however, prove any results concerning the quality of the obtained solution. Even the goal of obtaining a local maximizer is in general unattainable, and we must be content with convergence to a stationary point.

In the particular case when Q is the unit Euclidean ball in $\mathbf{R}^p$ and $f(x) = x^T Cx$ for some $p \times p$ symmetric positive definite matrix C, our gradient scheme specializes to the *power method*, which aims at maximizing the *Rayleigh quotient*

$$R(x) = \frac{x^T Cx}{x^T x}$$

and thus at computing the largest eigenvalue, and the corresponding eigenvector, of C.

By applying our general gradient scheme to our sparse PCA reformulations of the form (P), we obtain algorithms (Section 4) with per-iteration computational cost $O(npm)$.

We demonstrate on random Gaussian (Section 5.1) and gene expression data related to breast cancer (Section 5.2) that our methods are very efficient in practice. While achieving a balance between the explained variance and sparsity which is the same as or superior to the existing methods,

1. Our single-unit cardinality-penalized formulation is identical to that of d'Aspremont et al. (2008).
2. Stiefel manifold is the set of rectangular matrices with orthonormal columns.
3. Note that in the case $p > n$, it is recommended to factor the covariance matrix as $\Sigma = A^T A$ with $A \in \mathbf{R}^{n\times n}$, such that the dimension p in the reformulations equals at most n.

they are faster, often converging before some of the other algorithms manage to initialize. Additionally, in the case of gene expression data our approach seems to extract components with strongest biological content.

1.1 Notation

For convenience of the reader, and at the expense of redundancy, some of the less standard notation below is also introduced at the appropriate place in the text where it is used. Parameters $m \leq p \leq n$ are actual values of dimensions of spaces used in the paper. In the definitions below, we use these actual values (i.e., n, p and m) if the corresponding object we define is used in the text exclusively with them; otherwise we make use of the dummy variables k (representing p or n in the text) and l (representing m, p or n in the text).

Given a vector $y \in \mathbf{R}^k$, its j^{th} coordinate is denoted by y_j. With an abuse of notation, we may use subscripts to indicate a sequence of vectors, such as $y_1, y_2, \ldots, y_l$. In that case the j^{th} coordinates of these vectors are denoted by $y_{1j}, y_{2j}, \ldots, y_{lj}$. By y_i we refer to the i^{th} column of $Y \in \mathbf{R}^{k\times l}$. Consequently, the element of Y at position (i, j) can be written as y_{ij}.

By $\mathbf{E}$ we refer to a finite-dimensional vector space; $\mathbf{E}^*$ is its conjugate space, that is, the space of all linear functionals on $\mathbf{E}$. By $\langle s, x\rangle$ we denote the action of $s \in \mathbf{E}^*$ on $x \in \mathbf{E}$. For a self-adjoint positive definite linear operator $G : \mathbf{E} \to \mathbf{E}^*$ we define a pair of norms on $\mathbf{E}$ and $\mathbf{E}^*$ as follows

$$\begin{aligned} \|x\| &\stackrel{\text{def}}{=} \langle Gx, x\rangle^{1/2}, \quad x \in \mathbf{E}, \\ \|s\|_* &\stackrel{\text{def}}{=} \langle s, G^{-1}s\rangle^{1/2}, \quad s \in \mathbf{E}^*. \end{aligned} \tag{2}$$

Although the theory in Section 3 is developed in this general setting, the sparse PCA applications considered in this paper require either the choice $\mathbf{E} = \mathbf{E}^* = \mathbf{R}^p$ (see Section 3.3 and problems (8) and (13) in Section 2) or $\mathbf{E} = \mathbf{E}^* = \mathbf{R}^{p\times m}$ (see Section 3.4 and problems (16) and (20) in Section 2). In both cases we will let G be the corresponding identity operator for which we obtain

$$\langle x, y\rangle = \sum_i x_i y_i, \quad \|x\| = \langle x, x\rangle^{1/2} = \left(\sum_i x_i^2\right)^{1/2} = \|x\|_2, \quad x, y \in \mathbf{R}^p, \quad \text{and}$$

$$\langle X, Y\rangle = \operatorname{Tr} X^T Y, \quad \|X\| = \langle X, X\rangle^{1/2} = \left(\sum_{ij} x_{ij}^2\right)^{1/2} = \|X\|_F, \quad X, Y \in \mathbf{R}^{p\times m}.$$

Thus in the vector setting we work with the *standard Euclidean norm* and in the matrix setting with the *Frobenius norm*. The symbol Tr denotes the trace of its argument.

Furthermore, for $z \in \mathbf{R}^n$ we write $\|z\|_1 = \sum_i |z_i|$ (ℓ_1 norm) and by $\|z\|_0$ (ℓ_0 "norm") we refer to the number of nonzero coefficients, or *cardinality*, of z. By $\mathbf{S}^p$ we refer to the space of all $p \times p$ symmetric matrices; $\mathbf{S}^p_+$ (resp. $\mathbf{S}^p_{++}$) refers to the positive semidefinite (resp. definite) cone. Eigenvalues of matrix Y are denoted by $\lambda_i(Y)$, largest eigenvalue by $\lambda_{\max}(Y)$. Analogous notation with the symbol σ refers to singular values.

By $\mathcal{B}^k = \{y \in \mathbf{R}^k \mid y^T y \leq 1\}$ (resp. $\mathcal{S}^k = \{y \in \mathbf{R}^k \mid y^T y = 1\}$) we refer to the unit Euclidean ball (resp. sphere) in $\mathbf{R}^k$. If we write $\mathcal{B}$ and $\mathcal{S}$, then these are the corresponding objects in $\mathbf{E}$. The space of $n \times m$ matrices with unit-norm columns will be denoted by

$$[\mathcal{S}^n]^m = \{Y \in \mathbf{R}^{n\times m} \mid \operatorname{Diag}(Y^T Y) = I_m\},$$

where Diag$(\cdot)$ represents the diagonal matrix obtained by extracting the diagonal of the argument. *Stiefel manifold* is the set of rectangular matrices of fixed size with orthonormal columns:

$$\mathcal{S}_m^p = \{Y \in \mathbf{R}^{p\times m} \mid Y^T Y = I_m\}.$$

For $t \in \mathbf{R}$ we will further write $\text{sign}(t)$ for the sign of the argument and $t_+ = \max\{0,t\}$.

2. Some Formulations of the Sparse PCA Problem

In this section we propose four formulations of the sparse PCA problem, all in the form of the general optimization framework (P). The first two deal with the single-unit sparse PCA problem and the remaining two are their generalizations to the block case.

2.1 Single-unit Sparse PCA via ℓ_1-Penalty

Let us consider the optimization problem

$$\phi_{\ell_1}(\gamma) \stackrel{\text{def}}{=} \max_{z\in\mathcal{B}^n} \sqrt{z^T\Sigma z} - \gamma\|z\|_1, \tag{3}$$

with sparsity-controlling parameter $\gamma \geq 0$ and sample covariance matrix $\Sigma = A^T A$.

The solution $z^*(\gamma)$ of (3) in the case $\gamma = 0$ is equal to the right singular vector corresponding to $\sigma_{\max}(A)$, the largest singular value of A. It is the first principal component of the data matrix A. The optimal value of the problem is thus equal to

$$\phi_{\ell_1}(0) = (\lambda_{\max}(A^T A))^{1/2} = \sigma_{\max}(A).$$

Note that there is no reason to expect this vector to be sparse. On the other hand, for large enough γ, we will necessarily have $z^*(\gamma) = 0$, obtaining maximal sparsity. Indeed, since

$$\max_{z\neq 0} \frac{\|Az\|_2}{\|z\|_1} = \max_{z\neq 0} \frac{\|\sum_i z_i a_i\|_2}{\|z\|_1} \leq \max_{z\neq 0} \frac{\sum_i |z_i|\|a_i\|_2}{\sum_i |z_i|} = \max_i \|a_i\|_2 = \|a_{i^*}\|_2,$$

we get $\|Az\|_2 - \gamma\|z\|_1 < 0$ for all nonzero vectors z whenever γ is chosen to be strictly bigger than $\|a_{i^*}\|_2$. From now on we will assume that

$$\gamma < \|a_{i^*}\|_2. \tag{4}$$

Note that there is a trade-off between the value $\|Az^*(\gamma)\|_2$ and the sparsity of the solution $z^*(\gamma)$. The penalty parameter γ is introduced to "continuously" interpolate between the two extreme cases described above, with values in the interval $[0, \|a_{i^*}\|_2)$. It depends on the particular application whether sparsity is valued more than the explained variance, or vice versa, and to what extent. Due to these considerations, we will consider the solution of (3) to be a sparse principal component of A.

2.1.1 REFORMULATION

The reader will observe that the objective function in (3) is not convex, nor concave, and that the feasible set is of a high dimension if $p \ll n$. It turns out that these shortcomings are overcome by

considering the following reformulation:

$$\begin{aligned}\phi_{\ell_1}(\gamma) &= \max_{z\in\mathcal{B}^n} \|Az\|_2 - \gamma\|z\|_1 \\ &= \max_{z\in\mathcal{B}^n}\max_{x\in\mathcal{B}^p} x^T Az - \gamma\|z\|_1 \qquad (5)\\ &= \max_{x\in\mathcal{B}^p}\max_{z\in\mathcal{B}^n} \sum_{i=1}^n z_i(a_i^T x) - \gamma|z_i| \\ &= \max_{x\in\mathcal{B}^p}\max_{z'\in\mathcal{B}^n} \sum_{i=1}^n |z_i'|(|a_i^T x| - \gamma), \qquad (6)\end{aligned}$$

where $z_i = \mathrm{sign}(a_i^T x)z_i'$. In view of (4), there is some $x \in \mathcal{B}^n$ for which $a_i^T x > \gamma$. Fixing such x, solving the inner maximization problem for z' and then translating back to z, we obtain the closed-form solution

$$z_i^* = z_i^*(\gamma) = \frac{\mathrm{sign}(a_i^T x)[|a_i^T x| - \gamma]_+}{\sqrt{\sum_{k=1}^n [|a_k^T x| - \gamma]_+^2}}, \quad i = 1,\ldots,n. \qquad (7)$$

Problem (6) can therefore be written in the form

$$\boxed{\phi_{\ell_1}^2(\gamma) = \max_{x\in\mathcal{S}^p} \sum_{i=1}^n [|a_i^T x| - \gamma]_+^2.} \qquad (8)$$

Note that the objective function is differentiable and convex, and hence all local and global maxima must lie on the boundary, that is, on the unit Euclidean sphere $\mathcal{S}^p$. Also, in the case when $p \ll n$, formulation (8) requires to search a space of a much lower dimension than the initial problem (3).

2.1.2 SPARSITY

In view of (7), an optimal solution x^* of (8) defines a sparsity pattern of the vector z^*. In fact, the coefficients of z^* indexed by

$$I = \{i \mid |a_i^T x^*| > \gamma\}$$

are active while all others must be zero. Geometrically, active indices correspond to the defining hyperplanes of the polytope

$$\mathcal{D} = \{x \in \mathbf{R}^p \mid |a_i^T x| \le 1\}$$

that are (strictly) crossed by the line joining the origin and the point x^*/γ. Note that it is possible to say something about the sparsity of the solution even without the knowledge of x^*:

$$\gamma \ge \|a_i\|_2 \quad \Rightarrow \quad z_i^*(\gamma) = 0, \qquad i = 1,\ldots,n. \qquad (9)$$

2.2 Single-unit Sparse PCA via Cardinality Penalty

Instead of the ℓ_1-penalization, the authors of d'Aspremont et al. (2008) consider the formulation

$$\phi_{\ell_0}(\gamma) \stackrel{\mathrm{def}}{=} \max_{z\in\mathcal{B}^n} z^T\Sigma z - \gamma\,\|z\|_0, \qquad (10)$$

which directly penalizes the number of nonzero components (cardinality) of the vector z.

2.2.1 REFORMULATION

The reasoning of the previous section suggests the reformulation

$$\phi_{\ell_0}(\gamma) = \max_{x\in\mathcal{B}^p} \max_{z\in\mathcal{B}^n} (x^T A z)^2 - \gamma \|z\|_0, \tag{11}$$

where the maximization with respect to $z \in \mathcal{B}^n$ for a fixed $x \in \mathcal{B}^p$ has the closed form solution

$$z_i^* = z_i^*(\gamma) = \frac{[\mathrm{sign}((a_i^T x)^2 - \gamma)]_+ a_i^T x}{\sqrt{\sum_{k=1}^n [\mathrm{sign}((a_k^T x)^2 - \gamma)]_+ (a_k^T x)^2}}, \quad i = 1, \ldots, n. \tag{12}$$

In analogy with the ℓ_1 case, this derivation assumes that

$$\gamma < \|a_{i^*}\|_2^2,$$

so that there is $x \in \mathcal{B}^n$ such that $(a_i^T x)^2 - \gamma > 0$. Otherwise $z^* = 0$ is optimal. Formula (12) is easily obtained by analyzing (11) separately for fixed cardinality values of z. Hence, problem (10) can be cast in the following form

$$\boxed{\phi_{\ell_0}(\gamma) = \max_{x\in\mathcal{S}^p} \sum_{i=1}^n [(a_i^T x)^2 - \gamma]_+.} \tag{13}$$

Again, the objective function is convex, albeit nonsmooth, and the new search space is of particular interest if $p \ll n$. A different derivation of (13) for the $n = p$ case can be found in d'Aspremont et al. (2008).

2.2.2 SPARSITY

Given a solution x^* of (13), the set of active indices of z^* is given by

$$I = \{i \mid (a_i^T x^*)^2 > \gamma\}.$$

Geometrically, active indices correspond to the defining hyperplanes of the polytope

$$\mathcal{D} = \{x \in \mathbf{R}^p \mid |a_i^T x| \le 1\}$$

that are (strictly) crossed by the line joining the origin and the point $x^*/\sqrt{\gamma}$. As in the ℓ_1 case, we have

$$\gamma \ge \|a_i\|_2^2 \quad \Rightarrow \quad z_i^*(\gamma) = 0, \qquad i = 1, \ldots, n. \tag{14}$$

2.3 Block Sparse PCA via ℓ_1-Penalty

Consider the following block generalization of (5),

$$\phi_{\ell_1,m}(\gamma) \stackrel{\mathrm{def}}{=} \max_{\substack{X\in\mathcal{S}_m^p \\ Z\in[\mathcal{S}^n]^m}} \mathrm{Tr}(X^T A Z N) - \sum_{j=1}^m \gamma_j \sum_{i=1}^n |z_{ij}|, \tag{15}$$

where the m-dimensional vector $\gamma = [\gamma_1, \ldots, \gamma_m]^T$ is nonnegative and $N = \mathrm{Diag}(\mu_1, \ldots, \mu_m)$, with positive entries on the diagonal. The dimension m corresponds to the number of extracted components and is assumed to be smaller or equal to the rank of the data matrix, that is, $m \le \mathrm{Rank}(A)$.

Each parameter γ_j controls the sparsity of the corresponding component. It will be shown below that under some conditions on the parameters μ_j, the case $\gamma = 0$ recovers PCA. In that particular instance, any solution Z^* of (15) has orthonormal columns, although this is not explicitly enforced. For positive γ_j, the columns of Z^* are not expected to be orthogonal anymore. Most existing algorithms for computing several sparse principal components, for example, Zou et al. (2006), d'Aspremont et al. (2007) and Shen and Huang (2008), also do not impose orthogonal loading directions. Simultaneously enforcing sparsity and orthogonality seems to be a hard (and perhaps questionable) task.

2.3.1 REFORMULATION

Since problem (15) is completely decoupled in the columns of Z, that is,

$$\phi_{\ell_1,m}(\gamma) = \max_{X \in \mathcal{S}_m^p} \sum_{j=1}^{m} \max_{z_j \in \mathcal{S}^n} \mu_j x_j^T A z_j - \gamma_j \|z_j\|_1,$$

the closed-form solution (7) of (5) is easily adapted to the block formulation (15):

$$z_{ij}^* = z_{ij}^*(\gamma_j) = \frac{\operatorname{sign}(a_i^T x_j)[\mu_j |a_i^T x_j| - \gamma_j]_+}{\sqrt{\sum_{k=1}^n [\mu_j |a_k^T x_j| - \gamma_j]_+^2}}.$$

This leads to the reformulation

$$\boxed{\phi_{\ell_1,m}^2(\gamma) = \max_{X \in \mathcal{S}_m^p} \sum_{j=1}^{m} \sum_{i=1}^{n} [\mu_j |a_i^T x_j| - \gamma_j]_+^2,} \tag{16}$$

which maximizes a convex function $f : \mathbf{R}^{p \times m} \to \mathbf{R}$ on the Stiefel manifold $\mathcal{S}_m^p$.

2.3.2 SPARSITY

A solution X^* of (16) again defines the sparsity pattern of the matrix Z^*: the entry z_{ij}^* is active if

$$\mu_j |a_i^T x_j^*| > \gamma_j,$$

and equal to zero otherwise. For all $\gamma_j > \mu_j \max_i \|a_i\|_2$, the trivial solution $Z^* = 0$ is optimal.

2.3.3 BLOCK PCA

For $\gamma = 0$, problem (16) can be equivalently written in the form

$$\phi_{\ell_1,m}^2(0) = \max_{X \in \mathcal{S}_m^p} \operatorname{Tr}(X^T A A^T X N^2), \tag{17}$$

which has been well studied (see, e.g., Brockett 1991 and Absil et al. 2008). The solutions of (17) span the dominant m-dimensional invariant subspace of the matrix AA^T. Furthermore, if the parameters μ_j are all distinct, the columns of X^* are the m dominant eigenvectors of AA^T, that is, the m dominant left-eigenvectors of the data matrix A. The columns of the solution Z^* of (15) are thus the m dominant right singular vectors of A, that is, the PCA loading vectors. Such a matrix N

with distinct diagonal elements enforces the objective function in (17) to have isolated maximizers. In fact, if $N = I_m$, any point X^*U with X^* a solution of (17) and $U \in \mathcal{S}_m^m$ is also a solution of (17). In the case of sparse PCA, that is, $\gamma > 0$, the penalty term already ensures isolated maximizers, such that the diagonal elements of N do not have to be distinct. However, as it will be briefly illustrated in the forthcoming numerical experiments (Section 5), having distinct elements on the diagonal of N pushes towards sparse loading vectors that are more orthogonal.

2.4 Block Sparse PCA via Cardinality Penalty

The single-unit cardinality-penalized case can also be naturally extended to the block case:

$$\phi_{\ell_0,m}(\gamma) \stackrel{\text{def}}{=} \max_{\substack{X \in \mathcal{S}_m^p \\ Z \in [\mathcal{S}^n]^m}} \operatorname{Tr}(\operatorname{Diag}(X^T AZN)^2) - \sum_{j=1}^{m} \gamma_j \|z_j\|_0, \tag{18}$$

where the sparsity inducing vector $\gamma = [\gamma_1, \ldots, \gamma_m]^T$ is nonnegative and $N = \operatorname{Diag}(\mu_1, \ldots, \mu_m)$ with positive entries on the diagonal. In the case $\gamma = 0$, problem (20) is equivalent to (17), and therefore corresponds to PCA, provided that all μ_j are distinct.

2.4.1 REFORMULATION

Again, this block formulation is completely decoupled in the columns of Z,

$$\phi_{\ell_0,m}(\gamma) = \max_{X \in \mathcal{S}_m^p} \sum_{j=1}^{m} \max_{z_j \in \mathcal{S}^n} (\mu_j x_j^T A z_j)^2 - \gamma_j \|z_j\|_0,$$

so that the solution (12) of the single unit case provides the optimal columns z_i:

$$z_{ij}^* = z_{ij}^*(\gamma_j) = \frac{[\operatorname{sign}((\mu_j a_i^T x_j)^2 - \gamma_j)]_+ \mu_j a_i^T x_j}{\sqrt{\sum_{k=1}^{n} [\operatorname{sign}((\mu_j a_k^T x_j)^2 - \gamma_j)]_+ \mu_j^2 (a_k^T x_j)^2}}. \tag{19}$$

The reformulation of problem (18) is thus

$$\boxed{\phi_{\ell_0,m}(\gamma) = \max_{X \in \mathcal{S}_m^p} \sum_{j=1}^{m} \sum_{i=1}^{n} [(\mu_j a_i^T x_j)^2 - \gamma_j]_+,} \tag{20}$$

which maximizes a convex function $f : \mathbf{R}^{p \times m} \to \mathbf{R}$ on the Stiefel manifold $\mathcal{S}_m^p$.

2.4.2 SPARSITY

For a solution X^* of (20), the active entries z_{ij}^* of Z^* are given by the condition

$$(\mu_j a_i^T x_j^*)^2 > \gamma_j.$$

Hence for all $\gamma_j > \mu_j^2 \max_i \|a_i\|_2^2$, the optimal solution of (18) is $Z^* = 0$.

3. A Gradient Method for Maximizing Convex Functions

By $\mathbf{E}$ we denote an arbitrary finite-dimensional vector space; $\mathbf{E}^*$ is its conjugate, that is, the space of all linear functionals on $\mathbf{E}$. We equip these spaces with norms given by (2).

In this section we propose and analyze a simple gradient-type method for maximizing a convex function $f : \mathbf{E} \to \mathbf{R}$ on a compact set Q:

$$\boxed{f^* = \max_{x \in Q} f(x).} \tag{21}$$

Unless explicitly stated otherwise, we will *not* assume f to be differentiable. By $f'(x)$ we denote any subgradient of function f at x. By $\partial f(x)$ we denote its subdifferential.

At any point $x \in Q$ we introduce some measure for the first-order optimality conditions:

$$\Delta(x) \stackrel{\text{def}}{=} \max_{y \in Q} \langle f'(x), y - x \rangle.$$

It is clear that

$$\Delta(x) \geq 0, \tag{22}$$

with equality only at those points x where the gradient $f'(x)$ belongs to the normal cone to the set $\mathrm{Conv}(Q)$ at x.[4]

3.1 Algorithm

Consider the following simple algorithmic scheme.

Algorithm 1: Gradient scheme

input : $x_0 \in Q$
output: x_k (approximate solution of (21))
begin
  $k \longleftarrow 0$
  repeat
    $x_{k+1} \in \mathrm{Arg\,max}\{f(x_k) + \langle f'(x_k), y - x_k \rangle \mid y \in Q\}$
    $k \longleftarrow k + 1$
  until *a stopping criterion is satisfied*
end

Note that for example in the special case $Q = r\mathcal{S} \stackrel{\text{def}}{=} \{x \in \mathbf{E} \mid \|x\| = r\}$ or $Q = r\mathcal{B} \stackrel{\text{def}}{=} \{x \in \mathbf{E} \mid \|x\| \leq r\}$, the main step of Algorithm 1 can be written in an explicit form:

$$x_{k+1} = r \frac{G^{-1} f'(x_k)}{\|f'(x_k)\|_*}. \tag{23}$$

4. The normal cone to the set $\mathrm{Conv}(Q)$ at $x \in Q$ is *smaller* than the normal cone to the set Q. Therefore, the optimality condition $\Delta(x) = 0$ is *stronger* than the standard one.

3.2 Analysis

Our first convergence result is straightforward. Denote $\Delta_k \stackrel{\text{def}}{=} \min_{0\le i\le k} \Delta(x_i)$.

Theorem 1 *Let sequence $\{x_k\}_{k=0}^{\infty}$ be generated by Algorithm 1 as applied to a convex function f. Then the sequence $\{f(x_k)\}_{k=0}^{\infty}$ is monotonically increasing and $\lim_{k\to\infty} \Delta(x_k) = 0$. Moreover,*

$$\Delta_k \le \frac{f^* - f(x_0)}{k+1}.$$

Proof From convexity of f we immediately get

$$f(x_{k+1}) \ge f(x_k) + \langle f'(x_k), x_{k+1} - x_k\rangle = f(x_k) + \Delta(x_k),$$

and therefore, $f(x_{k+1}) \ge f(x_k)$ for all k. By summing up these inequalities for $k = 0, 1, \ldots, N-1$, we obtain

$$f^* - f(x_0) \ge f(x_k) - f(x_0) \ge \sum_{i=0}^{k} \Delta(x_i),$$

and the result follows. ■

For a sharper analysis, we need some technical assumptions on f and Q.

Assumption 1 *The norms of the subgradients of f are bounded from below on Q by a positive constant, that is,*

$$\delta_f \stackrel{def}{=} \min_{\substack{x\in Q \\ f'(x)\in\partial f(x)}} \|f'(x)\|_* > 0.$$

This assumption is not too binding because of the following result.

Proposition 2 *Assume that there exists a point $x' \notin Q$ such that $f(x') < f(x)$ for all $x \in Q$. Then*

$$\delta_f \ge \left[\min_{x\in Q} f(x) - f(x')\right] / \left[\max_{x\in Q} \|x - x'\|\right] > 0.$$

Proof Because f is convex, for any $x \in Q$ we have

$$0 < f(x) - f(x') \le \langle f'(x), x - x'\rangle \le \|f'(x)\|_* \|x - x'\|.$$

■

For our next convergence result we need to assume either strong convexity of f or strong convexity of the set $\mathrm{Conv}(Q)$.

Assumption 2 *Function f is* strongly convex, *that is, there exists a constant $\sigma_f > 0$ such that for any $x, y \in \mathbf{E}$*

$$f(y) \ge f(x) + \langle f'(x), y - x\rangle + \frac{\sigma_f}{2}\|y - x\|^2. \tag{24}$$

Note that convex functions satisfy inequality (24) with *convexity parameter* $\sigma_f = 0$.

Assumption 3 *The set* $\mathrm{Conv}(Q)$ *is* strongly convex*, that is, there is a constant* $\sigma_Q > 0$ *such that for any* $x, y \in \mathrm{Conv}(Q)$ *and* $\alpha \in [0,1]$ *the following inclusion holds:*

$$\alpha x + (1-\alpha)y + \frac{\sigma_Q}{2}\alpha(1-\alpha)\|x-y\|^2 \mathcal{S} \subset \mathrm{Conv}(Q). \tag{25}$$

Note that any set Q satisfies inclusion (25) with *convexity parameter* $\sigma_Q = 0$.

It can be shown (see Appendix A), that level sets of strongly convex functions with Lipschitz continuous gradient are again strongly convex. An example of such a function is the simple quadratic $x \mapsto \|x\|^2$. The level sets of this function correspond to Euclidean balls of varying sizes.

As we will see in Theorem 4, a better analysis of Algorithm 1 is possible if $\mathrm{Conv}(Q)$, the convex hull of the feasible set of problem (21), is strongly convex. Note that in the case of the two formulations (8) and (13) of the sparse PCA problem, the feasible set Q is the unit Euclidean sphere. Since the convex hull of the unit sphere is the unit ball, which is a strongly convex set, the feasible set of our sparse PCA formulations satisfies Assumption 3.

In the special case $Q = r\mathcal{S}$ for some $r > 0$, there is a simple proof that Assumption 3 holds with $\sigma_Q = \frac{1}{r}$. Indeed, for any $x, y \in \mathbf{E}$ and $\alpha \in [0,1]$, we have

$$\begin{aligned} \|\alpha x + (1-\alpha)y\|^2 &= \alpha^2\|x\|^2 + (1-\alpha)^2\|y\|^2 + 2\alpha(1-\alpha)\langle Gx, y\rangle \\ &= \alpha\|x\|^2 + (1-\alpha)\|y\|^2 - \alpha(1-\alpha)\|x-y\|^2. \end{aligned}$$

Thus, for $x, y \in r\mathcal{S}$ we obtain

$$\|\alpha x + (1-\alpha)y\| = \left[r^2 - \alpha(1-\alpha)\|x-y\|^2\right]^{1/2} \leq r - \frac{1}{2r}\alpha(1-\alpha)\|x-y\|^2.$$

Hence, we can take $\sigma_Q = \frac{1}{r}$.

The relevance of Assumption 3 is justified by the following technical observation.

Proposition 3 *If* f *is convex, then for any two subsequent iterates* x_k, x_{k+1} *of Algorithm 1*

$$\Delta(x_k) \geq \frac{\sigma_Q}{2}\|f'(x_k)\|_*\|x_{k+1} - x_k\|^2.$$

Proof We have noted in (22) that for convex f we have $\Delta(x_k) \geq 0$. We can thus concentrate on the situation when $\sigma_Q > 0$ and $f'(x_k) \neq 0$. Note that

$$\langle f'(x_k), x_{k+1} - y\rangle \geq 0 \quad \text{for all} \quad y \in \mathrm{Conv}(Q).$$

We will use this inequality with

$$y = y_\alpha \stackrel{\text{def}}{=} x_k + \alpha(x_{k+1} - x_k) + \frac{\sigma_Q}{2}\alpha(1-\alpha)\|x_{k+1} - x_k\|^2 \frac{G^{-1}f'(x_k)}{\|f'(x_k)\|_*}, \quad \alpha \in [0,1].$$

In view of (25), $y_\alpha \in \mathrm{Conv}(Q)$, and therefore

$$0 \geq \langle f'(x_k), y_\alpha - x_{k+1}\rangle = (1-\alpha)\langle f'(x_k), x_k - x_{k+1}\rangle + \frac{\sigma_Q}{2}\alpha(1-\alpha)\|x_{k+1} - x_k\|^2\|f'(x_k)\|_*.$$

Since α is an arbitrary value from $[0,1]$, the result follows. ■

We are now ready to refine our analysis of Algorithm 1.

Theorem 4 (Stepsize Convergence) *Let f be convex ($\sigma_f \geq 0$), and let either Assumption 2 ($\sigma_f > 0$) or Assumptions 1 ($\delta_f > 0$) and 3 ($\delta_Q > 0$) be satisfied. If $\{x_k\}$ is the sequence of points generated by Algorithm 1, then*

$$\sum_{k=0}^{\infty} \|x_{k+1} - x_k\|^2 \leq \frac{2(f^* - f(x_0))}{\sigma_Q \delta_f + \sigma_f}. \tag{26}$$

Proof Since f is convex, Proposition 3 gives

$$f(x_{k+1}) - f(x_k) \geq \Delta(x_k) + \frac{\sigma_f}{2}\|x_{k+1} - x_k\|^2 \geq \frac{1}{2}(\sigma_Q \delta_f + \sigma_f)\|x_{k+1} - x_k\|^2.$$

The additional assumptions of the theorem ensure that $\sigma_Q \delta_f + \delta_f > 0$. It remains to add the inequalities up for $k \geq 0$. ■

Theorem 4 gives an upper estimate on the number of iterations it takes for Algorithm 1 to produce a step of small size. Indeed,

$$k \geq \frac{2(f^* - f(x_0))}{\sigma_Q \delta_f + \sigma_f} \frac{1}{\varepsilon^2} - 1 \quad \Rightarrow \quad \min_{0 \leq i \leq k} \|x_{i+1} - x_i\| \leq \varepsilon.$$

It can be illustrated on simple examples that it is not in general possible to guarantee that the algorithm will produce iterates converging to a local maximizer. However, Theorem 4 guarantees that the set of the limit points is connected, and that all of them satisfy the first-order optimality condition. Also notice that, started from a local minimizer, the method will not move away.

3.2.1 TERMINATION

A reasonable stopping criterion for Algorithm 1 is the following: terminate once the relative change of the objective function becomes small:

$$\frac{f(x_{k+1}) - f(x_k)}{f(x_k)} \leq \varepsilon, \qquad \text{or equivalently,} \qquad f(x_{k+1}) \leq (1+\varepsilon) f(x_k).$$

3.3 Maximization with Spherical Constraints

Consider $\mathbf{E} = \mathbf{E}^* = \mathbf{R}^p$ with $G = I_p$ and $\langle s, x \rangle = \sum_i s_i x_i$, and let

$$Q = r\mathcal{S}^p = \{x \in \mathbf{R}^p \mid \|x\| = r\}.$$

Problem (21) takes on the form

$$\boxed{f^* = \max_{x \in r\mathcal{S}^p} f(x).} \tag{27}$$

Since Q is strongly convex ($\sigma_Q = \frac{1}{r}$), Theorem 4 is meaningful for any convex function f ($\sigma_f \geq 0$). The main step of Algorithm 1 can be written down explicitly (see (23)):

$$x_{k+1} = r \frac{f'(x_k)}{\|f'(x_k)\|_2}.$$

The following examples illustrate the connection of Algorithm 1 to classical methods.

Example 5 (Power Method) *In the special case of a quadratic objective function* $f(x) = \frac{1}{2}x^TCx$ *for some* $C \in \mathbf{S}^p_{++}$ *on the unit sphere* ($r = 1$)*, we have*

$$f^* = \tfrac{1}{2}\lambda_{max}(C),$$

and Algorithm 1 is equivalent to the power iteration method *for computing the largest eigenvalue of* C *(Golub and Van Loan, 1996). Hence for* $Q = \mathcal{S}^p$*, we can think of our scheme as a generalization of the power method. Indeed, our algorithm performs the following iteration:*

$$x_{k+1} = \frac{Cx_k}{\|Cx_k\|}, \quad k \geq 0.$$

Note that both δ_f *and* σ_f *are equal to the smallest eigenvalue of* C*, and hence the right-hand side of (26) is equal to*

$$\frac{\lambda_{max}(C) - x_0^TCx_0}{2\lambda_{min}(C)}. \tag{28}$$

Example 6 (Shifted Power Method) *If* C *is not positive semidefinite in the previous example, the objective function is not convex and our results are not applicable. However, this complication can be circumvented by instead running the algorithm with the shifted quadratic function*

$$\hat{f}(x) = \frac{1}{2}x^T(C + \omega I_p)x,$$

where $\omega > 0$ *satisfies* $\hat{C} = \omega I_p + C \in \mathbf{S}^p_{++}$*. On the feasible set, this change only adds a constant term to the objective function. The method, however, produces different sequence of iterates. Note that the constants* δ_f *and* σ_f *are also affected and, correspondingly, the estimate (28).*

The example above illustrates an easy "trick" to turn a convex convex objective function into a strongly convex one: one simply adds to the original objective function a strongly convex function that is constant on the boundary of the feasible set. The two formulations are equivalent since the objective functions differ only by a constant on the domain of interest. However, there is a clear trade-off. If the second term dominates the first term (say, by choosing very large ω), the algorithm will tend to treat the objective as a quadratic, and will hence tend to terminate in fewer iterations, nearer to the starting iterate. In the limit case, the method will not move away from the initial iterate.

3.4 Maximization with Orthonormality Constraints

Consider $\mathbf{E} = \mathbf{E}^* = \mathbf{R}^{p\times m}$, the space of $p \times m$ real matrices, with $m \leq p$. Note that for $m = 1$ we recover the setting of the previous section. We assume this space is equipped with the trace inner product: $\langle X, Y\rangle = \mathrm{Tr}(X^TY)$. The induced norm, denoted by $\|X\|_F \stackrel{\text{def}}{=} \langle X, X\rangle^{1/2}$, is the Frobenius norm (we let G be the identity operator). We can now consider various feasible sets, the simplest being a ball or a sphere. Due to nature of applications in this paper, let us concentrate on the situation when Q is a special subset of the sphere with radius $r = \sqrt{m}$, the Stiefel manifold:

$$Q = \mathcal{S}^p_m = \{X \in \mathbf{R}^{p\times m} \mid X^TX = I_m\}.$$

Problem (21) then takes on the following form:

$$\boxed{f^* = \max_{X\in\mathcal{S}^p_m} f(X).}$$

Using the duality of the nuclear and spectral matrix norms and Proposition 7 below it can be shown that $\mathrm{Conv}(Q)$ is equal to the unit spectral ball. It can be then further deduced that this set is not strongly convex ($\sigma_Q = 0$) and as a consequence, Theorem 4 is meaningful only if f is strongly convex ($\sigma_f > 0$). Of course, Theorem 1 applies also in the $\sigma_f = 0$ case.

At every iteration, Algorithm 1 needs to maximize a linear function over the Stiefel manifold. In the text that follows, it will be convenient to use the symbol $\mathrm{Polar}(C)$ for the U factor of the *polar decomposition* of matrix $C \in \mathbf{R}^{p\times m}$:

$$C = UP, \qquad U \in \mathcal{S}_m^p, \qquad P \in \mathbf{S}_+^m.$$

The complexity of the polar decomposition is $O(pm^2)$, with $p \geq m$. In view of the Proposition 7, the main step of Algorithm 1 can be written in the form

$$x_{k+1} = \mathrm{Polar}(f'(x_k)).$$

Proposition 7 *Let $C \in \mathbf{R}^{p\times m}$, with $m \leq p$, and denote by $\sigma_i(C)$, $i = 1,\ldots,m$, the singular values of C. Then*

$$\max_{X\in\mathcal{S}_m^p} \langle C, X\rangle = \sum_{i=1}^m \sigma_i(C) \qquad (= \|C\|_* = \mathrm{Tr}[(C^TC)^{1/2}]), \tag{29}$$

with maximizer $X^* = \mathrm{Polar}(C)$. If C is of full rank, then $\mathrm{Polar}(C) = C(C^TC)^{-1/2}$.

Proof Existence of the polar factorization in the nonsquare case is covered by Theorem 7.3.2 in Horn and Johnson (1985). Let $C = V\Sigma W^T$ be the singular value decomposition (SVD) of A; that is, V is $p \times p$ orthonormal, W is $m \times m$ orthonormal, and Σ is $p \times m$ diagonal with values $\sigma_i(A)$ on the diagonal. Then

$$\begin{aligned}
\max_{X\in\mathcal{S}_m^p} \langle C, X\rangle &= \max_{X\in\mathcal{S}_m^p} \langle V\Sigma W^T, X\rangle \\
&= \max_{X\in\mathcal{S}_m^p} \langle \Sigma, V^TXW\rangle \\
&= \max_{Z\in\mathcal{S}_m^p} \langle \Sigma, Z\rangle = \max_{Z\in\mathcal{S}_m^p} \sum_{i=1}^m \sigma_i(C) z_{ii} \leq \sum_i^m \sigma_i(C).
\end{aligned}$$

The third equality follows since the function $X \mapsto V^TXW$ maps $\mathcal{S}_m^p$ onto itself. Both factors of the polar decomposition of C can be easily read-off from the SVD. Indeed, if we let V' be the submatrix of V consisting of its first m columns and Σ' be the principal $m \times m$ submatrix of Σ, that is, a diagonal matrix with values $\sigma_i(C)$ on its diagonal, then $C = V'\Sigma'W^T = (V'W^T)(W\Sigma'W^T)$ and we can put $U = V'W^T$ and $P = W\Sigma'W^T$. To establish (29) it remains to note that

$$\langle C, U\rangle = \mathrm{Tr}\, P = \sum_i \lambda_i(P) = \sum_i \sigma_i(P) = \mathrm{Tr}(P^TP)^{1/2} = \mathrm{Tr}(C^TC)^{1/2} = \sum_i \sigma_i(C).$$

Finally, since $C^TC = PU^TUP = P^2$, we have $P = (C^TC)^{1/2}$, and in the full rank case we obtain $X^* = U = CP^{-1} = C(C^TC)^{-1/2}$. ■

Note that the block sparse PCA formulations (16) and (20) conform to this setting. Here is one more example:

Example 8 (Rectangular Procrustes Problem) *Let $C, X \in \mathbf{R}^{p\times m}$ and $D \in \mathbf{R}^{p\times p}$ and consider the following problem:*

$$\min\{\|C - DX\|_F^2 \mid X^T X = I_m\}. \tag{30}$$

Since $\|C-DX\|_F^2 = \|C\|_F^2 + \langle DX, DX\rangle - 2\langle CD, X\rangle$, by a similar shifting technique as in the previous example we can cast problem (30) in the following form

$$\max\{\omega\|X\|_F^2 - \langle DX, DX\rangle + 2\langle CD, X\rangle \mid X^T X = I_m\}.$$

For $\omega > 0$ large enough, the new objective function will be strongly convex. In this case our algorithm becomes similar to the gradient method proposed in Fraikin et al. (2008).

The standard Procrustes problem in the literature is a special case of (30) with $p = m$.

4. Algorithms for Sparse PCA

The solutions of the sparse PCA formulations of Section 2 provide locally optimal patterns of zeros and nonzeros for a vector $z \in \mathcal{S}^n$ (in the single-unit case) or a matrix $Z \in [\mathcal{S}^n]^m$ (in the block case). The sparsity-inducing penalty term used in these formulations biases however the values assigned to the nonzero entries, which should be readjusted by considering the sole objective of maximum variance. An algorithm for sparse PCA combines thus a method that identifies a "good" pattern of sparsity with a method that fills the active entries. In the sequel, we discuss the general block sparse PCA problem. The single-unit case is recovered in the particular case $m = 1$.

4.1 Methods for Pattern-finding

The application of our general method (Algorithm 1) to the four sparse PCA formulations of Section 2, that is, (8), (13), (16) and (20), leads to Algorithms 2, 3, 4 and 5 below, that provide a locally optimal pattern of sparsity for a matrix $Z \in [\mathcal{S}^n]^m$. This pattern is defined as a binary matrix $P \in \{0,1\}^{n\times m}$ such that $p_{ij} = 1$ if the loading z_{ij} is active and $p_{ij} = 0$ otherwise. So P is an indicator of the coefficients of Z that are zeroed by our method. The computational complexity of the single-unit algorithms (Algorithms 2 and 3) is $O(np)$ operations per iteration. The block algorithms (Algorithms 4 and 5) have complexity $O(npm)$ per iteration.

These algorithms need to be initialized at a point for which the associated sparsity pattern has *at least one* active element. In case of the single-unit algorithms, such an initial iterate $x \in \mathcal{S}^p$ is chosen parallel to the column of A with the largest norm, that is,

$$x = \frac{a_{i^*}}{\|a_{i^*}\|_2}, \quad \text{where} \quad i^* = \arg\max_i \|a_i\|_2. \tag{31}$$

For the block algorithms, a suitable initial iterate $X \in \mathcal{S}_m^p$ is constructed in a block-wise manner as $X = [x|X_\perp]$, where x is the unit-norm vector (31) and $X_\perp \in \mathcal{S}_{m-1}^p$ is orthogonal to x, that is, $x^T X_\perp = 0$.

The nonnegative parameters γ have to be chosen below the upper bounds derived in Section 2 and which are summarized in Table 1. Increasing the value of these parameters leads to solutions of smaller cardinality. There is however not explicit relationship between γ and the resulting cardinality. Since the proposed algorithms are fast, one can afford some trials and errors to reach a targeted cardinality. We however see it as an advantage not to enforce a fixed cardinality, since this information is often unknown a priori. As illustrated in the forthcoming numerical experiments (Section 5),

Algorithm 2	Single-unit ℓ_1	$\gamma \leq \max_i \|a_i\|_2$
Algorithm 3	Single-unit ℓ_0	$\gamma \leq \max_i \|a_i\|_2^2$
Algorithm 4	Block ℓ_1	$\gamma_j \leq \mu_j \max_i \|a_i\|_2$
Algorithm 5	Block ℓ_0	$\gamma_j \leq \mu_j^2 \max_i \|a_i\|_2^2$

Table 1: Theoretical upper-bounds on the sparsity parameters γ.

our algorithms are able to recover cardinalities that are best adapted to the model that underlies the data.

As previously explained, the parameters μ_j required by the block algorithms can be either identical (e.g., equal to one) or distinct (e.g., $\mu_j = \frac{1}{j}$). Since distinct μ_j leads to orthogonal loading vectors in the PCA case (i.e., $\gamma = 0$), they are expected to push towards orthogonality also in the sparse PCA case. Nevertheless, unless otherwise stated, the technical parameters μ_j will be set to one in what follows.

Let us finally mention that the input matrix A of these algorithms can be the data matrix itself as well as any matrix such that the factorization $\Sigma = A^T A$ of the covariance matrix holds. This property is very valuable when there is no access to the data and only the covariance matrix is available, or when the number of samples is greater than the number of variables. In this last case, the dimension p can be reduced to at most n by computing an eigenvalue decomposition or a Cholesky decomposition of the covariance matrix, for instance.

Algorithm 2: Single-unit sparse PCA method based on the ℓ_1-penalty (8)

input : Data matrix $A \in \mathbf{R}^{p \times n}$
Sparsity-controlling parameter $\gamma \geq 0$
Initial iterate $x \in \mathcal{S}^p$
output: A locally optimal sparsity pattern P
begin
  repeat
    $x \longleftarrow \sum_{i=1}^n [|a_i^T x| - \gamma]_+ \operatorname{sign}(a_i^T x) a_i$
    $x \longleftarrow \frac{x}{\|x\|}$
  until *a stopping criterion is satisfied*
  Construct vector $P \in \{0,1\}^n$ such that $\begin{cases} p_i = 1 & \text{if } |a_i^T x| > \gamma \\ p_i = 0 & \text{otherwise.} \end{cases}$
end

4.2 Post-processing

Once a "good" sparsity pattern P has been identified, the active entries of Z still have to be filled. To this end, we consider the optimization problem,

$$(X^*, Z^*) \stackrel{\text{def}}{=} \arg \max_{\substack{X \in \mathcal{S}_m^p \\ Z \in [\mathcal{S}^n]^m \\ Z_{P'} = 0}} \operatorname{Tr}(X^T A Z N), \tag{32}$$

Algorithm 3: Single-unit sparse PCA algorithm based on the ℓ_0-penalty (13)

input : Data matrix $A \in \mathbf{R}^{p\times n}$
Sparsity-controlling parameter $\gamma \geq 0$
Initial iterate $x \in \mathcal{S}^p$
output: A locally optimal sparsity pattern P
begin
  repeat
    $x \longleftarrow \sum_{i=1}^{n} [\mathrm{sign}((a_i^T x)^2 - \gamma)]_+ \, a_i^T x \, a_i$
    $x \longleftarrow \frac{x}{\|x\|}$
  until *a stopping criterion is satisfied*
  Construct vector $P \in \{0,1\}^n$ such that $\begin{cases} p_i = 1 & \text{if } (a_i^T x)^2 > \gamma \\ p_i = 0 & \text{otherwise.} \end{cases}$
end

Algorithm 4: Block sparse PCA algorithm based on the ℓ_1-penalty (16)

input : Data matrix $A \in \mathbf{R}^{p\times n}$
Sparsity-controlling vector $[\gamma_1, \ldots \gamma_m]^T \geq 0$
Parameters $\mu_1, \ldots, \mu_m > 0$
Initial iterate $X \in \mathcal{S}_m^p$
output: A locally optimal sparsity pattern P
begin
  repeat
    for $j = 1, \ldots, m$ **do**
      $x_j \longleftarrow \sum_{i=1}^{n} \mu_j [\mu_j |a_i^T x_j| - \gamma_j]_+ \, \mathrm{sign}(a_i^T x) a_i$
    $X \longleftarrow \mathrm{Polar}(X)$
  until *a stopping criterion is satisfied*
  Construct matrix $P \in \{0,1\}^{n\times m}$ such that $\begin{cases} p_{ij} = 1 & \text{if } \mu_j |a_i^T x_j| > \gamma_j \\ p_{ij} = 0 & \text{otherwise.} \end{cases}$
end

where $P' \in \{0,1\}^{n\times m}$ is the complement of P, $Z_{P'}$ denotes the entries of Z that are constrained to zero and $N = \mathrm{Diag}(\mu_1, \ldots, \mu_m)$ with strictly positive μ_i. Problem (32) assigns the active part of the loading vectors Z to maximize the variance explained by the resulting components. Without loss of generality, each column of P is assumed to contain active elements.

In the single-unit case $m = 1$, an explicit solution of (32) is available,

$$\begin{aligned} X^* &= u, \\ Z_P^* &= v \text{ and } Z_{P'}^* = 0, \end{aligned} \tag{33}$$

where σuv^T with $\sigma > 0$, $u \in \mathcal{B}^p$ and $v \in \mathcal{B}^{\|P\|_0}$ is a rank one singular value decomposition of the matrix A_P, that corresponds to the submatrix of A containing the columns related to the active entries. The post-processing (33) is equivalent to the *variational renormalization* proposed by Moghaddam et al. (2006).

Algorithm 5: Block sparse PCA algorithm based on the ℓ_0-penalty (20)

input : Data matrix $A \in \mathbf{R}^{p\times n}$
Sparsity-controlling vector $[\gamma_1,\dots\gamma_m]^T \geq 0$
Parameters $\mu_1,\dots,\mu_m > 0$
Initial iterate $X \in \mathcal{S}_m^p$
output: A locally optimal sparsity pattern P
begin
  repeat
    for $j = 1,\dots,m$ **do**
      $x_j \longleftarrow \sum_{i=1}^n \mu_j^2[\mathrm{sign}((\mu_j a_i^T x_j)^2 - \gamma_j)]_+ \, a_i^T x_j \, a_i$
    $X \longleftarrow \mathrm{Polar}(X)$
  until *a stopping criterion is satisfied*
  Construct matrix $P \in \{0,1\}^{n\times m}$ such that $\begin{cases} p_{ij} = 1 & \text{if } (\mu_j a_i^T x_j)^2 > \gamma_j \\ p_{ij} = 0 & \text{otherwise.} \end{cases}$
end

Although an exact solution of (32) is hard to compute in the block case $m > 1$, a local maximizer can be efficiently computed by optimizing alternatively with respect to one variable while keeping the other ones fixed. The following lemmas provide an explicit solution to each of these subproblems.

Lemma 9 *For a fixed $Z \in [\mathcal{S}^n]^m$, a solution X^* of*

$$\max_{X\in\mathcal{S}_m^p} \mathrm{Tr}(X^T AZN)$$

is provided by the U factor of the polar decomposition of the product AZN.

Proof See Proposition 7. ■

Lemma 10 *The solution*

$$Z^* \stackrel{def}{=} \arg \max_{\substack{Z\in[\mathcal{S}^n]^m \\ Z_{P'}=0}} \mathrm{Tr}(X^T AZN), \tag{34}$$

is at any point $X \in \mathcal{S}_m^p$ defined by the two conditions $Z_P^* = (A^T XND)_P$ and $Z_{P'}^* = 0$, where D is a positive diagonal matrix that normalizes each column of Z^* to unit norm, that is,

$$D \;=\; \mathrm{Diag}(NX^T AA^T XN)^{-\frac{1}{2}}.$$

Proof The Lagrangian of the optimization problem (34) is

$$\mathcal{L}(Z,\Lambda_1,\Lambda_2) = \mathrm{Tr}(X^T AZN) - \mathrm{Tr}(\Lambda_1(Z^T Z - I_m)) - \mathrm{Tr}(\Lambda_2^T Z),$$

where the Lagrangian multipliers $\Lambda_1 \in \mathbf{R}^{m\times m}$ and $\Lambda_2 \in \mathbf{R}^{n\times m}$ have the following properties: Λ_1 is an invertible diagonal matrix and $(\Lambda_2)_P = 0$. The first order optimality conditions of (34) are thus

$$\begin{aligned} A^T XN - 2Z\Lambda_1 - \Lambda_2 &= 0 \\ \mathrm{Diag}(Z^T Z) &= I_m \\ Z_P &= 0. \end{aligned}$$

Hence, any stationary point Z^* of (34) satisfies $Z^*_P = (A^T XND)_P$ and $Z^*_{P'} = 0$, where D is a diagonal matrix that normalizes the columns of Z^* to unit norm. The second order optimality condition imposes the diagonal matrix D to be positive. Such a D is unique and given by $D = \mathrm{Diag}(NX^T AA^T XN)^{-\frac{1}{2}}$. ■

The alternating optimization scheme is summarized in Algorithm 6, which computes a local solution of (32). A judicious initialization is provided by an accumulation point of the algorithm for pattern-finding, that is, Algorithms 4 and 5.

Algorithm 6: Alternating optimization scheme for solving (32)

input : Data matrix $A \in \mathbf{R}^{p\times n}$
Sparsity pattern $P \in \{0,1\}^{n\times m}$
Matrix $N = \mathrm{Diag}(\mu_1, \ldots, \mu_m)$
Initial iterate $X \in \mathcal{S}^p_m$
output: A local minimizer (X, Z) of (32)
begin
  repeat
    $Z \longleftarrow A^T XN$
    $Z \longleftarrow Z\ \mathrm{Diag}(Z^T Z)^{-\frac{1}{2}}$
    $Z_{\bar{P}} \longleftarrow 0$
    $X \longleftarrow \mathrm{Polar}(AZN)$
  until *a stopping criterion is satisfied*
end

It should be noted that Algorithm 6 is a postprocessing heuristic that, strictly speaking, is required only for the ℓ_1 block formulation (Algorithm 4). In fact, since the cardinality penalty only depends on the sparsity pattern P and not on the actual values assigned to Z_P, a solution (X^*, Z^*) of Algorithms 3 or 5 is also a local maximizer of (32) for the resulting pattern P. This explicit solution provides a good alternative to Algorithm 6. In the single unit case with ℓ_1 penalty (Algorithm 2), the solution (33) is available.

4.3 Sparse PCA Algorithms

To sum up, in this paper we propose four sparse PCA algorithms, each combining a method to identify a "good" sparsity pattern with a method to fill the active entries of the m loading vectors.

They are summarized in Table 2. The MATLAB code of these GPower[5] algorithms is available on the authors' websites.[6]

	Computation of P	Computation of Z_P
GPower_{ℓ_1}	Algorithm 2	Equation (33)
GPower_{ℓ_0}	Algorithm 3	Equation (12)
$\mathsf{GPower}_{\ell_1,m}$	Algorithm 4	Algorithm 6
$\mathsf{GPower}_{\ell_0,m}$	Algorithm 5	Equation (19)

Table 2: New algorithms for sparse PCA.

4.4 Deflation Scheme

For the sake of completeness, we recall a classical deflation process for computing m sparse principal components with a single-unit algorithm (d'Aspremont et al., 2007). Let $z \in \mathbf{R}^n$ be a unit-norm sparse loading vector of the data A. Subsequent directions can be sequentially obtained by computing a dominant sparse component of the residual matrix $A - xz^T$, where $x = Az$ is the vector that solves

$$\min_{x \in \mathbf{R}^p} \|A - xz^T\|_F.$$

Further deflation techniques for sparse PCA have been proposed by Mackey (2008).

4.5 Connection with Existing Sparse PCA Methods

As previously mentioned, our ℓ_0-based single-unit algorithm GPower_{ℓ_0} rests on the same reformulation (13) as the greedy algorithm proposed by d'Aspremont et al. (2008).

There is also a clear connection between both single-unit algorithms GPower_{ℓ_1} and GPower_{ℓ_0} and the rSVD algorithms of Shen and Huang (2008), which solve the optimization problems

$$\min_{\substack{z \in \mathbf{R}^n \\ x \in \mathcal{S}^p}} \|A - xz^T\|_F^2 + 2\gamma\|z\|_1 \quad \text{and} \quad \min_{\substack{z \in \mathbf{R}^n \\ x \in \mathcal{S}^p}} \|A - xz^T\|_F^2 + \gamma\|z\|_0$$

by alternating optimization over one variable (either x or z) while fixing the other one. It can be shown that an update on x amounts to the iterations of Algorithms 2 and 3, depending on the penalty type. Although rSVD and GPower were derived differently, it turns out that, except for the initialization and post-processing phases, the algorithms *are identical*. There are, however, several benefits to our approach: 1) we are able to analyze convergence properties of the method, 2) we show that the core algorithm can be derived as a special case of a generalization of the power method (and hence more applications are possible), 3) we give generalizations from single unit case to block case, 4) our approach uncovers the possibility of a very useful initialization technique, 5) we equip the method with a practical postprocessing phase, 6) we provide a link with the formulation of d'Aspremont et al. (2008).

5. Our algorithms are named GPower where the "G" stands for *generalized* or *gradient*.
6. Websites are `http://www.inma.ucl.ac.be/~richtarik` and `http://www.montefiore.ulg.ac.be/~journee`.

5. Numerical Experiments

In this section, we evaluate the proposed power algorithms against existing sparse PCA methods. Three competing methods are considered in this study: a greedy scheme aimed at computing a local maximizer of (10) (Approximate Greedy Search Algorithm, d'Aspremont et al. (2008)), the SPCA algorithm (Zou et al., 2006) and the sPCA-rSVD algorithm (Shen and Huang, 2008). We do not include the DSPCA algorithm (d'Aspremont et al., 2007) in our numerical study. This method solves a convex relaxation of the sparse PCA problem and has a large per-iteration computational complexity of $O(n^3)$ compared to the other methods. Table 3 lists the considered algorithms.

GPower$_{\ell_1}$	Single-unit sparse PCA via ℓ_1-penalty
GPower$_{\ell_0}$	Single-unit sparse PCA via ℓ_0-penalty
GPower$_{\ell_1,m}$	Block sparse PCA via ℓ_1-penalty
GPower$_{\ell_0,m}$	Block sparse PCA via ℓ_0-penalty
Greedy	Greedy method
SPCA	SPCA algorithm
rSVD$_{\ell_1}$	sPCA-rSVD algorithm with an ℓ_1-penalty ("soft thresholding")
rSVD$_{\ell_0}$	sPCA-rSVD algorithm with an ℓ_0-penalty ("hard thresholding")

Table 3: Sparse PCA algorithms we compare in this section.

These algorithms are compared on random data (Sections 5.1 and 5.2) as well as on real data (Sections 5.3 and 5.4). All numerical experiments are performed in MATLAB. The parameter ε in the stopping criterion of the GPower algorithms has been fixed to 10^{-4}. MATLAB implementations of the SPCA algorithm and the greedy algorithm have been rendered available by Zou et al. (2006) and d'Aspremont et al. (2008). We have, however, implemented the sPCA-rSVD algorithm on our own (Algorithm 1 in Shen and Huang 2008), and use it with the same stopping criterion as for the GPower algorithms. This algorithm initializes with the best rank-one approximation of the data matrix. This is done by a first run of the algorithm with the sparsity-inducing parameter γ that is set to zero.

Given a data matrix $A \in \mathbf{R}^{p\times n}$, the considered sparse PCA algorithms provide m unit-norm sparse loading vectors stored in the matrix $Z \in [\mathcal{S}^n]^m$. The samples of the associated components are provided by the m columns of the product AZ. The variance explained by these m components is an important comparison criterion of the algorithms. In the simple case $m = 1$, the variance explained by the component Az is

$$\mathrm{Var}(z) = z^T A^T A z.$$

When z corresponds to the first principal loading vector, the variance is $\mathrm{Var}(z) = \sigma_{\max}(A)^2$. In the case $m > 1$, the derived components are likely to be correlated. Hence, summing up the variance explained individually by each of the components overestimates the variance explained simultaneously by all the components. This motivates the notion of *adjusted variance* proposed by Zou et al. (2006). The adjusted variance of the m components $Y = AZ$ is defined as

$$\mathrm{AdjVar}\ Z = \mathrm{Tr}\, R^2,$$

where $Y = QR$ is the QR decomposition of the components sample matrix Y ($Q \in \mathcal{S}_m^p$ and R is an $m \times m$ upper triangular matrix).

5.1 Random Data Drawn from a Sparse PCA Model

In this section, we follow the procedure proposed by Shen and Huang (2008) to generate random data with a covariance matrix having sparse eigenvectors. To this end, a covariance matrix is first synthesized through the eigenvalue decomposition $\Sigma = VDV^T$, where the first m columns of $V \in \mathbf{R}^{n\times n}$ are pre-specified sparse orthonormal vectors. A data matrix $A \in \mathbf{R}^{p\times n}$ is then generated by drawing p samples from a zero-mean normal distribution with covariance matrix Σ, that is, $A \sim N(0,\Sigma)$.

Consider a setup with $n = 500, m = 2$ and $p = 50$, where the two orthonormal eigenvectors are specified as follows

$$\left\{\begin{array}{rcll} v_{1i} &=& \frac{1}{\sqrt{10}} & \text{for} \quad i = 1,\ldots,10, \\ v_{1i} &=& 0 & \text{otherwise,} \end{array}\right. \qquad \left\{\begin{array}{rcll} v_{2i} &=& \frac{1}{\sqrt{10}} & \text{for} \quad i = 11,\ldots,20, \\ v_{2i} &=& 0 & \text{otherwise,} \end{array}\right.$$

The remaining eigenvectors $v_j,\ j > 2$, are chosen arbitrarily, and the eigenvalues are fixed at the following values

$$\left\{\begin{array}{rcll} d_{11} &=& 400 & \\ d_{22} &=& 300 & \\ d_{jj} &=& 1, & \text{for} \quad j = 3,\ldots,500. \end{array}\right.$$

We generate 500 data matrices $A \in \mathbf{R}^{p\times n}$ and employ the four GPower algorithms as well as Greedy to compute two unit-norm sparse loading vectors $z_1, z_2 \in \mathbf{R}^{500}$, which are hoped to be close to v_1 and v_2. We consider the model underlying the data to be *successfully identified* (or *recovered*) when both quantities $|v_1^T z_1|$ and $|v_2^T z_2|$ are greater than 0.99.

Two simple alternative strategies are compared for choosing the sparsity-inducing parameters γ_1 and γ_2 required by the GPower algorithms. First, we choose them uniformly at random, between the theoretical bounds. Second, we fix them to reasonable a priori values; in particular, the midpoints of the corresponding admissible interval. For the block algorithm GPower$_{\ell_1,m}$, the parameter γ_2 is fixed at 10 percent of the corresponding upper bound. This value was chosen by limited trial and error to give good results for the particular data analyzed. We do not intend to suggest that this is a recommended choice in general. The values of the sparsity-inducing parameters for the ℓ_0-based GPower algorithms are systematically chosen as the squares of the values chosen for their ℓ_1 counterparts. More details on the selection of γ_1 and γ_2 are provided in Table 4. Concerning the parameters μ_1 and μ_2 used by the block algorithms, both situations $\mu_1 = \mu_2$ and $\mu_1 > \mu_2$ have been considered. Note that Greedy requires to specify the targeted cardinalities as an input, that is, ten nonzeros entries for both loading vectors.

In Table 5, we provide the average of the scalar products $|z_1^T z_2|$, $|v_1^T z_1|$ and $|v_2^T z_2|$ for 500 data matrices with the covariance matrix Σ. The proportion of successful identification of the vectors v_1 and v_2 is also given. The table shows that the GPower algorithms are robust with respect to the choice of the sparsity inducing parameters γ. Good values of γ_1 and γ_2 are easily found by trial and error. The chances of recovery of the sparse model underlying the data are rather good, and some versions of the algorithms successfully recover the sparse model even when the parameters γ are chosen at random. The GPower algorithms do not appear to be as successful as Greedy, which managed to correctly identify vectors v_1 and v_2 in all tests. Note that while the latter method requires the exact knowledge of the cardinality of each component, the GPower algorithms find the sparse model that fits the data best without this information. This property of the GPower algorithms is

Algorithm	Random	Fixed
GPower$_{\ell_1}$	γ_1 uniform distrib. on $[0, \max_i \|a_i\|_2]$ γ_2 uniform distrib. on $[0, \max_i \|a'_i\|_2]$	$\gamma_1 = \frac{1}{2}\max_i \|a_i\|_2$ $\gamma_2 = \frac{1}{2}\max_i \|a'_i\|_2$
GPower$_{\ell_0}$	$\sqrt{\gamma_1}$ uniform distrib. on $[0, \max_i \|a_i\|_2]$ $\sqrt{\gamma_2}$ uniform distrib. on $[0, \max_i \|a'_i\|_2]$	$\gamma_1 = \frac{1}{4}\max_i \|a_i\|_2^2$ $\gamma_2 = \frac{1}{4}\max_i \|a'_i\|_2^2$
GPower$_{\ell_1,m}$ with $\mu_1 = \mu_2 = 1$	γ_1 uniform distrib. on $[0, \max_i \|a_i\|_2]$ γ_2 uniform distrib. on $[0, \max_i \|a_i\|_2]$	$\gamma_1 = \frac{1}{2}\max_i \|a_i\|_2$ $\gamma_2 = \frac{1}{10}\max_i \|a_i\|_2$
GPower$_{\ell_0,m}$ with $\mu_1 = \mu_2 = 1$	$\sqrt{\gamma_1}$ uniform distrib. on $[0, \max_i \|a_i\|_2]$ $\sqrt{\gamma_2}$ uniform distrib. on $[0, \max_i \|a_i\|_2]$	$\gamma_1 = \frac{1}{4}\max_i \|a_i\|_2^2$ $\gamma_2 = \frac{1}{100}\max_i \|a_i\|_2^2$
GPower$_{\ell_1,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	γ_1 uniform distrib. on $[0, \max_i \|a_i\|_2]$ γ_2 uniform distrib. on $[0, \frac{1}{2}\max_i \|a_i\|_2]$	$\gamma_1 = \frac{1}{2}\max_i \|a_i\|_2$ $\gamma_2 = \frac{1}{20}\max_i \|a_i\|_2$
GPower$_{\ell_0,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	$\sqrt{\gamma_1}$ uniform distrib. on $[0, \max_i \|a_i\|_2]$ $\sqrt{\gamma_2}$ uniform distrib. on $[0, \frac{1}{2}\max_i \|a_i\|_2]$	$\gamma_1 = \frac{1}{4}\max_i \|a_i\|_2^2$ $\gamma_2 = \frac{1}{400}\max_i \|a_i\|_2^2$

Table 4: Details on the random and fixed choices of the sparsity-inducing parameters γ_1 and γ_2 leading to the results displayed in Table 5. Matrix A' used in the case of the single-unit algorithms denotes the residual matrix after one deflation step.

valuable in real-data settings, where little or nothing is known a priori about the cardinality of the components.

Looking at the values reported in Table 5, we observe that the block GPower algorithms are more likely to obtain loading vectors that are "more orthogonal" when using parameters μ_j which are distinct.

Algorithm	γ	$\lvert z_1^T z_2\rvert$	$\lvert v_1^T z_1\rvert$	$\lvert v_2^T z_2\rvert$	Chance of success
GPower$_{\ell_1}$	random	$15.8\ 10^{-3}$	0.9693	0.9042	0.71
GPower$_{\ell_0}$	random	$15.7\ 10^{-3}$	0.9612	0.8990	0.69
GPower$_{\ell_1,m}$ with $\mu_1 = \mu_2 = 1$	random	$10.1\ 10^{-3}$	0.8370	0.2855	0.06
GPower$_{\ell_0,m}$ with $\mu_1 = \mu_2 = 1$	random	$9.2\ 10^{-3}$	0.8345	0.3109	0.07
GPower$_{\ell_1,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	random	$1.8\ 10^{-4}$	0.8300	0.3191	0.09
GPower$_{\ell_0,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	random	$1.5\ 10^{-4}$	0.8501	0.3001	0.09
GPower$_{\ell_1}$	fixed	0	0.9998	0.9997	1
GPower$_{\ell_0}$	fixed	0	0.9998	0.9997	1
GPower$_{\ell_1,m}$ with $\mu_1 = \mu_2 = 1$	fixed	$4.25\ 10^{-2}$	0.9636	0.8114	0.63
GPower$_{\ell_0,m}$ with $\mu_1 = \mu_2 = 1$	fixed	$3.77\ 10^{-2}$	0.9663	0.7990	0.67
GPower$_{\ell_1,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	fixed	$1.8\ 10^{-3}$	0.9875	0.9548	0.89
GPower$_{\ell_0,m}$ with $\mu_1 = 1$ and $\mu_2 = 0.5$	fixed	$6.7\ 10^{-5}$	0.9937	0.9654	0.96
PCA	–	0	0.9110	0.9063	0
Greedy	–	0	0.9998	0.9997	1

Table 5: Average of the quantities $\lvert z_1^T z_2\rvert$, $\lvert v_1^T z_1\rvert$, $\lvert v_2^T z_2\rvert$ and proportion of successful identifications of the two dominant sparse eigenvectors of Σ by extracting two sparse principal components from 500 data matrices. The Greedy algorithm requires prior knowledge of the cardinalities of each component, while the GPower algorithms are very likely to identify the underlying sparse model without this information.

Table 5 does not include results for the SPCA algorithm because of our limited experience with it and the absence of such experiments in the literature. However, in view of the connections developed in Section 4.5, we do expect that the rSVD methods will exhibit similar flexibility to sparsity parameter tuning.

5.2 Random Data without Underlying Sparse PCA Model

All random data matrices $A \in \mathbf{R}^{p\times n}$ considered in this section are generated according to a Gaussian distribution, with zero mean and unit variance.

5.2.1 TRADE-OFF CURVES

Let us first compare the single-unit algorithms, which provide a unit-norm sparse loading vector $z \in \mathbf{R}^n$. We first plot the variance explained by the extracted component against the cardinality of the resulting loading vector z. For each algorithm, the sparsity-inducing parameter is incrementally increased to obtain loading vectors z with a cardinality that decreases from n to 1. The results displayed in Figure 1 are averages of computations on 100 random matrices with dimensions $p = 100$ and $n = 300$. The considered sparse PCA methods aggregate in two groups: GPower$_{\ell_1}$, GPower$_{\ell_0}$, Greedy and rSVD$_{\ell_0}$ outperform the SPCA and rSVD$_{\ell_1}$. It seems that these latter methods perform worse because of the ℓ_1 penalty term used in them. If one, however, post-processes the active part of z according to (33), as we do in GPower$_{\ell_1}$, all sparse PCA methods reach the same performance.

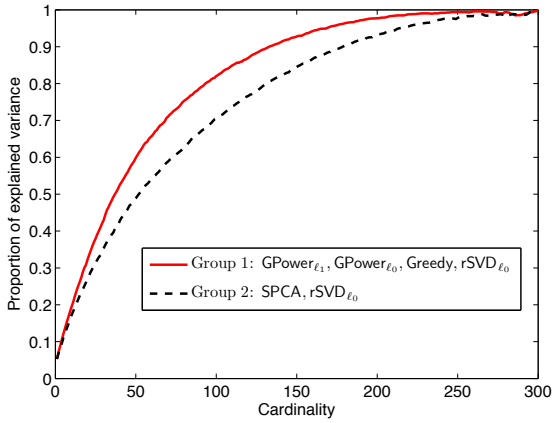


Figure 1: **Trade-off curves** between explained variance and cardinality. The vertical axis is the ratio $\mathrm{Var}(z_{\mathrm{sPCA}})/\mathrm{Var}(z_{\mathrm{PCA}})$, where the loading vector z_{sPCA} is computed by sparse PCA and z_{PCA} is the first principal loading vector. The considered algorithms aggregate in two groups: GPower$_{\ell_1}$, GPower$_{\ell_0}$, Greedy and rSVD$_{\ell_0}$ (top curve), and SPCA and rSVD$_{\ell_1}$ (bottom curve). For a fixed cardinality value, the methods of the first group explain more variance.

5.2.2 CONTROLLING SPARSITY WITH γ

Among the considered methods, the greedy approach is the only one to directly control the cardinality of the solution, that is, the desired cardinality is an input of the algorithm. The other methods require a parameter controlling the trade-off between variance and cardinality. Increasing this parameter leads to solutions with smaller cardinality, but the resulting number of nonzero elements can not be precisely predicted. In Figure 2, we plot the average relationship between the parameter γ and the resulting cardinality of the loading vector z for the two algorithms GPower$_{\ell_1}$ and GPower$_{\ell_0}$. In view of (9) (resp. (14)), the entries i of the loading vector z obtained by the GPower$_{\ell_1}$ (resp. GPower$_{\ell_0}$) algorithm satisfying

$$\|a_i\|_2 \leq \gamma \quad \text{(resp. } \|a_i\|_2^2 \leq \gamma\text{)} \tag{35}$$

have to be zero. Taking into account the distribution of the norms of the columns of A, this provides for every γ a theoretical upper bound on the expected cardinality of the resulting vector z.

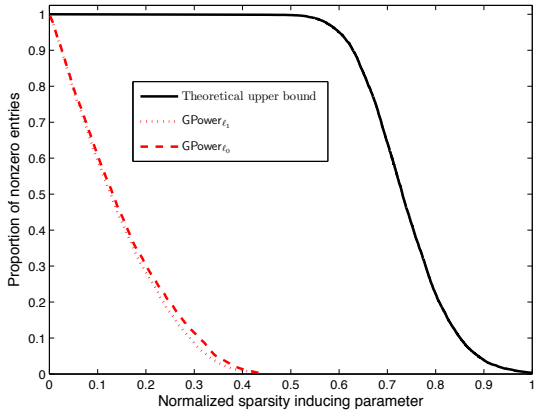


Figure 2: Dependence of cardinality on the value of the sparsity-inducing parameter γ. In case of the GPower$_{\ell_1}$ algorithm, the horizontal axis shows $\gamma/\|a_{i^*}\|_2$, whereas for the GPower$_{\ell_0}$ algorithm, we use $\sqrt{\gamma}/\|a_{i^*}\|_2$. The theoretical upper bound is therefore identical for both methods. The plots are averages based on 100 test problems of size $p = 100$ and $n = 300$.

5.2.3 GREEDY VERSUS THE REST

From the experiments reported above, Greedy and the GPower methods appear to have similar performance in terms of quality of the obtained solution. Moreover, Greedy computes a full path of solutions up to a chosen cardinality, and does not have to deal with the issue of tuning the sparsity parameter γ. The price of this significant advantage of Greedy is its heavy computational load. In order to compare the empirical computational complexities of different algorithms, we display in Figure 3 the average time required to extract one sparse component from Gaussian matrices of dimensions $p = 100$ and $n = 300$. One immediately notices that the greedy method slows down significantly as cardinality increases, whereas the speed of the other considered algorithms does not depend on cardinality. Since on average Greedy is much slower than the other methods, even for

low cardinalities, and because we aim at large-scale applications where the computational load of Greedy would be prohibitive, we discard it from the following numerical experiments.

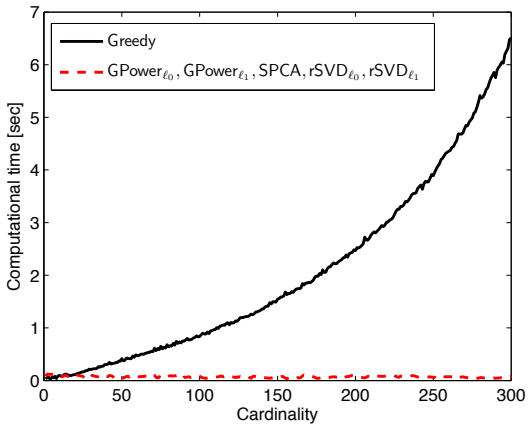


Figure 3: The computational complexity of Greedy grows significantly with cardinality of the resulting loading vector. The speed of the other methods is unaffected by the cardinality target. The single dashed line is representative of the speed of the methods GPower$_{\ell_1}$, GPower$_{\ell_0}$, SPCA, rSVD$_{\ell_0}$, rSVD$_{\ell_0}$ in this test.

5.2.4 SPEED AND SCALING TEST

In Tables 6 and 7 we compare the speed of the remaining algorithms. Table 6 deals with problems with a fixed aspect ratio $n/p = 10$, whereas in Table 7, p is fixed at 500, and exponentially increasing values of n are considered. For the GPower$_{\ell_1}$ method, the sparsity inducing parameter γ was set to 10% of the upper bound $\gamma_{\max} = \|a_{i^*}\|_2$. For the GPower$_{\ell_0}$ method, γ was set to 1% of $\gamma_{\max} = \|a_{i^*}\|_2^2$ in order to aim for solutions of comparable cardinalities (see (35)). These two parameters have also been used for the rSVD$_{\ell_1}$ and the rSVD$_{\ell_0}$ methods, respectively. Concerning SPCA, the sparsity parameter has been chosen by trial and error to get, on average, solutions with similar cardinalities as obtained by the other methods. The values displayed in Tables 6 and 7 correspond to the average running times of the algorithms on 100 test instances for each problem size. In both tables, the new methods GPower$_{\ell_1}$ and GPower$_{\ell_0}$ are the fastest. The difference in speed between GPower$_{\ell_1}$ and GPower$_{\ell_0}$ results from different approaches to fill the active part of z: GPower$_{\ell_1}$ requires to compute a rank-one approximation of a submatrix of A (see Equation (33)), whereas the explicit solution (12) is available to GPower$_{\ell_0}$. The linear complexity of the algorithms in the problem size n is clearly visible in Table 7.

5.2.5 DIFFERENT CONVERGENCE MECHANISMS

Figure 4 illustrates how the trade-off between explained variance and sparsity evolves in the time of computation for the two methods GPower$_{\ell_1}$ and rSVD$_{\ell_1}$. In case of the GPower$_{\ell_1}$ algorithm, the initialization point (31) provides a good approximation of the final cardinality. This method then works on maximizing the variance while keeping the sparsity at a low level throughout. The rSVD$_{\ell_1}$

$p \times n$	100×1000	250×2500	500×5000	750×7500	1000×10000
GPower$_{\ell_1}$	0.10	0.86	2.45	4.28	5.86
GPower$_{\ell_0}$	0.03	0.42	1.21	2.07	2.85
SPCA	0.24	2.92	14.5	40.7	82.2
rSVD$_{\ell_1}$	0.19	2.42	3.97	7.51	9.59
rSVD$_{\ell_0}$	0.18	2.14	3.85	6.94	8.34

Table 6: Average computational time for the extraction of one component (in seconds).

$p \times n$	500×1000	500×2000	500×4000	500×8000	500×16000
GPower$_{\ell_1}$	0.42	0.92	2.00	4.00	8.54
GPower$_{\ell_0}$	0.18	0.42	0.96	2.14	4.55
SPCA	5.20	7.20	12.0	22.6	44.7
rSVD$_{\ell_1}$	1.05	2.12	3.63	7.43	14.4
rSVD$_{\ell_0}$	1.02	1.97	3.45	6.58	13.2

Table 7: Average computational time for the extraction of one component (in seconds).

algorithm, in contrast, works in two steps. First, it maximizes the variance, without enforcing sparsity. This corresponds to computing the first principal component and requires thus a first run of the algorithm with random initialization and a sparsity inducing parameter set at zero. In the second run, this parameter is set to a positive value and the method works to rapidly decrease cardinality at the expense of only a modest decrease in explained variance. So, the new algorithm GPower$_{\ell_1}$ performs faster primarily because it combines the two phases into one, simultaneously optimizing the trade-off between variance and sparsity.

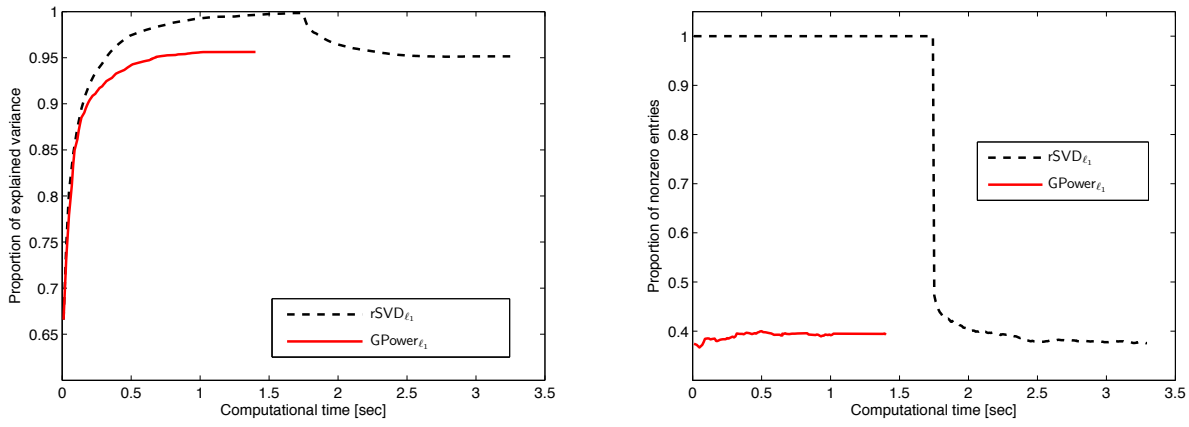


Figure 4: Evolution of explained variance (left) and cardinality (right) in time for the methods GPower$_{\ell_1}$ and rSVD$_{\ell_1}$ run on a test problem of size $p = 250$ and $n = 2500$. The rSVD$_{\ell_1}$ algorithm first solves unconstrained PCA, whereas GPower$_{\ell_1}$ immediately optimizes the trade-off between variance and sparsity.

5.2.6 EXTRACTING MORE COMPONENTS

Similar numerical experiments, which include the methods $\mathsf{GPower}_{\ell_1,m}$ and $\mathsf{GPower}_{\ell_0,m}$, have been conducted for the extraction of more than one component. A deflation scheme is used by the non-block methods to sequentially compute m components. These experiments lead to similar conclusions as in the single-unit case, that is, the methods GPower_{ℓ_1}, GPower_{ℓ_0}, $\mathsf{GPower}_{\ell_1,m}$, $\mathsf{GPower}_{\ell_0,m}$ and rSVD_{ℓ_0} outperform the SPCA and rSVD_{ℓ_1} approaches in terms of variance explained at a fixed cardinality. Again, these last two methods can be improved by postprocessing the resulting loading vectors with Algorithm 6, as it is done for $\mathsf{GPower}_{\ell_1,m}$. The average running times for problems of various sizes are listed in Table 8. The new power-like methods are significantly faster on all instances.

$p \times n$	50×500	100×1000	250×2500	500×5000	750×7500
GPower_{ℓ_1}	0.22	0.56	4.62	12.6	20.4
GPower_{ℓ_0}	0.06	0.17	2.15	6.16	10.3
$\mathsf{GPower}_{\ell_1,m}$	0.09	0.28	3.50	12.4	23.0
$\mathsf{GPower}_{\ell_0,m}$	0.05	0.14	2.39	7.7	12.4
SPCA	0.61	1.47	13.4	48.3	113.3
rSVD_{ℓ_1}	0.29	1.12	7.72	22.6	46.1
rSVD_{ℓ_0}	0.28	1.03	7.21	20.7	41.2

Table 8: Average computational time for the extraction of $m = 5$ components (in seconds).

5.2.7 COST AND BENEFITS OF THE POST-PROCESSING PHASE

Figure 5 illustrates the evolution of the relative increase of computational time as well as the relative improvement in terms of explained variance due to the post-processing phase for increasing values of γ. Only methods with iterative post-processing algorithms are considered, that is, GPower_{ℓ_1} (left-hand plot) and $\mathsf{GPower}_{\ell_1,m}$ (right-hand plot). In the single unit case, the post-processing phase, which amounts to a rank-one SVD of the truncated data matrix A_P, becomes less costly as the level of sparsity increases. As expected, the improvement of variance increases when γ gets larger, that is, when the ℓ_1-penalty biases more and more the values assigned to the non-zero entries of the vector z. A similar observation holds in the block case, excepted that the relative excess of computational time took by the post-processing increases with γ. This difference with the single-unit case results from the fact that the post-processing in the block case deals with sparse matrices of possibly large dimension, whereas in the single-unit case the problem is easily rewritten in terms of a full vector with a dimension that equals the number of nonzero elements. Overall, the postprocessing uses less that 10% of the time needed by the main routine, to improve the explained variance by up to 30%.

5.3 Pitprops Data

The "pitprops" data, which stores 180 observations of 13 variables, has been a standard benchmark to evaluate algorithms for sparse PCA (see, e.g., Jolliffe et al. 2003; Zou et al. 2006; Moghaddam et al. 2006; Shen and Huang 2008). Following these previous studies, we use the GPower algorithms to compute *six* sparse principal components of the data. For such more-samples-than-variables settings, it is customary to first factor the covariance matrix as $\Sigma = A^T A$ with $A \in \mathbf{R}^{13\times 13}$, such that the dimension p is virtually reduced to 13. This operation can be readily done through the eigenvalue decomposition of Σ.

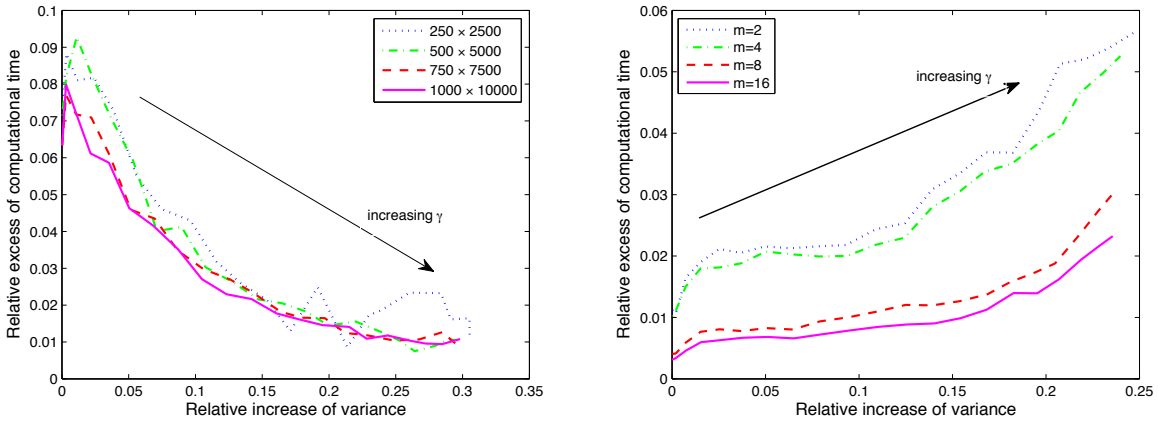


Figure 5: Effects of post-processing in the case of the algorithms GPower$_{\ell_1}$ (left-hand plot) and GPower$_{\ell_1,m}$ (right-hand plot) for increasing values of γ. In both plots, the horizontal axis is the percent increase of variance achieved by the postprocessing phase and the vertical axis is the percent increase in computational time due to post-processing. For GPower$_{\ell_1}$, several problem sizes are considered, whereas the curves for GPower$_{\ell_1,m}$ relate to matrices of dimension 500-by-5000 for several numbers m of extracted components. Each curve is an average on 25 random Gaussian data matrices.

In Table 9, we provide the total cardinality and the proportion of adjusted variance explained by six components computed with SPCA, rSVD$_{\ell_1}$, Greedy as well as our GPower algorithms. The results concerning SPCA, rSVD$_{\ell_1}$, Greedy correspond to the patterns of zeros and nonzeros proposed by Zou et al. (2006), Shen and Huang (2008) and Moghaddam et al. (2006), respectively. For fair comparison, the pattern related to SPCA and rSVD$_{\ell_1}$ have been post-processed with the approach proposed in Section 4.2. Concerning the Gpower algorithms, we fix the six parameters γ_j at the same ratio of their respective upper-bounds. For the block algorithm GPower$_{\ell_1,m}$, experiments have been conducted in both cases "identical μ_j" and "distinct μ_j".

Table 9 illustrates that better patterns can be identified with the GPower algorithms, that is, patterns that explain more variance with the same cardinality (and sometimes even with a smaller one). These results are furthermore likely to be improved by a fine tuning of the six parameters γ_j (i.e., by choosing them independently from each others).

5.4 Analysis of Gene Expression Data

Gene expression data results from DNA microarrays and provide the expression level of thousands of genes across several hundreds of experiments. The interpretation of these huge databases remains a challenge. Of particular interest is the identification of genes that are systematically coexpressed under similar experimental conditions. We refer to Riva et al. (2005) and references therein for more details on microarrays and gene expression data. PCA has been intensively applied in this context (e.g., Alter et al. 2003). Further methods for dimension reduction, such as independent component analysis (Liebermeister, 2002) or nonnegative matrix factorization (Brunet et al., 2004), have also

Method	Parameters	Total cardinality	Prop. of explained variance
rSVD$_{\ell_1}$	see Shen and Huang (2008)	25	0.7924
SPCA	see Zou et al. (2006)	18	0.7680
Greedy	cardinalities: 6-2-3-1-1-1	14	0.7150
	cardinalities: 5-2-2-1-1-1	12	0.5406
GPower$_{\ell_1}$	$\gamma_j/\bar{\gamma}_j = 0.22$, for $j = 1,\ldots,6$	25	0.8083
	$\gamma_j/\bar{\gamma}_j = 0.28$	18	0.7674
	$\gamma_j/\bar{\gamma}_j = 0.30$	15	0.7542
	$\gamma_j/\bar{\gamma}_j = 0.40$	13	0.7172
	$\gamma_j/\bar{\gamma}_j = 0.50$	11	0.6042
GPower$_{\ell_1,m}$	$\gamma_j/\bar{\gamma}_j = 0.17$, for $j = 1,\ldots,6$	25	0.7733
with $\mu_j = 1$	$\gamma_j/\bar{\gamma}_j = 0.25$	17	0.7708
	$\gamma_j/\bar{\gamma}_j = 0.3$	14	0.7508
	$\gamma_j/\bar{\gamma}_j = 0.4$	13	0.7076
	$\gamma_j/\bar{\gamma}_j = 0.45$	11	0.6603
GPower$_{\ell_1,m}$	$\gamma_j/\bar{\gamma}_j = 0.18$, for $j = 1,\ldots,6$	25	0.8111
	$\gamma_j/\bar{\gamma}_j = 0.25$	18	0.7849
with $\mu_j = \frac{1}{j}$	$\gamma_j/\bar{\gamma}_j = 0.30$	15	0.7610
	$\gamma_j/\bar{\gamma}_j = 0.35$	13	0.7323
	$\gamma_j/\bar{\gamma}_j = 0.40$	12	0.6656

Table 9: Extraction of 6 components from the pitprops data. For GPower$_{\ell_1}$, one defines the upper-bounds $\bar{\gamma}_j = \max_i \|a_i^{(j)}\|_2$, where $A^{(j)}$ is the residual data matrix after $j-1$ deflation steps. For GPower$_{\ell_1,m}$, the upper-bounds are $\bar{\gamma}_j = \mu_j \max_i \|a_i\|_2$.

been used on gene expression data. Sparse PCA, which extracts components involving a few genes only, is expected to enhance interpretation.

5.4.1 DATA SETS

The results below focus on four major data sets related to breast cancer. They are briefly detailed in Table 10.[7] Each sparse PCA algorithm computes ten components from these data sets, that is, $m = 10$.

Study	Samples (p)	Genes (n)	Reference
Vijver	295	13319	van de Vijver et al. (2002)
Wang	285	14913	Wang et al. (2005)
Naderi	135	8278	Naderi et al. (2007)
JRH-2	101	14223	Sotiriou et al. (2006)

Table 10: Breast cancer cohorts.

7. The normalized data sets have been kindly provided by Andrew Teschendorff.

5.4.2 SPEED

The average computational time required by the sparse PCA algorithms on each data set is displayed in Table 11. The indicated times are averages on all the computations performed to obtain cardinality ranging from n down to 1.

	Vijver	Wang	Naderi	JRH-2
GPower$_{\ell_1}$	5.92	5.33	2.15	2.69
GPower$_{\ell_0}$	4.86	4.93	1.33	1.73
GPower$_{\ell_1,m}$	5.40	4.37	1.77	1.14
GPower$_{\ell_0,m}$	5.61	7.21	2.25	1.47
SPCA	77.7	82.1	26.7	11.2
rSVD$_{\ell_1}$	10.19	9.97	3.96	4.43
rSVD$_{\ell_0}$	9.51	9.23	3.46	3.61

Table 11: Average computational times (in seconds) for the extraction of $m = 10$ components.

5.4.3 TRADE-OFF CURVES

Figure 6 plots the proportion of adjusted variance versus the cardinality for the "Vijver" data set. The other data sets have similar plots. As for the random test problems, this performance criterion does not discriminate among the different algorithms. All methods have in fact the same performance, provided that the SPCA and rSVD$_{\ell_1}$ approaches are used with postprocessing by Algorithm 6.

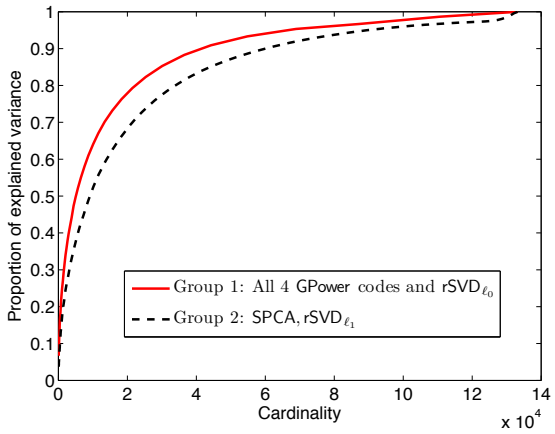


Figure 6: **Trade-off curves** between explained variance and cardinality for the "Vijver" data set. The vertical axis is the ratio $\mathrm{AdjVar}(Z_{\mathrm{sPCA}})/\mathrm{AdjVar}(Z_{\mathrm{PCA}})$, where the loading vectors Z_{sPCA} are computed by sparse PCA and Z_{PCA} are the m first principal loading vectors.

5.4.4 INTERPRETABILITY

A more interesting performance criterion is to estimate the biological interpretability of the extracted components. The *pathway enrichment index* (PEI) proposed by Teschendorff et al. (2007)

measures the statistical significance of the overlap between two kinds of gene sets. The first sets are inferred from the computed components by retaining the most expressed genes, whereas the second sets result from biological knowledge. For instance, metabolic pathways provide sets of genes known to participate together when a certain biological function is required. An alternative is given by the regulatory motifs: genes tagged with an identical motif are likely to be coexpressed. One expects sparse PCA methods to recover some of these biologically significant sets. Table 12 displays the PEI based on 536 metabolic pathways related to cancer. The PEI is the fraction of these 536 sets presenting a statistically significant overlap with the genes inferred from the sparse principal components. The values in Table 12 correspond to the largest PEI obtained among all possible cardinalities. Similarly, Table 13 is based on 173 motifs. More details on the selected pathways and motifs can be found in Teschendorff et al. (2007). This analysis clearly indicates that the sparse PCA methods perform much better than PCA in this context. Furthermore, the new GPower algorithms, and especially the block formulations, provide largest PEI values for both types of biological information. In terms of biological interpretability, they systematically outperform previously published algorithms.

	Vijver	Wang	Naderi	JRH-2
PCA	0.0728	0.0466	0.0149	0.0690
GPower$_{\ell_1}$	**0.1493**	0.1026	0.0728	0.1250
GPower$_{\ell_1}$	0.1250	0.1250	0.0672	0.1026
GPower$_{\ell_1,m}$	0.1418	0.1250	**0.1026**	**0.1381**
GPower$_{\ell_0,m}$	0.1362	**0.1287**	0.1007	0.1250
SPCA	0.1362	0.1007	0.0840	0.1007
rSVD$_{\ell_1}$	0.1213	0.1175	0.0914	0.0914
rSVD$_{\ell_0}$	0.1175	0.0970	0.0634	0.1063

Table 12: PEI-values based on a set of 536 cancer-related pathways.

	Vijver	Wang	Naderi	JRH-2
PCA	0.0347	0	0.0289	0.0405
GPower$_{\ell_1}$	0.1850	0.0867	0.0983	0.1792
GPower$_{\ell_0}$	0.1676	0.0809	0.0925	**0.1908**
GPower$_{\ell_1,m}$	**0.1908**	**0.1156**	**0.1329**	0.1850
GPower$_{\ell_0,m}$	0.1850	0.1098	**0.1329**	0.1734
SPCA	0.1734	0.0925	0.0809	0.1214
rSVD$_{\ell_1}$	0.1387	0.0809	0.1214	0.1503
rSVD$_{\ell_0}$	0.1445	0.0867	0.0867	0.1850

Table 13: PEI-values based on a set of 173 motif-regulatory gene sets.

6. Conclusion

We have proposed two single-unit and two block formulations of the sparse PCA problem and constructed reformulations with several favorable properties. First, the reformulated problems are of the form of maximization of a convex function on a compact set, with the feasible set being either a unit Euclidean sphere or the Stiefel manifold. This structure allows for the design and iteration complexity analysis of a simple gradient scheme which applied to our sparse PCA setting results in

four new algorithms for computing sparse principal components of a matrix $A \in \mathbf{R}^{p\times n}$. Second, our algorithms appear to be faster if either the objective function or the feasible set are strongly convex, which holds in the single-unit case and can be enforced in the block case. Third, the dimension of the feasible sets does not depend on n but on p and on the number m of components to be extracted. This is a highly desirable property if $p \ll n$. Last but not least, on random and real-life biological data, our methods systematically outperform the existing algorithms both in speed and trade-off performance. Finally, in the case of the biological data, the components obtained by our block algorithms deliver the richest biological interpretation as compared to the components extracted by the other methods.

Acknowledgments

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors. Research of Yurii Nesterov and Peter Richtárik has been supported by the grant "Action de recherche concertée ARC 04/09-315" from the "Direction de la recherche scientifique - Communauté française de Belgique". Michel Journée is a research fellow of the Belgian National Fund for Scientific Research (FNRS).

Appendix A.

In this appendix we characterize a class of functions with strongly convex level sets. First we need to collect some basic preliminary facts. All the inequalities of Proposition 11 are well-known in the literature.

Proposition 11 *(i) If f is a strongly convex function with convexity parameter σ_f, then for all x, y and $0 \leq \alpha \leq 1$,*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y) - \frac{\sigma_f}{2}\alpha(1-\alpha)\|x-y\|^2. \tag{36}$$

(ii) If f is a convex differentiable function and its gradient is Lipschitz continuous with constant L_f, then for all x and h,

$$f(x+h) \leq f(x) + \langle f'(x), h\rangle + \frac{L_f}{2}\|h\|^2, \tag{37}$$

and

$$\|f'(x)\|_* \leq \sqrt{2L_f(f(x)-f_*)}, \tag{38}$$

where $f_* \stackrel{def}{=} \min_{x\in\mathbf{E}} f(x)$.

We are now ready for the main result of this section.

Theorem 12 (Strongly Convex Level Sets) *Let $f : \mathbf{E} \to \mathbf{R}$ be a nonnegative strongly convex function with convexity parameter $\sigma_f > 0$. Also assume f has a Lipschitz continuous gradient with Lipschitz constant $L_f > 0$. Then for any $\omega > 0$, the set*

$$Q_\omega \stackrel{def}{=} \{x \mid f(x) \leq \omega\}$$

is strongly convex with convexity parameter

$$\sigma_{Q_\omega} = \frac{\sigma_f}{\sqrt{2\omega L_f}}.$$

Proof Consider any $x, y \in Q_\omega$, scalar $0 \leq \alpha \leq 1$ and let $z_\alpha = \alpha x + (1-\alpha)y$. Notice that by convexity, $f(z_\alpha) \leq \omega$. For any $u \in \mathbf{E}$,

$$\begin{aligned}
f(z_\alpha + u) &\stackrel{(37)}{\leq} f(z_\alpha) + \langle f'(z_\alpha), u\rangle + \frac{L_f}{2}\|u\|^2 \\
&\leq f(z_\alpha) + \|f'(z_\alpha)\|\|u\| + \frac{L_f}{2}\|u\|^2 \\
&\stackrel{(38)}{\leq} f(z_\alpha) + \sqrt{2L_f f(z_\alpha)}\|u\| + \frac{L_f}{2}\|u\|^2 \\
&= \left(\sqrt{f(z_\alpha)} + \sqrt{\tfrac{L_f}{2}}\|u\|\right)^2 \\
&\stackrel{(36)}{\leq} \left(\sqrt{\omega - \beta} + \sqrt{\tfrac{L_f}{2}}\|u\|\right)^2,
\end{aligned}$$

where

$$\beta = \frac{\sigma_f}{2}\alpha(1-\alpha)\|x-y\|^2. \tag{39}$$

In view of (25), it remains to show that the last displayed expression is bounded above by ω whenever u is of the form

$$u = \frac{\sigma_{Q_\omega}}{2}\alpha(1-\alpha)\|x-y\|^2 s = \frac{\sigma_f}{2\sqrt{2\omega L_f}}\alpha(1-\alpha)\|x-y\|^2 s, \tag{40}$$

for some $s \in \mathcal{S}$. However, this follows directly from concavity of the scalar function $g(t) = \sqrt{t}$:

$$\begin{aligned}
\sqrt{\omega - \beta} = g(\omega - \beta) &\leq g(\omega) - \langle g'(\omega), \beta\rangle \\
&= \sqrt{\omega} - \frac{\beta}{2\sqrt{\omega}} \\
&\stackrel{(39)}{\leq} \sqrt{\omega} - \frac{\sigma_f}{4\sqrt{\omega}}\alpha(1-\alpha)\|x-y\|^2 \\
&\stackrel{(40)}{\leq} \sqrt{\omega} - \sqrt{\frac{L_f}{2}}\|u\|.
\end{aligned}$$

■

Example 13 *Let $f(x) = \|x\|^2$. Note that $\sigma_f = L_f = 2$. If we let $\omega = r^2$, then*

$$Q_\omega = \{x \mid f(x) \leq \omega\} = \{x \mid \|x\| \leq r\} = r \cdot \mathcal{B}.$$

We have shown before (see the discussion immediately following Assumption 3), that the strong convexity parameter of this set is $\sigma_{Q_\omega} = \frac{1}{r}$. Note that we recover this as a special case of Theorem 12:

$$\sigma_{Q_\omega} = \frac{\sigma_f}{\sqrt{2\omega L_f}} = \frac{1}{r}.$$

References

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, January 2008.

O. Alter, P. O. Brown, and D. Botstein. Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms. *Proc Natl Acad Sci USA*, 100(6):3351–3356, 2003.

R. W. Brockett. Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems. *Linear Algebra Appl.*, 146:79–91, 1991.

J. P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proc Natl Acad Sci USA*, 101(12):4164–4169, 2004.

J. Cadima and I. T. Jolliffe. Loadings and correlations in the interpretation of principal components. *Journal of Applied Statistics*, 22:203–214, 1995.

A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *Siam Review*, 49:434–448, 2007.

A. d'Aspremont, F. R. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

C. Fraikin, Yu. Nesterov, and P. Van Dooren. A gradient-type algorithm optimizing the coupling between matrices. *Linear Algebra and its Applications*, 429(5-6):1229–1242, 2008.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

R. A. Horn and C. A. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, UK, 1985.

I. T. Jolliffe. Rotation of principal components: choice of normalization constraints. *Journal of Applied Statistics*, 22:29–35, 1995.

I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics*, 12(3):531–547, 2003.

W. Liebermeister. Linear modes of gene expression determined by independent component analysis. *Bioinformatics*, 18(1):51–60, 2002.

L. Mackey. Deflation methods for sparse PCA. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1017–1024, 2008.

B. Moghaddam, Y. Weiss, and S. Avidan. Spectral bounds for sparse PCA: Exact and greedy algorithms. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 915–922. MIT Press, Cambridge, MA, 2006.

A. Naderi, A. E. Teschendorff, N. L. Barbosa-Morais, S. E. Pinder, A. R. Green, D. G. Powe, J. F. R. Robertson, S. Aparicio, I. O. Ellis, J. D. Brenton, and C. Caldas. A gene expression signature to predict survival in breast cancer across independent data sets. *Oncogene*, 26:1507–1516, 2007.

B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall Inc., Englewood Cliffs, N.J., 1980. ISBN 0-13-880047-2. Prentice-Hall Series in Computational Mathematics.

A. Riva, A.-S. Carpentier, B. Torrésani, and A. Hénaut. Comments on selected fundamental aspects of microarray analysis. *Computational Biology and Chemistry*, 29(5):319–336, 2005.

H. Shen and J. Z. Huang. Sparse principal component analysis via regularized low rank matrix approximation. *Journal of Multivariate Analysis*, 99(6):1015–1034, 2008.

C. Sotiriou, P. Wirapati, S. Loi, A. Harris, S. Fox, J. Smeds, H. Nordgren, P. Farmer, V. Praz, B. Haibe-Kains, C. Desmedt, D. Larsimont, F. Cardoso, H. Peterse, D. Nuyten, M. Buyse, M. J. Van de Vijver, J. Bergh, M. Piccart, and M. Delorenzi. Gene expression profiling in breast cancer: understanding the molecular basis of histologic grade to improve prognosis. *J Natl Cancer Inst*, 98(4):262–272, 2006.

A. Teschendorff, M. Journée, P.-A. Absil, R. Sepulchre, and C. Caldas. Elucidating the altered transcriptional programs in breast cancer using independent component analysis. *PLoS Computational Biology*, 3(8):1539–1554, 2007.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(2):267–288, 1996.

M. J. van de Vijver, Y. D. He, L. J. van't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend, and R. Bernards. A gene-expression signature as a predictor of survival in breast cancer. *N Engl J Med*, 347(25): 1999–2009, 2002.

Y. Wang, J. G. Klijn, Y. Zhang, A. M. Sieuwerts, M. P. Look, F. Yang, D. Talantov, M. Timmermans, M. E. Meijer-van Gelder, J. Yu, T. Jatkoe, E. M. Berns, D. Atkins, and J. A. Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365(9460):671–679, 2005.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.

Journal of Machine Learning Research 11 (2010) 555-580 Submitted 12/08; Revised 9/09; Published 2/10

Approximate Tree Kernels

Konrad Rieck RIECK@CS.TU-BERLIN.DE
Technische Universität Berlin
Franklinstraße 28/29
10587 Berlin, Germany

Tammo Krueger TAMMO.KRUEGER@FIRST.FRAUNHOFER.DE
Fraunhofer Institute FIRST
Kekuléstraße 7
12489 Berlin, Germany

Ulf Brefeld BREFELD@YAHOO-INC.COM
Yahoo! Research
Avinguda Diagonal 177
08018 Barcelona, Spain

Klaus-Robert Müller KLAUS-ROBERT.MUELLER@TU-BERLIN.DE
Technische Universität Berlin
Franklinstraße 28/29
10587 Berlin, Germany

Editor: John Shawe-Taylor

Abstract

Convolution kernels for trees provide simple means for learning with tree-structured data. The computation time of tree kernels is quadratic in the size of the trees, since all pairs of nodes need to be compared. Thus, large parse trees, obtained from HTML documents or structured network data, render convolution kernels inapplicable. In this article, we propose an effective approximation technique for parse tree kernels. The approximate tree kernels (ATKs) limit kernel computation to a sparse subset of relevant subtrees and discard redundant structures, such that training and testing of kernel-based learning methods are significantly accelerated. We devise linear programming approaches for identifying such subsets for supervised and unsupervised learning tasks, respectively. Empirically, the approximate tree kernels attain run-time improvements up to three orders of magnitude while preserving the predictive accuracy of regular tree kernels. For unsupervised tasks, the approximate tree kernels even lead to more accurate predictions by identifying relevant dimensions in feature space.

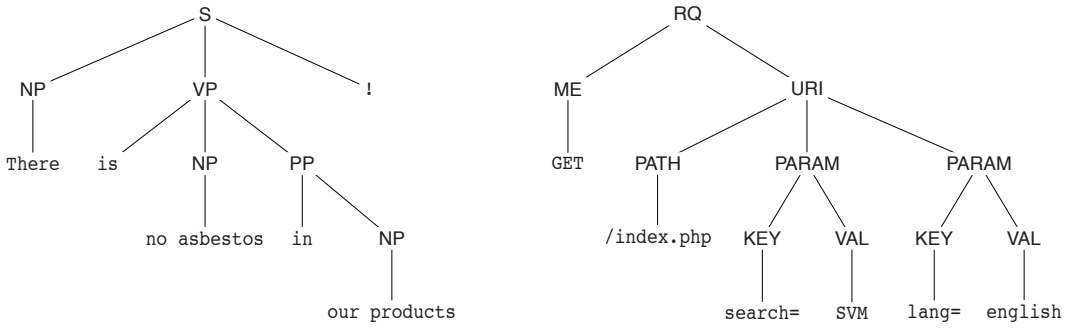
Keywords: tree kernels, approximation, kernel methods, convolution kernels

1. Introduction

Learning from tree-structured data is an elementary problem in machine learning, as trees arise naturally in many real-world applications. Exemplary applications involve parse trees in natural language processing, HTML documents in information retrieval, molecule structures in computational chemistry, and structured network data in computer security (e.g., Manning and Schütze, 1999; Kashima and Koyanagi, 2002; Moschitti, 2006b; Cilia and Moschitti, 2007; Düssel et al., 2008; Rieck et al., 2008; Bockermann et al., 2009).

In general, trees carry hierarchical information reflecting the underlying dependency structure of a domain at hand—an appropriate representation of which is often indispensable for learning accurate prediction models. For instance, shallow representations of trees such as flat feature vectors often fail to capture the underlying dependencies. Thus, the prevalent tools for learning with structured data are kernel functions, which implicitly assess pairwise similarities of structured objects and thereby avoid explicit representations (see Müller et al., 2001; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004). Kernel functions for structured data can be constructed using the convolution of local kernels defined over sub-structures (Haussler, 1999). A prominent example for such a convolution is the parse tree kernel proposed by Collins and Duffy (2002) which determines the similarity of trees by counting shared subtrees.

The computation of parse tree kernels, however, is inherently quadratic in the number of tree nodes, as it builds on dynamic programming to compute the contribution of shared subtrees. Allocating and updating tables for dynamic programming is feasible for small tree sizes, say less than 200 nodes, so that tree kernels have been widely applied in natural language processing, for example, for question classification (Zhang and Lee, 2003) and parse tree reranking (Collins and Duffy, 2002). Figure 1(a) shows an exemplary parse tree of natural language text. Large trees involve computations that exhaust available resources in terms of memory and run-time. For example, the computation of a parse tree kernel for two HTML documents comprising 10,000 nodes each, requires about 1 gigabyte of memory and takes over 100 seconds on a recent computer system. Given that kernel computations are performed millions of times in large-scale learning, it is evident that regular tree kernels are an inappropriate choice in many learning tasks.



(a) Parse tree for a sentence

(b) Parse tree for a HTTP request

Figure 1: Parse trees for natural language text and the HTTP network protocol.

The limitation of regular tree kernels becomes apparent when considering learning tasks based on formal grammars, such as web spam detection. In web spam detection one seeks to find arrays of linked fraudulent web pages, so-called link farms, that deteriorate the performance of search engines by manipulating search results (e.g., Wu and Davison, 2005; Drost and Scheffer, 2005; Castillo et al., 2006). Besides being densely linked, these web pages share an important property: they are generated automatically according to templates. Hence, a promising approach for web spam detection is the analysis of structure in web pages using parse trees of HTML documents. Unfortunately, HTML documents can grow almost arbitrarily large and render the computation of conventional tree kernels impossible.

Moreover, tree-structured data also arises as part of unsupervised learning problems, such as clustering and anomaly detection. For instance, a critical task in computer security is the automatic detection of novel network attacks (Eskin et al., 2002). Current detection techniques fail to cope with the increasing amount and diversity of network threats, as they depend on manually generated detection rules. As an alternative, one-class learning methods have been successfully employed for identifying anomalies in the context of intrusion detection (e.g., Eskin et al., 2002; Kruegel and Vigna, 2003; Rieck and Laskov, 2007). Due to the already mentioned run-time constraints these approaches focus on shallow features, although efficient methods for extracting syntactical structure from network data are available (e.g., Pang et al., 2006; Borisov et al., 2007; Wondracek et al., 2008). Similar to natural language, network protocols are defined in terms of grammars and individual communication can be represented as parse trees, see Figure 1(b).

To alleviate the limitations of regular tree kernels, we propose *approximate tree kernels* (ATKs) which approximate the kernel computation and thereby allow for efficient learning with arbitrary sized trees in supervised and in unsupervised settings. The efficacy gained by approximate tree kernels rests on a two-stage process: A sparse set of relevant subtrees rooted at appropriate grammar symbols is determined from a small sample of trees, *prior* to subsequent training and testing processes. By decoupling the selection of symbols from the kernel computation, both, run-time and memory requirements are significantly reduced. In the supervised setting, the subset of symbols is optimized with respect to its ability to discriminate between the involved classes, while for the unsupervised setting the optimization is performed with respect to node occurrences in order to minimize the expected run-time. The corresponding optimization problems are translated into linear programs that can be efficiently solved with standard techniques.

Experiments conducted on question classification, web spam detection and network intrusion detection demonstrate the expressiveness and efficiency of our novel approximate kernels. Throughout all our experiments, approximate tree kernels are significantly faster than regular convolution kernels. Depending on the size of the trees, we observe run-time and memory improvements up to 3 orders of magnitude. Furthermore, the approximate tree kernels not only consistently yield the same predictive performance as regular tree kernels, but even outperform their exact counterparts in some tasks by identifying informative dimensions in feature space.

The remainder of this article is organized as follows. We introduce regular parse tree kernels in Section 2 and present our main contribution, the approximate tree kernels, in Section 3. We study the characteristics of approximate kernels on artificial data in Section 4 and report on real-world applications in Section 5. Finally, Section 6 concludes.

2. Kernels for Parse Trees

Let $G = (S, P, s)$ be a grammar with production rules P and a start symbol s defined over a set S of non-terminal and terminal symbols (Hopcroft and Motwani, 2001). A tree X is called a parse tree of G if X is derived by assembling productions $p \in P$ such that every node $x \in X$ is labeled with a symbol $\ell(x) \in S$. To navigate in a parse tree, we address the i-th child of a node x by x_i and denote the number of children by $|x|$. The number of nodes in X is indicated by $|X|$ and the set of all possible trees is given by $\mathcal{X}$.

A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric and positive semi-definite function, which implicitly computes an inner product in a reproducing kernel Hilbert space (Vapnik, 1995). A generic technique for defining kernel functions over structured data is the convolution of local kernels defined

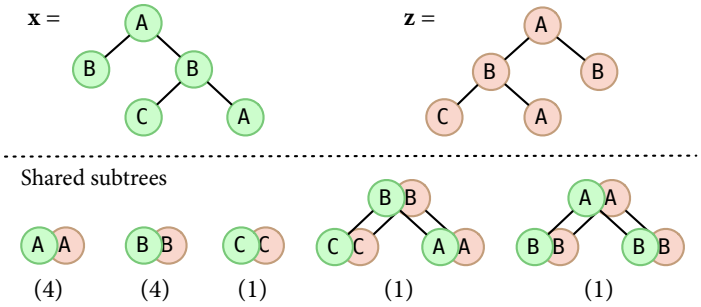


Figure 2: Shared subtrees in two parse trees. The numbers in brackets indicate the number of occurrences for each shared subtree pair.

over sub-structures (Haussler, 1999). Collins and Duffy (2002) apply this concept to parse trees by counting shared subtrees. Given two parse trees X and Z, their *parse tree kernel* is given by

$$k(X,Z) = \sum_{x \in X} \sum_{z \in Z} c(x,z), \tag{1}$$

where the counting function c recursively determines the number of shared subtrees rooted in the tree nodes x and z.

The function c is defined as $c(x,z) = 0$ if x and z are not derived from the same production and $c(x,z) = \lambda$ if x and z are leave nodes of the same production. In all other cases, the definition of the counting function c follows a recursive rule given by

$$c(x,z) = \lambda \prod_{i=1}^{|x|} (1 + c(x_i, z_i)) . \tag{2}$$

The trade-off parameter $0 < \lambda \leq 1$ balances the contribution of subtrees, such that small values of λ decay the contribution of lower nodes in large subtrees (see Collins and Duffy, 2002). Figure 2 illustrates two simple parse trees and the corresponding shared subtrees.

Several extensions and refinements of the parse tree kernel have been proposed in the literature to increase its expressiveness for specific learning tasks. For example, setting the constant term in the product of Equation (2) to zero restricts the counting function to take only complete subtrees into account (Vishwanathan and Smola, 2003). Kashima and Koyanagi (2002) extend the counting function to generic trees—not necessarily derived from a grammar—by considering ordered subsets of child nodes for computing the kernel. Further extensions to the counting function proposed by Moschitti (2006b) allow for controlling the vertical as well as the horizontal contribution of subtree counts. Moreover, Moschitti and Zanzotto (2007) extend the parse tree kernel to operate on pairs of trees for deriving relations between sentences in tasks such as textual entailment recognition. However, all of the above mentioned extensions depend on dynamic programming over all pairs of nodes and thus yield prohibitive run-time and memory requirements if large trees are considered.

Selecting discriminative subtrees for tree kernels has been first studied by Suzuki et al. (2004) in the domain of natural language processing. A feature selection procedure based on statistical tests is embedded in the dynamic programming, such that relevant substructures are identified during computation of the parse tree kernel (see also Suzuki and Isozaki, 2005). While this procedure

significantly improves the expressiveness of corresponding tree kernels, the entanglement of feature selection and dynamic programming unfortunately prevents any improvement of run-time and memory requirements over regular tree kernels.

First steps toward run-time improvement of tree kernels have been devised by Moschitti (2006a), who introduces an alternative algorithm limiting computation to node pairs with matching grammar symbols. Although this extension reduces the run-time, it does not alleviate prohibitive memory requirements, as counts for all possible node pairs need to be stored. Also note, that for large trees with $|X| \gg |S|$, only a minor speed-up is gained, since only a small fraction of node pairs can be discarded from the kernel computation. Nevertheless, this algorithm is an efficient approach when learning with small trees as in natural language processing.

3. Approximate Tree Kernels

The previous section argues that the computation of regular tree kernels using dynamic programming is infeasible for large tree structures. In this section we introduce approximate tree kernels to significantly decrease this computational burden. Our approximation of tree kernels is based on the observation that trees often contain redundant parts that are not only irrelevant for the learning task but also slow-down the kernel computation unnecessarily. As an example for redundancy in trees let us consider the task of web spam detection. While few HTML elements, such as header and meta tags, are indicative for web spam templates, the majority of formatting tags is irrelevant and may even harm performance. We exploit this observation by restricting the kernel computation to a sparse set of subtrees rooted in only a few grammar symbols.

In general, selecting relevant subtrees for the kernel computation requires efficient means for enumerating subtrees. The amount of generic subtrees contained in a single parse tree is exponential in the number of nodes and thus intractable for large tree structures. Consequently, we refrain from exhaustive enumeration and limit the selection to subtrees rooted at particular grammar symbols. We introduce a selection function $\omega : S \to \{0,1\}$, which controls whether subtrees rooted in nodes with the symbol $s \in S$ contribute to the convolution ($\omega(s) = 1$) or not ($\omega(s) = 0$). By means of ω, approximate tree kernels are defined as follows.

Definition 1 *Given a selection function $\omega : S \to \{0,1\}$, the approximate tree kernel is defined as*

$$\hat{k}_\omega(X,Z) = \sum_{s \in S} \omega(s) \sum_{\substack{x \in X \\ \ell(x)=s}} \sum_{\substack{z \in Z \\ \ell(z)=s}} \tilde{c}(x,z), \tag{3}$$

where the approximate counting function $\tilde{c}$ is defined as (i) $\tilde{c}(x,z) = 0$ if x and z are not derived from the same production, (ii) $\tilde{c}(x,z) = 0$ if x or z has not been selected, that is, $\omega(\ell(x)) = 0$ or $\omega(\ell(z)) = 0$, and (iii) $c(x,z) = \lambda$ if x and z are leave nodes of the same production. In all other cases, the approximate counting function $\tilde{c}$ is defined as

$$\tilde{c}(x,z) = \lambda \prod_{i=1}^{|x|} (1 + \tilde{c}(x_i, z_i)) .$$

For the task at hand, the selection function ω needs to be adapted to the tree data before the resulting approximate tree kernel can be applied together with learning algorithms. Note that the exact parse tree kernel in Equation (1) is obtained as a special case of Equation (3) if $\omega(s) = 1$ for all symbols $s \in S$. Irrespectively of the actual choice of ω, Proposition 2 shows that the approximate kernel $\hat{k}$ is a valid kernel function.

Proposition 2 *The approximate tree kernel in Definition 1 is a kernel function.*

Proof Let $\phi(X)$ be the vector of frequencies of all subtrees occurring in X as defined by Collins and Duffy (2002). Then, by definition, $\hat{k}_\omega$ can always be written as

$$\hat{k}_\omega(X,Z) = \langle P_\omega \phi(X), P_\omega \phi(Z) \rangle,$$

where P_ω projects the dimensions of $\phi(X)$ on the subtrees rooted in symbols selected by ω. For any ω, the projection P_ω is independent of the actual X and Z, and hence $\hat{k}_\omega$ is a valid kernel. ■

Proposition 2 allows to view approximate tree kernels as feature selection techniques. If the selection function effectively identifies relevant symbols, the dimensionality of the feature space is reduced and the kernel provides access to a refined data representation. We will address this point in Section 4 and 5 where we study eigendecompositions in the induced features spaces. Additionally, we note that the approximate tree kernel realizes a run-time speed-up by a factor of q_ω, which depends on the number of selected symbols in ω and the amount of subtrees rooted at these symbols. For two particular trees X, Z we can state the following simple proposition.

Proposition 3 *The approximate tree kernel $\hat{k}_\omega(X,Z)$ can be computed q_ω times faster than $k(X,Z)$.*

Proof Let $\#_s(X)$ denote the occurrences of nodes $x \in X$ with $\ell(x) = s$. Then the speed-up q_ω realized by $\hat{k}_\omega$ is lower bounded by

$$q_\omega \geq \frac{\sum_{s \in S} \#_s(X) \, \#_s(Z)}{\sum_{s \in S} \omega(s) \, \#_s(X) \, \#_s(Z)} \tag{4}$$

as all nodes with identical symbols in X and Z are paired. For the trivial case where for all elements $\omega(s) = 1$, the factor q_ω equals 1 and the run-time is identical to the parse tree kernel. In all other cases $q_\omega > 1$ holds since at least one symbol is discarded from the denominator in Equation (4). ■

The quality of $\hat{k}_\omega$ and also q_ω depend on the actual choice of ω. If the selection function ω discards redundant and irrelevant subtrees from the kernel computation the approximate kernel can not only be computed faster but also preserves the discriminative expressiveness of the regular tree kernel. Sections 3.1 and 3.2 deal with adapting ω for supervised and unsupervised learning tasks, respectively. Although, the resulting optimization problems are quadratic in the number of instances, selecting symbols can be performed on a small fraction of the data prior to the actual learning process; hence, performance gains achieved in the latter are not affected by the initial selection process. We show in Section 5 that reasonable approximations can be achieved for moderate sample sizes.

3.1 The Supervised Setting

In the supervised setting, we are given n labeled parse trees $(X_1, y_1), \ldots, (X_n, y_n)$ with $y_i \in \mathcal{Y}$. For binary classification we may have $\mathcal{Y} = \{-1, 1\}$ while a multi-class scenario with κ classes gives rise to the set $\mathcal{Y} = \{1, 2, \ldots, \kappa\}$. In the supervised case, the aim of the approximation is to preserve the discriminative power of the regular kernel by selecting a sparse but expressive subset of grammar symbols. We first note that Y with elements $[Y_{ij}]_{i,j=1,\ldots,n}$ given by

$$Y_{ij} = [\![y_i = y_j]\!] - [\![y_i \neq y_j]\!], \tag{5}$$

represents an optimal kernel matrix where $[\![u]\!]$ is an indicator function returning 1 if u is true and 0 otherwise. For binary classification problems, Equation (5) can also be computed by the outer product $Y = yy^{\mathsf{T}}$, where $y = (y_1, \ldots, y_n)^{\mathsf{T}}$.

Inspired by kernel target alignment (Cristianini et al., 2001), a simple measure of the similarity of the approximate kernel $\hat{K}_\omega = [\hat{k}_\omega(X_i, X_j)]_{i,j=1,\ldots,n}$ and the optimal Y is provided by the Frobenius inner product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ that is defined as $\langle A, B \rangle_{\mathcal{F}} = \sum_{ij} a_{ij} b_{ij}$. We have,

$$\langle Y, \hat{K}_\omega \rangle_{\mathcal{F}} = \sum_{y_i = y_j} \hat{k}_{ij} - \sum_{y_i \neq y_j} \hat{k}_{ij}. \tag{6}$$

The right hand side of Equation (6) measures the within class (first term) and the between class (second term) similarity. Approximate kernels that discriminate well between the involved classes realize large values of $\langle Y, \hat{K}_\omega \rangle_{\mathcal{F}}$, hence maximizing Equation (6) with respect to ω suffices for finding approximate kernels with high discriminative power.

We arrive at the following integer linear program that has to be maximized with respect to ω to align $\hat{K}_\omega$ to the labels y,

$$\max_{\omega \in \{0,1\}^{|S|}} \quad \sum_{\substack{i,j=1 \\ i \neq j}}^{n} y_i y_j \hat{k}_\omega(X_i, X_j). \tag{7}$$

Note that we exclude diagonal elements, that is, $\hat{k}_\omega(X_i, X_i)$, from Equation (7), as the large self-similarity induced by the parse tree kernel impacts numerical stability on large tree structures (see Collins and Duffy, 2002).

Optimizing Equation (7) directly will inevitably reproduce the regular convolution kernel as all subtrees contribute positively to the kernel computation. As a remedy, we restrict the number of supporting symbols of the approximation by a pre-defined constant N. Moreover, instead of optimizing the integer program directly, we propose to use a relaxed variant thereof, where a threshold is used to discretize ω. Consequently, we obtain the following relaxed linear program that can be solved with standard solvers.

Optimization Problem 1 (Supervised Setting) *Given a labeled training sample of size n and let $N \in \mathbb{N}$. The optimal selection function ω^* can be computed by solving*

$$\begin{aligned} \omega^* = \operatorname*{argmax}_{\omega \in [0,1]^{|S|}} \quad & \sum_{\substack{i,j=1 \\ i \neq j}}^{n} y_i y_j \sum_{s \in S} \omega(s) \sum_{\substack{x \in X_i \\ \ell(x) = s}} \sum_{\substack{z \in X_j \\ \ell(z) = s}} \tilde{c}(x, z) \\ \textit{subject to} \quad & \sum_{s \in S} \omega(s) \leq N, \end{aligned} \tag{8}$$

where the counting function $\tilde{c}$ is given in Definition (3).

3.2 The Unsupervised Setting

Optimization Problem 1 identifies N symbols providing the highest discriminative power given a labeled training set. In the absence of labels, for instance in an anomaly detection task, the operational goal needs to be changed. In these cases, the only accessible information for finding ω are the tree structures themselves. Large trees are often characterized by redundant substructures that strongly impact run-time performance while encoding only little information. Moreover, syntactical

structure present in all trees of a data set do not provide any information suitable for learning. As a consequence, we seek a selection of symbols using a constraint on the expected run-time and thereby implicitly discard redundant and costly structures with small contribution to the kernel value.

Given unlabeled parse trees $X_1, X_2, \ldots, X_n$, we introduce a function $f(s)$ which measures the average frequency of node comparisons for the symbol s in the training set, defined as

$$f(s) = \frac{1}{n^2} \sum_{i,j=1}^{n} \#_s(X_i) \#_s(X_j). \tag{9}$$

Using the average comparison frequency f, we bound the expected run-time of the approximate kernel by $\rho \in (0,1]$, the ratio of expected node comparisons with respect to the exact parse tree kernel. The following optimization problem details the computation of the optimal ω^* for unsupervised settings.

Optimization Problem 2 (Unsupervised Setting) *Given an unlabeled training sample of size n and a comparison ratio $\rho \in (0,1]$.*

$$\begin{aligned} \omega^* = \operatorname*{argmax}_{\omega \in [0,1]^{|S|}} & \sum_{\substack{i,j=1 \\ i \neq j}}^{n} \sum_{s \in S} \omega(s) \sum_{\substack{x \in X_i \\ \ell(x)=s}} \sum_{\substack{z \in X_j \\ \ell(z)=s}} \tilde{c}(x,z) \\ \textit{subject to} \quad & \frac{\sum_{s \in S} \omega(s) f(s)}{\sum_{s \in S} f(s)} \leq \rho, \end{aligned} \tag{10}$$

where the counting function $\tilde{c}$ is given in Definition 3 and f defined as in Equation (9).

The optimal selection function ω^* gives rise to a tree kernel that approximates the regular kernel as close as possible, while on average considering a fraction of ρ node pairs for computing the similarities. Analogously to the supervised setting, we solve the relaxed variant of the integer program and use a threshold to discretize the resulting ω.

3.3 Extensions

The supervised and unsupervised formulations in Optimization Problem 1 and 2 build on different constraints for determining a selection of appropriate symbols. Depending on the learning task at hand, these constraints are exchangeable, such that approximate tree kernels in supervised settings may also be restricted to the ratio ρ of expected node comparisons and the unsupervised formulation can be alternatively conditioned on a fixed number of symbols N. Both constraints may even be applied jointly to limit expected run-time and the number of selected symbols. For our presentation, we have chosen a constraint on the number of symbols in the supervised setting, as it provides an intuitive measure for the degree of approximation. By contrast, we employ a constraint on the expected number of node comparisons in the unsupervised formulation, as in the absence of labels, it is easier to bound the run-time, irrespective of the number of selected symbols.

Further extensions incorporating prior knowledge into the proposed approximations are straight forward. For instance, the approximation procedure can be refined using logical compounds based on conjunctions and disjunctions of symbols. If the activation of symbol s_j requires the activation of s_{j+1}, the constraint $\omega(s_j) - \omega(s_{j+1}) = 0$ can be included in Equation (8) and (10). A conjunction (AND) of m symbols can then be efficiently encoded by $m-1$ additional constraints as

$$\forall_{j=1}^{m-1} \omega(s_j) - \omega(s_{j+1}) = 0.$$

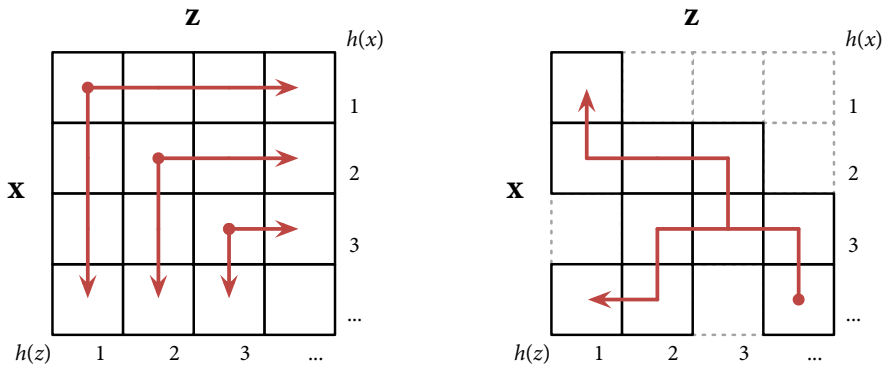
For a disjunction (OR) of symbols $s_j, \ldots, s_{j+m}$ the following constraint guarantees that at least one representative of the group is active in the solution

$$\omega(s_j) + \omega(s_{j+1}) + \ldots + \omega(s_{j+m}) \geq 1. \tag{11}$$

Alternatively, Equation (11) can be turned into an exclusive disjunction (XOR) of symbols by changing the inequality into an equality constraint.

3.4 Implementation

A standard technique for computing tree kernels is dynamic programming, where a table of $|X| \times |Z|$ elements is used to store the counts of subtrees during recursive evaluations. The kernel computation is then carried out in either a systematic or a structural fashion, where intermediate results are stored in the table as sketched in Figure 3. The systematic variant processes the subtrees with ascending height, such that at a particular height, all counts for lower subtrees can be looked up in the table (Shawe-Taylor and Cristianini, 2004). For the structural variant, the dynamic programming table acts as a cache, which stores previous results when computing the recursive counting directly (Moschitti, 2006a). This latter approach has the advantage that only matching subtrees are considered and mismatching nodes do not contribute to the run-time as in the systematic variant.



(a) Systematic mode (b) Structural mode

Figure 3: Dynamic programming for tree kernels. The convolution of subtree counts is computed in a systematic (Shawe-Taylor and Cristianini, 2004) or structural manner (Moschitti, 2006a). The term $h(x)$ denotes the height of the subtree rooted in $x \in X$.

We now provide details on an implementation of approximate tree kernels using the structural approach. The inputs of our implementation are parse trees X, Z and a selection function ω which has been determined in advance using the supervised or unsupervised approximation (Optimization Problems 1 and 2). The implementation proceeds by first generating pairs of matching nodes from the trees X, Z, similarly to the algorithm proposed by Moschitti (2006a). However, pairs whose symbols are not selected by ω are omitted. The computation of the tree kernel is then carried out by looping over the node pairs and counting the number of shared subtrees rooted at each pair. An exemplary implementation of the approximate kernel is given in Algorithms 1, 2 and 3.

Algorithm 1 Approximate Tree Kernel

```
1: function KERNEL(X, Z, ω)
2:     L ← GENERATEPAIRS(X, Z, ω)
3:     k ← 0
4:     for (x, z) ← L do                          ▷ Loop over selected pairs of nodes
5:         k ← k + COUNT(x, z)
6:     return k
```

Algorithm 1 realizes a generic tree kernel, which determines the number of shared subtrees by looping over a list of node pairs. Algorithm 2 shows the corresponding analogue of the counting function in Equation (2) which is called during each iteration of the loop. While the standard implementation of the parse tree kernel (e.g., Shawe-Taylor and Cristianini, 2004; Moschitti, 2006a) uses a dynamic programming table to store the contribution of subtree counts, we employ a hash table denoted by H. A hash table guarantees constant reading and writing of intermediate results yet it grows with the number of selected node pairs and thereby reduces memory in comparison to a standard table of all possible pairs. Note that if all symbols in ω are selected, H realizes the standard dynamic programming approach.

Algorithm 2 Counting Function

```
1: function COUNT(x, z)
2:     if x and z have different productions then
3:         return 0
4:     if x or z is a leaf node then
5:         return λ
6:     if (x, z) stored in hash table H then
7:         return H(x, z)                         ▷ Read dynamic programming cell
8:     c ← 1
9:     for i ← 1 to |x| do
10:        c ← c · (1 + COUNT(x_i, z_i))
11:    H(x, z) ← λc                               ▷ Write dynamic programming cell
12:    return H(x, z)
```

Algorithm 3 implements the function for generating pairs of nodes with selected symbols. The function first sorts the tree nodes using a predefined order in line 2–3. For our implementation we apply a standard lexicographic sorting on the symbols of nodes. Algorithm 3 then proceeds by generating a set of matching node pairs L, satisfying the invariant that included pairs $(x,z) \in L$ have matching symbols (i.e., $\ell(x) = \ell(z)$) and are selected via ω (i.e., $\omega(x) = 1$). The generation of pairs is realized analogously to merging sorted arrays (see Knuth, 1973). The function removes elements from the lists of sorted nodes N_X and N_Z in parallel until a matching and selected pair (x,z) is discovered. With a slight abuse of notation, all available node pairs (a,b) with label $\ell(x)$ are then added to L and removed from N_X and N_Z in lines 12–14 of Algorithm 3.

Algorithm 3 Node Pair Generation

```
1: function GENERATEPAIRS(X, Z, ω)
2:     N_X ← SORTNODES(X)
3:     N_Z ← SORTNODES(Z)
4:     while N_X and N_Z not empty do
5:         x ← head of N_X
6:         z ← head of N_Z
7:         if ℓ(x) < ℓ(z) or ω(x) = 0 then
8:             remove x from N_X                    ▷ x mismatches or not selected
9:         else if ℓ(x) > ℓ(z) or ω(z) = 0 then
10:            remove z from N_Z                    ▷ y mismatches or not selected
11:        else
12:            N ← { (a,b) ∈ N_X × N_Z with label ℓ(x) }
13:            L ← L ∪ N                            ▷ Add all pairs with label ℓ(x)
14:            remove N from N_X and N_Z
15:    return L
```

3.5 Application Setup

In contrast to previous work on feature selection for tree kernels (see Suzuki et al., 2004), the efficiency of our approximate tree kernels is rooted in *decoupling* the selection of symbols from later application of the learned kernel function. In particular, our tree kernels are applied in a two-stage process as detailed in the following.

1. *Selection stage*. In the first stage, a sparse selection ω of grammar symbols is determined on a sample of tree data, where depending on the learning setting either Optimization Problem 1 or 2 is solved by linear programming. As solving both problems involves computing exact tree kernels, the selection is optimized on a small fraction of the trees. To limit memory requirements, the sample may be further filtered to contain only trees of reasonable sizes.

2. *Application stage*. In the subsequent application stage, the approximate tree kernels are employed together with learning algorithms using the efficient implementation detailed in the previous section. The optimized ω reduces the run-time and memory requirements of the kernel computation, such that learning with trees of almost arbitrary size becomes feasible.

The approximate tree kernels involve the parameter λ as defined in Equation (2). The parameter controls the contribution of subtrees; values close to zero emphasize shallow subtrees and $\lambda = 1$ corresponds to a uniform weighting of all subtrees. To avoid repeatedly solving Optimization Problem 1 or 2 for different values of λ, we fix $\lambda = 1$ in the selection stage and perform model selection only in the application stage for $\lambda \in [10^{-4}, 10^{0}]$. This procedure ensures that the approximate tree kernels are first determined over equally weighted subtrees, hence allowing for an unbiased optimization in the selection phase. A potential refinement of λ is postponed to the application stage to exploit performance gains of the approximate tree kernel. Note that if prior knowledge is available, this may be reflected by a different choice of λ in the selection stage.

4. Experiments on Artificial Data

Before studying the expressiveness and performance of approximate tree kernels in real-word applications, we aim at gaining insights into the approximation process. We thus conduct experiments using artificial data generated by the following probabilistic grammar, where A, B, C, D denote non-terminal symbols and a, b terminal symbols. The start symbol is S.

$$S \xrightarrow{[1.0]} A\ B \tag{*1}$$
$$A \xrightarrow{[0.2|0.2|0.6]} A\ A \mid C\ D \mid a \tag{*2}$$
$$B \xrightarrow{[0.2|0.2|0.6]} B\ B \mid D\ C \mid b \tag{*3}$$
$$C \xrightarrow{[0.3|0.3|0.3]} A\ B \mid A \mid B \tag{*4}$$
$$D \xrightarrow{[0.3|0.3|0.3]} B\ A \mid A \mid B \tag{*5}$$

Parse trees are generated from the above grammar by applying the rule $S \to AB$ and randomly choosing matching production rules according to their probabilities until all branches end in terminal nodes. Recursions are included in (*2)–(*5) to ensure that symbols occur at different positions and depths in the parse trees.

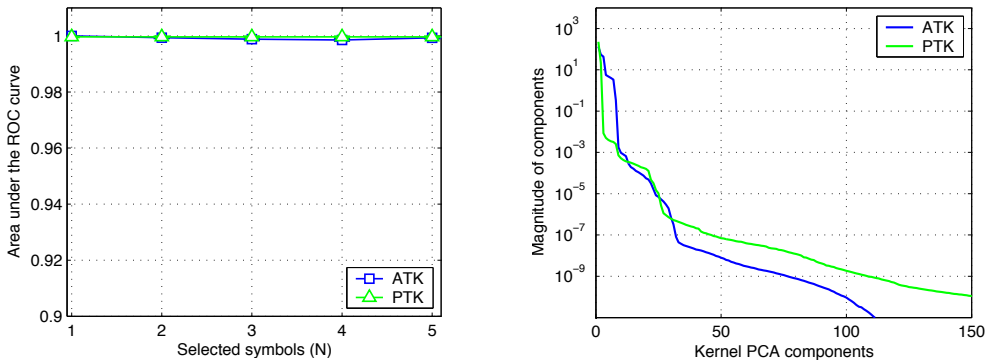
4.1 A Supervised Learning Task

To generate a simple supervised classification task, we assign the first rule in (*4) as an indicator of the positive class and the first rule in (*5) as one of the negative class. We then prepare our data set, such that one but not two of the rules are contained in each parse tree. That is, positive examples possess the rule $C \to AB$ and negative instances exhibit the rule $D \to BA$. Note that due to the symmetric design of the production rules, the two classes can not be distinguished from the symbols C and D alone but from the respective production rules.

Using this setup, we generate training, validation, and test sets consisting of 500 positive and negative trees each. We then apply the two-stage process detailed in Section 3.5: First, the selection function ω is adapted by solving Optimization Problem 1 using a sample of 250 randomly drawn trees from the training set. Second, a Support Vector Machine (SVM) is trained on the training data and applied to the test set, where the optimal regularization parameter of the SVM and the depth parameter λ are selected using the validation set. We report on averages over 10 repetitions and error bars indicate standard errors.

The classification performance of the SVM for the two kernel functions is depicted in Figure 4, where the number of selected symbols N for the approximate kernel is given on the x-axis and the attained area under the ROC curve (AUC) is shown on the y-axis. The parse tree kernel (PTK) leads to a perfect discrimination between the two classes, yet the approximate tree kernel (ATK) performs equally well, irrespectively of the number of selected symbols. That is, the approximation captures the discriminant subtrees rooted at either the symbol C or D in all settings. This selection of discriminative subtrees is also reflected in the optimal value of the depth parameter λ determined during the model selection. While for the exact tree kernel the optimal λ is 10^{-2}, the approximate kernel yields best results with $\lambda = 10^{-3}$, thus putting emphasis on shallow subtrees and the discriminative production rules rooted at C and D.

To analyze the feature space induced by the selection of subtrees, we perform a kernel principle component analysis (PCA) (see Schölkopf et al., 1998; Braun et al., 2008) for the exact and the



(a) Classification performance (b) Kernel PCA plot for $N = 1$

Figure 4: Classification performance and kernel PCA plot for the supervised toy data.

approximate tree kernel. Figure 4(b) shows the sorted magnitudes of the principal components in feature space. Although the differences are marginal, comparing the spectra allows for viewing the approximate kernel as a denoised variant of the regular parse tree kernel. The variance of smaller components is shifted towards leading principle components, resulting in a dimensionality reduction in feature space (see Mika et al., 1999).

4.2 An Unsupervised Learning Task

In order to obtain an unsupervised learning task, we modify the artificial grammar to reflect the notion of anomaly detection. First, we incorporate redundancy into the grammar by increasing the probability of irrelevant production rules in (*4)–(*5) as follows

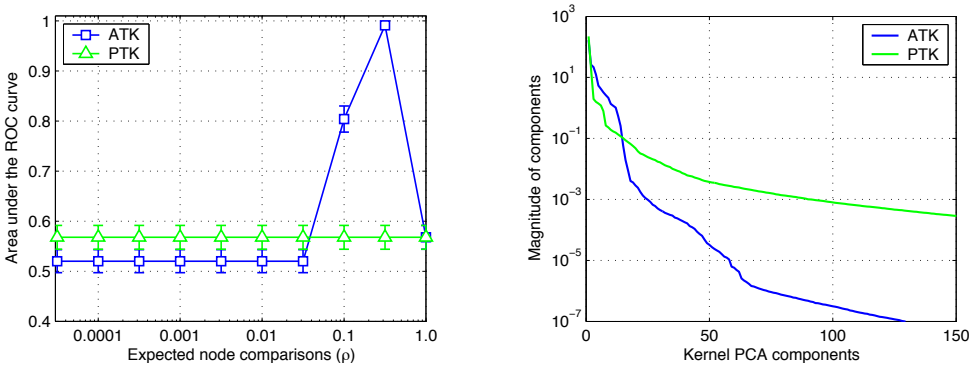
$$C \xrightarrow{[0.1|0.4|0.4]} A\ B \mid A \mid B \qquad (*4)$$

$$D \xrightarrow{[0.1|0.4|0.4]} B\ A \mid A \mid B \qquad (*5)$$

Second, we sample the parse trees such that training, validation, and testing sets contain 99% positive and 1% negative instances each, thus matching the anomaly detection scenario. We pursue the same two-stage procedure as in the previous section but first solve Optimization Problem 2 for adapting the approximate tree kernel to the unlabeled data and then employ a one-class SVM (Schölkopf et al., 1999) for training and testing.

Figure 5(a) shows the detection performance for the parse tree kernel and the approximate tree kernel for varying values of ρ. The parse tree kernel reaches an AUC value of 57%. Surprisingly, we observe a substantial gain in performance for the approximate kernel, leading to an almost perfect separation of the two classes for $\rho = 0.3$. Moreover, for the approximate kernel shallow subtrees are sufficient for detection of anomalies which is indicated by an optimal $\lambda = 10^{-3}$, whereas for the exact kernel subtrees of all depths need to be considered due to an optimal $\lambda = 1$.

The high detection performances can be explained by considering a kernel PCA of the two tree kernels in Figure 5(b). The redundant production rules introduce irrelevant and noisy dimensions into the feature space induced by the parse tree kernel. Clearly, for $\rho = 0.3$, the approximate tree kernel effectively reduces the intrinsic dimensionality by shifting the variance towards leading



(a) Detection performance (b) Kernel PCA plot for $\rho = 0.3$

Figure 5: Detection performance and kernel PCA plot for the unsupervised toy data.

components. Compared to the exact kernel, the resulting eigenspectrum of the approximate kernel possesses more explanatory and fewer noisy components.

5. Real-world Experiments

We now proceed to study the expressiveness, stability, and run-time performance of approximate tree kernels in real-world applications, namely supervised learning tasks dealing with question classification and web spam detection, respectively, and an unsupervised learning task on intrusion detection for HTTP and FTP traffic. In all experiments we employ the exact parse tree kernel and state-of-the-art implementations as baseline methods.

- **Question Classification.** Question classification is an important step for automatic answering of questions (Voorhees, 2004). The task is to categorize a user-supplied question into predefined semantic categories. We employ the data collection by Li and Roth (2002) consisting of 6,000 English questions assigned to six classes (abbreviation, entity, description, human, location, numeric value). Each question is transformed to a respective parse tree using the MEI Parser[1] (Charniak, 1999). For simplicity, we learn a discrimination between the category "entity" (1,339 instances) and all other categories using a two-class Support Vector Machine (SVM).

- **Web Spam Detection.** Web spam refers to fraudulent HTML documents, which yield high ranks in search engines through massive amounts of links. The detection of so-called link farms is essential for providing proper search results and protecting users from fraud. We use the web spam data as described by Castillo et al. (2006). The collection consists of HTML documents from normal and spam websites in the UK. All sites are examined by humans and manually annotated. We use a fault-tolerant HTML parser[2] to obtain parse trees from HTML documents. From the top 20 sites of both classes we sample 5,000 parse trees covering 974 web spam documents and 4,026 normal HTML pages. Again, we use a two-class SVM as the underlying learning algorithm.

1. Maximum-Entropy-Inspired Parser, see `ftp://ftp.cs.brown.edu/pub/nlparser`.
2. Beautiful Soup Parser, see `http://www.crummy.com/software/BeautifulSoup`.

- **Intrusion Detection.** Intrusion detection aims to automatically identify unknown attacks in network traffic. As labels for such data are hard to obtain, unsupervised learning has been a major focus in intrusion detection research (e.g., Eskin et al., 2002; Kruegel and Vigna, 2003; Rieck and Laskov, 2007; Laskov et al., 2008). Thus, for our experiments we employ a one-class SVM (Schölkopf et al., 1999) in the variant of Tax and Duin (1999) to detect anomalies in network traffic of the protocols HTTP and FTP. Network traffic for HTTP is recorded at the Fraunhofer FIRST institute, while FTP traffic is obtained from the Lawrence Berkeley National Laboratory [3] (see Paxson and Pang, 2003). Both traffic traces cover a period of 10 days. Attacks are additionally injected into the traffic using popular hacking tools.[4] The network data is converted to parse trees using the protocol grammars provided in the specifications (see Fielding et al., 1999; Postel and Reynolds, 1985). From the generated parse trees for each protocol we sample 5,000 instances and add 89 attacks for HTTP and 62 for FTP, respectively. This setting is similar to the data sets used in the DARPA intrusion detection evaluation (Lippmann et al., 2000).

Figure 6 shows the distribution of tree sizes in terms of nodes for each of the three learning tasks. For question classification, the largest tree comprises 113 nodes, while several parse trees in the web spam and intrusion detection data consist of more than 5,000 nodes.

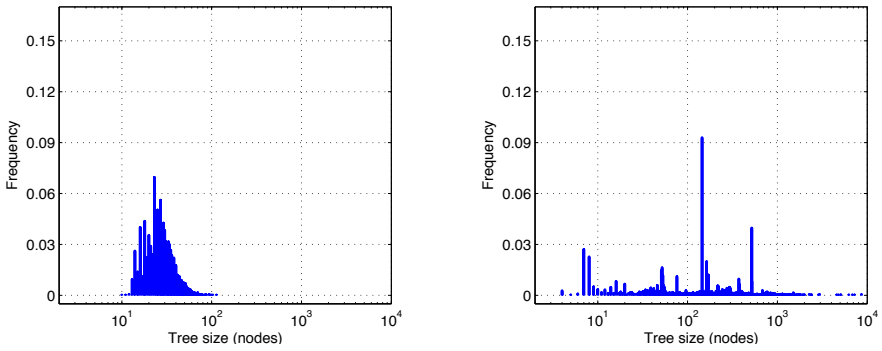
For each learning task, we pursue the two-stage procedure described in Section 3.5 and conduct the following experimental procedure: parse trees are randomly drawn from each data set and split into training, validation and test partitions consisting of 1,000 trees each. If not stated otherwise, we first draw 250 instances at random from the training set for the selection stage, where we solve Optimization Problem 1 or 2 with fixed $\lambda = 1$. In the application stage, the resulting approximate kernels are then compared to exact kernels using SVMs as underlying learning methods. Model selection is performed for the regularization parameter of the SVM and the depth parameter λ. We measure the area under the ROC curve of the resulting classifiers and report on averages over 10 repetitions with error bars indicate standard errors. In all experiments we make use of the LIBSVM library developed by Chang and Lin (2000).

5.1 Results for Question Classification

We first study the expressiveness of the approximate tree kernel and the exact parse tree kernel for the question classification task. We thus vary the number of selected symbols in Optimization Problem 1 and report on the achieved classification performance for the approximate tree kernel for varying N and the exact tree kernel in Figure 7(a).

As expected, the approximation becomes more accurate for increasing values of N, meaning that the more symbols are included in the approximation, the better is the resulting discrimination. However, the curve saturates to the performance of the regular parse tree kernel for selecting 7 and more symbols. The selected symbols are NP, VP, PP, S1, SBARQ, SQ, and TERM. The symbols NP, PP, and VP capture the coarse semantics of the considered text, while SBARQ and SQ correspond to the typical structure of questions. Finally, the symbol TERM corresponds to terminal symbols and contains the actual sequence of tokens including interrogative pronouns. The optimal depth λ for the approximate kernel is again lower with 10^{-2} in comparison to the optimal value of 10^{-1} for the exact kernel, as discriminative substructures are close to the selected symbols.

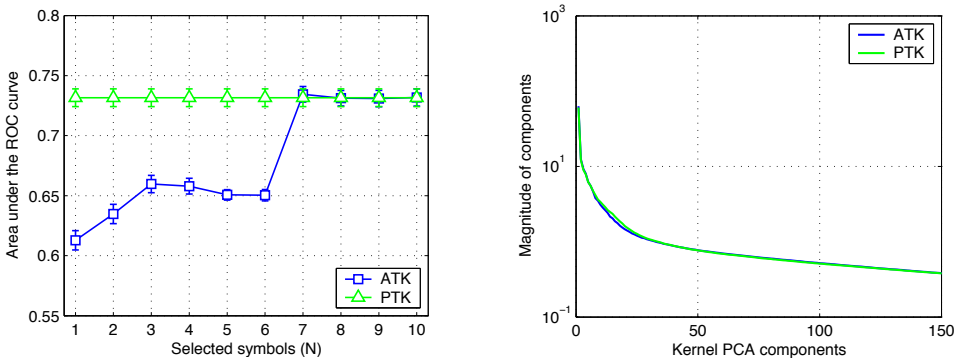
3. LBNL-FTP-PKT, `http://www-nrg.ee.lbl.gov/anonymized-traces.html`.
4. Metasploit Framework, see `http://www.metasploit.org`.



(a) Question classification (b) Web spam detection

(c) Intrusion detection (HTTP) (d) Intrusion detection (FTP)

Figure 6: Tree sizes for question classification, web spam detection and intrusion detection.

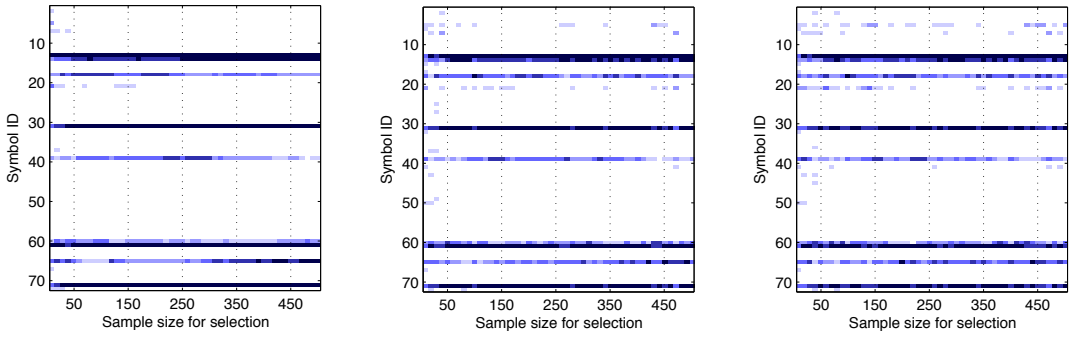


(a) Classification performance (b) Kernel PCA plot for $N = 7$

Figure 7: Classification performance and kernel PCA plot for the question classification task.

Figure 7(b) shows the eigenspectra of the parse tree kernel and its approximate variant with the above 7 selected symbols. Even though the proposed kernel is only an approximation of the regular tree kernel, their eigenspectra are nearly identical. That is, the approximate tree kernel leads to a nearly identical feature space to its exact counterpart.

The above experiments demonstrate the ability of the approximation to select discriminative symbols, yet it is not clear how the expressiveness of the approximate kernels depends on the re-



(a) Selection (b) Selection with 5% label noise (c) Selection with 10% label noise

Figure 8: Stability plot for the question classification task.

duced sample size in the selection stage. To examine this issue, we keep $N = 7$ fixed and vary the amount of data supplied for adapting the approximate tree kernel. Figure 8(a) displays the assignments of the selection function ω, where the size of the provided data is shown on the x-axis and the IDs of the grammar symbols are listed on the y-axis. The intensity of each point reflects the average number of times the corresponding symbol has been chosen in five repetitions. The selection remains stable for sample sizes of 150 parse trees, where consistently the correct 7 symbols are identified. Even if label noise is injected in the data, the approximation remains stable if at least 150 trees are considered for the selection as depicted in Figure 8(b) and 8(c).

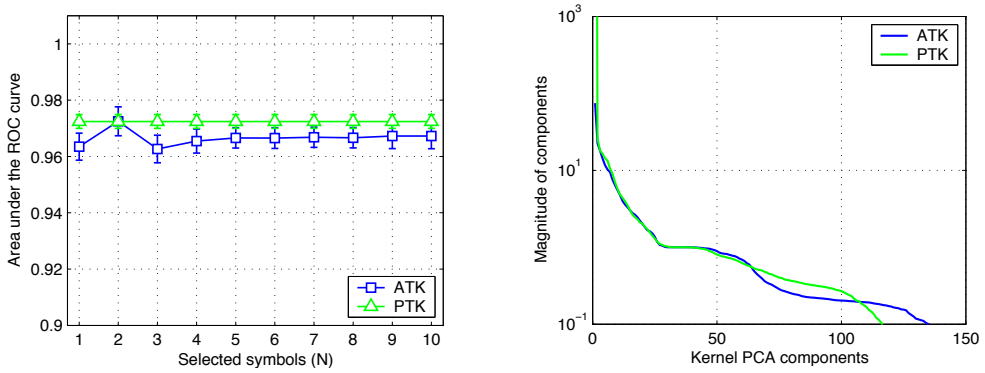
The results on question classification show that exploiting the redundancy in parse trees can be beneficial even when dealing with small trees. Approximate tree kernels identify a simplified representation that proves robust against label noise and leads to the same classification rate compared to regular parse tree kernels.

5.2 Results for Web Spam Detection

We now study approximate tree kernels for web spam detection. Unfortunately, training SVMs using the exact parse tree kernel proves intractable for many large trees in the data due to their excessive memory requirements. We thus exclude trees from the web spam data set with more than 1,500 nodes for the following experiments. Again, we vary the number of symbols to be selected and measure the corresponding AUC value over 10 repetitions.

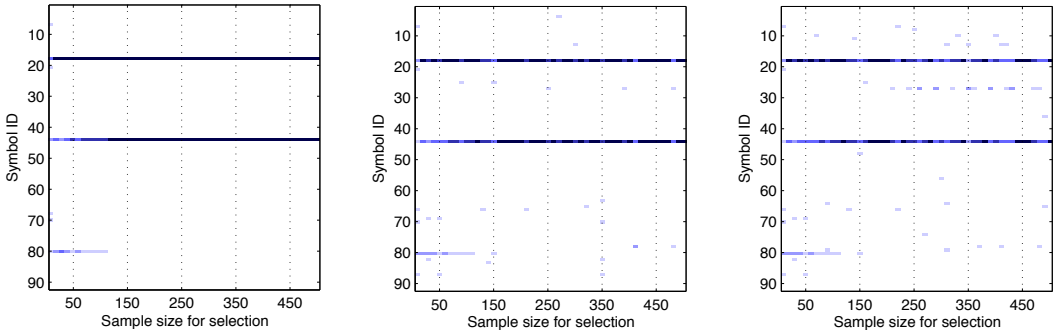
The results are shown in Figure 9(a). The approximation is consistently on par with the regular parse tree kernel for four and more selected labels, as the differences in this interval are not significant. However, the best result is obtained for selecting only two symbols. The approximation picks the tags HTML and BODY. We credit this finding to the usage of templates in spam websites inducing a strict order of high-level tags in the documents. In particular, header and meta tags occurring in subtrees below the HTML tag are effective for detecting spam templates. As a consequence, the optimal $\lambda = 10^{-1}$ for the approximate kernel effectively captures discriminative features reflecting web spam templates rooted at the HTML and BODY tag. The eigendecomposition of the two kernels in Figure 9(b) hardly show any differences. As for the question classification task, the exact and approximate tree kernels share the same expressiveness.

Figure 10 shows the stability of the selection function for varying amounts of data considered in the selection stage where N is fixed to 2. The selection saturates for samples containing at least 120



(a) Classification performance (b) Kernel PCA plot for $N = 2$

Figure 9: Classification performance and kernel PCA plot for the web spam detection task.



(a) Selection (b) Selection with 5% label noise (c) Selection with 10% label noise

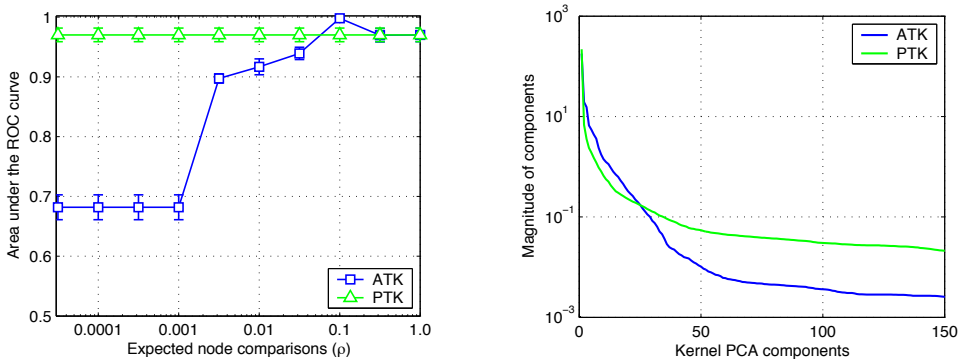
Figure 10: Stability plot for the web spam detection task.

parse trees and the two symbols HTML and BODY are chosen consistently. Moreover, the selection of symbols for web spam detection proves robust against label noise. Even for a noise ratio of 10% which corresponds to flipping every tenth label, the same symbols are selected. This result confirms the property of web spam to be generated from templates which provides a strong feature for discrimination even in presence of label noise.

5.3 Results for Intrusion Detection

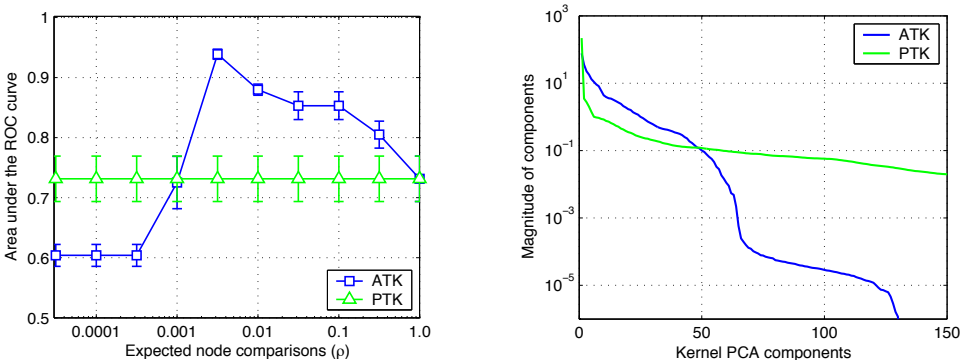
In this section, we study the expressiveness of approximate kernels for unsupervised intrusion detection. Since label information is not available, we adapt the selection function to the data using Optimization Problem 2. The resulting approximate tree kernels are then employed together with a one-class SVM for the detection of attacks in HTTP and FTP parse trees. To determine the impact of the approximation on the detection performance, we vary the number of expected node comparisons, that is, variable ρ in Optimization Problem 2. We again exclude trees comprising more than 1,500 nodes due to prohibitive memory requirements for the exact tree kernel.

Figures 11 (HTTP) and 12 (FTP) show the observed detection rates on the left for the approximate and the exact tree kernel. Clearly, the approximate tree kernel performs identically to its exact counterpart if the ratio of node comparisons ρ equals 100%. However, when the number of



(a) Detection performance (b) Kernel PCA plot for $\rho = 0.1$

Figure 11: Detection performance and analysis of the intrusion detection task (HTTP).



(a) Detection performance (b) Kernel PCA plot for $\rho = 0.003$

Figure 12: Detection performance and analysis of the intrusion detection task (FTP).

comparisons is restricted to only a fraction, the approximate kernel significantly outperforms the exact parse tree kernel and leads to a superior detection rate. The approximate tree kernel realizes an AUC improvement of 1% for HTTP data. For the FTP protocol, the differences are more severe: the approximate kernel outperforms its exact counterpart and yields an AUC improvement of 20%. The optimal depth parameter λ for the approximate kernel is 10^{-2} for HTTP and 10^{-2} for FTP, while the exact tree kernel requires $\lambda = 10^{-1}$ in the optimal setting. This result demonstrates that the approximation identifies relevant grammar symbols by focusing on shallow subtrees comprising discriminative patterns.

These gains in performance can be explained by looking at the respective eigenspectra, depicted in Figures 11(b) and 12(b). Compared to the regular kernel, the approximate kernel yields remarkably fewer noisy components. This is particularly the case for FTP traffic. Moreover, the variance is shifted toward only a few leading components. The approximate tree kernel performs a dimensionality reduction by suppressing noisy and redundant parts of the feature space. For HTTP traffic, such redundancy is for instance induced by common web browsers like Internet Explorer and Mozilla Firefox whose header attributes constitute a good portion of the resulting parse trees. This syntactical information is delusive in the context of intrusion detection and hence their removal

improves the detection performance. Note that the observed gain in performance is achieved using only less than 10% of the grammar symbols. That is, the approximation is not only more accurate than the exact parse tree kernel but also concisely represented.

5.4 Run-time Performance

As we have seen in the previous sections, approximate kernels can lead to a concise description of the task at hand by selecting discriminative substructures in data. In this section we compare the run-time and memory requirements of the approximate tree kernel with state-of-the-art implementations of the exact tree kernels. We first measure the time for selection, training, and testing phases using SVMs as underlying learning methods. For all data sets, we use 250 randomly drawn trees in the selection stage where training and test sets consist of 1,000 instances each. Again, we exclude large trees with more than 1,500 nodes because of excessive memory requirements.

Selection stage on 250 parse trees	
Question classification	17s $\pm$0
Web spam detection	144s $\pm$28
Intrusion detection (HTTP)	43s $\pm$7
Intrusion detection (FTP)	31s $\pm$2

Table 1: Selection stage prior to training and testing phase.

The run-time for the selection of symbols prior to application of the SVMs are presented in Table 1. For all three data sets, a selection is determined in less than 3 minutes, demonstrating the advantage of phrasing the selection as a simple linear program. Table 5.4 lists the training and testing times using the approximate tree kernel (ATK) and a fast implementation for the exact tree kernel (PTK2) devised by Moschitti (2006a). As expected, the size of the trees influences the observed results. For the small trees in the question classification task we record run-time improvements by a factor of 1.7 while larger trees in the other tasks give rise to speed-up factors between $2.8 - 13.8$. Note that the total run-time of the application stage is only marginally affected by the initial selection stage that is performed only once prior to the learning process. For example, in the task of web spam detection a speed-up of roughly 10 is attained for the full experimental evaluation, as the selection is performed once, whereas 25 runs of training and testing are necessary for model selection.

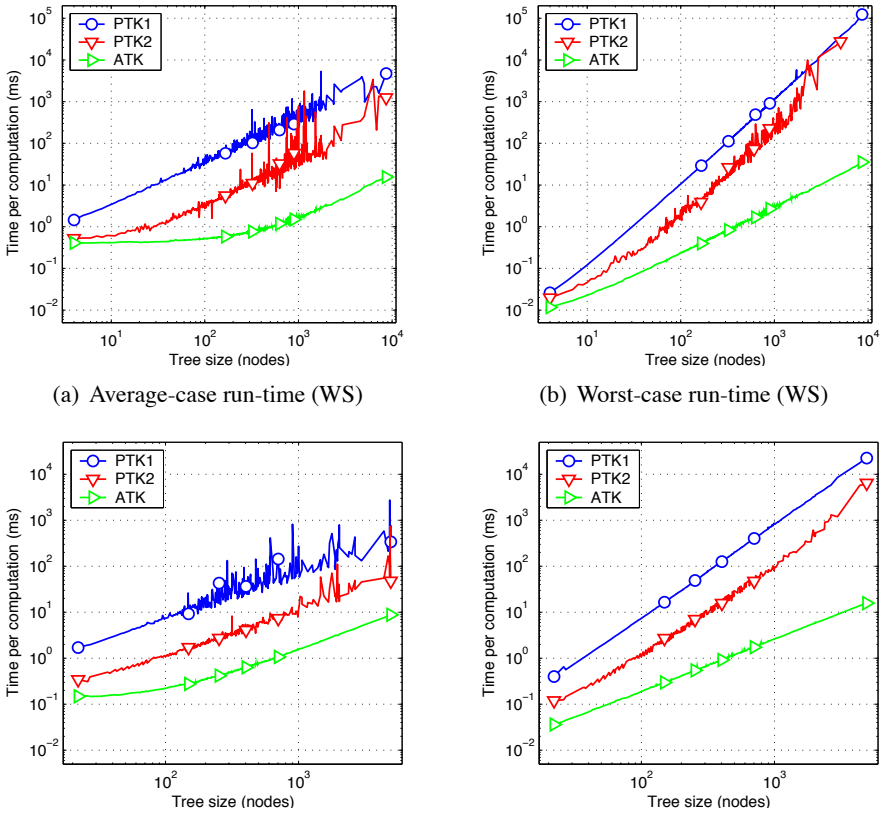
However, the interpretability of the results reported in Table 5.4 is limited because parse trees containing more than 1,500 nodes have been excluded from the experiment and the true performance gain induced by approximate tree kernels cannot be estimated. Moreover, the reported training and testing times refer to a particular learning method and cannot be transferred to other methods and applications, such as clustering and regression tasks. To address these issues, we study the run-time performance and memory consumption of tree kernels explicitly—independently of a particular learning method. Notice that for these experiments we include parse trees of all sizes. As baselines, we include a standard implementation of the parse tree kernel (PTK1) detailed by Shawe-Taylor and Cristianini (2004) and the improved variant (PTK2) proposed by Moschitti (2006a).

For each kernel, we estimate the average run-time and memory requirements by computing kernels between reference trees of fixed sizes and 100 randomly drawn trees. We also consider the worst-case scenario for each data set, which occurs if kernels are computed between identical parse trees, thus realizing the maximal number of matching node pairs. We focus in our experiments on

	ATK	*PTK2*	*Speed-up*
Training time on 1,000 parse trees			
Question classification	42s ±4	72s ±7	1.7×
Web spam detection	111s ±17	1,487s ±435	13.4×
Intrusion detection (HTTP)	123s ±20	349s ±80	2.8×
Intrusion detection (FTP)	125s ±14	517s ±129	5.8×
Testing time on 1,000 parse trees			
Question classification	40s ±4	70s ±2	1.8×
Web spam detection	112s ±18	1,542s ±471	13.8×
Intrusion detection (HTTP)	81s ±14	225s ±71	2.8×
Intrusion detection (FTP)	107s ±15	455s ±112	4.1×

Table 2: Training and testing time of SVMs using the exact and the approximate tree kernel.

the learning tasks of web spam and intrusion detection (HTTP), where results for question classification and FTP are analogous.



(a) Average-case run-time (WS)

(b) Worst-case run-time (WS)

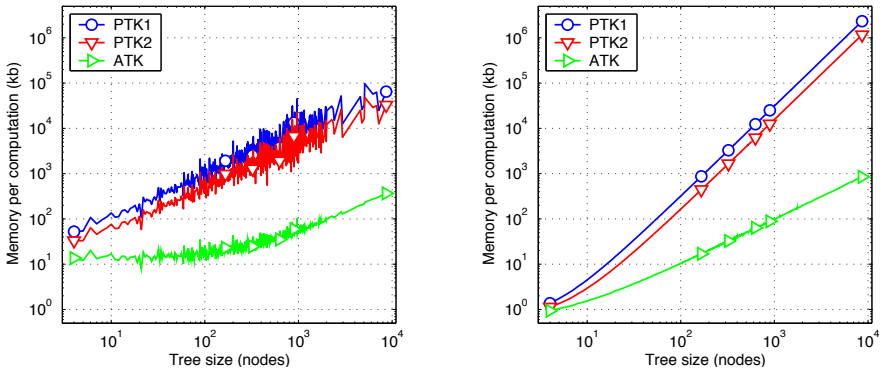
(c) Average-case run-time (ID)

(d) Worst-case run-time (ID)

Figure 13: Run-times for web spam (WS) and intrusion detection (ID).

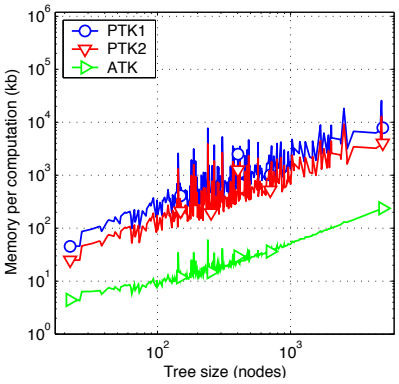
Figure 13 illustrates the run-time performance of the approximate and the two exact tree kernels. The run-time is given in milliseconds (ms) per kernel computation on the y-axis and the size of the

considered trees is shown on the x-axis. Both axes are presented in log-scale. Although the improved variant by Moschitti (PTK2) is significantly faster than the standard implementation, neither of the two show compelling run-times in both tasks. For both implementations of the regular tree kernel, a single kernel computation can take more than 10 seconds, thus rendering large-scale applications infeasible. By contrast, the approximate tree kernel computes similarities between trees up to three orders of magnitude faster and yields a worst-case computation time of less than 40 ms for the web spam detection task and less than 20 ms for the intrusion detection task. The worst-case analysis shows that the exact tree kernel scales quadratically in the number of nodes whereas the approximate tree kernel is computed in sub-quadratic time in the size of the trees.

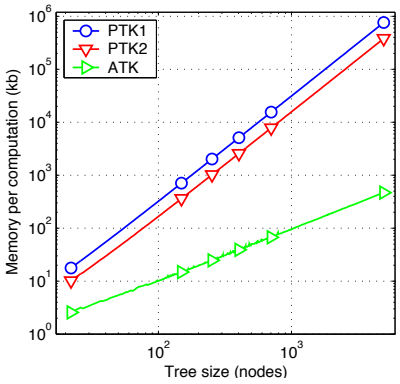


(a) Average-case memory (WS)

(b) Worst-case memory (WS)



(c) Average-case memory (ID))



(d) Worst-case memory (ID)

Figure 14: Memory requirements for web spam (WS) and intrusion detection (ID).

Figure 14 reports on average and worst-case memory requirements for the web spam detection and intrusion detection task. The memory consumption in kilobytes is depicted on the y-axis and the size of the considered trees is shown on the x-axis. Both axes are given in logarithmic scale. In all figures, the curves of the approximate kernel are significantly below the variants of the parse tree kernel. The allocated memory for the regular tree kernel exceeds 1 gigabytes in both learning tasks, which is clearly prohibitive for a single kernel computation. In contrast the approximate tree kernel requires at most 800 kilobytes. For the worst-case estimation, the memory consumption of the exact kernel scales quadratically in the number of tree nodes while the approximate tree kernel scales sub-quadratically due to the sparse selection of symbols.

6. Conclusions

Learning with large trees render regular parse tree kernels inapplicable due to quadratic run-time and memory requirements. As a remedy, we propose to approximate regular tree kernels. Our approach splits into a selection and an application stage. In the selection stage, the computation of tree kernels is narrowed to a sparse subset of subtrees rooted in appropriate grammar symbols. The symbols are chosen according to their discriminative power for supervised settings and to minimize the expected number of node comparisons for unsupervised settings, respectively. We derive linear programming approaches to identify such symbols, where the resulting optimization problems can be solved with standard techniques. In the subsequent application stage, learning algorithms benefit from the initial selection because run-time and memory requirements for the kernel computation are significantly reduced.

We evaluate the approximate trees kernels with SVMs as underlying learning algorithms for question classification, web spam detection and intrusion detection. In all experiments, the approximate tree kernels not only replicate the predictive performances of exact kernels but also provide concise representations by operating on only 2–10% of the available grammar symbols. The resulting approximate kernels lead to significant improvements in terms of run-time and memory requirements. For large trees, the approximation reduces a single kernel computation from 1 gigabyte to less than 800 kilobytes, accompanied by run-time improvements up to three orders of magnitude. We also observe improvements for parse trees generated for sentences in natural language, however, at a smaller scale. The most dramatic results are obtained for intrusion detection. Here, a kernel PCA shows that approximate tree kernels effectively identify relevant dimensions in feature space and discard redundant and noisy subspaces from the kernel computation. Consequently, the approximate kernels perform more efficiently and more accurately than their exact counterparts achieving AUC improvements of up to 20%.

To the best of our knowledge, we present the first efficient approach to learning with large trees containing thousands of nodes. In view of the many large-scale applications comprising structured data , the presented work provides means for efficient and accurate learning with large structures. Although we focus on classification, approximate tree kernels are easily leveraged to other kernel-based learning tasks, such as regression and clustering, using the introduced techniques. Moreover, the devised approximate tree kernels build on the concept of convolution over local kernel functions. Our future work will focus on transferring attained performance gains to the framework of convolution kernels, aiming at rendering learning with various types of complex structured data feasible in large-scale applications.

Acknowledgments

The authors would like to thank the anonymous reviewers for helpful comments and suggestions. Furthermore, we like to thank Patrick Düssel and René Gerstenberger for providing efficient implementations of network protocol parsers. The authors gratefully acknowledge the funding from the Bundesministerium für Bildung und Forschung under the project REMIND (FKZ 01-IS07007A) and from the FP7-ICT Program of the European Community under the PASCAL2 Network of Excellence, ICT-216886. Most of the work was done when UB was at TU Berlin.

References

C. Bockermann, M. Apel, and M. Meier. Learning SQL for database intrusion detection using context-sensistive modelling. In *Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, 2009. to appear.

N. Borisov, D.J. Brumley, H. Wang, J. Dunagan, P. Joshi, and C. Guo. Generic application-level protocol analyzer and its language. In *Proc. of Network and Distributed System Security Symposium (NDSS)*, 2007.

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.

C. Castillo, D. Donato, L. Becchetti, P. Boldi, S. Leonardi, M. Santini, and S. Vigna. A reference collection for web spam. *SIGIR Forum*, 40(2):11–24, 2006. URL `http://portal.acm.org/citation.cfm?id=1189703`.

C.-C. Chang and C.-J. Lin. LIBSVM: Introduction and benchmarks. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, 2000.

E. Charniak. A maximum-entropy-inspired parser. Technical Report CS-99-12, Brown University, 1999.

E. Cilia and A. Moschitti. Advanced tree-based kernels for protein classification. In *Artificial Intelligence and Human-Oriented Computing (AI*IA), 10th Congress*, LNCS, pages 218–229, 2007.

M. Collins and N. Duffy. Convolution kernel for natural language. In *Advances in Neural Information Proccessing Systems (NIPS)*, volume 16, pages 625–632, 2002.

N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola. On kernel target alignment. In *Advances in Neural Information Proccessing Systems (NIPS)*, volume 14, pages 367–737, 2001.

I. Drost and T. Scheffer. Thwarting the nigritude ultramarine: Learning to identify link spam. In *Proc. of the Eurpoean Conference on Machine Learning (ECML)*, 2005.

P. Düssel, C. Gehl, P. Laskov, and K. Rieck. Incorporation of application layer protocol syntax into anomaly detection. In *Proc. of International Conference on Information Systems Security (ICISS)*, pages 188–202, 2008.

E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. *Applications of Data Mining in Computer Security*, chapter A geometric framework for unsupervised anomaly detection: detecting intrusions in unlabeled data. Kluwer, 2002.

R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. URL `http://www.ietf.org/rfc/rfc2616.txt`. Updated by RFC 2817.

D. Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, UC Santa Cruz, July 1999.

J.E. Hopcroft and J.D. Motwani, R. Ullmann. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2 edition, 2001.

H. Kashima and T. Koyanagi. Kernels for semi-structured data. In *International Conference on Machine Learning (ICML)*, pages 291–298, 2002.

D. Knuth. *The Art of Computer Programming*, volume 3. Addison-Wesley, 1973.

C. Kruegel and G. Vigna. Anomaly detection of web-based attacks. In *Proc. of 10th ACM Conf. on Computer and Communications Security*, pages 251–261, 2003.

P. Laskov, K. Rieck, and K.-R. Müller. Machine learning for intrusion detection. In *Mining Massive Data Sets for Security*, pages 366–373. IOS press, 2008.

X. Li and D. Roth. Learning question classifiers. In *International Conference on Computational Linguistics (ICCL)*, pages 1–7, 2002. doi: http://dx.doi.org/10.3115/1072228.1072378.

R. Lippmann, J.W. Haines, D.J. Fried, J. Korba, and K. Das. The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks*, 34(4):579–595, 2000.

C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

S. Mika, B. Schölkopf, A.J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de–noising in feature spaces. In M.S. Kearns, S.A. Solla, and D.A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 536–542. MIT Press, 1999.

A. Moschitti. Making tree kernels practical for natural language processing. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006a.

A. Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *European Conference on Machine Learning (ECML)*, 2006b.

A. Moschitti and F.M. Zanzotto. Fast and effective kernels for relation learning from texts. In *International Conference on Machine Learning (ICML)*, 2007.

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.

R. Pang, V. Paxson, R. Sommer, and L.L. Peterson. binpac: a yacc for writing application protocol parsers. In *Proc. of ACM Internet Measurement Conference*, pages 289–300, 2006.

V. Paxson and R. Pang. A high-level programming environment for packet trace anonymization and transformation. In *Proc. of Applications, Technologies, Architectures, and Protocols for Computer Communications SIGCOMM*, pages 339 – 351, 2003.

J. Postel and J. Reynolds. File Transfer Protocol. RFC 959 (Standard), October 1985. URL `http://www.ietf.org/rfc/rfc959.txt`. Updated by RFCs 2228, 2640, 2773, 3659.

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, U. Brefeld, and T. Krueger. Approximate kernels for trees. Technical Report FIRST 5/2008, Fraunhofer Institute FIRST, September 2008.

B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A.J. 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.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. TR 87, Microsoft Research, Redmond, WA, 1999.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

J. Suzuki and H. Isozaki. Sequence and tree kernels with statistical feature mining. In *Advances in Neural Information Proccessing Systems (NIPS)*, volume 17, 2005.

J. Suzuki, H. Isozaki, and E. Maeda. Convolution kernels with feature selection for natural language processing tasks. In *Annual Meeting on Association for Computational Linguistics (ACL)*, 2004.

D.M.J. Tax and R.P.W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20 (11–13):1191–1199, 1999.

V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

S.V.N. Vishwanathan and A.J. Smola. Fast kernels for string and tree matching. In *Advances in Neural Information Proccessing Systems (NIPS)*, pages 569–576, 2003.

E.M. Voorhees. Overview of the trec 2004 question answering track. In *Proc. of the Thirteenth Text Retreival Conference (TREC)*, 2004.

G. Wondracek, P.M. Comparetti, C. Kruegel, and E. Kirda. Automatic network protocol analysis. In *Proc. of Network and Distributed System Security Symposium (NDSS)*, 2008.

B. Wu and B. D. Davison. Identifying link farm spam pages. In *WWW '05: Special Interest Tracks, 14th International Conference on World Wide Web*, 2005.

D. Zhang and W. S. Lee. Question classification using support vector machines. In *Annual International ACM SIGIR Conference*, pages 26–32, 2003.

Journal of Machine Learning Research 11 (2010) 581-602 Submitted 7/09; Revised 12/09; Published 2/10

On Finding Predictors for Arbitrary Families of Processes

Daniil Ryabko DANIIL@RYABKO.NET
INRIA Lille-Nord Europe
40, avenue Halley
Parc Scientifique de la Haute Borne
59650 Villeneuve d'Ascq, France

Editor: Gábor Lugosi

Abstract

The problem is sequence prediction in the following setting. A sequence $x_1,\dots,x_n,\dots$ of discrete-valued observations is generated according to some unknown probabilistic law (measure) μ. After observing each outcome, it is required to give the conditional probabilities of the next observation. The measure μ belongs to an arbitrary but known class $\mathcal{C}$ of stochastic process measures. We are interested in predictors ρ whose conditional probabilities converge (in some sense) to the "true" μ-conditional probabilities, if any $\mu \in \mathcal{C}$ is chosen to generate the sequence. The contribution of this work is in characterizing the families $\mathcal{C}$ for which such predictors exist, and in providing a specific and simple form in which to look for a solution. We show that if any predictor works, then there exists a Bayesian predictor, whose prior is discrete, and which works too. We also find several sufficient and necessary conditions for the existence of a predictor, in terms of topological characterizations of the family $\mathcal{C}$, as well as in terms of local behaviour of the measures in $\mathcal{C}$, which in some cases lead to procedures for constructing such predictors.

It should be emphasized that the framework is completely general: the stochastic processes considered are not required to be i.i.d., stationary, or to belong to any parametric or countable family.

Keywords: sequence prediction, time series, online prediction, Bayesian prediction

1. Introduction

Given a sequence $x_1,\dots,x_n$ of observations $x_i \in \mathcal{X}$, where $\mathcal{X}$ is a finite set, we want to predict what are the probabilities of observing $x_{n+1} = x$ for each $x \in \mathcal{X}$, or, more generally, probabilities of observing different $x_{n+1},\dots,x_{n+h}$, before x_{n+1} is revealed, after which the process continues. It is assumed that the sequence is generated by some unknown stochastic process μ, a probability measure on the space of one-way infinite sequences $\mathcal{X}^\infty$. The goal is to have a predictor whose predicted probabilities converge (in a certain sense) to the correct ones (that is, to μ-conditional probabilities). In general this goal is impossible to achieve if nothing is known about the measure μ generating the sequence. In other words, one cannot have a predictor whose error goes to zero for any measure μ. The problem becomes tractable if we assume that the measure μ generating the data belongs to some known class $\mathcal{C}$. The questions addressed in this work are a part of the following general problem: given an arbitrary set $\mathcal{C}$ of measures, how can we find a predictor that performs well when the data is generated by any $\mu \in \mathcal{C}$, and whether it is possible to find such a predictor at all. An example of a generic property of a class $\mathcal{C}$ that allows for construction of a predictor, is that $\mathcal{C}$ is countable. Clearly, this condition is very strong. An example, important from the applications

point of view, of a class $\mathcal{C}$ of measures for which predictors are known, is the class of all stationary measures. The general question, however, is very far from being answered.

The contribution of this work to solving this question is, first, in that we provide a specific form in which to look for a predictor. More precisely, we show that if a predictor that predicts every $\mu \in \mathcal{C}$ exists, then such a predictor can also be obtained as a weighted sum of countably many elements of $\mathcal{C}$. This result can also be viewed as a justification of the Bayesian approach to sequence prediction: if there exists a predictor which predicts well every measure in the class, then there exists a Bayesian predictor (with a rather simple prior) that has this property too. In this respect it is important to note that the result obtained about such a Bayesian predictor is pointwise (holds for every μ in $\mathcal{C}$), and stretches far beyond the set its prior is concentrated on. Next, we derive some characterizations of families $\mathcal{C}$ for which a predictor exist. We first analyze what is furnished by the notion of separability, when a suitable topology can be found: we find that it is a sufficient but not always a necessary condition. We then derive some sufficient conditions for the existence of a predictor which are based on local (truncated to the first n observation) behaviour of measures in the class $\mathcal{C}$. Necessary conditions cannot be obtained in this way (as we demonstrate), but sufficient conditions, along with rates of convergence and construction of predictors, can be found.

The *motivation* for studying predictors for arbitrary classes $\mathcal{C}$ of processes is two-fold. First of all, prediction is a basic ingredient for constructing intelligent systems. Indeed, in order to be able to find optimal behaviour in an unknown environment, an intelligent agent must be able, at the very least, to predict how the environment is going to behave (or, to be more precise, how relevant parts of the environment are going to behave). Since the response of the environment may in general depend on the actions of the agent, this response is necessarily non-stationary for explorative agents. Therefore, one cannot readily use prediction methods developed for stationary environments, but rather has to find predictors for the classes of processes that can appear as a possible response of the environment.

Apart from this, the problem of prediction itself has numerous applications in such diverse fields as data compression, market analysis, bioinformatics, and many others. It seems clear that prediction methods constructed for one application cannot be expected to be optimal when applied to another. Therefore, an important question is how to develop specific prediction algorithms for each of the domains.

1.1 Prior Work

As it was mentioned, if the class $\mathcal{C}$ of measures is countable (that is, if $\mathcal{C}$ can be represented as $\mathcal{C} := \{\mu_k : k \in \mathbb{N}\}$), then there exists a predictor which performs well for any $\mu \in \mathcal{C}$. Such a predictor can be obtained as a Bayesian mixture $\rho_S := \sum_{k\in\mathbb{N}} w_k \mu_k$, where w_k are summable positive real weights, and it has very strong predictive properties; in particular, ρ_S predicts every $\mu \in \mathcal{C}$ in total variation distance, as follows from the result of Blackwell and Dubins (1962). Total variation distance measures the difference in (predicted and true) conditional probabilities of all future events, that is, not only the probabilities of the next observations, but also of observations that are arbitrary far off in the future (see formal definitions below). In the context of sequence prediction the measure ρ_S was first studied by Solomonoff (1978). Since then, the idea of taking a convex combination of a finite or countable class of measures (or predictors) to obtain a predictor permeates most of the research on sequential prediction (see, for example, Cesa-Bianchi and Lugosi, 2006) and more general learning problems (Hutter, 2005; Ryabko and Hutter, 2008a). In practice it is clear that, on

the one hand, countable models are not sufficient, since already the class $\mu_p, p \in [0,1]$ of Bernoulli i.i.d. processes, where p is the probability of 0, is not countable. On the other hand, prediction in total variation can be too strong to require; predicting probabilities of the next observation may be sufficient, maybe even not on every step but in the Cesaro sense. A key observation here is that a predictor $\rho_S = \sum w_k \mu_k$ may be a good predictor not only when the data is generated by one of the processes $\mu_k, k \in \mathbb{N}$, but when it comes from a much larger class. Let us consider this point in more detail. Fix for simplicity $\mathcal{X} = \{0,1\}$. The Laplace predictor

$$\lambda(x_{n+1} = 0 | x_1, \ldots, x_n) = \frac{\#\{i \leq n : x_i = 0\} + 1}{n + |\mathcal{X}|} \tag{1}$$

predicts any Bernoulli i.i.d. process: although convergence in total variation distance of conditional probabilities does not hold, predicted probabilities of the next outcome converge to the correct ones. Moreover, generalizing the Laplace predictor, a predictor λ_k can be constructed for the class M_k of all k-order Markov measures, for any given k. As was found by Ryabko (1988), the combination $\rho_R := \sum w_k \lambda_k$ is a good predictor not only for the set $\cup_{k \in \mathbb{N}} M_k$ of all finite-memory processes, but also for any measure μ coming from a much larger class: that of all stationary measures on $\mathcal{X}^\infty$. Here prediction is possible only in the Cesaro sense (more precisely, ρ_R predicts every stationary process in expected time-average Kullback-Leibler divergence, see definitions below). The Laplace predictor itself can be obtained as a Bayes mixture over all Bernoulli i.i.d. measures with uniform prior on the parameter p (the probability of 0). However, as was observed in Hutter (2007) (and as is easy to see), the same (asymptotic) predictive properties are possessed by a Bayes mixture with a countably supported prior which is dense in $[0,1]$ (e.g., taking $\rho := \sum w_k \delta_k$ where $\delta_k, k \in \mathbb{N}$ ranges over all Bernoulli i.i.d. measures with rational probability of 0). For a given k, the set of k-order Markov processes is parametrized by finitely many $[0,1]$-valued parameters. Taking a dense subset of the values of these parameters, and a mixture of the corresponding measures, results in a predictor for the class of k-order Markov processes. Mixing over these (for all $k \in \mathbb{N}$) yields, as in Ryabko (1988), a predictor for the class of all stationary processes. Thus, for the mentioned classes of processes, a predictor can be obtained as a Bayes mixture of countably many measures in the class. An additional reason why this kind of analysis is interesting is because of the difficulties arising in trying to construct Bayesian predictors for classes of processes that can not be easily parametrized. Indeed, a natural way to obtain a predictor for a class $\mathcal{C}$ of stochastic processes is to take a Bayesian mixture of the class. To do this, one needs to define the structure of a probability space on $\mathcal{C}$. If the class $\mathcal{C}$ is well parametrized, as is the case with the set of all Bernoulli i.i.d. process, then one can integrate with respect to the parametrization. In general, when the problem lacks a natural parametrization, although one can define the structure of the probability space on the set of (all) stochastic process measures in many different ways, the results one can obtain will then be with probability 1 with respect to the prior distribution (see, for example, Jackson et al., 1999). Pointwise consistency cannot be assured (see, for example, Diaconis and Freedman, 1986) in this case, meaning that some (well-defined) Bayesian predictors are not consistent on some (large) subset of $\mathcal{C}$. Results with prior probability 1 can be hard to interpret if one is not sure that the structure of the probability space defined on the set $\mathcal{C}$ is indeed a natural one for the problem at hand (whereas if one does have a natural parametrization, then usually results for every value of the parameter can be obtained, as in the case with Bernoulli i.i.d. processes mentioned above). The results of the present work show that when a predictor exists it can indeed be given as a Bayesian predictor, which predicts every (and not almost every) measure in the class, while its support is only a countable set.

A related question is formulated as a question about two individual measures, rather than about a class of measures and a predictor. Namely, one can ask under which conditions one stochastic process predicts another. In Blackwell and Dubins (1962) it was shown that if one measure is absolutely continuous with respect to another, than the latter predicts the former (the conditional probabilities converge in a very strong sense). In Ryabko and Hutter (2007, 2008b) a weaker form of convergence of probabilities (in particular, convergence of expected average KL divergence) is obtained under weaker assumptions.

1.2 The Results

First, we show that if there is a predictor that performs well for every measure coming from a class $\mathcal{C}$ of processes, then a predictor can also be obtained as a convex combination $\sum_{k\in\mathbb{N}} w_k\mu_k$ for some $\mu_k \in \mathcal{C}$ and some $w_k > 0$, $k \in \mathbb{N}$. This holds if the prediction quality is measured by either total variation distance, or expected average KL divergence: one measure of performance that is very strong, the other rather weak. The analysis for the total variation case relies on the fact that if ρ predicts μ in total variation distance, then μ is absolutely continuous with respect to ρ, so that $\rho(x_{1..n})/\mu(x_{1..n})$ converges to a positive number with μ-probability 1 and with a positive ρ-probability. However, if we settle for a weaker measure of performance, such as expected average KL divergence, measures $\mu \in \mathcal{C}$ are typically singular with respect to a predictor ρ. Nevertheless, since ρ predicts μ we can show that $\rho(x_{1..n})/\mu(x_{1..n})$ decreases subexponentially with n (with high probability or in expectation); then we can use this ratio as an analogue of the density for each time step n, and find a convex combination of countably many measures from $\mathcal{C}$ that has desired predictive properties for each n. Combining these predictors for all n results in a predictor that predicts every $\mu \in \mathcal{C}$ in average KL divergence. The proof techniques developed have a potential to be used in solving other questions concerning sequence prediction, in particular, the general question of how to find a predictor for an arbitrary class $\mathcal{C}$ of measures.

We then exhibit some sufficient conditions on the class $\mathcal{C}$, under which a predictor for all measures in $\mathcal{C}$ exists. It is important to note that none of these conditions relies on a parametrization of any kind. The conditions presented are of two types: conditions on asymptotic behaviour of measures in $\mathcal{C}$, and on their local (restricted to first n observations) behaviour. Conditions of the first type concern separability of $\mathcal{C}$ with respect to the total variation distance and the expected average KL divergence. We show that in the case of total variation separability is a necessary and sufficient condition for the existence of a predictor, whereas in the case of expected average KL divergence it is sufficient but is not necessary.

The conditions of the second kind concern the "capacity" of the sets $\mathcal{C}^n := \{\mu^n : \mu \in \mathcal{C}\}$, $n \in \mathbb{N}$, where μ^n is the measure μ restricted to the first n observations. Intuitively, if $\mathcal{C}^n$ is small (in some sense), then prediction is possible. We measure the capacity of $\mathcal{C}^n$ in two ways. The first way is to find the maximum probability given to each sequence $x_1,\dots,x_n$ by some measure in the class, and then take a sum over $x_1,\dots,x_n$. Denoting the obtained quantity c_n, one can show that it grows polynomially in n for some important classes of processes, such as i.i.d. or Markov processes. We show that, in general, if c_n grows subexponentially then a predictor exists that predicts any measure in $\mathcal{C}$ in expected average KL divergence. On the other hand, exponentially growing c_n are not sufficient for prediction. A more refined way to measure the capacity of $\mathcal{C}^n$ is using a concept of channel capacity from information theory, which was developed for a closely related problem of finding optimal codes for a class of sources. We extend corresponding results from information

theory to show that sublinear growth of channel capacity is sufficient for the existence of a predictor, in the sense of expected average divergence. Moreover, the obtained bounds on the divergence are optimal up to an additive logarithmic term.

The rest of the paper is organized as follows. Section 2 introduces the notation and definitions. In Section 3 we show that if any predictor works than there is a Bayesian one that works, while in Section 4 we provide several characterizations of predictable classes of processes. Section 4.1 is concerned with separability, while Section 4.2 analyzes conditions based on local behaviour of measures. Finally, Section 5 provides outlook and discussion.

As running examples that illustrate the results of each section we use countable classes of measures, the family of all Bernoulli i.i.d. processes, and that of all stationary processes.

2. Preliminaries

Let $\mathcal{X}$ be a finite set. The notation $x_{1..n}$ is used for $x_1,\dots,x_n$. We consider stochastic processes (probability measures) on $(\mathcal{X}^\infty,\mathcal{F})$, where $\mathcal{F}$ is the sigma-field generated by the cylinder sets $[x_{1..n}]$, $x_i\in\mathcal{X}, n\in\mathbb{N}$, where $[x_{1..n}]$ is the set of all infinite sequences that start with $x_{1..n}$. Since we are only interested in those measures on $(\mathcal{X}^\infty,\mathcal{F})$ which are probability measures (the measure of $\mathcal{X}^\infty$ equals 1), we call them simply *measures*. For a finite set A denote $|A|$ its cardinality. We use $\mathbf{E}_\mu$ for expectation with respect to a measure μ.

Next we introduce the criteria of the quality of prediction used in this paper. For two measures μ and ρ we are interested in how different the μ- and ρ-conditional probabilities are, given a data sample $x_{1..n}$. Introduce the *(conditional) total variation* distance

$$v(\mu,\rho,x_{1..n}) := \sup_{A\in\mathcal{F}} |\mu(A|x_{1..n})-\rho(A|x_{1..n})|.$$

Definition 1 *We say that ρ predicts μ in total variation if*

$$v(\mu,\rho,x_{1..n})\to 0 \ \mu\text{-a.s.}$$

This convergence is rather strong. In particular, it means that ρ-conditional probabilities of arbitrary far-off events converge to μ-conditional probabilities. Moreover, ρ predicts μ in total variation if Blackwell and Dubins (1962) and only if Kalai and Lehrer (1994) μ is absolutely continuous with respect to ρ:

Theorem 2 (Blackwell and Dubins, 1962; Kalai and Lehrer, 1994) *If ρ, μ are arbitrary probability measures on $(\mathcal{X}^\infty,\mathcal{F})$, then ρ predicts μ in total variation if and only if μ is absolutely continuous with respect to ρ.*

Thus, for a class $\mathcal{C}$ of measures there is a predictor ρ that predicts every $\mu\in\mathcal{C}$ in total variation if and only if every $\mu\in\mathcal{C}$ has a density with respect to ρ. Although such sets of processes are rather large, they do not include even such basic examples as the set of all Bernoulli i.i.d. processes. That is, there is no ρ that would predict in total variation every Bernoulli i.i.d. process measure δ_p, $p\in[0,1]$, where p is the probability of 0. Therefore, perhaps for many (if not most) practical applications this measure of the quality of prediction is too strong, and one is interested in weaker measures of performance.

For two measures μ and ρ introduce the *expected cumulative Kullback-Leibler divergence (KL divergence)* as

$$d_n(\mu,\rho) := \mathbf{E}_\mu \sum_{t=1}^{n} \sum_{a\in\mathcal{X}} \mu(x_t = a|x_{1..t-1}) \log \frac{\mu(x_t = a|x_{1..t-1})}{\rho(x_t = a|x_{1..t-1})}.$$

In words, we take the expected (over data) average (over time) KL divergence between μ- and ρ-conditional (on the past data) probability distributions of the next outcome.

Definition 3 *We say that ρ predicts μ in expected average KL divergence if*

$$\frac{1}{n} d_n(\mu,\rho) \to 0.$$

This measure of performance is much weaker, in the sense that it requires good predictions only one step ahead, and not on every step but only on average; also, the convergence is not with probability 1, but in expectation. With prediction quality so measured, predictors exist for relatively large classes of measures; most notably, Ryabko (1988) provides a predictor which predicts every stationary process in expected average KL divergence. A simple but useful identity that we will need (in the context of sequence prediction introduced also by Ryabko, 1988) is the following

$$d_n(\mu,\rho) = -\sum_{x_{1..n}\in\mathcal{X}^n} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})}, \tag{2}$$

where on the right-hand side we have simply the KL divergence between measures μ and ρ restricted to the first n observations.

Thus, the results of this work will be established with respect to two very different measures of prediction quality, one of which is very strong and the other rather weak. This suggests that the facts established reflect some fundamental properties of the problem of prediction, rather than those pertinent to particular measures of performance. On the other hand, it remains open to extend the results below to different measures of performance.

3. Fully Nonparametric Bayes Predictors

In this section we show that if there is a predictor that predicts every μ in some class $\mathcal{C}$, then there is a Bayesian mixture of countably many elements from $\mathcal{C}$ that predicts every $\mu \in \mathcal{C}$ too. This is established for the two notions of prediction quality that were introduced: total variation and expected average KL divergence. After the theorems we present some examples of families of measures for which predictors exist.

Theorem 4 *Let $\mathcal{C}$ be a set of probability measures on $(\mathcal{X}^\infty, \mathcal{F})$. If there is a measure ρ such that ρ predicts every $\mu \in \mathcal{C}$ in total variation, then there is a sequence $\mu_k \in \mathcal{C}$, $k \in \mathbb{N}$ such that the measure $\nu := \sum_{k\in\mathbb{N}} w_k \mu_k$ predicts every $\mu \in \mathcal{C}$ in total variation, where w_k are any positive weights that sum to 1.*

This relatively simple fact can be proven in different ways, relying on the mentioned equivalence (Blackwell and Dubins, 1962; Kalai and Lehrer, 1994) of the statements "ρ predicts μ in total variation distance" and "μ is absolutely continuous with respect to ρ." The proof presented below is not the shortest possible, but it uses ideas and techniques that are then generalized to the case

of prediction in expected average KL-divergence, which is more involved, since in all interesting cases all measures $\mu \in \mathcal{C}$ are singular with respect to any predictor that predicts all of them. Another proof of Theorem 4 can be obtained from Theorem 7 in the next section. Yet another way would be to derive it from algebraic properties of the relation of absolute continuity, given in Plesner and Rokhlin (1946).

Proof We break the (relatively easy) proof of this theorem into three steps, which will make the proof of the next theorem more understandable.

Step 1: densities. For any $\mu \in \mathcal{C}$, since ρ predicts μ in total variation, by Theorem 2, μ has a density (Radon-Nikodym derivative) f_μ with respect to ρ. Thus, for the (measurable) set T_μ of all sequences $x_1, x_2, ... \in \mathcal{X}^\infty$ on which $f_\mu(x_{1,2,...}) > 0$ (the limit $\lim_{n\to\infty} \frac{\rho(x_{1..n})}{\mu(x_{1..n})}$ exists and is finite and positive) we have $\mu(T_\mu) = 1$ and $\rho(T_\mu) > 0$. Next we will construct a sequence of measures $\mu_k \in \mathcal{C}$, $k \in \mathbb{N}$ such that the union of the sets T_{μ_k} has probability 1 with respect to every $\mu \in \mathcal{C}$, and will show that this is a sequence of measures whose existence is asserted in the theorem statement.

Step 2: a countable cover and the resulting predictor. Let $\varepsilon_k := 2^{-k}$ and let $m_1 := \sup_{\mu\in\mathcal{C}} \rho(T_\mu)$. Clearly, $m_1 > 0$. Find any $\mu_1 \in \mathcal{C}$ such that $\rho(T_{\mu_1}) \geq m_1 - \varepsilon_1$, and let $T_1 = T_{\mu_1}$. For $k > 1$ define $m_k := \sup_{\mu\in\mathcal{C}} \rho(T_\mu \backslash T_{k-1})$. If $m_k = 0$ then define $T_k := T_{k-1}$, otherwise find any μ_k such that $\rho(T_{\mu_k} \backslash T_{k-1}) \geq m_k - \varepsilon_k$, and let $T_k := T_{k-1} \cup T_{\mu_k}$. Define the predictor ν as $\nu := \sum_{k\in\mathbb{N}} w_k \mu_k$.

Step 3: ν predicts every $\mu \in \mathcal{C}$. Since the sets $T_1, T_2 \backslash T_1, \ldots, T_k \backslash T_{k-1}, \ldots$ are disjoint, we must have $\rho(T_k \backslash T_{k-1}) \to 0$, so that $m_k \to 0$ (since $m_k \leq \rho(T_k \backslash T_{k-1}) + \varepsilon_k \to 0$). Let

$$T := \cup_{k\in\mathbb{N}} T_k.$$

Fix any $\mu \in \mathcal{C}$. Suppose that $\mu(T_\mu \backslash T) > 0$. Since μ is absolutely continuous with respect to ρ, we must have $\rho(T_\mu \backslash T) > 0$. Then for every $k > 1$ we have

$$m_k = \sup_{\mu'\in\mathcal{C}} \rho(T_{\mu'} \backslash T_{k-1}) \geq \rho(T_\mu \backslash T_{k-1}) \geq \rho(T_\mu \backslash T) > 0,$$

which contradicts $m_k \to 0$. Thus, we have shown that

$$\mu(T \cap T_\mu) = 1. \tag{3}$$

Let us show that every $\mu \in \mathcal{C}$ is absolutely continuous with respect to ν. Indeed, fix any $\mu \in \mathcal{C}$ and suppose $\mu(A) > 0$ for some $A \in \mathcal{F}$. Then from (3) we have $\mu(A \cap T) > 0$, and, by absolute continuity of μ with respect to ρ, also $\rho(A \cap T) > 0$. Since $T = \cup_{k\in\mathbb{N}} T_k$, we must have $\rho(A \cap T_k) > 0$ for some $k \in \mathbb{N}$. Since on the set T_k the measure μ_k has non-zero density f_{μ_k} with respect to ρ, we must have $\mu_k(A \cap T_k) > 0$. (Indeed, $\mu_k(A \cap T_k) = \int_{A\cap T_k} f_{\mu_k} d\rho > 0$.) Hence,

$$\nu(A \cap T_k) \geq w_k \mu_k(A \cap T_k) > 0,$$

so that $\nu(A) > 0$. Thus, μ is absolutely continuous with respect to ν, and so, by Theorem 2, ν predicts μ in total variation distance. ■

Thus, examples of families $\mathcal{C}$ for which there is a ρ that predicts every $\mu \in \mathcal{C}$ in total variation, are limited to families of measures which have a density with respect to some measure ρ. On the one hand, from statistical point of view, such families are rather large: the assumption that the probabilistic law in question has a density with respect to some (nice) measure is a standard one

in statistics. It should also be mentioned that such families can easily be uncountable. On the other hand, even such basic examples as the set of all Bernoulli i.i.d. measures does not allow for a predictor that predicts every measure in total variation. Indeed, all these processes are singular with respect to one another; in particular, each of the non-overlapping sets T_p of all sequences which have limiting fraction p of 0s has probability 1 with respect to one of the measures and 0 with respect to all others; since there are uncountably many of these measures, there is no measure ρ with respect to which they all would have a density (since such a measure should have $\rho(T_p) > 0$ for all p) . As it was mentioned, predicting in total variation distance means predicting with arbitrarily growing horizon (Kalai and Lehrer, 1994), while prediction in expected average KL divergence is only concerned with the probabilities of the next observation, and only on time and data average. For the latter measure of prediction quality, consistent predictors exist not only for the class of all Bernoulli processes, but also for the class of all stationary processes (Ryabko, 1988). The next theorem establishes the result similar to Theorem 4 for expected average KL divergence.

Theorem 5 *Let $\mathcal{C}$ be a set of probability measures on $(\mathcal{X}^\infty, \mathcal{F})$. If there is a measure ρ such that ρ predicts every $\mu \in \mathcal{C}$ in expected average KL divergence, then there exist a sequence $\mu_k \in \mathcal{C}, k \in \mathbb{N}$ and a sequence $w_k > 0, k \in \mathbb{N}$, such that $\sum_{k,\in\mathbb{N}} w_k = 1$, and the measure $\nu := \sum_{k\in\mathbb{N}} w_k \mu_k$ predicts every $\mu \in \mathcal{C}$ in expected average KL divergence.*

A difference worth noting with respect to the formulation of Theorem 4 (apart from a different measure of divergence) is in that in the latter the weights w_k can be chosen arbitrarily, while in Theorem 5 this is not the case. In general, the statement "$\sum_{k\in\mathbb{N}} w_k \nu_k$ predicts μ in expected average KL divergence for some choice of w_k, $k \in \mathbb{N}$" does not imply "$\sum_{k\in\mathbb{N}} w'_k \nu_k$ predicts μ in expected average KL divergence for every summable sequence of positive $w'_k, k \in \mathbb{N}$," while the implication trivially holds true if the expected average KL divergence is replaced by the total variation. This is illustrated in the last example of this section. An interesting related question (which is beyond the scope of this paper) is how to chose the weights to optimize the behaviour of a predictor before asymptotic.

The idea of the proof of Theorem 5 is as follows. For every μ and every n we consider the sets T^n_μ of those $x_{1..n}$ on which μ is greater than ρ. These sets have to have (from some n on) a high probability with respect to μ. Then since ρ predicts μ in expected average KL divergence, the ρ-probability of these sets cannot decrease exponentially fast (that is, it has to be quite large). (The sequences $\mu(x_{1..n})/\rho(x_{1..n}), n \in \mathbb{N}$ will play the role of densities of the proof of Theorem 4, and the sets T^n_μ the role of sets T_μ on which the density is non-zero.) We then use, for each given n, the same scheme to cover the set $\mathcal{X}^n$ with countably many T^n_μ, as was used in the proof of Theorem 4 to construct a countable covering of the set $\mathcal{X}^\infty$, obtaining for each n a predictor ν_n. Then the predictor ν is obtained as $\sum_{n\in\mathbb{N}} w_n \nu_n$, where the weights decrease subexponentially. The latter fact ensures that, although the weights depend on n, they still play no role asymptotically. The technically most involved part of the proof is to show that the sets T^n_μ in asymptotic have sufficiently large weights in those countable covers that we construct for each n. This is used to demonstrate the implication "if a set has a high μ probability, then its ρ-probability does not decrease too fast, provided some regularity conditions." The proof is broken into the same steps as the (simpler) proof of Theorem 4, to make the analogy explicit and the proof more understandable.

Proof Define the weights $w_k := wk^{-2}$, where w is the normalizer $6/\pi^2$.

Step 1: densities. Define the sets

$$T_\mu^n := \left\{ x_{1..n} \in \mathcal{X}^n : \mu(x_{1..n}) \geq \frac{1}{n}\rho(x_{1..n}) \right\}. \tag{4}$$

Using Markov's inequality, we derive

$$\mu(\mathcal{X}^n \backslash T_\mu^n) = \mu\left(\frac{\rho(x_{1..n})}{\mu(x_{1..n})} > n \right) \leq \frac{1}{n} E_\mu \frac{\rho(x_{1..n})}{\mu(x_{1..n})} = \frac{1}{n}, \tag{5}$$

so that $\mu(T_\mu^n) \to 1$. (Note that if μ is singular with respect to ρ, as is typically the case, then $\frac{\rho(x_{1..n})}{\mu(x_{1..n})}$ converges to 0 μ-a.e. and one can replace $\frac{1}{n}$ in (4) by 1, while still having $\mu(T_\mu^n) \to 1$.)

Step 2n: a countable cover, time n. Fix an $n \in \mathbb{N}$. Define $m_1^n := \max_{\mu \in \mathcal{C}} \rho(T_\mu^n)$ (since $\mathcal{X}^n$ are finite all suprema are reached). Find any μ_1^n such that $\rho_1^n(T_{\mu_1^n}^n) = m_1^n$ and let $T_1^n := T_{\mu_1^n}^n$. For $k > 1$, let $m_k^n := \max_{\mu \in \mathcal{C}} \rho(T_\mu^n \backslash T_{k-1}^n)$. If $m_k^n > 0$, let μ_k^n be any $\mu \in \mathcal{C}$ such that $\rho(T_{\mu_k^n}^n \backslash T_{k-1}^n) = m_k^n$, and let $T_k^n := T_{k-1}^n \cup T_{\mu_k^n}^n$; otherwise let $T_k^n := T_{k-1}^n$. Observe that (for each n) there is only a finite number of positive m_k^n, since the set $\mathcal{X}^n$ is finite; let K_n be the largest index k such that $m_k^n > 0$. Let

$$\nu_n := \sum_{k=1}^{K_n} w_k \mu_k^n.$$

As a result of this construction, for every $n \in \mathbb{N}$ every $k \leq K_n$ and every $x_{1..n} \in T_k^n$ using (4) we obtain

$$\nu_n(x_{1..n}) \geq w_k \frac{1}{n} \rho(x_{1..n}). \tag{6}$$

Step 2: the resulting predictor. Finally, define

$$\nu := \frac{1}{2}\gamma + \frac{1}{2} \sum_{n \in \mathbb{N}} w_n \nu_n, \tag{7}$$

where γ is the i.i.d. measure with equal probabilities of all $x \in \mathcal{X}$ (that is, $\gamma(x_{1..n}) = |\mathcal{X}|^{-n}$ for every $n \in \mathbb{N}$ and every $x_{1..n} \in \mathcal{X}^n$). We will show that ν predicts every $\mu \in \mathcal{C}$, and then in the end of the proof (Step r) we will show how to replace γ by a combination of a countable set of elements of $\mathcal{C}$ (in fact, γ is just a regularizer which ensures that ν-probability of any word is never too close to 0).

Step 3: ν predicts every $\mu \in \mathcal{C}$. Fix any $\mu \in \mathcal{C}$. Introduce the parameters $\varepsilon_\mu^n \in (0,1)$, $n \in \mathbb{N}$, to be defined later, and let $j_\mu^n := 1/\varepsilon_\mu^n$. Observe that $\rho(T_k^n \backslash T_{k-1}^n) \geq \rho(T_{k+1}^n \backslash T_k^n)$, for any $k > 1$ and any $n \in \mathbb{N}$, by definition of these sets. Since the sets $T_k^n \backslash T_{k-1}^n$, $k \in \mathbb{N}$ are disjoint, we obtain $\rho(T_k^n \backslash T_{k-1}^n) \leq 1/k$. Hence, $\rho(T_\mu^n \backslash T_j^n) \leq \varepsilon_\mu^n$ for some $j \leq j_\mu^n$, since otherwise $m_j^n = \max_{\mu \in \mathcal{C}} \rho(T_\mu^n \backslash T_{j_\mu^n}^n) > \varepsilon_\mu^n$ so that $\rho(T_{j_\mu^n+1}^n \backslash T_{j_\mu^n}^n) > \varepsilon_\mu^n = 1/j_\mu^n$, which is a contradiction. Thus,

$$\rho(T_\mu^n \backslash T_{j_\mu^n}^n) \leq \varepsilon_\mu^n. \tag{8}$$

We can upper-bound $\mu(T^n_\mu \backslash T^n_{j^n_\mu})$ as follows. First, observe that

$$d_n(\mu,\rho) = -\sum_{x_{1..n}\in T^n_\mu \cap T^n_{j^n_\mu}} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})} - \sum_{x_{1..n}\in T^n_\mu \backslash T^n_{j^n_\mu}} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})} - \sum_{x_{1..n}\in \mathcal{X}^n \backslash T^n_\mu} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})} = I + II + III. \quad (9)$$

Then, from (4) we get

$$I \geq -\log n. \quad (10)$$

Observe that for every $n \in \mathbb{N}$ and every set $A \subset \mathcal{X}^n$, using Jensen's inequality we can obtain

$$-\sum_{x_{1..n}\in A} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})} = -\mu(A) \sum_{x_{1..n}\in A} \frac{1}{\mu(A)} \mu(x_{1..n}) \log \frac{\rho(x_{1..n})}{\mu(x_{1..n})} \geq -\mu(A) \log \frac{\rho(A)}{\mu(A)} \geq -\mu(A) \log \rho(A) - \frac{1}{2}. \quad (11)$$

Thus, from (11) and (8) we get

$$II \geq -\mu(T^n_\mu \backslash T^n_{j^n_\mu}) \log \rho(T^n_\mu \backslash T^n_{j^n_\mu}) - 1/2 \geq -\mu(T^n_\mu \backslash T^n_{j^n_\mu}) \log \varepsilon^n_\mu - 1/2. \quad (12)$$

Furthermore,

$$III \geq \sum_{x_{1..n}\in \mathcal{X}^n \backslash T^n_\mu} \mu(x_{1..n}) \log \mu(x_{1..n}) \geq \mu(\mathcal{X}^n \backslash T^n_\mu) \log \frac{\mu(\mathcal{X}^n \backslash T^n_\mu)}{|\mathcal{X}^n \backslash T^n_\mu|} \geq -\frac{1}{2} - \mu(\mathcal{X}^n \backslash T^n_\mu) n \log |\mathcal{X}| \geq -\frac{1}{2} - \log |\mathcal{X}|, \quad (13)$$

where in the second inequality we have used the fact that entropy is maximized when all events are equiprobable, in the third one we used $|\mathcal{X}^n \backslash T^n_\mu| \leq |\mathcal{X}|^n$, while the last inequality follows from (5). Combining (9) with the bounds (10), (12) and (13) we obtain

$$d_n(\mu,\rho) \geq -\log n - \mu(T^n_\mu \backslash T^n_{j^n_\mu}) \log \varepsilon^n_\mu - 1 - \log |\mathcal{X}|,$$

so that

$$\mu(T^n_\mu \backslash T^n_{j^n_\mu}) \leq \frac{1}{-\log \varepsilon^n_\mu} \Big(d_n(\mu,\rho) + \log n + 1 + \log |\mathcal{X}| \Big). \quad (14)$$

Since $d_n(\mu,\rho) = o(n)$, we can define the parameters ε^n_μ in such a way that $-\log \varepsilon^n_\mu = o(n)$ while at the same time the bound (14) gives $\mu(T^n_\mu \backslash T^n_{j^n_\mu}) = o(1)$. Fix such a choice of ε^n_μ. Then, using $\mu(T^n_\mu) \to 1$, we can conclude

$$\mu(\mathcal{X}^n \backslash T^n_{j^n_\mu}) \leq \mu(\mathcal{X}^n \backslash T^n_\mu) + \mu(T^n_\mu \backslash T^n_{j^n_\mu}) = o(1). \quad (15)$$

We proceed with the proof of $d_n(\mu,\nu)=o(n)$. For any $x_{1..n}\in T^n_{j^n_\mu}$ we have

$$\nu(x_{1..n})\geq\frac{1}{2}w_n\nu_n(x_{1..n})\geq\frac{1}{2}w_nw_{j^n_\mu}\frac{1}{n}\rho(x_{1..n})=\frac{w_nw}{2n}(\varepsilon^n_\mu)^2\rho(x_{1..n}),\tag{16}$$

where the first inequality follows from (7), the second from (6), and in the equality we have used $w_{j^n_\mu}=w/(j^n_\mu)^2$ and $j^n_\mu=1/\varepsilon^\mu_n$. Next we use the decomposition

$$d_n(\mu,\nu)=-\sum_{x_{1..n}\in T^n_{j^n_\mu}}\mu(x_{1..n})\log\frac{\nu(x_{1..n})}{\mu(x_{1..n})}-\sum_{x_{1..n}\in\mathcal{X}^n\setminus T^n_{j^n_\mu}}\mu(x_{1..n})\log\frac{\nu(x_{1..n})}{\mu(x_{1..n})}=I+II.\tag{17}$$

From (16) we find

$$\begin{aligned}I&\leq-\log\left(\frac{w_nw}{2n}(\varepsilon^n_\mu)^2\right)-\sum_{x_{1..n}\in T^n_{j^n_\mu}}\mu(x_{1..n})\log\frac{\rho(x_{1..n})}{\mu(x_{1..n})}\\&=(1+3\log n-2\log\varepsilon^n_\mu-2\log w)+\left(d_n(\mu,\rho)+\sum_{x_{1..n}\in\mathcal{X}^n\setminus T^n_{j^n_\mu}}\mu(x_{1..n})\log\frac{\rho(x_{1..n})}{\mu(x_{1..n})}\right)\\&\leq o(n)-\sum_{x_{1..n}\in\mathcal{X}^n\setminus T^n_{j^n_\mu}}\mu(x_{1..n})\log\mu(x_{1..n})\\&\leq o(n)+\mu(\mathcal{X}^n\setminus T^n_{j^n_\mu})n\log|\mathcal{X}|=o(n),\end{aligned}\tag{18}$$

where in the second inequality we have used $-\log\varepsilon^n_\mu=o(n)$ and $d_n(\mu,\rho)=o(n)$, in the last inequality we have again used the fact that the entropy is maximized when all events are equiprobable, while the last equality follows from (15). Moreover, from (7) we find

$$II\leq\log 2-\sum_{x_{1..n}\in\mathcal{X}^n\setminus T^n_{j^n_\mu}}\mu(x_{1..n})\log\frac{\gamma(x_{1..n})}{\mu(x_{1..n})}\leq 1+n\mu(\mathcal{X}^n\setminus T^n_{j^n_\mu})\log|\mathcal{X}|=o(n),\tag{19}$$

where in the last inequality we have used $\gamma(x_{1..n})=|\mathcal{X}|^{-n}$ and $\mu(x_{1..n})\leq 1$, and the last equality follows from (15).

From (17), (18) and (19) we conclude $\frac{1}{n}d_n(\nu,\mu)\to 0$.

Step r: the regularizer γ. It remains to show that the i.i.d. regularizer γ in the definition of ν (7), can be replaced by a convex combination of a countably many elements from $\mathcal{C}$. Indeed, for each $n\in\mathbb{N}$, denote

$$A_n:=\{x_{1..n}\in\mathcal{X}^n:\exists\mu\in\mathcal{C}\,\mu(x_{1..n})\neq 0\},$$

and let for each $x_{1..n}\in\mathcal{X}^n$ the measure $\mu_{x_{1..n}}$ be any measure from $\mathcal{C}$ such that $\mu_{x_{1..n}}(x_{1..n})\geq\frac{1}{2}\sup_{\mu\in\mathcal{C}}\mu(x_{1..n})$. Define

$$\gamma'_n(x'_{1..n}):=\frac{1}{|A_n|}\sum_{x_{1..n}\in A_n}\mu_{x_{1..n}}(x'_{1..n}),$$

for each $x'_{1..n}\in A^n$, $n\in\mathbb{N}$, and let $\gamma':=\sum_{k\in\mathbb{N}}w_k\gamma'_k$. For every $\mu\in\mathcal{C}$ we have

$$\gamma'(x_{1..n})\geq w_n|A_n|^{-1}\mu_{x_{1..n}}(x_{1..n})\geq\frac{1}{2}w_n|\mathcal{X}|^{-n}\mu(x_{1..n})$$

for every $n \in \mathbb{N}$ and every $x_{1..n} \in A_n$, which clearly suffices to establish the bound $II = o(n)$ as in (19). ■

Example: countable classes of measures. A very simple but rich example of a class $\mathcal{C}$ that satisfies the conditions of both the theorems above, is any countable family $\mathcal{C} = \{\mu_k : k \in \mathbb{N}\}$ of measures. In this case, any mixture predictor $\rho := \sum_{k\in\mathbb{N}} w_k \mu_k$ predicts all $\mu \in \mathcal{C}$ both in total variation and in expected average KL divergence. A particular instance, that has gained much attention in the literature, is the family of all computable measures. Although countable, this family of processes is rather rich. The problem of predicting all computable measures was introduced in Solomonoff (1978), where a mixture predictor was proposed.

Example: Bernoulli i.i.d. processes. Consider the class $\mathcal{C}_B = \{\mu_p : p \in [0,1]\}$ of all Bernoulli i.i.d. processes: $\mu_p(x_k = 0) = p$ independently for all $k \in \mathbb{N}$. Clearly, this family is uncountable. Moreover, each set

$$T_p := \{x \in \mathcal{X}^\infty : \text{ the limiting fraction of 0s in } x \text{ equals } p\},$$

has probability 1 with respect to μ_p and probability 0 with respect to any $\mu_{p'} : p' \neq p$. Since the sets T_p, $p \in [0,1]$ are non-overlapping, there is no measure ρ for which $\rho(T_p) > 0$ for all $p \in [0,1]$. That is, there is no measure ρ with respect to which all μ_p are absolutely continuous. Therefore, by Theorem 2, a predictor that predicts any $\mu \in \mathcal{C}_B$ in total variation does not exist, demonstrating that this notion of prediction is rather strong. However, we know (e.g., Krichevsky, 1993) that the Laplace predictor (1) predicts every Bernoulli i.i.d. process in expected average KL divergence (and not only). Hence, Theorem 4 implies that there is a countable mixture predictor for this family too. Let us find such a predictor. Let $\mu_q : q \in Q$ be the family of all Bernoulli i.i.d. measures with rational probability of 0, and let $\rho := \sum_{q\in Q} w_q \mu_q$, where w_q are arbitrary positive weights that sum to 1. Let μ_p be any Bernoulli i.i.d. process. Let $h(p,q)$ denote the divergence $p\log(p/q) + (1-p)\log(1-p/1-q)$. For each ε we can find a $q \in Q$ such that $h(p,q) < \varepsilon$. Then

$$\frac{1}{n} d_n(\mu_p, \rho) = \frac{1}{n}\mathbf{E}_{\mu_p} \log \frac{\log \mu_p(x_{1..n})}{\log \rho(x_{1..n})} \leq \frac{1}{n}\mathbf{E}_{\mu_p} \log \frac{\log \mu_p(x_{1..n})}{w_q \log \mu_q(x_{1..n})}$$
$$= -\frac{\log w_q}{n} + h(p,q) \leq \varepsilon + o(1). \quad (20)$$

Since this holds for each ε, we conclude that $\frac{1}{n} d_n(\mu_p, \rho) \to 0$ and ρ predicts every $\mu \in \mathcal{C}_B$ in expected average KL divergence.

Example: stationary processes. In Ryabko (1988) a predictor ρ_R was constructed which predicts every stationary process $\rho \in \mathcal{C}_S$ in expected average KL divergence. (This predictor is obtained as a mixture of predictors for k-order Markov sources, for all $k \in \mathbb{N}$.) Therefore, Theorem 5 implies that there is also a countable mixture predictor for this family of processes. Such a predictor can be constructed as follows (the proof in this example is based on the proof in Ryabko and Astola, 2006, Appendix 1). Observe that the family $\mathcal{C}_k$ of k-order stationary binary-valued Markov processes is parametrized by 2^k $[0,1]$-valued parameters: probability of observing 0 after observing $x_{1..k}$, for each $x_{1..k} \in \mathcal{X}^k$. For each $k \in \mathbb{N}$ let μ_q^k, $q \in Q^{2^k}$ be the (countable) family of all stationary k-order Markov processes with rational values of all the parameters. We will show that any predictor $\nu := \sum_{k\in\mathbb{N}} \sum_{q\in Q^{2^k}} w_k w_q \mu_q^k$, where w_k, $k \in \mathbb{N}$ and $w_q, q \in Q^{2^k}$, $k \in \mathbb{N}$ are any sequences of positive real weights that sum to 1, predicts every stationary $\mu \in \mathcal{C}_S$ in expected average KL divergence. For

$\mu \in \mathcal{C}_S$ and $k \in \mathbb{N}$ define the k-order conditional Shannon entropy $h_k(\mu) := \mathbf{E}_\mu \log \mu(x_{k+1}|x_{1..k})$. We have $h_{k+1}(\mu) \geq h_k(\mu)$ for every $k \in \mathbb{N}$ and $\mu \in \mathcal{C}_S$, and the limit

$$h_\infty(\mu) := \lim_{k\to\infty} h_k(\mu) \tag{21}$$

is called the limit Shannon entropy; see, for example, Gallager (1968). Fix some $\mu \in \mathcal{C}_S$. It is easy to see that for every $\varepsilon > 0$ and every $k \in \mathbb{N}$ we can find a k-order stationary Markov measure $\mu^k_{q_\varepsilon}$, $q_\varepsilon \in Q^{2^k}$ with rational values of the parameters, such that

$$\mathbf{E}_\mu \log \frac{\mu(x_{k+1}|x_{1..k})}{\mu^k_{q_\varepsilon}(x_{k+1}|x_{1..k})} < \varepsilon. \tag{22}$$

We have

$$\begin{aligned}
\frac{1}{n} d_n(\mu,\nu) &\leq -\frac{\log w_k w_{q_\varepsilon}}{n} + \frac{1}{n} d_n(\mu,\mu^k_{q_\varepsilon}) \\
&= O(k/n) + \frac{1}{n}\mathbf{E}_\mu \log \mu(x_{1..n}) - \frac{1}{n}\mathbf{E}_\mu \log \mu^k_{q_\varepsilon}(x_{1..n}) \\
&= o(1) + h_\infty(\mu) - \frac{1}{n}\mathbf{E}_\mu \sum_{k=1}^{n} \log \mu^k_{q_\varepsilon}(x_t|x_{1..t-1}) \\
&= o(1) + h_\infty(\mu) - \frac{1}{n}\mathbf{E}_\mu \sum_{t=1}^{k} \log \mu^k_{q_\varepsilon}(x_t|x_{1..t-1}) - \frac{n-k}{n}\mathbf{E}_\mu \log \mu^k_{q_\varepsilon}(x_{k+1}|x_{1..k}) \\
&\leq o(1) + h_\infty(\mu) - \frac{n-k}{n}(h_k(\mu) - \varepsilon),
\end{aligned} \tag{23}$$

where the first inequality is derived analogously to (20), the first equality follows from (2), the second equality follows from the Shannon-McMillan-Breiman theorem (e.g., Gallager, 1968), that states that $\frac{1}{n}\log\mu(x_{1..n}) \to h_\infty(\mu)$ in expectation (and a.s.) for every $\mu \in \mathcal{C}_S$, and (2); in the third equality we have used the fact that $\mu^k_{q_\varepsilon}$ is k-order Markov and μ is stationary, whereas the last inequality follows from (22). Finally, since the choice of k and ε was arbitrary, from (23) and (21) we obtain $\lim_{n\to\infty} \frac{1}{n} d_n(\mu,\nu) = 0$.

Example: weights may matter. Finally, we provide an example that illustrates the difference between the formulations of Theorems 4 and 5: in the latter the weights are not arbitrary. We will construct a sequence of measures $\nu_k, k \in \mathbb{N}$, a measure μ, and two sequences of positive weights w_k and w'_k with $\sum_{k\in\mathbb{N}} w_k = \sum_{k\in\mathbb{N}} w'_k = 1$, for which $\nu := \sum_{k\in\mathbb{N}} w_k \nu_k$ predicts μ in expected average KL divergence, but $\nu' := \sum_{k\in\mathbb{N}} w'_k \nu_k$ does not. Let ν_k be a deterministic measure that first outputs k 0s and then only 1s, $k \in \mathbb{N}$. Let $w_k = w/k^2$ with $w = 6/\pi^2$ and $w'_k = 2^{-k}$. Finally, let μ be a deterministic measure that outputs only 0s. We have $d_n(\mu,\nu) = -\log(\sum_{k\geq n} w_k) \leq -\log(wn^{-2}) = o(n)$, but $d_n(\mu,\nu') = -\log(\sum_{k\geq n} w'_k) = -\log(2^{-n+1}) = n-1 \neq o(n)$, proving the claim.

4. Characterizing Predictable Classes

Knowing that a mixture of a countable subset gives a predictor if there is one, a notion that naturally comes to mind, when trying to characterize families of processes for which a predictor exists, is separability. Can we say that there is a predictor for a class $\mathcal{C}$ of measures if and only if $\mathcal{C}$ is separable? Of course, to talk about separability we need a suitable topology on the space of all

measures, or at least on $\mathcal{C}$. If the formulated questions were to have a positive answer, we would need a different topology for each of the notions of predictive quality that we consider. Sometimes these measures of predictive quality indeed define a nice enough structure of a probability space, but sometimes they do not. The question whether there exists a topology on $\mathcal{C}$, separability with respect to which is equivalent to the existence of a predictor, is already more vague and less appealing. Nonetheless, in the case of total variation distance we obviously have a candidate topology: that of total variation distance, and indeed separability with respect to this topology is equivalent to the existence of a predictor, as the next theorem shows. This theorem also implies Theorem 4, thereby providing an alternative proof for the latter. In the case of expected average KL divergence the situation is different. While one can introduce a topology based on it, separability with respect to this topology turns out to be a sufficient but not a necessary condition for the existence of a predictor, as is shown in Theorem 9.

4.1 Separability

Definition 6 (unconditional total variation distance) *Introduce the (unconditional) total variation distance*

$$v(\mu,\rho) := \sup_{A\in\mathcal{F}} |\mu(A)-\rho(A)|.$$

Theorem 7 *Let $\mathcal{C}$ be a set of probability measures on $(\mathcal{X}^\infty,\mathcal{F})$. There is a measure ρ such that ρ predicts every $\mu\in\mathcal{C}$ in total variation if and only if $\mathcal{C}$ is separable with respect to the topology of total variation distance. In this case, any measure ν of the form $\nu=\sum_{k=1}^\infty w_k\mu_k$, where $\{\mu_k : k\in\mathbb{N}\}$ is any dense countable subset of $\mathcal{C}$ and w_k are any positive weights that sum to 1, predicts every $\mu\in\mathcal{C}$ in total variation.*

Proof *Sufficiency and the mixture predictor.* Let $\mathcal{C}$ be separable in total variation distance, and let $\mathcal{D}=\{\nu_k : k\in\mathbb{N}\}$ be its dense countable subset. We have to show that $\nu:=\sum_{k\in\mathbb{N}} w_k\nu_k$, where w_k are any positive real weights that sum to 1, predicts every $\mu\in\mathcal{C}$ in total variation. To do this, it is enough to show that $\mu(A)>0$ implies $\nu(A)>0$ for every $A\in\mathcal{F}$ and every $\mu\in\mathcal{C}$. Indeed, let A be such that $\mu(A)=\varepsilon>0$. Since $\mathcal{D}$ is dense in $\mathcal{C}$, there is a $k\in\mathbb{N}$ such that $v(\mu,\nu_k)<\varepsilon/2$. Hence $\nu_k(A)\geq\mu(A)-v(\mu,\nu_k)\geq\varepsilon/2$ and $\nu(A)\geq w_k\nu_k(A)\geq w_k\varepsilon/2>0$.

Necessity. For any $\mu\in\mathcal{C}$, since ρ predicts μ in total variation, μ has a density (Radon-Nikodym derivative) f_μ with respect to ρ. We can define L_1 distance with respect to ρ as $L_1^\rho(\mu,\nu)=\int_{\mathcal{X}^\infty}|f_\mu-f_\nu|d\rho$. The set of all measures that have a density with respect to ρ, is separable with respect to this distance (for example, a dense countable subset can be constructed based on measures whose densities are step-functions, that take only rational values, see, e.g., Kolmogorov and Fomin, 1975); therefore, its subset $\mathcal{C}$ is also separable. Let $\mathcal{D}$ be any dense countable subset of $\mathcal{C}$. Thus, for every $\mu\in\mathcal{C}$ and every ε there is a $\mu'\in\mathcal{D}$ such that $L_1^\rho(\mu,\mu')<\varepsilon$. For every measurable set A we have

$$|\mu(A)-\mu'(A)|=\left|\int_A f_\mu d\rho-\int_A f_{\mu'}d\rho\right|\leq\int_A|f_\mu-f_{\mu'}|d\rho\leq\int_{\mathcal{X}^\infty}|f_\mu-f_{\mu'}|d\rho<\varepsilon.$$

Therefore, $v(\mu,\mu')=\sup_{A\in\mathcal{F}}|\mu(A)-\mu'(A)|<\varepsilon$, and the set $\mathcal{C}$ is separable in total variation distance. ■

Definition 8 (asymptotic KL "distance" D) *Define asymptotic expected average KL divergence between measures μ and ρ as*

$$D(\mu,\rho)=\limsup_{n\to\infty}\frac{1}{n}d_n(\mu,\rho). \tag{24}$$

Theorem 9 *For any set $\mathcal{C}$ of probability measures on $(\mathcal{X}^\infty,\mathcal{F})$, separability with respect to the asymptotic expected average KL divergence D is a sufficient but not a necessary condition for the existence of a predictor:*

(i) If there exists a countable set $\mathcal{D}:=\{\nu_k : k\in\mathbb{N}\}\subset\mathcal{C}$, such that for every $\mu\in\mathcal{C}$ and every $\varepsilon>0$ there is a measure $\mu'\in\mathcal{D}$, such that $D(\mu,\mu')<\varepsilon$, then every measure ν of the form $\nu=\sum_{k=1}^\infty w_k\mu_k$, where w_k are any positive weights that sum to 1, predicts every $\mu\in\mathcal{C}$ in expected average KL divergence.

(ii) There is an uncountable set $\mathcal{C}$ of measures, and a measure ν, such that ν predicts every $\mu\in\mathcal{C}$ in expected average KL divergence, but $\mu_1\neq\mu_2$ implies $D(\mu_1,\mu_2)=\infty$ for every $\mu_1,\mu_2\in\mathcal{C}$; in particular, $\mathcal{C}$ is not separable with respect to D.

Proof *(i)* Fix $\mu\in\mathcal{C}$. For every $\varepsilon>0$ pick $k\in\mathbb{N}$ such that $D(\mu,\nu_k)<\varepsilon$. We have

$$d_n(\mu,\nu)=\mathbf{E}_\mu\log\frac{\mu(x_{1..n})}{\nu(x_{1..n})}\le\mathbf{E}_\mu\log\frac{\mu(x_{1..n})}{w_k\nu_k(x_{1..n})}=-\log w_k+d_n(\mu,\nu_k)\le n\varepsilon+o(n).$$

Since this holds for every ε, we conclude $\frac{1}{n}d_n(\mu,\nu)\to 0$.

(ii) Let $\mathcal{C}$ be the set of all deterministic sequences (measures concentrated on just one sequence) such that the number of 0s in the first n symbols is less than $\sqrt{n}$. Clearly, this set is uncountable. It is easy to check that $\mu_1\neq\mu_2$ implies $D(\mu_1,\mu_2)=\infty$ for every $\mu_1,\mu_2\in\mathcal{C}$, but the predictor ν, given by $\nu(x_n=0):=1/n$ independently for different n, predicts every $\mu\in\mathcal{C}$ in expected average KL divergence. ■

Examples. Basically, the examples of the preceding section carry over here. Indeed, the example of countable families is trivially also an example of separable (with respect to either of the considered topologies) family. For Bernoulli i.i.d. and k-order Markov processes, the (countable) sets of processes that have rational values of the parameters, considered in the previous section, are dense both in the topology of the parametrization and with respect to the asymptotic average divergence D. It is also easy to check from the arguments presented in the corresponding example of Section 3, that the family of all k-order stationary Markov processes with rational values of the parameters, where we take all $k\in\mathbb{N}$, is dense with respect to D in the set $\mathcal{C}_S$ of all stationary processes, so that $\mathcal{C}_S$ is separable with respect to D. Thus, the sufficient but not necessary condition of separability is satisfied in this case. On the other hand, neither of these latter families is separable with respect to the topology of total variation distance.

4.2 Conditions Based on the Local Behaviour of Measures

Next we provide some sufficient conditions for the existence of a predictor based on local characteristics of the class of measures, that is, measures truncated to the first n observations. First of all, it must be noted that necessary and sufficient conditions cannot be obtained this way. The basic

example is that of a family $\mathcal{C}_0$ of all deterministic sequences that are 0 from some time on. This is a countable class of measures which is very easy to predict. Yet, the class of measures on $\mathcal{X}^n$, obtained by truncating all measures in $\mathcal{C}_0$ to the first n observations, coincides with what would be obtained by truncating all deterministic measures to the first n observations, the latter class being obviously not predictable at all (see also examples below). Nevertheless, considering this kind of local behaviour of measures, one can obtain not only sufficient conditions for the existence of a predictor, but also rates of convergence of the prediction error. It also gives some ideas of how to construct predictors, for the cases when the sufficient conditions obtained are met.

For a class $\mathcal{C}$ of stochastic processes and a sequence $x_{1..n} \in \mathcal{X}^n$ introduce the coefficients

$$c_{x_{1..n}}(\mathcal{C}) := \sup_{\mu\in\mathcal{C}} \mu(x_{1..n}).$$

Define also the normalizer

$$c_n(\mathcal{C}) := \sum_{x_{1..n}\in\mathcal{X}^n} c_{x_{1..n}}(\mathcal{C}).$$

Definition 10 (NML estimate) *The normalized maximum likelihood estimator λ is defined (e.g., Krichevsky, 1993) as*

$$\lambda_{\mathcal{C}}(x_{1..n}) := \frac{1}{c_n(\mathcal{C})} c_{x_{1..n}}(\mathcal{C}),$$

for each $x_{1..n} \in \mathcal{X}^n$.

The family $\lambda_{\mathcal{C}}(x_{1..n})$ (indexed by n) in general does not immediately define a stochastic process over $\mathcal{X}^\infty$ ($\lambda_{\mathcal{C}}$ are not consistent for different n); thus, in particular, using average KL divergence for measuring prediction quality would not make sense, since

$$d_n(\mu(\cdot|x_{1..n-1}), \lambda_{\mathcal{C}}(\cdot|x_{1..n-1}))$$

can be negative, as the following example shows.

Example: negative d_n for NML estimates. Let the processes $\mu_i, i \in \{1,\dots,4\}$ be defined on the steps $n = 1,2$ as follows. $\mu_1(00) = \mu_2(01) = \mu_4(11) = 1$, while $\mu_3(01) = \mu_3(00) = 1/2$. We have $\lambda_{\mathcal{C}}(1) = \lambda_{\mathcal{C}}(0) = 1/2$, while $\lambda_{\mathcal{C}}(00) = \lambda_{\mathcal{C}}(01) = \lambda_{\mathcal{C}}(11) = 1/3$. If we define $\lambda_{\mathcal{C}}(x|y) = \lambda_{\mathcal{C}}(yx)/\lambda_{\mathcal{C}}(y)$, we obtain $\lambda_{\mathcal{C}}(1|0) = \lambda_{\mathcal{C}}(0|0) = 2/3$. Then $d_2(\mu_3(\cdot|0), \lambda_{\mathcal{C}}(\cdot|0)) = \log 3/4 < 0$.

Yet, by taking an appropriate mixture, it is still possible to construct a predictor (a stochastic process) based on λ, that predicts all the measures in the class.

Definition 11 (predictor ρ_c) *Let $w := 6/\pi^2$ and let $w_k := \frac{w}{k^2}$. Define a measure μ_k as follows. On the first k steps it is defined as $\lambda_{\mathcal{C}}$, and for $n > k$ it outputs only zeros with probability 1; so, $\mu_k(x_{1..k}) = \lambda_{\mathcal{C}}(x_{1..k})$ and $\mu_k(x_n = 0) = 1$ for $n > k$. Define the measure ρ_c as*

$$\rho_c = \sum_{k=1}^{\infty} w_k \mu_k.$$

Thus, we have taken the normalized maximum likelihood estimates λ_n for each n and continued them arbitrarily (actually, by a deterministic sequence) to obtain a sequence of measures on $(\mathcal{X}^\infty, \mathcal{F})$ that can be summed.

Theorem 12 *For any set $\mathcal{C}$ of probability measures on $(\mathcal{X}^\infty, \mathcal{F})$, the predictor ρ_c defined above satisfies*

$$\frac{1}{n} d_n(\mu, \rho_c) \le \frac{\log c_n(\mathcal{C})}{n} + O\left(\frac{\log n}{n}\right); \tag{25}$$

in particular, if

$$\log c_n(\mathcal{C}) = o(n), \tag{26}$$

then ρ_c predicts every $\mu \in \mathcal{C}$ in expected average KL divergence.

Proof Indeed,

$$\begin{aligned}\frac{1}{n} d_n(\mu, \rho_c) = \frac{1}{n} \mathbf{E} \log \frac{\mu(x_{1..n})}{\rho_c(x_{1..n})} &\le \frac{1}{n} \mathbf{E} \log \frac{\mu(x_{1..n})}{w_n \mu_n(x_{1..n})} \\ &\le \frac{1}{n} \log \frac{c_n(\mathcal{C})}{w_n} = \frac{1}{n}(\log c_n(\mathcal{C}) + 2\log n + \log w). \end{aligned} \tag{27}$$

■

Example: i.i.d., finite-memory. To illustrate the applicability of the theorem we first consider the class of i.i.d. processes $\mathcal{C}_B$ over the binary alphabet $\mathcal{X} = \{0,1\}$. It is easy to see that, for each $x_1, \dots, x_n$,

$$\sup_{\mu \in \mathcal{C}_B} \mu(x_{1..n}) = (k/n)^k (1 - k/n)^{n-k},$$

where $k = \#\{i \le n : x_i = 0\}$ is the number of 0s in $x_1, \dots, x_n$. For the constants $c_n(\mathcal{C})$ we can derive

$$\begin{aligned} c_n(C) = \sum_{x_{1..n} \in \mathcal{X}^n} \sup_{\mu \in \mathcal{C}_B} \mu(x_{1..n}) &= \sum_{x_{1..n} \in \mathcal{X}^n} (k/n)^k (1 - k/n)^{n-k} \\ &= \sum_{k=0}^{n} \binom{n}{k} (k/n)^k (1 - k/n)^{n-k} \le \sum_{k=0}^{n} \sum_{t=0}^{n} \binom{n}{k} (k/n)^t (1 - k/n)^{n-t} = n + 1, \end{aligned}$$

so that $c_n(C) \le n + 1$.

In general, for the class $\mathcal{C}_k$ of *processes with memory k* over a finite space $\mathcal{X}$ we can get polynomial $c_n(\mathcal{C})$ (see, for example, Krichevsky, 1993, and also Ryabko and Hutter, 2007). Thus, with respect to finite-memory processes, the conditions of Theorem 12 leave ample space for the growth of $c_n(\mathcal{C})$, since (26) allows subexponential growth of $c_n(\mathcal{C})$. Moreover, these conditions are tight, as the following example shows.

Example: exponential coefficients are not sufficient. Observe that the condition (26) cannot be relaxed further, in the sense that exponential coefficients c_n are not sufficient for prediction. Indeed, for the class of all deterministic processes (that is, each process from the class produces some fixed sequence of observations with probability 1) we have $c_n = 2^n$, while obviously for this class a predictor does not exist.

Example: stationary processes. For the set of all stationary processes we can obtain $c_n(C) \ge 2^n/n$ (as is easy to see by considering periodic n-order Markov processes, for each $n \in \mathbb{N}$), so that the conditions of Theorem 12 are not satisfied. This cannot be fixed, since uniform rates of convergence cannot be obtained for this family of processes, as was shown in Ryabko (1988).

4.2.1 OPTIMAL RATES OF CONVERGENCE

A natural question that arises with respect to the bound (25) is whether it can be matched by a lower bound. This question is closely related to the optimality of the normalized maximum likelihood estimates used in the construction of the predictor. In general, since NML estimates are not optimal, neither are the rates of convergence in (25). To obtain (close to) optimal rates one has to consider a different measure of capacity.

To do so, we make the following connection to a problem in information theory. Let $\mathcal{P}(\mathcal{X}^\infty)$ be the set of all stochastic processes (probability measures) on the space $(\mathcal{X}^\infty, \mathcal{F})$, and let $\mathcal{P}(\mathcal{X})$ be the set of probability distributions over a (finite) set $\mathcal{X}$. For a class $\mathcal{C}$ of measures we are interested in a predictor that has a small (or minimal) worst-case (with respect to the class $\mathcal{C}$) probability of error. Thus, we are interested in the quantity

$$\inf_{\rho\in\mathcal{P}(\mathcal{X}^\infty)} \sup_{\mu\in\mathcal{C}} D(\mu,\rho), \tag{28}$$

where the infimum is taken over all stochastic processes ρ, and D is the asymptotic expected average KL divergence (24). (In particular, we are interested in the conditions under which the quantity (28) equals zero.) This problem has been studied for the case when the probability measures are over a finite set $\mathcal{X}$, and D is replaced simply by the KL divergence d between the measures. Thus, the problem was to find the probability measure ρ (if it exists) on which the following minimax is attained

$$R(A) := \inf_{\rho\in\mathcal{P}(\mathcal{X})} \sup_{\mu\in A} d(\mu,\rho), \tag{29}$$

where $A \subset \mathcal{P}(\mathcal{X})$. This problem is closely related to the problem of finding the best code for the class of sources A, which was its original motivation. The normalized maximum likelihood distribution considered above does not in general lead to the optimum solution for this problem. The optimum solution is obtained through the result that relates the minimax (29) to the so-called channel capacity.

Definition 13 (Channel capacity) *For a set A of measures on a finite set $\mathcal{X}$ the* channel capacity *of A is defined as*

$$C(A) := \sup_{P\in\mathcal{P}_0(A)} \sum_{\mu\in S(P)} P(\mu) d(\mu,\rho_P),$$

where $\mathcal{P}_0(A)$ is the set of all probability distributions on A that have a finite support, $S(P)$ is the (finite) support of a distribution $P\in\mathcal{P}_0(A)$, and $\rho_P = \sum_{\mu\in S(P)} P(\mu)\mu$.

It is shown in Ryabko (1979) and Gallager (1976 (revised 1979) that $C(A) = R(A)$, thus reducing the problem of finding a minimax to an optimization problem. For probability measures over infinite spaces this result ($R(A) = C(A)$) was generalized by Haussler (1997), but the divergence between probability distributions is measured by KL divergence (and not asymptotic average KL divergence), which gives infinite $R(A)$, for example, already for the class of i.i.d. processes.

However, truncating measures in a class $\mathcal{C}$ to the first n observations, we can use the results about channel capacity to analyze the predictive properties of the class. Moreover, the rates of convergence that can be obtained along these lines are close to optimal. In order to pass from measures minimizing the divergence for each individual n to a process that minimizes the divergence for all n we use the same idea as when constructing the process ρ_c.

Theorem 14 *Let $\mathcal{C}$ be a set of measures on $(\mathcal{X}^\infty, \mathcal{F})$, and let $\mathcal{C}^n$ be the class of measures from $\mathcal{C}$ restricted to $\mathcal{X}^n$. There exists a measure ρ_C such that*

$$\frac{1}{n} d_n(\mu, \rho_C) \le \frac{C(\mathcal{C}^n)}{n} + O\left(\frac{\log n}{n}\right);$$

in particular, if $C(\mathcal{C}^n)/n \to 0$, then ρ_C predicts every $\mu \in \mathcal{C}$ in expected average KL divergence. Moreover, for any measure ρ_C and every $\varepsilon > 0$ there exists $\mu \in \mathcal{C}$ such that

$$\frac{1}{n} d_n(\mu, \rho_C) \ge \frac{C(\mathcal{C}^n)}{n} - \varepsilon.$$

Proof As shown in Gallager (1976 (revised 1979), for each n there exists a sequence ν_k^n, $k \in \mathbb{N}$ of measures on $\mathcal{X}^n$ such that

$$\lim_{k\to\infty} \sup_{\mu\in\mathcal{C}^n} d_n(\mu, \nu_k^n) \to C(\mathcal{C}^n).$$

For each $n \in \mathbb{N}$ find an index k_n such that

$$|\sup_{\mu\in\mathcal{C}^n} d_n(\mu, \nu_{k_n}^n) - C(\mathcal{C}^n)| \le 1.$$

Define the measure ρ_n as follows. On the first n symbols it coincides with $\nu_{k_n}^n$ and $\rho_n(x_m = 0) = 1$ for $m > n$. Finally, set $\rho_C = \sum_{n=1}^\infty w_n \rho_n$, where $w_k = \frac{w}{n^2}, w = 6/\pi^2$. We have to show that $\lim_{n\to\infty} \frac{1}{n} d_n(\mu, \rho_C) = 0$ for every $\mu \in \mathcal{C}$. Indeed, similarly to (27), we have

$$\begin{aligned}\frac{1}{n} d_n(\mu, \rho_C) &= \frac{1}{n} \mathbf{E}_\mu \log \frac{\mu(x_{1..n})}{\rho_C(x_{1..n})} \\ &\le \frac{\log w_k^{-1}}{n} + \frac{1}{n} \mathbf{E}_\mu \log \frac{\mu(x_{1..n})}{\rho_n(x_{1..n})} \le \frac{\log w + 2\log n}{n} + \frac{1}{n} d_n(\mu, \rho_n) \\ &\le o(1) + \frac{C(\mathcal{C}^n)}{n}.\end{aligned}$$

The second statement follows from the fact (Ryabko, 1979; Gallager, 1976 (revised 1979) that $C(\mathcal{C}^n) = R(\mathcal{C}^n)$ (cf. (29)). ■

Thus, if the channel capacity $C(\mathcal{C}^n)$ grows sublinearly, a predictor can be constructed for the class of processes $\mathcal{C}$. In this case the problem of constructing the predictor is reduced to finding the channel capacities for different n and finding the corresponding measures on which they are attained or approached.

Examples. For the class of all Bernoulli i.i.d. processes, the channel capacity $C(\mathcal{C}_B^n)$ is known to be $O(\log n)$ (Krichevsky, 1993). For the family of all stationary processes it is $O(n)$, so that the conditions of Theorem 14 are satisfied for the former but not for the latter.

We also remark that the requirement of a sublinear channel capacity cannot be relaxed, in the sense that a linear channel capacity is not sufficient for prediction, since it is the maximal possible capacity for a set of measures on $\mathcal{X}^n$, achieved, for example, on the set of all measures, or on the set of all deterministic sequences.

5. Discussion

The first possible extension of the results of the paper that comes to mind is to find out whether the same holds for other measures of performance, such as prediction in KL divergence without time-averaging, or with probability 1 rather then in expectation, or with respect to other measures of prediction error, such as absolute distance. (See Ryabko and Hutter, 2007 for a discussion of different measures of performance and relations between them.) Maybe the same results can be obtained in more general formulations, for example, using f-divergences of Csiszar (1967).

More generally, the questions we addressed in this work are a part of a larger problem: given an arbitrary class $\mathcal{C}$ of stochastic processes, find the best predictor for it. We have considered two subproblems: first, in which form to look for a predictor if one exists. Here we have shown that if any predictor works then a Bayesian one works too. The second one is to characterize families of processes for which a predictor exists. Here we have analyzed what the notion of separability furnishes in this respect, as well as identified some simple sufficient conditions based on the local behaviour of measures in the class. Another approach would be to identify the conditions which two measures μ and ρ have to satisfy in order for ρ to predict μ. For prediction in total variation such conditions have been identified (Blackwell and Dubins, 1962; Kalai and Lehrer, 1994) and, in particular, in the context of the present work, they turn out to be very useful. Kalai and Lehrer (1994) also provide some characterization for the case of a weaker notion of prediction: difference between conditional probabilities of the next (several) outcomes (weak merging of opinions). In Ryabko and Hutter (2008b) some sufficient conditions are found for the case of prediction in expected average KL divergence, and prediction in average KL divergence with probability 1. Of course, another very natural approach to the general problem posed above is to try and find predictors (in the form of algorithms) for some particular classes of processes which are of practical interest. Towards this end, we have found a rather simple form that some solution to this question has if a solution exists: a Bayesian predictor whose prior is concentrated on a countable set. We have also identified some sufficient conditions under which a predictor can actually be constructed (e.g., using NML estimates). However, the larger question of how to construct an optimal predictor for an arbitrary given family of processes, remains open.

Taking an even more general perspective, one can consider the problem of finding the best response to the actions of a (stochastic) environment, which itself responds to the actions of a learner. Allowing into consideration environments that change their behaviour in response to the action of the learner, clearly makes the problem much more difficult, but it also dramatically extends the range of applications. For this general problem one can pose the same questions: given a set $\mathcal{C}$ of environments, how can we construct a learner that is (asymptotically) optimal if any environment from $\mathcal{C}$ is chosen to generate the data? One can consider Bayesian learners for this formulation too (Hutter, 2005); it would be interesting to find out whether one can show that, when there is a learner which is optimal in every environment from $\mathcal{C}$, then there is a Bayesian learner with a countably supported prior that has this property too.

Acknowledgments

This work has been partially supported by French National Research Agency (ANR), project EXPLO-RA ANR-08-COSI-004. Some preliminary results were reported at UAI'09 (Ryabko, 2009) and ALT'08 (Ryabko, 2008).

References

D. Blackwell and L. Dubins. Merging of opinions with increasing information. *Annals of Mathematical Statistics*, 33:882–887, 1962.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006. ISBN 0521841089.

I. Csiszar. Information-type measures of difference of probability distributions and indirect observations. *Studia Sci. Math. Hungar*, 2:299–318, 1967.

P. Diaconis and D. Freedman. On the consistency of Bayes estimates. *Annals of Statistics*, 14(1): 1–26, 1986.

R. G. Gallager. *Information Theory and Reliable Communication*. John Wiley & Sons, New York, NY, USA, 1968.

R. G. Gallager. Source coding with side information and universal coding. Technical Report LIDS-P-937, M.I.T., 1976 (revised 1979).

D. Haussler. A general minimax result for relative entropy. *IEEE Trans. on Information Theory*, 43 (4):1276–1280, 1997.

M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin, 2005.

M. Hutter. On universal prediction and Bayesian confirmation. *Theoretical Computer Science*, 348 (1):33–48, 2007.

M. Jackson, E. Kalai, and R. Smorodinsky. Bayesian representation of stochastic processes under learning: de Finetti revisited. *Econometrica*, 67(4):875–794, 1999.

E. Kalai and E. Lehrer. Weak and strong merging of opinions. *Journal of Mathematical Economics*, 23:73–86, 1994.

A. N. Kolmogorov and S. V. Fomin. *Elements of the Theory of Functions and Functional Analysis*. Dover, 1975.

R. Krichevsky. *Universal Compression and Retrival*. Kluwer Academic Publishers, 1993.

A.I. Plesner and V.A. Rokhlin. Spectral theory of linear operators, II. *Uspekhi Matematicheskikh Nauk*, 1:71–191, 1946.

B. Ryabko. Coding of a source with unknown but ordered probabilities. *Problems of Information Transmission*, 15(2):134–138, 1979.

B. Ryabko. Prediction of random sequences and universal coding. *Problems of Information Transmission*, 24:87–96, 1988.

B. Ryabko and J. Astola. Universal codes as a basis for time series testing. *Statistical Methodology*, 3:375–397, 2006.

D. Ryabko. Some sufficient conditions on an arbitrary class of stochastic processes for the existence of a predictor. In *Proc. 19th International Conf. on Algorithmic Learning Theory (ALT'08)*, LNAI 5254, pages 169–182, Budapest, Hungary, 2008. Springer, Berlin.

D. Ryabko. Characterizing predictable classes of processes. In A. Ng J. Bilmes, editor, *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*, Montreal, Canada, 2009.

D. Ryabko and M. Hutter. On sequence prediction for arbitrary measures. In *Proc. 2007 IEEE International Symposium on Information Theory*, pages 2346–2350, Nice, France, 2007. IEEE. ISBN 1-4244-1429-6.

D. Ryabko and M. Hutter. On the possibility of learning in reactive environments with arbitrary dependence. *Theoretical Computer Science*, 405(3):274–284, 2008a.

D. Ryabko and M. Hutter. Predicting non-stationary processes. *Applied Mathematics Letters*, 21 (5):477–482, 2008b.

R. J. Solomonoff. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Information Theory*, IT-24:422–432, 1978.

Journal of Machine Learning Research 11 (2010) 603-624 Submitted 11/07; Revised 5/09; Published 2/10

A Rotation Test to Verify Latent Structure

Patrick O. Perry PATPERRY@STANFORD.EDU
Art B. Owen OWEN@STAT.STANFORD.EDU
Department of Statistics
Stanford Unversity
Stanford CA, 94305, USA

Editor: Aapo Hyvärinen

Abstract

In multivariate regression models we have the opportunity to look for hidden structure unrelated to the observed predictors. However, when one fits a model involving such latent variables it is important to be able to tell if the structure is real, or just an artifact of correlation in the regression errors. We develop a new statistical test based on random rotations for verifying the existence of latent variables. The rotations are carefully constructed to rotate orthogonally to the column space of the regression model. We find that only non-Gaussian latent variables are detectable, a finding that parallels a well known phenomenon in independent components analysis. We base our test on a measure of non-Gaussianity in the histogram of the principal eigenvector components instead of on the eigenvalue. The method finds and verifies some latent dichotomies in the microarray data from the AGEMAP consortium.

Keywords: independent components analysis, Kronecker covariance, latent variables, projection pursuit, transposable data

1. Introduction

The problem we consider here is one of verifying statistically that an apparent latent variable is real. The context is a microarray study, although the ideas are applicable for other high throughput biological settings, and more generally for problems where a large number of mutually correlated variables has been observed.

To fix ideas, suppose we have a matrix $Y \in \mathbb{R}^{n\times N}$ of gene expression data. Each of N genes has been measured on n microarrays. We can often assume that the n arrays come from statistically independent trials, but the N genes on any single array have a rich and unknown correlation structure. There is also an $n \times p$ matrix X of predictor variables to relate to the genes ($p < n$). We are interested in which of the predictors significantly affect the response.

The motivating context is the AGEMAP project of Zahn et al. (2007) for which $n = 40$, $N = 8932$, and $p = 3$, a project whose primary goal is to find which genes are statistically correlated with age. The covariate matrix $X \in \mathbb{R}^{n\times 3}$ has columns for intercept, age, and sex. In matrix form, we model $Y = XB + E$ where B is a p-by-N matrix of regression coefficients and E is an n-by-N matrix of Gaussian or approximately Gaussian errors.

The residuals from the linear model showed a sharp dichotomy, splitting the n AGEMAP subjects into two groups. The split could not be explained by the measured variables and it strongly suggests the presence of a binary latent variable. Binary and other latent variables can arise when

the microarray data are generated at different times, by different technicians, or at different sites. Sometimes a suspected latent variable can be confirmed by looking more closely at the data or lab notes. At that point we might identify the cause and use it as an ordinary regression variable. In other cases we may not be able to pinpoint the cause, but we still want statistical confirmation that it is real. When we are confident that the variable is real then it makes sense to use a model that includes one or more latent variables.

A natural way to test for a latent variable is to compute a singular value decomposition of the residual matrix $\widehat{E} = Y - X\widehat{B}$ and decide that a latent variable is present when the largest singular value of $\widehat{E}$ is sufficiently large. Equivalently, such a test is based on the eigenvalues of the covariance matrix of the rows of $\widehat{E}$. As we show below, a test based on those eigenvalues cannot work when the rows of E are correlated Gaussian random vectors. Adding a Gaussian latent variable simply changes the covariance structure and hence is not detectable. The situation is similar to that faced in independent components analysis (Hyvarinen et al., 2001) which becomes degenerate for Gaussian components.

While the eigenvalues offer no possibility to confirm the presence of a latent variable in correlated Gaussian noise, the eigenvectors of the covariance matrix do. Our tests are based on eigenvectors, rejecting the null hypothesis when the components of an eigenvector differ significantly from what we would get with Gaussian errors. Such measures have been derived for exploratory projection pursuit by Friedman (1987), although it is important to note that Friedman's measures by themselves do not measure of statistical significance. We cannot apply standard tests of Gaussianity such as the Anderson-Darling test, because such tests require an IID sample and the components of the eigenvector are not IID. Instead we show how to use a test based on random rotations of the data. When the noise is independent across observations and comes from a multivariate Gaussian distribution, then under the null hypothesis of no latent variable, random rotations don't change the distribution of our test statistic. Importantly, our test is still valid when there are arbitrary correlations between the columns of E.

Rotation tests have been used before, by Langsrud (2005). Our main contribution is to extend rotation tests to the context of regression for the explicit purpose of detecting latent variables. We show how to apply rotations orthogonally to a given linear model and we combine rotation tests with measures of non-Gaussianity of eigenvectors.

Our principal focus is on testing for the presence versus the absence of one latent variable. The regression model is usually used when we expect no latent variables. The presence of even one latent variable would make it reasonable to switch to a factor model. We also consider sequential tests for the correct number of latent variables when there is at least one of them.

The outline of the paper is as follows: Section 2 introduces the AGEMAP data as a motivating example and introduces the regression model mixing measured and latent predictors. Section 3 develops rotation tests for the existence of latent structure in the residual matrix from a regression. The tests surveyed in Langsrud (2005) need to be modified in order to rotate orthogonally to the regression model. We also show that Gaussian latent vectors cannot be detected, and then present some test statistics for non-Gaussian latent vectors. Section 4 presents numerical simulations on examples where we know the structure. The test is able to identify large latent variables and we find that it gives reliable p-values when no latent variables are present. Section 5 discusses fitting and validating latent variables for the AGEMAP data. Section 6 presents our conclusions.

2. Background

Here we describe the AGEMAP data and introduce regression models that include both measured and latent variables.

2.1 The AGEMAP Data

The motivating application arises from the AGEMAP (Zahn et al., 2007) study of aging in mice. AGEMAP is a large microarray study conducted at the National Institute on Aging and analyzed in collaboration with the Kim lab in Stanford's department of developmental biology. The primary focus of the AGEMAP analysis was to find which genes have expression levels that change with age.

Mice of ages 1, 6, 16, and 24 months were included. There were five male mice at each age and five female mice at each age. For each of these 40 mice, 16 microarrays were prepared, one for each of 16 tissues. The tissues considered were: adrenenal glands, bone marrow, cerebellum, cerebrum, eye, gonad (ovaries/testis), heart, hippocampus, kidney, liver, lung, muscle, spleen, spinal cord, striatum, and thymus.

From each of the $40 \times 16 = 640$ microarrays, values for 8932 genes were obtained. The microarrays had more than 8932 probes, but data from multiple probes corresponding to any single gene have been averaged.

We arrange the data into tissue specific matrices $Y^{(k)}$ for $k = 1, \ldots, 16$. Each $Y^{(k)}$ has 8932 columns. The matrix $Y^{(k)}$ has n_k rows, one for each sample of tissue type k. The values of n_k are unequal due to missing data. We have $32 \le n_k \le 40$ and the average sample size is 38.625. The entry $Y_{ij}^{(k)}$ is the logarithm of the expression value for mouse i and gene j in tissue k.

2.2 Regression with Measured and Latent Variables

For a single tissue type we can drop the superscript k. Zahn et al. (2007) used multiple regression analysis to investigate the effects of aging on gene expression. The regression model for gene j is

$$Y_{ij} = \beta_{0j} + \beta_{1j} A_i + \beta_{2j} S_i + \varepsilon_{ij}, \quad 1 \le i \le n \tag{1}$$

where Y_{ij} is log expression, A_i is the age in months of mouse i, and S_i is 1 if mouse i is female and is 0 otherwise. The random error term is denoted by ε_{ij}. We write all N regression models simultaneously as

$$Y = XB + E$$

where B is a p by N matrix of regression coefficients and E is an n by N matrix of random noise. The n by p design matrix X has a column for each covariate.

Now suppose that there is a latent variable taking the value U_i for the array of the ith mouse. Then adding the latent variable to the regression (1) yields

$$Y_{ij} = \beta_{0j} + \beta_{1j} A_i + \beta_{2j} S_i + \gamma_j U_i + \varepsilon_{ij}, \quad 1 \le i \le n,$$

where both γ_j and U_i are unknown. In matrix notation we have

$$Y = XB + U\Gamma + E \tag{2}$$

where $U \in \mathbb{R}^{n\times 1}$ and $\Gamma \in \mathbb{R}^{1\times N}$. If there are ℓ latent variables then the model remains as shown in (2), except that now $U \in \mathbb{R}^{n\times \ell}$ and $\Gamma \in \mathbb{R}^{\ell\times N}$.

Each individual regression includes more parameters than observations, having n latent values U_i and a coefficient γ_j. But in aggregate only $\ell(N+n)$ parameters are added to the regression for Nn observations, so the model is not saturated for small ℓ.

2.3 Forcing Identifiability

The latent variable model in (2) is not identifiable. To see why, note that if we were to replace U by $U + X\theta$ and B by $B - \theta\Gamma$, for some $\theta \in \mathbb{R}^{p\times N}$ then we would get the same residuals. A similar indeterminacy arises from replacing U by UC and Γ by $C^{-1}\Gamma$ for an invertible $C \in \mathbb{R}^{p\times p}$. We will sometimes assume that the latent variables U satisfy $X^\mathsf{T}U = 0$, $U^\mathsf{T}U = I_\ell$, and $\Gamma^\mathsf{T}\Gamma = D = \mathrm{diag}(d_1,\ldots,d_\ell)$ where $d_1 > d_2 > \cdots > d_\ell > 0$. This makes the model identifiable apart from the signs of the columns of U. Those can be specified by making the first nonzero value in each column positive. The existence of a latent term is not affected by identifiability of U, so we won't have to force U to be identifiable to detect a latent variable. We will also assume that $X^\mathsf{T}X$ has full rank p (this can be easily arranged by removing redundant predictors).

2.4 Noise Model and Estimation

In this section we describe the noise matrix E. One construction would be to assume that the entries of E are all independent and normally distributed with mean 0 and a different variance for each gene. We do not believe that the normality assumption causes serious difficulty for the AGEMAP data. But, assuming zero correlations among genes on the same array is not tenable. We assume instead that the rows of E are independent draws from the $\mathcal{N}(0,\Sigma_N)$ distribution where $\Sigma_N \in \mathbb{R}^{N\times N}$ is a gene-gene covariance matrix.

We will make frequent use of Kronecker notation. The random matrix $S \sim \mathcal{N}(M, A\otimes B)$ if its elements have a joint normal distribution with $\mathbb{E}(S_{ij}) = M_{ij}$ and $\mathrm{Cov}(S_{ij}, S_{kl}) = A_{ik}B_{jl}$ for matrices M, A, and B of appropriate dimensionality.

Our model for the error is that $E \sim \mathcal{N}(0, I_n \otimes \Sigma_N)$. Then model (2) may be written as

$$Y \sim \mathcal{N}(XB + U\Gamma, I_n \otimes \Sigma_N).$$

The identifiability restrictions of Section 2.3 are as before. It will often be simpler to introduce an $n \times N$ matrix Z with IID $\mathcal{N}(0,1)$ entries and note that

$$Y \overset{d}{=} XB + U\Gamma + Z\Sigma_N^{\mathsf{T}/2} \tag{3}$$

where $\Sigma_N^{1/2} \in \mathbb{R}^{N\times N}$ satisfies $\Sigma_N^{1/2}(\Sigma_N^{1/2})^\mathsf{T} = \Sigma_N$ and $\Sigma_N^{\mathsf{T}/2}$ is a shorthand for $(\Sigma_N^{1/2})^\mathsf{T}$. Similarly $A^{-\mathsf{T}}$ means $(A^{-1})^\mathsf{T}$ for invertible A.

Because of our orthogonality constraint, $X^\mathsf{T}U = 0$, the least squares estimate of B is unaffected by the latent variable. That is $\widehat{B} = (X^\mathsf{T}X)^{-1}X^\mathsf{T}Y$. We find that $\widehat{B}$ is normally distributed with $\mathbb{E}(\widehat{B}) = B$ and the covariance between $\hat\beta_{ij}$ and $\hat\beta_{kl}$ is $((X^\mathsf{T}X)^{-1})_{ik}(\Sigma_N)_{jl}$. We summarize this via

$$\widehat{B} \sim \mathcal{N}(B, (X^\mathsf{T}X)^{-1} \otimes \Sigma_N).$$

We can estimate the latent term $\widehat{U}\widehat{\Gamma}$ from the residual matrix $\widehat{E} = Y - X\widehat{B}$. The least squares estimates correspond to Principal Components Analysis (PCA), and can be gotten from truncating

the singular value decomposition of $\widehat{E}$ to ℓ terms. This least squares procedure has been described by Gabriel (1978), who also incorporates column based covariates analogous to the row based ones in X. Another alternative is to use Independent Components Analysis (ICA). If there is some prior knowledge about the distribution of the possible latent factors, this knowledge can be used in the estimation procedure, for example by choosing a particular test statistic to use in Section 3.

3. Rotation Tests for Structure in the Residual Matrix

We have two tasks when dealing with a latent term. The primary task is to determine whether any latent structure exists in the residual matrix $Y - X\widehat{B}$. A secondary task is to estimate that latent structure when we believe it exists.

The main complication is that Σ_N, the noise covariance, is unknown. When Σ_N is known, we can multiply both sides (3) from the right by $(\Sigma_N^{\mathsf{T}/2})^{-1}$ and obtain a model with the same regression variables, the same number of factors, and errors that are IID $\mathcal{N}(0,1)$. Then if $n \gg N$ classical methods due first to Gollob (1968) and refined by Mandel (1971) based on nested hypothesis testing, may be applied. If instead $n \ll N$ the resulting IID $\mathcal{N}(0,1)$ errors can be handled by recent developments in random matrix theory due to Baik and Silverstein (2006) and Paul (2007a). For example, in this setting Rao and Edelman (2008) apply an approach based on the Akaike information criterion (AIC).

Most methods for identifying latent structure only look at the singular values of the residual matrix $\widehat{E}$. Since the correlation in the residuals is nontrivial and unknown, our setting leads us to consider other functions of $\widehat{E}$.

We will use the following elementary formula. If $W \sim \mathcal{N}(0, \Psi \otimes \Phi)$ then

$$BWC^{\mathsf{T}} \sim \mathcal{N}(0, (B\Psi B^{\mathsf{T}}) \otimes (C\Phi C^{\mathsf{T}})), \tag{4}$$

so long as the product matrix BWC^{T} is well defined.

3.1 Rotations Under the Null Hypothesis

Under the null hypothesis of no latent variable, our error term is $E \sim \mathcal{N}(0, I \otimes \Sigma_N)$. For any $n \times n$ orthogonal matrix $\mathbb{O}$, we find that $\mathbb{O}E \sim \mathcal{N}(0, [\mathbb{O}I\mathbb{O}^{\mathsf{T}}] \otimes \Sigma_N) = \mathcal{N}(0, I \otimes \Sigma_N)$ so that $\mathbb{O}E \stackrel{d}{=} E$. The original residual matrix, E, and the rotated residual matrix, $\mathbb{O}E$ have the same distribution. This fact provides our starting point.

We refer to orthogonal matrices as rotations. A stricter usage of the term requires $\det(\mathbb{O}) = 1$ but following Langsrud (2005) we allow $\det(\mathbb{O}) = -1$ as well. Such reflections, as they are sometimes called, also preserve the distribution of E so they are worth including. In a rotation test we compare some aspect of the data to its value under repeated random rotations of the data. Such rotation tests are analogous to the more familiar permutation tests. Rotation tests were first introduced by Wedderburn (1975) and Heiberger (1978). A recent survey appears in Langsrud (2005) who focuses on multiple testing issues.

Here we give a self-contained derivation of rotation tests for multiple regression. The regression context requires us to make some modifications to the method.

The data are $Y = XB + E$ with $E \stackrel{d}{=} U\Gamma + Z\Sigma_N^{\mathsf{T}/2}$ where Z_{ij} are IID $\mathcal{N}(0,1)$. The matrix X has rank $p < n$ and hat matrix $H = X(X^{\mathsf{T}}X)^{-1}X^{\mathsf{T}}$. The residual matrix is $\widehat{E} = (I-H)Y = (I-H)XB + (I-H)E = (I-H)E$. We will apply many random rotations to $\widehat{E}$. This is mathematically equivalent

to rotating both X and Y each time, and then taking the residuals from the rotated variables, but of course it is faster to simply rotate the residuals. To prove equivalence:

Proposition 1 *For integers $n > p > 0$ and $N \geq 1$, let $X \in \mathbb{R}^{n\times p}$, $Y \in \mathbb{R}^{n\times N}$, $B \in \mathbb{R}^{p\times N}$ and $E \in \mathbb{R}^{n\times N}$ satisfy $Y = XB + E$. Suppose that $H = X(X^\mathsf{T}X)^{-1}X^\mathsf{T}$ has rank p. Let $\mathbb{O} \in \mathbb{R}^{n\times n}$ be an orthogonal matrix. Put $\widetilde{Y} = \mathbb{O}Y$, $\widetilde{X} = \mathbb{O}X$, $\widetilde{H} = \widetilde{X}(\widetilde{X}^\mathsf{T}\widetilde{X})^{-1}\widetilde{X}^\mathsf{T}$ and $\widetilde{E} = (I - \widetilde{H})\widetilde{Y}$. Then $\widetilde{E} = \mathbb{O}(I-H)Y$.*

Proof The rotated hat matrix satisfies $\widetilde{H} = \mathbb{O}X(X^\mathsf{T}\mathbb{O}^\mathsf{T}\mathbb{O}X)^{-1}X^\mathsf{T}\mathbb{O}^\mathsf{T} = \mathbb{O}H\mathbb{O}^\mathsf{T}$. The new residual is $\widetilde{E} = \widetilde{Y} - \widetilde{H}\widetilde{Y}$. Now $\widetilde{H}\widetilde{Y} = \mathbb{O}H\mathbb{O}^\mathsf{T}\mathbb{O}Y = \mathbb{O}HY$. Finally $\widetilde{E} = \mathbb{O}Y - \mathbb{O}HY = \mathbb{O}(I-H)Y$. ■

In the regression context, randomly rotating the residuals does not generally preserve their distribution. First because $(I-H)(I-H)^\mathsf{T} = (I-H)$, we find that under the null hypothesis ($U = 0$):

$$\widehat{E} \sim \mathcal{N}(0, (I-H)\otimes\Sigma_N).$$

But then, for the rotated residual we have

$$\mathbb{O}\widehat{E} \sim \mathcal{N}(0, [\mathbb{O}(I-H)\mathbb{O}^\mathsf{T}]\otimes\Sigma_N),$$

by Equation (4). These distributions do not usually coincide.

We will fix this problem by restricting attention to a special subset of rotation matrices. The desired rotations $\mathbb{O}$ satisfy $\mathbb{O}H\mathbb{O}^\mathsf{T} = H$ (equivalently $\mathbb{O}(I-H)\mathbb{O}^\mathsf{T} = (I-H)$) for then the rotation does not change the distribution of $\widehat{E}$. The rotations we want will fix X but rotate the space orthogonal to X. Specific construction details follow.

Because H has p eigenvalues equal to 1 and $n-p$ eigenvalues equal to 0 we may write it as $H = Q_1Q_1^\mathsf{T}$ where $Q_1 \in \mathbb{R}^{n\times p}$ satisfies $Q_1^\mathsf{T}Q_1 = I_p$. Let $Q_2 \in \mathbb{R}^{n\times(n-p)}$ be a matrix such that $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$ is orthogonal. For our construction, we let $\mathbb{O}_* \in \mathbb{R}^{(n-p)\times(n-p)}$ be an orthogonal matrix and then take

$$\mathbb{O} = Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*Q_2^\mathsf{T}. \tag{5}$$

The matrices produced by Equation (5) are orthogonal and satisfy $\mathbb{O}H\mathbb{O}^\mathsf{T} = H$. We summarize as follows:

Proposition 2 *Let $Y \sim \mathcal{N}(XB, I_n\otimes\Sigma_N)$ where $X \in \mathbb{R}^{n\times p}$ has rank $p < n$ and $B \in \mathbb{R}^{p\times N}$. Let $\widehat{E} = (I-H)Y$ where $H = X(X^\mathsf{T}X)^{-1}X^\mathsf{T}$ and let $\widetilde{E} = \mathbb{O}\widehat{E}$ where $\mathbb{O}$ satisfies* (5)*. Then both $\widehat{E}$ and $\widetilde{E}$ have the $\mathcal{N}(0, (I-H)\otimes\Sigma_N)$ distribution.*

Proof Let $E = Y - XB \sim \mathcal{N}(0, I_n\otimes\Sigma_N)$. Then $\widehat{E} = (I-H)Y = (I-H)E \sim \mathcal{N}(0, (I-H)\otimes\Sigma_N)$ as above. Now suppose that $\mathbb{O}$ satisfies (5). Then $\widetilde{E} = \mathbb{O}(I-H)E \sim \mathcal{N}(0, [\mathbb{O}(I-H)\mathbb{O}^\mathsf{T}]\otimes\Sigma_N)$. Next

$$\begin{aligned}
\mathbb{O}(I-H)\mathbb{O}^\mathsf{T} &= \left(Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*Q_2^\mathsf{T}\right)(I-H)\left(Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*Q_2^\mathsf{T}\right)\\
&= \left(Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*Q_2^\mathsf{T}\right)Q_2Q_2^\mathsf{T}\left(Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*^\mathsf{T}Q_2^\mathsf{T}\right)\\
&= Q_2\mathbb{O}_*\mathbb{O}_*^\mathsf{T}Q_2^\mathsf{T}\\
&= I-H,
\end{aligned}$$

and so $\widetilde{E} \overset{d}{=} \widehat{E}$. ■

To complete this section we show that Equation (5) generates all of the desired rotations.

Proposition 3 *Let $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \in \mathbb{R}^{n\times n}$ be an orthogonal matrix, where $Q_1 \in \mathbb{R}^{n\times p}$ for $n > p > 0$. Let $\mathbb{O} \in \mathbb{R}^{n\times n}$ be an orthogonal matrix and write $H = Q_1 Q_1^\mathsf{T}$. If $\mathbb{O}H\mathbb{O}^\mathsf{T} = H$ then $\mathbb{O} = Q_1\mathbb{O}_\circ Q_1^\mathsf{T} + Q_2\mathbb{O}_* Q_2^\mathsf{T}$ where $\mathbb{O}_\circ \in \mathbb{R}^{p\times p}$ and $\mathbb{O}_* \in \mathbb{R}^{(n-p)\times(n-p)}$ are orthogonal matrices.*

Proof First, any orthogonal matrix $\mathbb{O}$ can be written as $\mathbb{O} = QPQ^\mathsf{T}$ where

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}$$

is orthogonal, and partitioned with $P_{11} \in \mathbb{R}^{p\times p}$, $P_{22} \in \mathbb{R}^{(n-p)\times(n-p)}$ and so on. We may take $P = Q^\mathsf{T}\mathbb{O}Q$. Now $\mathbb{O}H\mathbb{O}^\mathsf{T} = QPQ^\mathsf{T}Q_1Q_1^\mathsf{T}QP^\mathsf{T}Q^\mathsf{T} = FF^\mathsf{T}$where

$$F = QPQ^\mathsf{T}Q_1 = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}\begin{pmatrix} Q_1^\mathsf{T} \\ Q_2^\mathsf{T} \end{pmatrix} Q_1 = Q_1P_{11} + Q_2P_{21}.$$

Assume that $\mathbb{O}H\mathbb{O}^\mathsf{T} = H$. Then $Q_1^\mathsf{T}\mathbb{O}H\mathbb{O}^\mathsf{T}Q_1 = I_p$ so that $(Q_1^\mathsf{T}F)(Q_1^\mathsf{T}F)^\mathsf{T} = I_p$. But $Q_1^\mathsf{T}F = P_{11}$. Therefore $P_{11}P_{11}^\mathsf{T} = I_p$, or in other words P_{11} is an orthogonal matrix. The columns of P_{11} are unit vectors and so are those of P. Therefore $P_{21} = 0$. Similarly $P_{12} = 0$ and P_{22} is an orthogonal matrix. Taking $\mathbb{O}_\circ = P_{11}$ and $\mathbb{O}_* = P_{22}$ completes the proof. ■

From Proposition 3 we see that the suitable rotations take the form $\mathbb{O} = Q_1\mathbb{O}_\circ Q_1^\mathsf{T} + Q_2\mathbb{O}_* Q_2^\mathsf{T}$. In Equation (5), we only use $\mathbb{O}_\circ = I_p$. We don't need to vary that part of the rotation because in our application we work with

$$\mathbb{O}(I-H) = \mathbb{O}Q_2Q_2^\mathsf{T} = (Q_1\mathbb{O}_\circ Q_1^\mathsf{T} + Q_2\mathbb{O}_* Q_2^\mathsf{T})Q_2Q_2^\mathsf{T} = Q_2\mathbb{O}_* Q_2^\mathsf{T}.$$

The choice of $\mathbb{O}_\circ$ does not affect the value of $\widetilde{E} = \mathbb{O}(I-H)Y$, and so we may simply take $\mathbb{O}_\circ = I_p$.

3.2 Testing for the Existence of Structure

Here we construct a test for latent structure in the residual matrix. The null hypothesis is $H_0 : U = 0$ and the alternative is $H_1 : U \neq 0$. We will construct a rotation test by modifying the rotation tests in Langsrud (2005). As in the discussion of the randomization tests in Lehmann and Romano (2005) we find a group of rotations. It is easy to show that $O_H = \{Q_1Q_1^\mathsf{T} + Q_2\mathbb{O}_*Q_2^\mathsf{T} \mid \mathbb{O}_* \in \mathbb{R}^{(n-p)\times(n-p)}, \mathbb{O}_*^\mathsf{T}\mathbb{O}_* = I_{n-p}\}$ is a group under multiplication, because orthogonal $r-p$ by $r-p$ matrices are a group. Southworth et al. (2009) give a cautionary note on randomizations without a group structure.

Proposition 2 allows us to perform a test of the hypothesis as follows: let $T : \mathbb{R}^{n\times N} \to \mathbb{R}$ be any statistic of the residual matrix. Specific examples are given in Section 3.4. We generate independent n by n random rotations $\mathbb{O}_1, \ldots, \mathbb{O}_{R-1}$ uniformly from O_H. Then, construct a p-value as

$$\hat{p} = \frac{1}{R}\Big(1 + \sum_{i=1}^{R-1} 1\{T(\mathbb{O}_i\widehat{E}) \geq T(\widehat{E})\}\Big). \tag{6}$$

Since under the null hypothesis $\mathbb{O}_i\widehat{E}$ and $\widehat{E}$ have the same distribution and since O_H is a group, this gives us a valid p-value. The leading 1 in $\hat{p}$ counts the observed $\widehat{E}$ and prevents us from claiming $\hat{p}$ below $1/R$ if we have only seen R rotations (including the original one).

3.3 Gaussian Alternative

Here we show that when Σ_N is unknown, a Gaussian latent variable cannot be detected. This was remarked on by Snee (1982) and it is well known in the independent component analysis community; see Hyvarinen et al. (2001).

To see how the problem manifests, consider the model in (3) with just one latent variable with entries $U_i \sim \mathcal{N}(0,1)$ independently of Z. The regression model XB is assumed nonrandom. To focus on essentials we do not initially impose the normalization $U^\mathsf{T}U = I_\ell$.

Proposition 4 *For positive integers N, n, and ℓ, let $Z \sim \mathcal{N}(0, I_n \otimes I_N)$ independently of $U \sim \mathcal{N}(0, I_n \otimes I_\ell)$. Let $\Gamma \in \mathbb{R}^{\ell\times N}$. Then $U\Gamma + Z\Sigma_N^{\mathsf{T}/2} \overset{d}{=} Z\widetilde{\Sigma}_N^{\mathsf{T}/2}$ where $\widetilde{\Sigma}_N = \Sigma_N + \Gamma^\mathsf{T}\Gamma$.*

Proof We only need to show that $U\Gamma + Z\Sigma_N^{\mathsf{T}/2}$ has the same distribution as $Z\widetilde{\Sigma}_N^{\mathsf{T}/2}$. The matrix $U\Gamma + Z\Sigma_N^{\mathsf{T}/2}$ has independent identically distributed rows from $\mathcal{N}(0, \Sigma_N + \Gamma^\mathsf{T}\Gamma)$. Therefore $U\Gamma + Z\Sigma_N^{\mathsf{T}/2}$ has representation $\widetilde{Z}\widetilde{\Sigma}_N^{\mathsf{T}/2}$ where $\widetilde{Z} \in \mathbb{R}^{N\times n}$ has IID entries from $\mathcal{N}(0,1)$. This $\widetilde{Z}$ has the same distribution as Z. ■

If we do normalize U then nothing essential changes. We replace U by UC for a random normalizing matrix $C \in \mathbb{R}^{\ell\times\ell}$ that is independent of Z (that is, $UC \sim \mathcal{N}(0, I_n \otimes I_\ell)$ and $U^\mathsf{T}U = I$). Then we compensate by replacing Γ by $C^{-1}\Gamma$ and get $U\Gamma + Z\Sigma_N^{\mathsf{T}/2} \overset{d}{=} Z\widetilde{\Sigma}_N$ where $\widetilde{\Sigma}_N = \Sigma_N + \Gamma^\mathsf{T}C^{-\mathsf{T}}C^{-1}\Gamma$ is now random.

The implication of Proposition 4 is that if we don't know anything about Σ_N, or Γ, then a Gaussian latent variable is impossible to detect. There is no mathematical difference between a Gaussian latent vector and a changed correlation structure. Put another way, such latent variables are already well accounted for in the correlation structure.

Latent variables of practical interest typically exhibit non-Gaussian traits like clumping or outliers. Also, if a latent variable corresponds to a roughly-linear time trend, then it will be nearly uniformly distributed if the points are sampled at regular time intervals. Therefore this restriction still leaves many interesting testing problems.

3.4 Choice of the Test Statistic, T

Since a Gaussian latent variable is covered by the correlation model and is not detectable, any effective test statistic T must be tuned for non-Gaussian latent variables. A non-Gaussian latent variable makes for an error term $U\Gamma + Z\Sigma_N^{\mathsf{T}/2}$ that does not have a rotationally invariant distribution.

In principle any function $T(E)$ can be used. However, the choice of T will often be dictated by what we deem to be interesting structure. Here we describe four different possibilities for T. We first apply a simplification procedure to reduce E to a vector $u(E)$. Then, we apply a function to reduce u to a scalar. The end result is a scalar-valued function $T(E)$.

We would like $u(E)$ to be representative of latent structure in E. An obvious choice is the first left singular vector of E, which corresponds to the first principal component of E. A second choice is to apply ICA to E, treating the columns as mixtures of n-dimensional sources, and have $u(E)$ be the first estimated source. In both cases $u(E)$ is a unit vector.

We cannot simply use a test for normality of the components of $u(E)$, such as the Anderson-Darling test, because the components we get are not independent $\mathcal{N}(0,1)$ even under H_0.

Instead, we propose two functions for reducing $u = u(E)$ to a scalar. The first is the L^1 norm of the vector:

$$T_{L^1}(u) = \sum_{i=1}^{n} |u_i|.$$

As a point of reference, for independent $u_i \sim \mathcal{N}(0, \frac{1}{n})$ we would get $T_{L^1} \doteq \sqrt{2n/\pi}$. So, the expected L^1 norm of a uniformly distributed n-dimensional unit vector is approximately $\sqrt{2n/\pi} \doteq 0.798\sqrt{n}$. Larger values of T_{L^1} correspond to distributions whose expected absolute value is large compared to their root mean square. Uniform distributions on $[-1,1]$ or $\{-1,1\}$ behave this way. Conversely, small values of T_{L^1} arise from very heavy tailed distributions like the Cauchy which have outliers.

The rotation based p-value (6) is sensitive to large values of T_{L^1} and should therefore catch dichotomies and light tailed latent variables. To detect heavy tailed alternatives we could use (6) with $1/T_{L^1}$. Because we are potentially interested in both kinds of non-Gaussian latent structure we take

$$\hat{p} = \frac{2}{R} \min\left(1 + \sum_{i=1}^{R-1} 1\{\widetilde{T}_i \geq \widehat{T}\},\ 1 + \sum_{i=1}^{R-1} 1\{\widetilde{T}_i \leq \widehat{T}\}\right), \tag{7}$$

where $\widehat{T} = T(\widehat{E})$ and $\widetilde{T}_i = T(\mathbb{O}_i \widehat{E})$ and $T(\cdot)$ subsumes all the computation in T_{L^1}. The leading 2 in (7) compensates for using the more extreme of two tails.

The second test statistic comes from Exploratory Projection Pursuit (Friedman, 1987). T_{EPP} is a distance measure on densities, represented as a Legendre-series and then truncated to 4 terms:

$$T_{\text{EPP}}(u) = \sum_{j=1}^{4} \left(j + \frac{1}{2}\right) \left(\mathbb{E}P_j(R)\right)^2$$

where, P_j is the j-th Legendre polynomial, R is a random variable uniformly distributed over the discrete set $\{2\Phi(u_i) - 1\}_{i=1}^n$, Φ is the cumulative distribution function of the $\mathcal{N}(0,1)$ distribution, and $\mathbb{E}$ denotes expectation over the randomness in R. The Legendre polynomials can be computed using the recurrence relation

$$\begin{aligned} P_0(x) &= 1, \\ P_1(x) &= x, \quad \text{and}, \\ (j+1)P_{j+1}(x) &= (2j+1)xP_j(x) - jP_{j-1}(x). \end{aligned}$$

In computing T_{EPP} we use

$$\mathbb{E}P_j(R) = \frac{1}{n} \sum_{i=1}^{n} P_j(2\Phi(u_i) - 1).$$

T_{EPP} is designed to be close to 0 when the histogram of u looks Gaussian, and it gets bigger the more "non-Gaussian" u is. The full derivation of T_{EPP} is given by Friedman (1987).

Because only large values of T_{EPP} are interesting we use (6) directly without making a two-tailed modification.

We can combine PCA or ICA with $T_{\text{EPP}}(u)$ or $T_{L^1}(u)$ and get four different test statistics. We write $T_{\text{EPP}}(E)$ or $T_{L^1}(E)$ when the context dictating PCA or ICA for u is clear. The choice of T is independent of the procedure for estimating the latent variable. In particular, it is possible to detect the existence of latent structure in $\widehat{E}$ using a PCA-based test statistic and then fit the structure using ICA. Indeed, in simulations it turns out to be better to use PCA for u. This will be further explored in Section 4.

3.5 Identifying the Rank of the Latent Term

When we are able to reject the null hypothesis we conclude that some latent structure exists, but we do not know the rank of U. To estimate the number of latent variables we consider a sequential approach, based on subtracting estimated latent variables and looking for latent structure in the residuals.

First, we fit the model $\widehat{Y}_0 = X_0\widehat{B}_0$ and get the residual matrix $\widehat{E}_0$. The subscripts denote a model with 0 latent terms. Next, we test for latent structure in $\widehat{E}_0$. If we determine that any structure exists, we fit a single latent variable $\hat{u}_1$.

At this stage, we treat $\hat{u}_1$ as a known covariate. We create a new covariate matrix $X_1 = [X_0 \;\; \hat{u}_1]$ by appending $\hat{u}_1$ as a column onto the old covariate matrix. Now, we fit the a new model $\widehat{Y}_1 = X_1\widehat{B}_1$ and get a new residual matrix $\widehat{E}_1$. We proceed in a sequential matter: test for structure in $\widehat{E}_i$; upon identifying structure, fit a single latent variable, treat it as a covariate, and get a new residual matrix $\widehat{E}_{i+1}$. We stop when there is no latent structure in $\widehat{E}_i$.

In the context of PCA, fitting $\hat{u}_2$ sequentially after adjusting $\widehat{E}$ for $\hat{u}_1$ is equivalent to fitting $\hat{u}_1$ and $\hat{u}_2$ simultaneously. For other estimation methods, this equivalence may not hold. The procedure we describe is still valid, but there may be some loss in power. If this is a concern, the practitioner can adjust the testing procedure accordingly.

One has to be careful when testing for more than one latent term. In particular, for some settings when $n \ll N$, it is impossible to consistently estimate the latent variables $\hat{u}_i$. When we do not have a good estimate of $\hat{u}_i$, treating it as a known covariate will introduce a potentially serious error. When this happens, only the p-value for the first term is reliable. This point is illustrated in the example of Section 4.3, where the error in the p-value distribution was small.

3.6 Caveats

When we reject the null hypothesis, then either there is strong enough latent structure in the data, or the noise is far from Gaussian. Therefore, rejecting the null hypothesis is *necessary* to deem latent structure to be real, but not sufficient. Often there is ambiguity between what constitutes non-Gaussianity and what can be explained by a latent variable. An outlier can be modeled using a latent variable that has support on a single observation. Bi-modal noise can be re-cast as a clumping latent effect.

3.7 Related Work

Rank determination methods have been the subject of much interest in crop science. For a recent survey see Crossa and Cornelius (2002). Those methods tend to focus on the amount of variance explained by the first principal component. In an eigen-analysis of $Y - X\widehat{B}$, they focus on the size of the eigenvalues. There has been considerable difficulty with getting tests to have the right level, as described for example by dos S Dias and Krzanowski (2003). The core problem is that there is no good way to count the degrees of freedom for such data sets, despite recent progress in random matrix theory including El Karoui (2007), Paul (2007b), and Nadler (2007). Owen and Perry (2009) apply a cross-validation-based approach to rank determination for the truncated SVD and non-negative matrix factorization. That work requires independent noise, not the correlated noise we consider here. Efron (2009) uses permutations to test whether some microarrays are independent of each other.

4. Empirical Testing

In this section we examine the performance of rotation tests on constructed examples where we know the answer. Some readers may prefer to read the real data example of Section 5 first. In our constructed examples, the response satisfies

$$Y \sim \mathcal{N}(XB + U\Gamma, I_n \otimes \Sigma_N),$$

with parameters described below.

4.1 Microarray Model

This example is designed to resemble microarray studies. We take $p = 2, n = 20$ and $N = 256$. This value of N is small, to allow a larger number of simulated cases. The matrix X has a first column of 1s. The second column has values $1, \ldots, n$. We take B to be a $p \times N$ matrix of 0s. Having $B = 0$ is no loss of generality, because the analysis works on residuals after regression on X and the residuals are unaffected by B.

We construct latent variables $U \in \mathbb{R}^{n\times 3}$. The first latent variable, u_1, is the first column of U and has independent elements distributed as Cauchy random variables. The second latent variable, u_2, has elements which are either -1 or 1 with equal probability. The third latent variable, u_3, has elements that are independent and exponentially distributed with mean 1. Thus, u_1 is an "outlier" effect, u_2 is a "clumping" effect, and u_3 is some other latent effect.

The latent coefficient matrix, Γ, has independent elements distributed as $\mathcal{N}(0,1)$. We do not think that non-normal Γ would make the signal artificially easy to detect, but taking Gaussian Γ removes any such worry. As described, U, Γ, and X do not satisfy the identifiability conditions of Section 2.3. The existence of an unnormalized latent variable implies that a normalized one exists, and so the testing problem is unaffected.

For Σ_N we need a 256×256 correlation matrix. The true correlation patterns for microarray data are not known. The sample sizes to date are far too small to allow confident description of the patterns. Owen (2005) looks at what gene-gene correlations are like in real data. We mimic two features of microarray data. First, genes are often thought to belong to relatively small clusters. Second, the mean of the squared estimated off-diagonal sample correlations is often seen to be a small multiple of $1/n$. The value $1/n$ is very close to what we would expect in the event that all true correlations were zero. To encode the first property, we take

$$\Sigma_{ij} = \begin{cases} 1 & i = j, \\ \rho & \lfloor (i-1)/32 \rfloor = \lfloor (j-1)/32 \rfloor, \quad \& \quad i \neq j, \\ \rho & i - j \equiv 0 \bmod 32, \quad \& \quad i \neq j, \\ 0 & \text{else.} \end{cases}$$

In words, gene i belongs to two clusters: one cluster of 8 genes corresponding to the least-significant digit of $i-1$ in base 32, and one cluster of 32 genes corresponding to the most-significant digit of $i-1$ in base 32. Gene i has 38 non-zero correlations with other genes. The value of $\rho > 0$ is chosen so that signal is about 30% of the noise:

$$\frac{\sum_{i=1}^N \sum_{j=1}^N \Sigma_{ij}^2 - N}{N(N-1)/n} = 0.30.$$

Thus $38\rho^2 = (N-1)/(0.30\,n)$ so $\rho = \sqrt{0.30\,(N-1)/(38n)} \doteq 0.317$.

4.2 Rotation Tests

The true model has three latent variables. We are interested in what happens when testing for the first, second, third, and fourth latent terms. We look at two different choices for the test statistic: T_{EPP} in conjunction with PCA, and T_{EPP} in conjunction with ICA. The results are summarized in ROC curves in Figure 1.

The upper-right panel shows the results from testing the residual matrix after one latent term has been removed. Here, we see that testing with u_{PCA} results in about 75% of the replicates having estimated p-values less than 0.2, while testing with u_{ICA}, results in about 55%. Generally, the PCA test gave us higher power. We also found that FastICA can get stuck in a local minimum. This is what lead to its surprisingly poor performance in the upper left panel. The latent variables of the randomly-rotated data are more non-Gaussian than the latent variable estimated from the original data.

The lower-right panel shows the estimated p-values after three latent terms have been fit and removed from the residual matrix. As expected, the estimated p-values are close to the specified false-positive rates. Comparing the lower left panel to the others, we see, unsurprisingly that the smallest latent vector is hardest to detect while the largest is easiest to detect. Finally, testing for a fourth latent variable gives us a uniform p-value, which is exactly what we want since there are only three latent terms.

A word is in order about how we removed the first latent variable when testing for the presence of the second. We tried removing vectors as estimated by PCA and also by ICA. There was not much difference in performance, and PCA has the computational advantage that the estimated second vector does not change when we remove the first. Therefore when testing for the k'th vector, whether by ICA or PCA, we always used PCA to remove the first $k-1$ of them.

4.3 Testing Under and Near the Null Hypothesis

In the previous simulation, the signal-to-noise ratio between the latent effect terms and the random error is relatively high, and so the p-values for non-existent latent terms are faithful. In this simulation, we demonstrate that the p-values for testing for multiple latent variables are slightly liberal if the signal strength is too weak, but these p-values are still within tolerable accuracy.

We generate an $n \times N$ data matrix Y according to the model

$$Y = (N\lambda)^{1/2} u\gamma^{\mathsf{T}} + Z\Sigma_N^{\mathsf{T}/2},$$

with $n = 20$ and $N = 200$. There is a single latent variable u which has elements equal to -1 or $+1$ with equal probability. The coefficient vector γ is a uniformly distributed random unit vector in $\mathbb{R}^N$. The noise covariance Σ_N is a diagonal matrix with $(\Sigma_N)_{ii}$ independent from all other entries and exponentially-distributed with mean 1; $\Sigma_N^{1/2}$ is its square root. The noise variable matrix Z has IID $\mathcal{N}(0,1)$ elements. We choose λ to be a fixed scalar, specified below.

The theory in Section 3.2 tells us that the p-value from a rotation test of a single latent term is uniformly-distributed when $\lambda = 0$. However, it tells us nothing about p-values for a second term. Regardless of the value of λ, we would like them to be uniformly distributed, so that the test is faithful to the specified false positive rate. The issue is whether errors in the estimated first latent vector spoil the test for the second. Results in Onatski (2007) suggest that as the sample size goes to infinity, p-values from the second and higher terms will be faithful when we fit with PCA. Our sample size is only 20, so we do an empirical test.

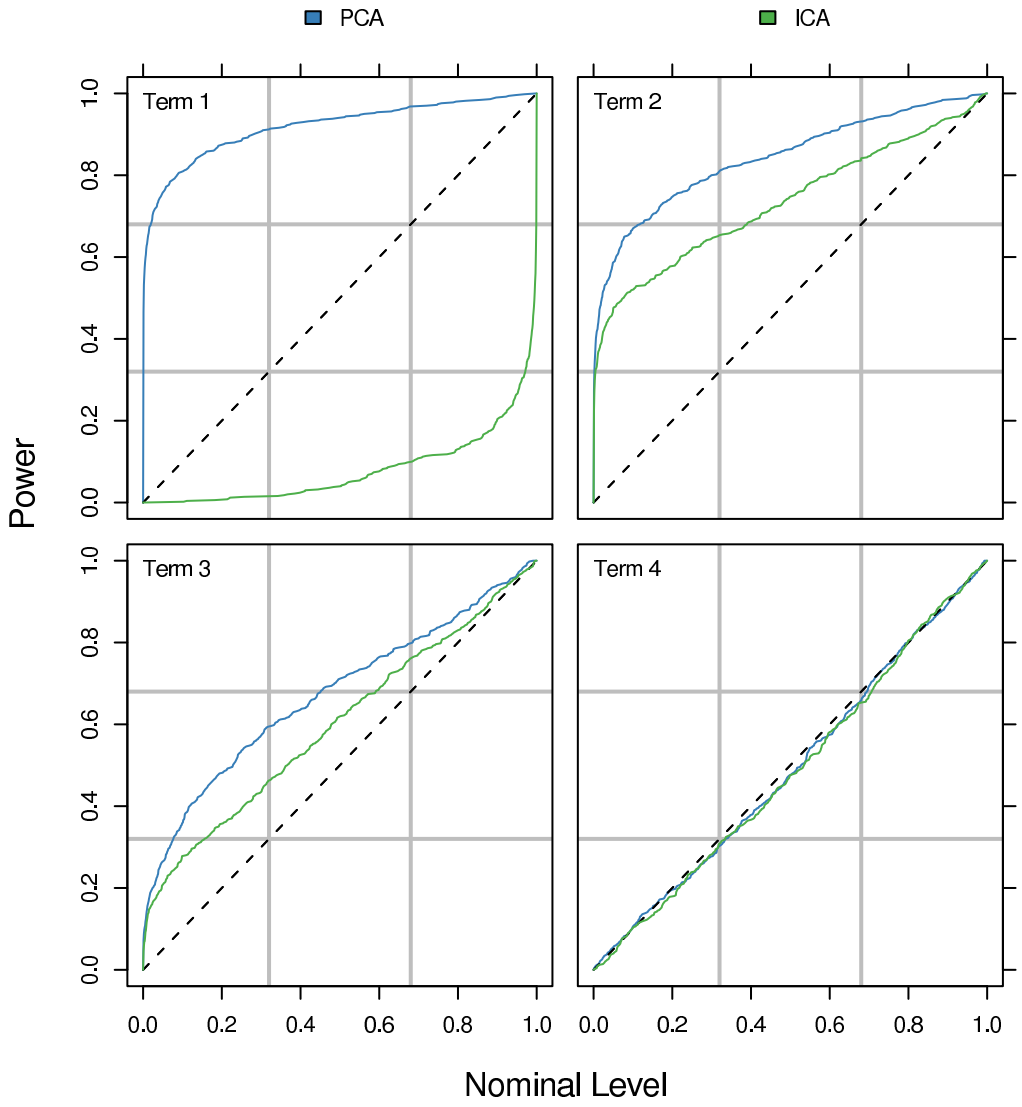


Figure 1: MULTI-FACTOR ROTATION TESTS WITH PCA AND ICA. We simulate data from a model with three latent terms and then apply rotation tests for latent structure. The test statistic is $T_{\mathrm{EPP}}(u)$ applied to either the first principal component u_{PCA}, or the first independent component u_{ICA} (via FastICA) of the residual matrix. The plots show estimated ROC curves after $0, 1, 2$, or 3 principal components have been fit and removed. The x axis of each plot is the specified false-positive rate. The y axis is the proportion of replicates with an estimated p-value below that level, using 500 total replicates of the data set. The plots are discussed further in the text.

For all λ in the set $\{0, 0.5, 1, 5, 10, 50, 100, 500, 1000\}$, we perform the following simulation, which we repeat 1000 times:

1) Generate data Y as described above.

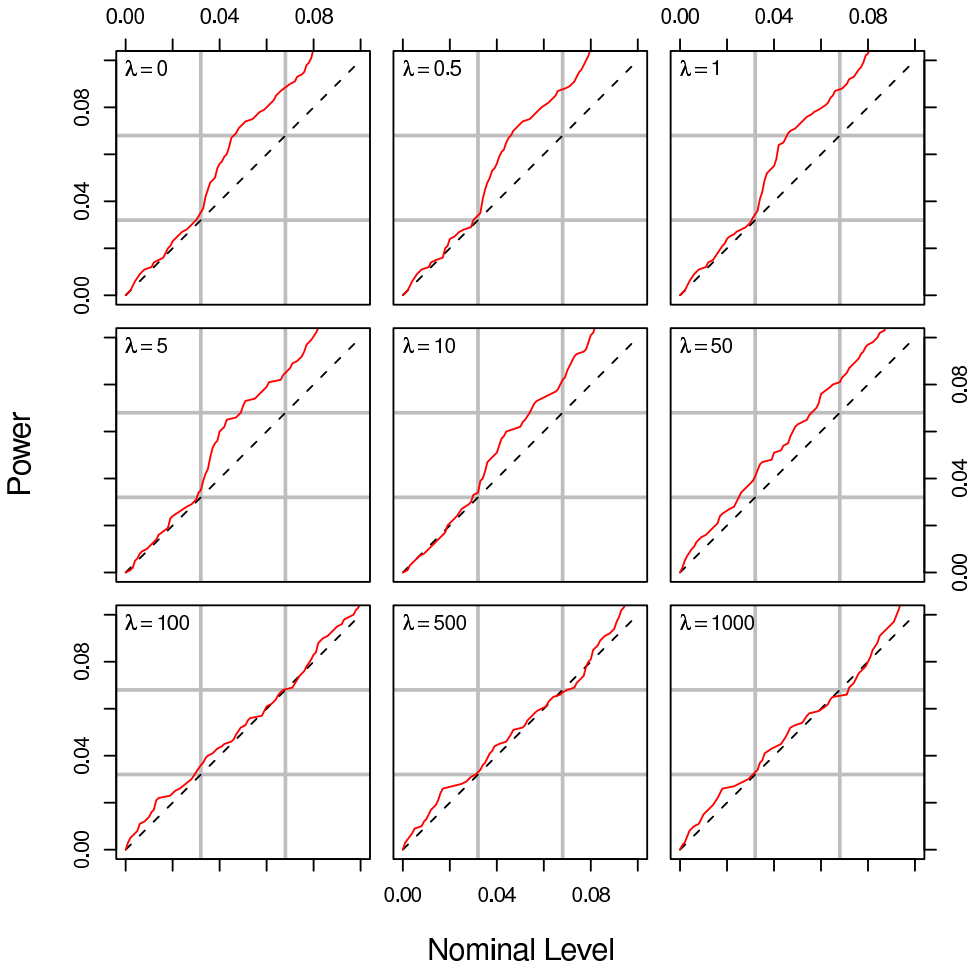


Figure 2: TESTING FOR A NON-EXISTENT SECOND TERM. We estimate a factor generated according to $(N\lambda)^{1/2}u\gamma^{\mathsf{T}}$ and then test for a second term. Depending on the factor strength (which is related to our ability to estimate), the p-values for validating the second term may be slightly liberal. This seems to be the case for $\lambda \leq 50$. For $\lambda \geq 100$, the p-values appear to be faithful.

2) Fit a single latent variable $\hat{u}$, using the first term in the SVD of Y.
3) Construct the matrix of residuals as $\widehat{E} = Y - \hat{u}\hat{u}^{\mathsf{T}}Y$.
4) Test for the existence of more latent terms using a PCA-based rotation test using T_{EPP} as our test statistic and treating $\hat{u}$ as a known covariate. Record the p-value estimated from 999 random rotations.

We would like to assess the implications of treating $\hat{u}$ as a known covariate. When λ is big, this is a reasonable assumption since the term is easy to estimate, but when λ is small this is not the case.

We summarize the results in Figure 2. We can see that for $\lambda \leq 50$, the small p-values are slightly liberal. When $\lambda \geq 100$, the p-values appear to be faithful. When the first latent variable is strong, then we have a reliable test for the second.

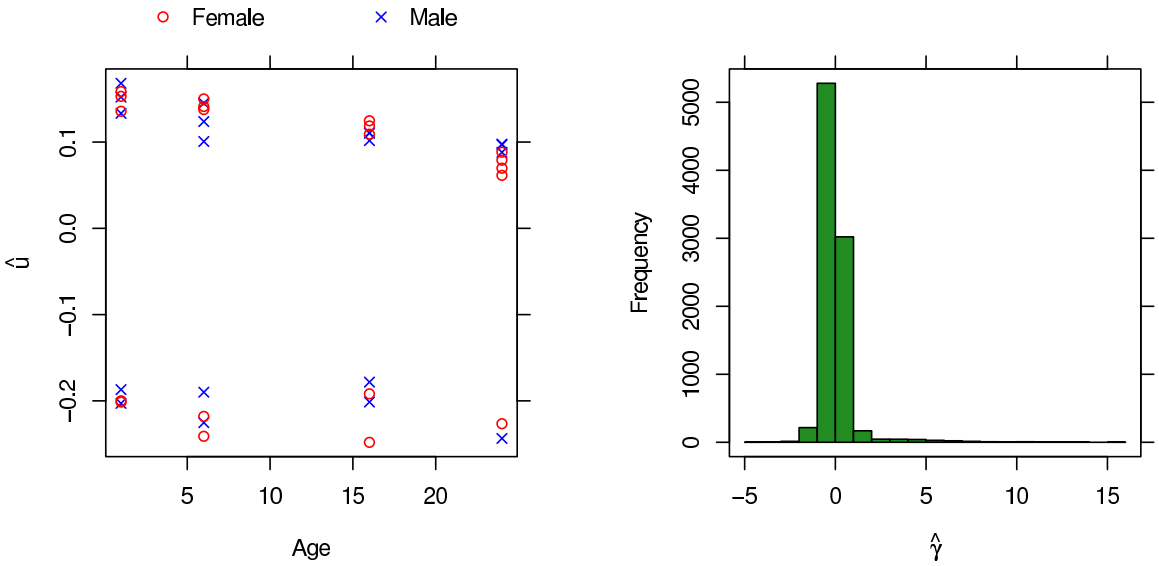


Figure 3: CEREBELLUM LATENT VARIABLE. The left plot shows the latent variable estimated for the cerebellum in each of 39 mice plotted versus the ages of those mice. We can see an apparent dichotomy that is unrelated to gender. The right plot shows a histogram of the regression coefficients for the latent variable. The long tail of the histogram indicates that a large number of genes (about 100) are related to the dichotomy.

5. Latent Variables for the AGEMAP Mice

Now that we have seen how rotation tests work in simulations, we apply them to the data described in the beginning of the paper. Recall that in the AGEMAP data set there are 16 tissue types and 32–40 mice per tissue with known age and sex. Here we will see patterns that certainly appear unlikely to be artifacts. Then we verify them by the rotation test.

We fit 16 regression models of gene activation on age and sex with one latent variable, one for each tissue. The result is that for each tissue type from $k = 1, \ldots, 16$, we have an estimated latent variables vector $U^{(k)} \in \mathbb{R}^{n_k}$.

The latent variables for tissue 2 (the cerebellum) have a striking pattern. There is one value $\widehat{U}_i^{(2)}$ for each of $n_2 = 39$ mice for which a cerebellum array was available. Figure 3 shows that latent variable plotted versus age and with plot symbols encoding the sex of the mouse. It is clear that the mice are split into two different groups, one with a high value of the latent variable and one low.

Often when one sees two distinct groups in microarray data, they correspond to male versus female samples, and certain genes that are sex related, such as on those on the Y chromosome in males or Xist genes that silence a second X chromosome for females. That cannot be the case here because the estimated latent variable is orthogonal to both the sex and age variables by construction, meaning the sum of its coefficients over male samples must equal the negative of the sum over females.

There are high and low values for the latent variable for the cerebellum. The second panel of Figure 4 shows the histogram of these latent values. It is clearly bimodal. The other 15 panels in Figure 4 show the corresponding histograms for the other 15 tissues.

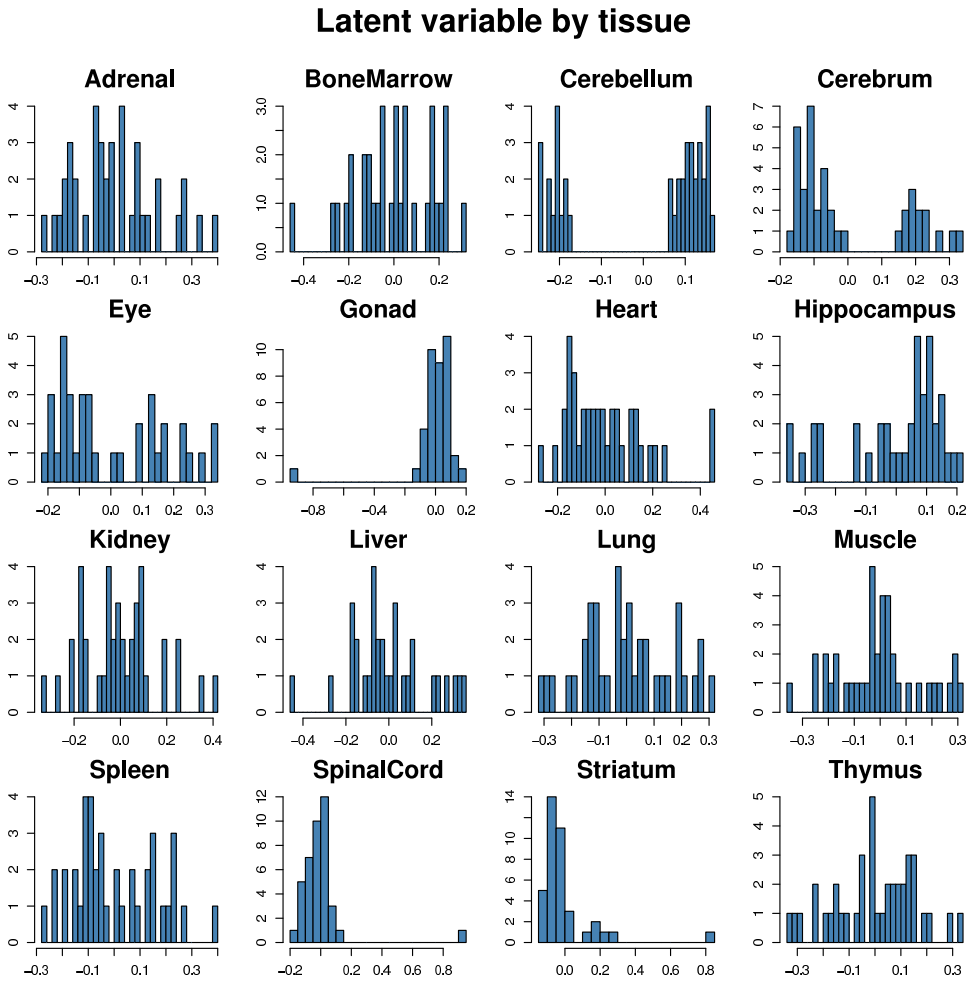


Figure 4: LATENT VARIABLES BY TISSUE. This figure shows histograms of the latent variables found in microarray data from 16 mouse tissues. In each histogram the latent variable values from up to 40 mice are given.

Some of the other histograms have interesting and interpretable structure too. The histograms for spinal tissue, gonad and striatum all show outliers. The biggest latent effect in these tissues is that the expression of one mouse was quite different from the other mice and that difference is reflected in a large number of genes. It is not simply one unusual animal. Three different mice were the outliers in the three different tissues.

The histogram for the cerebrum shows an apparent dichotomy, similar to but less pronounced than the one for the cerebellum. For both of these tissues, the latent variable is splitting the mice into two groups. Both dichotomies are somewhat imbalanced with one group roughly twice as large as the other. Such an effect would be explainable if the same latent factor were affecting both of these brain tissues. Figure 5 plots the estimated latent variable from the cerebrum versus that for the cerebellum. There is one point for each of the 39 mice in which both tissues were measured.

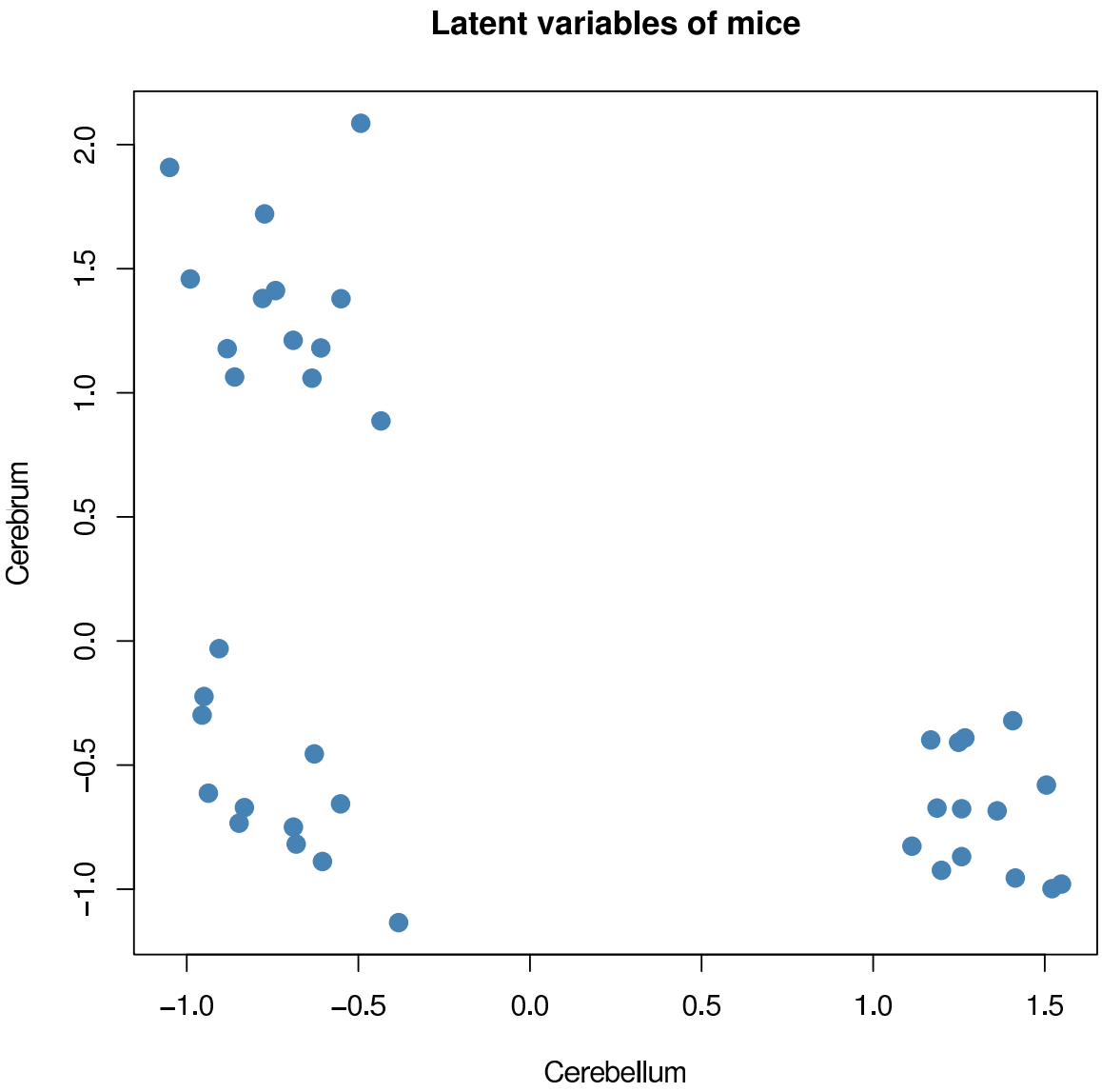


Figure 5: LATENT VARIABLES OF MICE. This figure plots the latent variable from the cerebrum versus that from the cerebellum for the 39 mice for which both arrays were available. There appear to be three kinds of mice in this population.

Figure 5 shows that the apparent dichotomy in the cerebrum is not the same as the one in the cerebellum. The pattern is not a simple double dichotomy either. Rather there appears to be a trichotomy. It is visually striking that there are no mice in the upper right hand corner of Figure 5. The counts of the four corners of Figure 5 are set out in Table 1.

About one third of the mice have the rare cerebellum type, one third have the rare cerebrum type and the remaining mice the common form for both tissues. Were the types independent we would expect about one ninth of the mice to be rare for both. A p value based on Fisher's exact test is

Count	Common cerebellum	Rare cerebellum
Rare cerebrum	13	0
Common cerebrum	12	14

Table 1: Counts of mice in the corners of Figure 5.

0.00094. While we can't be sure that the next mouse won't be the rare type for both brain parts, the failure to observe one here is statistically significant.

Although we don't have laboratory notes to identify the meaning of these groupings, the fact that the joint behavior of two dichotomies from different tissues forms such an interesting pattern lends additional support to the rotation results.

We applied rotation tests for all 16 tissues in the AGEMAP data set using two different measures of non-normality.

We demonstrate the calculation of p-values for the spleen and cerebellum data in Figure 6. The rotation distributions for the other 15 tissues have approximately the same shape, so we do not display them here. Instead, we summarize the results in Table 2. The estimated latent variables for the cerebellum, cerebrum, and eye tissues exhibited dichotomies. This shows up in significantly high values of T_{L^1} and T_{EPP}. The gonad, spinal cord, and striatum latent variables have clear outliers, which manifest as a significantly low values of T_{L^1}, and a significantly high values of T_{EPP}. The spleen latent variable potentially has an outlier at age 5 months, and is found to be marginally significant according to T_{EPP}, and not significant according to T_{L^1}. The discrepancy is because of the two- versus one-tailed p-value.

The only case where T_{L^1} and T_{EPP} give drastically different results is with the latent variable estimated from the hippocampus data. Using T_{L^1}, the variable is nowhere near significant ($\hat{p} = 0.586$), but using T_{EPP}, the variable is unquestionably significant ($\hat{p} < 0.001$). T_{EPP} finds the skewed histogram interesting, while T_{L^1} does not. A possible explanation for why the latent variable is insignificant according to T_{L^1} is this: T_{L^1} is simultaneously measuring presence of outliers and presence of clumping. Outliers correspond to low values of T_{L^1}, and clumping corresponds to high values of T_{L^1}. In the hippocampus data, we see both outliers *and* clumping. The two features "cancel out", giving a moderate value of T_{L^1}. T_{EPP}, on the other hand, does not distinguish between the different kinds of non-Gaussianity. The two features act in tandem to give a high value of the test statistic.

6. Conclusions

We find that it is possible to test for latent variables in correlated Gaussian noise by a rotation test using a projection pursuit index applied to the components of the first singular vector, instead of the usual test based on the size of the largest singular value. This test detects the lack of rotational invariance of the matrix of errors. The rotations must be done orthogonally to the regression variables. Testing for one latent variable is theoretically justified and reliable. Testing for additional terms is possible, but can give somewhat liberal p-values if the signal strength is too weak.

For microarray data, a normal distribution is often a very reasonable model. Some researchers apply transformations for the explicit purpose of making the data more normally distributed. For data that is not close to normally distributed a strategy of looking for latent variables by measuring how non-Gaussian they are is not recommended. It might uncover eigenvectors with especially

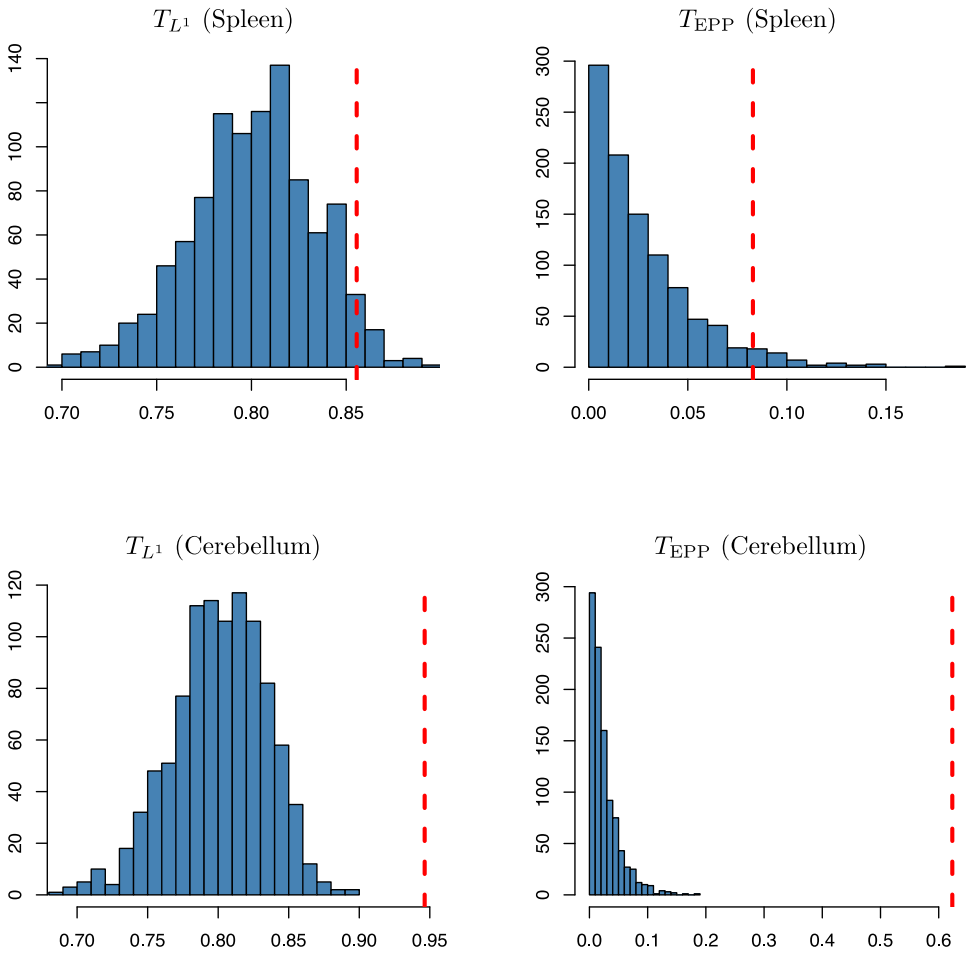


Figure 6: This figure shows histograms of 1000 realizations of the test statistics T_{L^1} and T_{EPP} after applying random rotations to the estimated latent variable. The top row comes from the spleen data, and the bottom row from the cerebellum. The dashed line shows the value of the test statistic at the observed data. In the top-left plot, the area in the right tail is 0.049, giving a two-sided p-value of 0.098. In the top-right plot, the tail area and the one-sided p-value are equal to 0.045. Formal descriptions of the p-values can be found in Equations (6) and (7). The latent variable is found to be barely significant at the 0.05 level according to T_{EPP}, but not according to T_{L^1}. In the bottom plots, the observed data falls at the extreme of the rotation histograms, and is found to be strongly significant in both cases.

non-Gaussian components but their interpretation is more difficult without a Gaussian background to compare them to.

Our original interest was to see if thousands of genes could be used to define a genomic "true age" of a sample of mouse tissue as a latent variable in the residuals from a regression that did not include age. It turned out that the dominant latent variable bore no resemblance to chronological

		T_{L^1}		T_{EPP}	
Tissue	R^2	T	$\hat{p}$	T	$\hat{p}$
Adrenal	0.109	0.7986	0.816	0.0204	0.460
Bone Marrow	0.227	0.8098	0.988	0.0035	0.931
Cerebellum	0.593	0.9464	0.002**	0.6233	0.001**
Cerebrum	0.520	0.8961	0.002**	0.4344	0.001**
Eye	0.165	0.8927	0.004*	0.2343	0.001**
Gonad	0.197	0.4788	0.002**	0.3899	0.001**
Heart	0.287	0.7866	0.562	0.0611	0.117
Hippocampus	0.208	0.8222	0.586	0.2281	0.001**
Kidney	0.247	0.7702	0.320	0.0145	0.598
Liver	0.311	0.7758	0.384	0.0608	0.170
Lung	0.169	0.8085	0.944	0.0130	0.625
Muscle	0.318	0.7601	0.172	0.0583	0.121
Spleen	0.328	0.8555	0.098	0.0829	0.045*
Spinal Cord	0.309	0.4641	0.002**	0.3864	0.001**
Striatum	0.319	0.5851	0.002**	0.4020	0.001**
Thymus	0.266	0.8125	0.838	0.0264	0.386

Table 2: Test statistics and p-values from the rotation tests applied to the AGEMAP data. In all cases, 1000 random rotation were used to construct the a reference histogram, and approximate p-values were estimated. Significant results at the 0.05 level are marked with a single asterisk. In instances where the observed test statistic was at the extreme end of the histogram, we have marked the p-values with two asterisks. In the second column of the table, we indicate how much of the residual is explained by the latent term. We can see that high R^2 does not necessarily indicate significance according to the rotation test. A Bonferroni correction for multiple testing would multiply the p-values by 32 and would find most of the same latent variables significant.

age. We never uncovered a biological explanation for the dichotomies and other latent variables that we saw. But, the rotation tests confirm that these striking anomalies would not arise from correlated Gaussian noise. Several of the tissues did not have apparent latent variables. Accordingly results like those in Table 2 help one focus on where to search for physical causes underlying apparent latent variables.

It may happen that a latent variable is statistically significant when judged by a rotation test but only explains a negligible amount of the response variation. This seems unlikely to happen in practice and did not happen for the AGEMAP data, according to the R^2 column in Table 2. But, when it does happen one can always declare the variable statistically but not practically significant.

It is natural to ask if rotation tests extend to nonlinear models. Our method is strongly geared to linear models because of the way we construct our rotations, so we see no straightforward extension.

Acknowledgments

We thank Stuart Kim, Jacob Zahn, and four anonymous referees for their comments. This work was supported by the NSF under grants DMS-0906056 and DMS-0604939.

References

J. Baik and J.W. Silverstein. Eigenvalues of large sample covariance matrices of spiked population models. *Journal of Multivariate Analysis*, 97(6):1382–1408, 2006.

J. Crossa and P. L. Cornelius. Linear-bilinear models for the analysis of genotype-environment interaction data. In M. S. Kang, editor, *Quantitative Genetics, Genomics and Plant Breeding in the 21st Century, an International Symposium*, pages 305–322, Wallingford UK, 2002. CAB International.

C. T. dos S Dias and W. J. Krzanowski. Model Selection and Cross Validation in Additive Main Effect and Multiplicative Interaction Models. *Crop Science*, 43(3):865–873, 2003.

B. Efron. Are a set of microarrays independent of each other? *Annals of Applied Statistics*, 3(3): 922–942, 2009.

N. El Karoui. Tracy-Widom limit for the largest eigenvalue of a large class of complex Wishart matrices. *Annals of Probability*, 35(2):663–714, 2007.

J. H. Friedman. Exploratory projection pursuit. *Journal of the American Statistical Association*, 82: 249–266, 1987.

K. R. Gabriel. Least squares approximation of matrices by additive and multiplicative models. *Journal of the Royal Statistical Society, B*, 40(2):186–196, 1978.

H. F. Gollob. A statistical model which combines features of factor analytic and analysis of variance techniques. *Psychometrika*, 33:73–115, 1968.

R. M. Heiberger. Algorithm AS 127: Generation of random orthogonal matrices. *Applied Statistics*, 27(2):199–206, 1978.

A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. J. Wiley New York, 2001.

Ø. Langsrud. Rotation tests. *Statistics and Computing*, 15:53–60, 2005.

E.L. Lehmann and J.P. Romano. *Testing Statistical Hypotheses*. Springer, 2005.

J. Mandel. A new analysis of variance model for non-additive data. *Technometrics*, 13:1–18, 1971.

B. Nadler. personal communication. Discussions at AIM workshop on Random Matrices and Higher Dimensional Inference, April 2007.

A. Onatski. Asymptotics of the principal components estimator of large factor models with weak factors and i.i.d. Gaussian noise. Available at http://www.columbia.edu/˜ao2027/inference45.pdf, August 2007.

A. B. Owen. Variance of the number of false discoveries. *Journal of the Royal Statistical Society, Series B*, 67:411–426, 2005.

A. B. Owen and P. O. Perry. Bi-cross-validation of the SVD and the non-negative matrix factorization. *Annals of applied statistics*, 3(2):564–594, 2009.

D. Paul. Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. *Statistica Sinica*, 17(4):1617–1642, 2007a.

D. Paul. Spiked covariance: some extensions. Unpublished technical note, June 2007b.

N.R. Rao and A. Edelman. Sample eigenvalue based detection of high dimensional signals in white noise using relatively few samples. *IEEE Transactions on Signal Processing*, 56(7 Part 1):2625–2638, 2008.

R. D. Snee. Nonadditivity in a Two-Way Classification: Is It Interaction or Nonhomogeneous Variance? *Journal of the American Statistical Association*, 77(379):515–519, 1982.

L. K. Southworth, S. K. Kim, and A. B. Owen. Properties of balanced permutations. *Journal of Computational Biology*, 16(4):625–638, 2009.

R. W. M. Wedderburn. Random rotations and multivariate normal simulation. Technical report, Rothamsted Experimental Station, 1975.

J.M. Zahn, S. Poosala, A.B. Owen, D.K. Ingram, A. Lustig, A. Carter, A.T. Weeraratna, D.D. Taub, M. Gorospe, K. Mazan-Mamczarz, et al. AGEMAP: A Gene Expression Database for Aging in Mice. *PLoS Genet*, 3(11):e201, 2007.

Journal of Machine Learning Research 11 (2010) 625-660 Submitted 8/09; Published 2/10

Why Does Unsupervised Pre-training Help Deep Learning?

Dumitru Erhan* DUMITRU.ERHAN@UMONTREAL.CA
Yoshua Bengio YOSHUA.BENGIO@UMONTREAL.CA
Aaron Courville AARON.COURVILLE@UMONTREAL.CA
Pierre-Antoine Manzagol PIERRE-ANTOINE.MANZAGOL@UMONTREAL.CA
Pascal Vincent PASCAL.VINCENT@UMONTREAL.CA
Département d'informatique et de recherche opérationnelle
Université de Montréal
2920, chemin de la Tour
Montréal, Québec, H3T 1J8, Canada

Samy Bengio BENGIO@GOOGLE.COM
Google Research
1600 Amphitheatre Parkway
Mountain View, CA, 94043, USA

Editor: Léon Bottou

Abstract

Much recent research has been devoted to learning algorithms for deep architectures such as Deep Belief Networks and stacks of auto-encoder variants, with impressive results obtained in several areas, mostly on vision and language data sets. The best results obtained on supervised learning tasks involve an unsupervised learning component, usually in an unsupervised pre-training phase. Even though these new algorithms have enabled training deep models, many questions remain as to the nature of this difficult learning problem. The main question investigated here is the following: how does unsupervised pre-training work? Answering this questions is important if learning in deep architectures is to be further improved. We propose several explanatory hypotheses and test them through extensive simulations. We empirically show the influence of pre-training with respect to architecture depth, model capacity, and number of training examples. The experiments confirm and clarify the advantage of unsupervised pre-training. The results suggest that unsupervised pre-training guides the learning towards basins of attraction of minima that support better generalization from the training data set; the evidence from these results supports a regularization explanation for the effect of pre-training.

Keywords: deep architectures, unsupervised pre-training, deep belief networks, stacked denoising auto-encoders, non-convex optimization

1. Introduction

Deep learning methods aim at learning feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower level features. They include learning methods for a wide array of *deep architectures* (Bengio, 2009 provides a survey), including neural networks with many hidden layers (Bengio et al., 2007; Ranzato et al., 2007; Vincent et al., 2008; Collobert and Weston, 2008) and graphical models with many levels of hidden variables (Hinton et al., 2006),

∗. Part of this work was done while Dumitru Erhan was at Google Research.

among others (Zhu et al., 2009; Weston et al., 2008). Theoretical results (Yao, 1985; Håstad, 1986; Håstad and Goldmann, 1991; Bengio et al., 2006), reviewed and discussed by Bengio and LeCun (2007), suggest that in order to learn the kind of complicated functions that can represent high-level abstractions (e.g., in vision, language, and other AI-level tasks), one may need *deep architectures*. The recent surge in experimental work in the field seems to support this notion, accumulating evidence that in challenging AI-related tasks—such as computer vision (Bengio et al., 2007; Ranzato et al., 2007; Larochelle et al., 2007; Ranzato et al., 2008; Lee et al., 2009; Mobahi et al., 2009; Osindero and Hinton, 2008), natural language processing (NLP) (Collobert and Weston, 2008; Weston et al., 2008), robotics (Hadsell et al., 2008), or information retrieval (Salakhutdinov and Hinton, 2007; Salakhutdinov et al., 2007)—deep learning methods significantly out-perform comparable but shallow competitors, and often match or beat the state-of-the-art.

These recent demonstrations of the potential of deep learning algorithms were achieved despite the serious challenge of training models with many layers of adaptive parameters. In virtually all instances of deep learning, the objective function is a highly non-convex function of the parameters, with the potential for many distinct *local minima* in the model parameter space. The principal difficulty is that not all of these minima provide equivalent generalization errors and, we suggest, that for deep architectures, the standard training schemes (based on random initialization) tend to place the parameters in regions of the parameters space that generalize poorly—as was frequently observed empirically but rarely reported (Bengio and LeCun, 2007).

The breakthrough to effective training strategies for deep architectures came in 2006 with the algorithms for training deep belief networks (DBN) (Hinton et al., 2006) and stacked auto-encoders (Ranzato et al., 2007; Bengio et al., 2007), which are all based on a similar approach: greedy layer-wise unsupervised pre-training followed by supervised fine-tuning. Each layer is pre-trained with an unsupervised learning algorithm, learning a nonlinear transformation of its input (the output of the previous layer) that captures the main variations in its input. This unsupervised pre-training sets the stage for a final training phase where the deep architecture is fine-tuned with respect to a supervised training criterion with gradient-based optimization. While the improvement in performance of trained deep models offered by the pre-training strategy is impressive, little is understood about the mechanisms underlying this success.

The objective of this paper is to explore, through extensive experimentation, how unsupervised pre-training works to render learning deep architectures more effective and why they appear to work so much better than traditional neural network training methods. There are a few reasonable hypotheses why unsupervised pre-training might work. One possibility is that unsupervised pre-training acts as a kind of network pre-conditioner, putting the parameter values in the appropriate range for further supervised training. Another possibility, suggested by Bengio et al. (2007), is that unsupervised pre-training initializes the model to a point in parameter space that somehow renders the optimization process more effective, in the sense of achieving a lower minimum of the empirical cost function.

Here, we argue that our experiments support a view of unsupervised pre-training as an unusual form of *regularization*: minimizing variance and introducing bias towards configurations of the parameter space that are useful for unsupervised learning. This perspective places unsupervised pre-training well within the family of recently developed semi-supervised methods. The unsupervised pre-training approach is, however, unique among semi-supervised training strategies in that it acts by defining a particular initialization point for standard supervised training rather than either modifying the supervised objective function (Barron, 1991) or explicitly imposing constraints on the parame-

ters throughout training (Lasserre et al., 2006). This type of initialization-as-regularization strategy has precedence in the neural networks literature, in the shape of the early stopping idea (Sjöberg and Ljung, 1995; Amari et al., 1997), and in the Hidden Markov Models (HMM) community (Bahl et al., 1986; Povey and Woodland, 2002) where it was found that first training an HMM as a generative model was essential (as an initialization step) before fine-tuning it discriminatively. We suggest that, in the highly non-convex situation of training a deep architecture, defining a particular initialization point *implicitly* imposes constraints on the parameters in that it specifies which minima (out of a very large number of possible minima) of the cost function are allowed. In this way, it may be possible to think of unsupervised pre-training as being related to the approach of Lasserre et al. (2006).

Another important and distinct property of the unsupervised pre-training strategy is that in the standard situation of training using stochastic gradient descent, the beneficial generalization effects due to pre-training do not appear to diminish as the number of labeled examples grows very large. We argue that this is a consequence of the combination of the non-convexity (multi-modality) of the objective function and the dependency of the stochastic gradient descent method on example ordering. We find that early changes in the parameters have a greater impact on the final region (basin of attraction of the descent procedure) in which the learner ends up. In particular, unsupervised pre-training sets the parameter in a region from which better basins of attraction can be reached, in terms of generalization. Hence, although unsupervised pre-training is a regularizer, it can have a positive effect on the training objective when the number of training examples is large.

As previously stated, this paper is concerned with an experimental assessment of the various competing hypotheses regarding the role of unsupervised pre-training in the recent success of deep learning methods. To this end, we present a series of experiments design to pit these hypotheses against one another in an attempt to resolve some of the mystery surrounding the effectiveness of unsupervised pre-training.

In the first set of experiments (in Section 6), we establish the effect of unsupervised pre-training on improving the generalization error of trained deep architectures. In this section we also exploit dimensionality reduction techniques to illustrate how unsupervised pre-training affects the location of minima in parameter space.

In the second set of experiments (in Section 7), we directly compare the two alternative hypotheses (pre-training as a pre-conditioner; and pre-training as an optimization scheme) against the hypothesis that unsupervised pre-training is a regularization strategy. In the final set of experiments, (in Section 8), we explore the role of unsupervised pre-training in the online learning setting, where the number of available training examples grows very large. In these experiments, we test key aspects of our hypothesis relating to the topology of the cost function and the role of unsupervised pre-training in manipulating the region of parameter space from which supervised training is initiated.

Before delving into the experiments, we begin with a more in-depth view of the challenges in training deep architectures and how we believe unsupervised pre-training works towards overcoming these challenges.

2. The Challenges of Deep Learning

In this section, we present a perspective on why standard training of deep models through gradient backpropagation appears to be so difficult. First, it is important to establish what we mean in stating that training is difficult.

We believe the central challenge in training deep architectures is dealing with the strong dependencies that exist during training between the parameters across layers. One way to conceive the difficulty of the problem is that we must simultaneously:

1. adapt the lower layers in order to provide adequate input to the final (end of training) setting of the upper layers

2. adapt the upper layers to make good use of the final (end of training) setting of the lower layers.

The second problem is easy on its own (i.e., when the final setting of the other layers is known). It is not clear how difficult is the first one, and we conjecture that a particular difficulty arises when both sets of layers must be learned jointly, as the gradient of the objective function is limited to a local measure given the current setting of other parameters. Furthermore, because with enough capacity the top two layers can easily overfit the training set, training error does not necessarily reveal the difficulty in optimizing the lower layers. As shown in our experiments here, the standard training schemes tend to place the parameters in regions of the parameters space that generalize poorly.

A separate but related issue appears if we focus our consideration of traditional training methods for deep architectures on stochastic gradient descent. A sequence of examples along with an online gradient descent procedure defines a trajectory in parameter space, which converges in some sense (the error does not improve anymore, maybe because we are near a local minimum). The hypothesis is that small perturbations of that trajectory (either by initialization or by changes in which examples are seen when) have more effect early on. Early in the process of following the stochastic gradient, changes in the weights tend to increase their magnitude and, consequently, the amount of non-linearity of the network increases. As this happens, the set of regions accessible by stochastic gradient descent on samples of the training distribution becomes smaller. Early on in training small perturbations allow the model parameters to switch from one basin to a nearby one, whereas later on (typically with larger parameter values), it is unlikely to "escape" from such a basin of attraction. Hence the early examples can have a larger influence and, in practice, trap the model parameters in particular regions of parameter space that correspond to the specific and arbitrary ordering of the training examples.[1] An important consequence of this phenomenon is that even in the presence of a very large (effectively infinite) amounts of supervised data, stochastic gradient descent is subject to a degree of *overfitting* to the training data presented early in the training process. In that sense, unsupervised pre-training interacts intimately with the optimization process, and when the number of training examples becomes large, its positive effect is seen not only on generalization error but also on training error.

1. This process seems similar to the "critical period" phenomena observed in neuroscience and psychology (Bornstein, 1987).

3. Unsupervised Pre-training Acts as a Regularizer

As stated in the introduction, we believe that greedy layer-wise unsupervised pre-training overcomes the challenges of deep learning by introducing a useful prior to the *supervised fine-tuning* training procedure. We claim that the regularization effect is a consequence of the pre-training procedure establishing an initialization point of the fine-tuning procedure inside a region of parameter space in which the parameters are henceforth restricted. The parameters are restricted to a relatively small volume of parameter space that is delineated by the boundary of the *local basin of attraction* of the supervised fine-tuning cost function.

The pre-training procedure increases the magnitude of the weights and in standard deep models, with a sigmoidal nonlinearity, this has the effect of rendering both the function more nonlinear and the cost function locally more complicated with more topological features such as peaks, troughs and plateaus. The existence of these topological features renders the parameter space locally more difficult to travel significant distances via a gradient descent procedure. This is the core of the restrictive property imposed by the pre-training procedure and hence the basis of its regularizing properties.

But unsupervised pre-training restricts the parameters to particular regions: those that correspond to capturing structure in the input distribution $P(X)$. To simply state that unsupervised pre-training is a regularization strategy somewhat undermines the significance of its effectiveness. Not all regularizers are created equal and, in comparison to standard regularization schemes such as L_1 and L_2 parameter penalization, unsupervised pre-training is dramatically effective. We believe the credit for its success can be attributed to the unsupervised training criteria optimized during unsupervised pre-training.

During each phase of the greedy unsupervised training strategy, layers are trained to represent the dominant factors of variation extant in the data. This has the effect of leveraging knowledge of X to form, at each layer, a representation of X consisting of statistically reliable features of X that can then be used to predict the output (usually a class label) Y. This perspective places unsupervised pre-training well within the family of learning strategies collectively know as semi-supervised methods. As with other recent work demonstrating the effectiveness of semi-supervised methods in regularizing model parameters, we claim that the effectiveness of the unsupervised pre-training strategy is limited to the extent that learning $P(X)$ is helpful in learning $P(Y|X)$. Here, we find transformations of X—learned features—that are predictive of the main factors of variation in $P(X)$, and when the pre-training strategy is effective,[2] some of these learned features of X are also predictive of Y. In the context of deep learning, the greedy unsupervised strategy may also have a special function. To some degree it resolves the problem of simultaneously learning the parameters at all layers (mentioned in Section 2) by introducing a proxy criterion. This proxy criterion encourages significant factors of variation, present in the input data, to be represented in intermediate layers.

To clarify this line of reasoning, we can formalize the effect of unsupervised pre-training in inducing a prior distribution over the parameters. Let us assume that parameters are forced to be chosen in a bounded region $\mathcal{S} \subset \mathbb{R}^d$. Let $\mathcal{S}$ be split in regions $\{R_k\}$ that are the basins of attraction of descent procedures in the training error (note that $\{R_k\}$ depends on the training set, but the dependency decreases as the number of examples increases). We have $\cup_k R_k = \mathcal{S}$ and $R_i \cap R_j = \emptyset$ for $i \neq j$. Let $v_k = \int 1_{\theta \in R_k} d\theta$ be the volume associated with region R_k (where θ are our model's

2. Acting as a form of (data-dependent) "prior" on the parameters, as we are about to formalize.

parameters). Let r_k be the probability that a purely random initialization (according to our initialization procedure, which factorizes across parameters) lands in R_k, and let π_k be the probability that pre-training (following a random initialization) lands in R_k, that is, $\sum_k r_k = \sum_k \pi_k = 1$. We can now take into account the initialization procedure as a regularization term:

$$\text{regularizer} = -\log P(\theta).$$

For pre-trained models, the prior is

$$P_{\text{pre-training}}(\theta) = \sum_k 1_{\theta \in R_k} \pi_k / v_k.$$

For the models without unsupervised pre-training, the prior is

$$P_{\text{no-pre-training}}(\theta) = \sum_k 1_{\theta \in R_k} r_k / v_k.$$

One can verify that $P_{\text{pre-training}}(\theta \in R_k) = \pi_k$ and $P_{\text{no-pre-training}}(\theta \in R_k) = r_k$. When π_k is tiny, the penalty is high when $\theta \in R_k$, with unsupervised pre-training. The derivative of this regularizer is zero almost everywhere because we have chosen a uniform prior inside each region R_k. Hence, to take the regularizer into account, and having a generative model $P_{\text{pre-training}}(\theta)$ for θ (i.e., this is the unsupervised pre-training procedure), it is reasonable to sample an initial θ from it (knowing that from this point on the penalty will not increase during the iterative minimization of the training criterion), and this is exactly how the pre-trained models are obtained in our experiments.

Note that this formalization is just an illustration: it is there to simply show how one could conceptually think of an initialization point as a regularizer and should not be taken as a literal interpretation of how regularization is explicitly achieved, since we do not have an analytic formula for computing the π_k's and v_k's. Instead these are implicitly defined by the whole unsupervised pre-training procedure.

4. Previous Relevant Work

We start with an overview of the literature on semi-supervised learning (SSL), since the SSL framework is essentially the one in which we operate as well.

4.1 Related Semi-Supervised Methods

It has been recognized for some time that generative models are less prone to overfitting than discriminant ones (Ng and Jordan, 2002). Consider input variable X and target variable Y. Whereas a discriminant model focuses on $P(Y|X)$, a generative model focuses on $P(X,Y)$ (often parametrized as $P(X|Y)P(Y)$), that is, it also cares about getting $P(X)$ right, which can reduce the freedom of fitting the data when the ultimate goal is only to predict Y given X.

Exploiting information about $P(X)$ to improve generalization of a classifier has been the driving idea behind semi-supervised learning (Chapelle et al., 2006). For example, one can use unsupervised learning to map X into a representation (also called embedding) such that two examples $\mathbf{x}_1$ and $\mathbf{x}_2$ that belong to the same cluster (or are reachable through a short path going through neighboring examples in the training set) end up having nearby embeddings. One can then use supervised learning (e.g., a linear classifier) in that new space and achieve better generalization in many cases (Belkin

and Niyogi, 2002; Chapelle et al., 2003). A long-standing variant of this approach is the application of Principal Components Analysis as a pre-processing step before applying a classifier (on the projected data). In these models the data is first transformed in a new representation using unsupervised learning, and a supervised classifier is stacked on top, learning to map the data in this new representation into class predictions.

Instead of having separate unsupervised and supervised components in the model, one can consider models in which $P(X)$ (or $P(X,Y)$) and $P(Y|X)$ share parameters (or whose parameters are connected in some way), and one can trade-off the supervised criterion $-\log P(Y|X)$ with the unsupervised or generative one ($-\log P(X)$ or $-\log P(X,Y)$). It can then be seen that the generative criterion corresponds to a particular form of prior (Lasserre et al., 2006), namely that the structure of $P(X)$ is connected to the structure of $P(Y|X)$ in a way that is captured by the shared parametrization. By controlling how much of the generative criterion is included in the total criterion, one can find a better trade-off than with a purely generative or a purely discriminative training criterion (Lasserre et al., 2006; Larochelle and Bengio, 2008).

In the context of deep architectures, a very interesting application of these ideas involves adding an unsupervised embedding criterion at each layer (or only one intermediate layer) to a traditional supervised criterion (Weston et al., 2008). This has been shown to be a powerful semi-supervised learning strategy, and is an alternative to the kind of algorithms described and evaluated in this paper, which also combine unsupervised learning with supervised learning.

In the context of scarcity of labelled data (and abundance of unlabelled data), deep architectures have shown promise as well. Salakhutdinov and Hinton (2008) describe a method for learning the covariance matrix of a Gaussian Process, in which the usage of unlabelled examples for modeling $P(X)$ improves $P(Y|X)$ quite significantly. Note that such a result is to be expected: with few labelled samples, modeling $P(X)$ usually helps. Our results show that even in the context of *abundant labelled data*, unsupervised pre-training still has a pronounced positive effect on generalization: a somewhat surprising conclusion.

4.2 Early Stopping as a Form of Regularization

We stated that pre-training as initialization can be seen as restricting the optimization procedure to a relatively small volume of parameter space that corresponds to a local basin of attraction of the supervised cost function. Early stopping can be seen as having a similar effect, by constraining the optimization procedure to a region of the parameter space that is close to the initial configuration of parameters. With τ the number of training iterations and η the learning rate used in the update procedure, $\tau\eta$ can be seen as the reciprocal of a regularization parameter. Indeed, restricting either quantity restricts the area of parameter space reachable from the starting point. In the case of the optimization of a simple linear model (initialized at the origin) using a quadratic error function and simple gradient descent, early stopping will have a similar effect to traditional regularization.

Thus, in both pre-training and early stopping, the parameters of the supervised cost function are constrained to be close to their initial values.[3] A more formal treatment of early stopping as regularization is given by Sjöberg and Ljung (1995) and Amari et al. (1997). There is no equivalent treatment of pre-training, but this paper sheds some light on the effects of such initialization in the case of deep architectures.

3. In the case of pre-training the "initial values" of the parameters for the supervised phase are those that were obtained at the end of pre-training.

5. Experimental Setup and Methodology

In this section, we describe the setting in which we test the hypothesis introduced in Section 3 and previously proposed hypotheses. The section includes a description of the deep architectures used, the data sets and the details necessary to reproduce our results.

5.1 Models

All of the successful methods (Hinton et al., 2006; Hinton and Salakhutdinov, 2006; Bengio et al., 2007; Ranzato et al., 2007; Vincent et al., 2008; Weston et al., 2008; Ranzato et al., 2008; Lee et al., 2008) in the literature for training deep architectures have something in common: they rely on an unsupervised learning algorithm that provides a training signal at the level of a single layer. Most work in two main phases. In a first phase, *unsupervised pre-training*, all layers are initialized using this layer-wise unsupervised learning signal. In a second phase, *fine-tuning*, a global training criterion (a prediction error, using labels in the case of a supervised task) is minimized. In the algorithms initially proposed (Hinton et al., 2006; Bengio et al., 2007; Ranzato et al., 2007), the unsupervised pre-training is done in a greedy layer-wise fashion: at stage k, the k-th layer is trained (with respect to an unsupervised criterion) using as input the output of the previous layer, and while the previous layers are kept fixed.

We shall consider two deep architectures as representatives of two families of models encountered in the deep learning literature.

5.1.1 DEEP BELIEF NETWORKS

The first model is the Deep Belief Net (DBN) by Hinton et al. (2006), obtained by training and stacking several layers of Restricted Boltzmann Machines (RBM) in a greedy manner. Once this stack of RBMs is trained, it can be used to initialize a multi-layer neural network for classification.

An RBM with n hidden units is a Markov Random Field (MRF) for the joint distribution between hidden variables h_i and observed variables x_j such that $P(\mathbf{h}|\mathbf{x})$ and $P(\mathbf{x}|\mathbf{h})$ factorize, that is, $P(\mathbf{h}|\mathbf{x}) = \prod_i P(h_i|\mathbf{x})$ and $P(\mathbf{x}|\mathbf{h}) = \prod_j P(x_j|\mathbf{h})$. The sufficient statistics of the MRF are typically h_i, x_j and $h_i x_j$, which gives rise to the following joint distribution:

$$P(\mathbf{x},\mathbf{h}) \propto e^{\mathbf{h}'W\mathbf{x}+b'\mathbf{x}+c'\mathbf{h}}$$

with corresponding parameters $\theta = (W,b,c)$ (with $'$ denoting transpose, c_i associated with h_i, b_j with x_j, and W_{ij} with $h_i x_j$). If we restrict h_i and x_j to be binary units, it is straightforward to show that

$$\begin{aligned} P(\mathbf{x}|\mathbf{h}) &= \prod_j P(x_j|\mathbf{h}) \quad \text{with} \\ P(x_j = 1|\mathbf{h}) &= \text{sigmoid}(b_j + \sum_i W_{ij} h_i). \end{aligned}$$

where $\text{sigmoid}(\mathbf{a}) = 1/(1+\exp(-\mathbf{a}))$ (applied element-wise on a vector $\mathbf{a}$), and $P(\mathbf{h}|\mathbf{x})$ also has a similar form:

$$\begin{aligned} P(\mathbf{h}|\mathbf{x}) &= \prod_i P(h_i|\mathbf{x}) \quad \text{with} \\ P(h_i = 1|\mathbf{x}) &= \text{sigmoid}(c_i + \sum_j W_{ij} x_j). \end{aligned}$$

The RBM form can be generalized to other conditional distributions besides the binomial, including continuous variables. Welling et al. (2005) describe a generalization of RBM models to conditional distributions from the exponential family.

RBM models can be trained by approximate stochastic gradient descent. Although $P(\mathbf{x})$ is not tractable in an RBM, the Contrastive Divergence estimator (Hinton, 2002) is a good stochastic approximation of $\frac{\partial \log P(\mathbf{x})}{\partial \theta}$, in that it very often has the same sign (Bengio and Delalleau, 2009).

A DBN is a multi-layer generative model with layer variables h_0 (the input or visible layer), h_1, h_2, etc. The top two layers have a joint distribution which is an RBM, and $P(h_k|h_{k+1})$ are parametrized in the same way as for an RBM. Hence a 2-layer DBN is an RBM, and a stack of RBMs share parametrization with a corresponding DBN. The contrastive divergence update direction can be used to initialize each layer of a DBN as an RBM, as follows. Consider the first layer of the DBN trained as an RBM P_1 with hidden layer h_1 and visible layer v_1. We can train a second RBM P_2 that models (in its visible layer) the samples h_1 from $P_1(h_1|v_1)$ when v_1 is sampled from the training data set. It can be shown that this maximizes a lower bound on the log-likelihood of the DBN. The number of layers can be increased greedily, with the newly added top layer trained as an RBM to model the samples produced by chaining the posteriors $P(h_k|h_{k-1})$ of the lower layers (starting from h_0 from the training data set).

The parameters of a DBN or of a stack of RBMs also correspond to the parameters of a deterministic feed-forward multi-layer neural network. The i-th unit of the k-th layer of the neural network outputs $\hat{h}_{ki} = \text{sigmoid}(c_{ki} + \sum_j W_{kij}\hat{h}_{k-1,j})$, using the parameters c_k and W_k of the k-th layer of the DBN. Hence, once the stack of RBMs or the DBN is trained, one can use those parameters to initialize the first layers of a corresponding multi-layer neural network. One or more additional layers can be added to map the top-level features $\hat{h}_k$ to the predictions associated with a target variable (here the probabilities associated with each class in a classification task). Bengio (2009) provides more details on RBMs and DBNs, and a survey of related models and deep architectures.

5.1.2 STACKED DENOISING AUTO-ENCODERS

The second model, by Vincent et al. (2008), is the so-called Stacked Denoising Auto-Encoder (SDAE). It borrows the greedy principle from DBNs, but uses denoising auto-encoders as a building block for unsupervised modeling. An auto-encoder learns an encoder $h(\cdot)$ and a decoder $g(\cdot)$ whose composition approaches the identity for examples in the training set, that is, $g(h(\mathbf{x})) \approx \mathbf{x}$ for $\mathbf{x}$ in the training set.

Assuming that some constraint prevents $g(h(\cdot))$ from being the identity for arbitrary arguments, the auto-encoder has to capture statistical structure in the training set in order to minimize reconstruction error. However, with a high capacity code ($h(\mathbf{x})$ has too many dimensions), a regular auto-encoder could potentially learn a trivial encoding. Note that there is an intimate connection between minimizing reconstruction error for auto-encoders and contrastive divergence training for RBMs, as both can be shown to approximate a log-likelihood gradient (Bengio and Delalleau, 2009).

The *denoising auto-encoder* (Vincent et al., 2008; Seung, 1998; LeCun, 1987; Gallinari et al., 1987) is a stochastic variant of the ordinary auto-encoder with the distinctive property that even with a high capacity model, it cannot learn the identity mapping. A denoising autoencoder is explicitly trained to denoise a corrupted version of its input. Its training criterion can also be viewed as a variational lower bound on the likelihood of a specific generative model. It has been shown on an array of data sets to perform significantly better than ordinary auto-encoders and similarly or better

than RBMs when stacked into a deep supervised architecture (Vincent et al., 2008). Another way to prevent regular auto-encoders with more code units than inputs to learn the identity is to restrict the capacity of the representation by imposing sparsity on the code (Ranzato et al., 2007, 2008).

We now summarize the training algorithm of the Stacked Denoising Auto-Encoders. More details are given by Vincent et al. (2008). Each denoising auto-encoder operates on its inputs $\mathbf{x}$, either the raw inputs or the outputs of the previous layer. The denoising auto-encoder is trained to reconstruct $\mathbf{x}$ from a stochastically corrupted (noisy) transformation of it. The output of each denoising auto-encoder is the "code vector" $h(\mathbf{x})$, not to confuse with the reconstruction obtained by applying the decoder to that code vector. In our experiments $h(\mathbf{x}) = \text{sigmoid}(\mathbf{b} + W\mathbf{x})$ is an ordinary neural network layer, with hidden unit biases $\mathbf{b}$, and weight matrix W. Let $C(\mathbf{x})$ represent a stochastic corruption of $\mathbf{x}$. As done by Vincent et al. (2008), we set $C_i(\mathbf{x}) = x_i$ or 0, with a random subset (of a fixed size) selected for zeroing. We have also considered a salt and pepper noise, where we select a random subset of a fixed size and set $C_i(\mathbf{x}) = \text{Bernoulli}(0.5)$. The denoised "reconstruction" is obtained from the noisy input with $\hat{\mathbf{x}} = \text{sigmoid}(\mathbf{c} + W^T h(C(\mathbf{x})))$, using biases $\mathbf{c}$ and the transpose of the feed-forward weights W. In the experiments on images, both the raw input x_i and its reconstruction $\hat{x}_i$ for a particular pixel i can be interpreted as a Bernoulli probability for that pixel: the probability of painting the pixel as black at that location. We denote $\text{CE}(\mathbf{x}||\hat{\mathbf{x}}) = \sum_i \text{CE}(x_i||\hat{x}_i)$ the sum of the component-wise cross-entropy between the Bernoulli probability distributions associated with each element of $\mathbf{x}$ and its reconstruction probabilities $\hat{\mathbf{x}}$: $\text{CE}(\mathbf{x}||\hat{\mathbf{x}}) = -\sum_i (x_i \log \hat{x}_i + (1 - x_i) \log (1 - \hat{x}_i))$. The Bernoulli model only makes sense when the input components and their reconstruction are in $[0, 1]$; another option is to use a Gaussian model, which corresponds to a Mean Squared Error (MSE) criterion.

With either DBN or SDAE, an output logistic regression layer is added after unsupervised training. This layer uses softmax (multinomial logistic regression) units to estimate $P(\text{class}|\mathbf{x}) = \text{softmax}_{\text{class}}(\mathbf{a})$, where a_i is a linear combination of outputs from the top hidden layer. The whole network is then trained as usual for multi-layer perceptrons, to minimize the output (negative log-likelihood) prediction error.

5.2 Data Sets

We experimented on three data sets, with the motivation that our experiments would help understand previously presented results with deep architectures, which were mostly with the MNIST data set and variations (Hinton et al., 2006; Bengio et al., 2007; Ranzato et al., 2007; Larochelle et al., 2007; Vincent et al., 2008):

MNIST the digit classification data set by LeCun et al. (1998), containing 60,000 training and 10,000 testing examples of 28x28 handwritten digits in gray-scale.

InfiniteMNIST a data set by Loosli et al. (2007), which is an extension of `MNIST` from which one can obtain a quasi-infinite number of examples. The samples are obtained by performing random elastic deformations of the original `MNIST` digits. In this data set, there is only one set of examples, and the models will be compared by their (online) performance on it.

Shapeset is a synthetic data set with a controlled range of geometric invariances. The underlying task is binary classification of 10×10 images of triangles and squares. The examples show

images of shapes with many variations, such as size, orientation and gray-level. The data set is composed of 50000 training, 10000 validation and 10000 test images.[4]

5.3 Setup

The models used are

1. Deep Belief Networks containing Bernoulli RBM layers,
2. Stacked Denoising Auto-Encoders with Bernoulli input units, and
3. standard feed-forward multi-layer neural networks,

each with 1–5 hidden layers. Each hidden layer contains the same number of hidden units, which is a hyperparameter. The other hyperparameters are the unsupervised and supervised learning rates, the L_2 penalty / weight decay,[5] and the fraction of stochastically corrupted inputs (for the SDAE). For `MNIST`, the number of supervised and unsupervised passes through the data (epochs) is 50 and 50 per layer, respectively. With `InfiniteMNIST`, we perform 2.5 million unsupervised updates followed by 7.5 million supervised updates.[6] The standard feed-forward networks are trained using 10 million supervised updates. For `MNIST`, model selection is done by choosing the hyperparameters that optimize the supervised (classification) error on the validation set. For `InfiniteMNIST`, we use the average online error over the last million examples for hyperparameter selection. In all cases, purely stochastic gradient updates are applied.

The experiments involve the training of deep architectures with a variable number of layers with and without unsupervised pre-training. For a given layer, weights are initialized using random samples from uniform$[-1/\sqrt{k}, 1/\sqrt{k}]$, where k is the number of connections that a unit receives from the previous layer (the fan-in). Either supervised gradient descent or unsupervised pre-training follows.

In most cases (for `MNIST`), we first launched a number of experiments using a cross-product of hyperparameter values[7] applied to 10 different random initialization seeds. We then selected the hyperparameter sets giving the best validation error for each combination of model (with or without pre-training), number of layers, and number of training iterations. Using these hyper-parameters, we launched experiments using an additional 400 initialization seeds. For `InfiniteMNIST`, only one seed is considered (an arbitrarily chosen value).

In the discussions below we sometimes use the word **apparent local minimum** to mean the solution obtained after training, when no further noticeable progress seems achievable by stochastic gradient descent. It is possible that these are not really near a true local minimum (there could be a tiny ravine towards significant improvement, not accessible by gradient descent), but it is clear that these end-points represent regions where gradient descent is stuck. Note also that when we write of number of layers it is to be understood as the number of *hidden* layers in the network.

4. The data set can be downloaded from `http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/ShapesetDataForJMLR`.
5. A penalizing term $\lambda||\theta||_2^2$ is added to the supervised objective, where θ are the weights of the network, and λ is a hyper-parameter modulating the strength of the penalty.
6. The number of examples was chosen to be as large as possible, while still allowing for the exploration a variety of hyper-parameters.
7. Number of hidden units $\in \{400, 800, 1200\}$; learning rate $\in \{0.1, 0.05, 0.02, 0.01, 0.005\}$; ℓ_2 penalty coefficient $\lambda \in \{10^{-4}, 10^{-5}, 10^{-6}, 0\}$; pre-training learning rate $\in \{0.01, 0.005, 0.002, 0.001, 0.0005\}$; corruption probability $\in \{0.0, 0.1, 0.25, 0.4\}$; tied weights $\in \{\text{yes}, \text{no}\}$.

6. The Effect of Unsupervised Pre-training

We start by a presentation of large-scale simulations that were intended to confirm some of the previously published results about deep architectures. In the process of analyzing them, we start making connections to our hypotheses and motivate the experiments that follow.

6.1 Better Generalization

When choosing the number of units per layer, the learning rate and the number of training iterations to optimize classification error on the validation set, unsupervised pre-training gives substantially lower test classification error than no pre-training, for the same depth or for smaller depth on various vision data sets (Ranzato et al., 2007; Bengio et al., 2007; Larochelle et al., 2009, 2007; Vincent et al., 2008) no larger than the MNIST digit data set (experiments reported from 10,000 to 50,000 training examples).

Such work was performed with only one or a handful of different random initialization seeds, so one of the goals of this study was to ascertain the effect of the random seed used when initializing ordinary neural networks (deep or shallow) and the pre-training procedure. For this purpose, between 50 and 400 different seeds were used to obtain the graphics on MNIST.

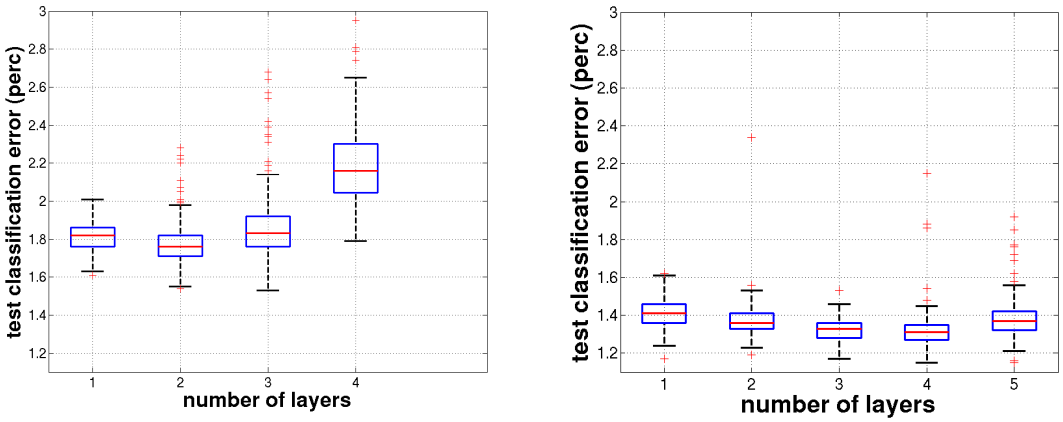


Figure 1: Effect of depth on performance for a model trained (**left**) without unsupervised pre-training and (**right**) with unsupervised pre-training, for 1 to 5 hidden layers (networks with 5 layers failed to converge to a solution, without the use of unsupervised pre-training). Experiments on MNIST. Box plots show the distribution of errors associated with 400 different initialization seeds (top and bottom quartiles in box, plus outliers beyond top and bottom quartiles). Other hyperparameters are optimized away (on the validation set). *Increasing depth seems to increase the probability of finding poor apparent local minima*.

Figure 1 shows the resulting distribution of test classification error, obtained with and without pre-training, as we increase the depth of the network. Figure 2 shows these distributions as histograms in the case of 1 and 4 layers. As can be seen in Figure 1, unsupervised pre-training allows

classification error to go down steadily as we move from 1 to 4 hidden layers, whereas without pre-training the error goes up after 2 hidden layers. It should also be noted that we were unable to effectively train 5-layer models without use of unsupervised pre-training. Not only is the error obtained on average with unsupervised pre-training systematically lower than without the pre-training, it appears also more robust to the random initialization. With unsupervised pre-training the variance stays at about the same level up to 4 hidden layers, with the number of bad outliers growing slowly.

Contrast this with the case without pre-training: the variance and number of bad outliers grows sharply as we increase the number of layers beyond 2. The gain obtained with unsupervised pre-training is more pronounced as we increase the number of layers, as is the gain in robustness to random initialization. This can be seen in Figure 2. The increase in error variance and mean for deeper architectures without pre-training suggests that **increasing depth increases the probability of finding poor apparent local minima** when starting from random initialization. It is also interesting to note the low variance and small spread of errors obtained with 400 seeds with unsupervised pre-training: it suggests that **unsupervised pre-training is robust with respect to the random initialization seed** (the one used to initialize parameters before pre-training).

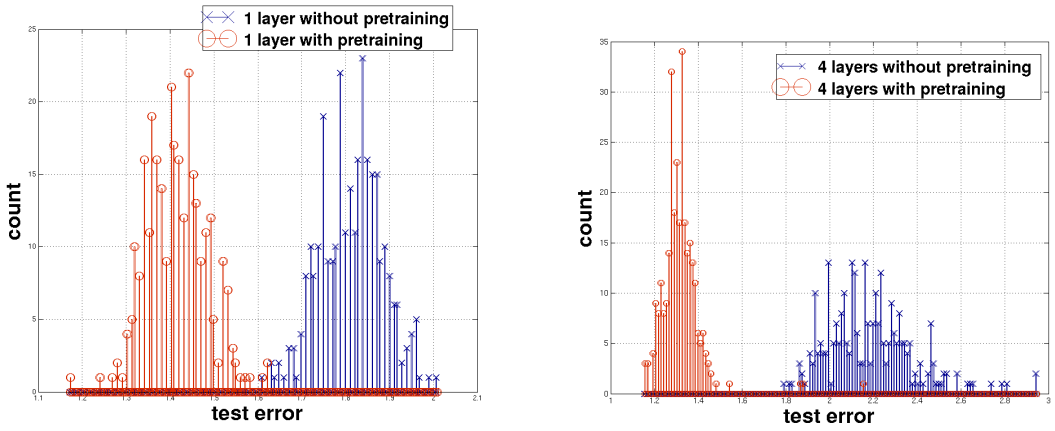


Figure 2: Histograms presenting the test errors obtained on `MNIST` using models trained with or without pre-training (400 different initializations each). **Left**: 1 hidden layer. **Right**: 4 hidden layers.

These experiments show that the variance of final test error with respect to initialization random seed is larger without pre-training, and this effect is magnified for deeper architectures. It should however be noted that there is a limit to the success of this technique: performance degrades for 5 layers on this problem.

6.2 Visualization of Features

Figure 3 shows the weights (called filters) of the first layer of the DBN before and after supervised fine-tuning. For visualizing what units do on the 2nd and 3rd layer, we used the activation maximization technique described by Erhan et al. (2009): to visualize what a unit responds most to, the method looks for the bounded input pattern that maximizes the activation of a given unit. This is an

optimization problem which is solved by performing gradient ascent in the space of the inputs, to find a local maximum of the activation function. Interestingly, nearly the same maximal activation input pattern is recovered from most random initializations of the input pattern.

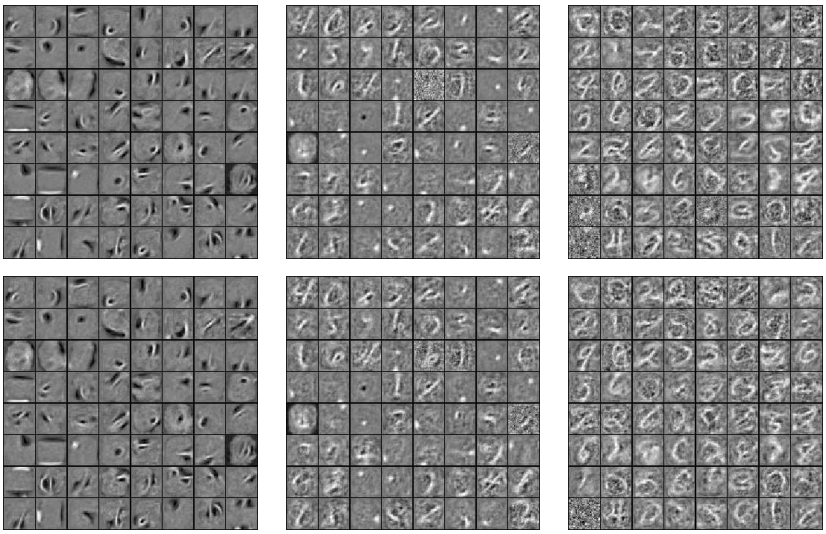


Figure 3: Visualization of filters learned by a DBN trained on `InfiniteMNIST`. The top figures contain a visualization of filters after pre-training, while the bottoms ones picture the same units after supervised fine-tuning; from left to right: units from the 1st, 2nd and 3rd layers, respectively.

For comparison, we have also visualized the filters of a network for 1–3 layers in which no pre-training was performed (Figure 4). While the first layer filters do seem to correspond to localized features, 2nd and 3rd layers are not as interpretable anymore. Qualitatively speaking, filters from the bottom row of Figure 3 and those from Figure 4 have little in common, which is an interesting conclusion in itself. In addition, there seems to be more interesting visual structures in the features learned in networks with unsupervised pre-training.

Several interesting conclusions can be drawn from Figure 3. First, supervised fine-tuning (after unsupervised pre-training), even with 7.5 million updates, does not change the weights in a significant way (at least visually): they seem stuck in a certain region of weight space, and the sign of weights does not change after fine-tuning (hence the same pattern is seen visually). Second, different layers change differently: the first layer changes least, while supervised training has more effect when performed on the 3rd layer. Such observations are consistent with the predictions made by our hypothesis: namely that the early dynamics of stochastic gradient descent, the dynamics induced by unsupervised pre-training, can "lock" the training in a region of the parameter space that is essentially inaccessible for models that are trained in a purely supervised way.

Finally, the features increase in complexity as we add more layers. First layer weights seem to encode basic stroke-like detectors, second layer weights seem to detect digit parts, while top layer weights detect entire digits. The features are more complicated as we add more layers, and displaying only one image for each "feature" does not do justice to the non-linear nature of that

feature. For example, it does not show the *set of patterns* on which the feature is highly active (or highly inactive).

While Figures 3–4 show only the filters obtained on `InfiniteMNIST`, the visualizations are similar when applied on `MNIST`. Likewise, the features obtained with SDAE result in qualitatively similar conclusions; Erhan et al. (2009) gives more details.

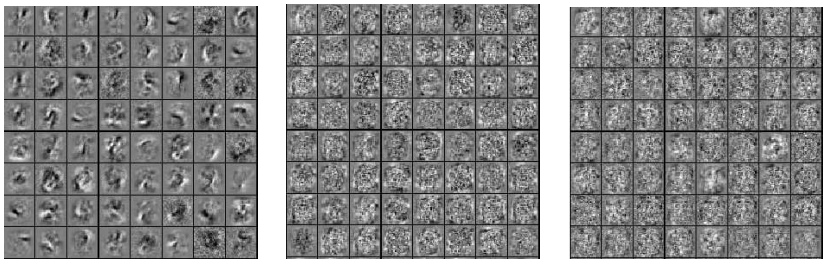


Figure 4: Visualization of filters learned by a network without pre-training, trained on `InfiniteMNIST`. The filters are shown after supervised training; from left to right: units from the 1st, 2nd and 3rd layers, respectively.

6.3 Visualization of Model Trajectories During Learning

Visualizing the learned features allows for a qualitative comparison of the training strategies for deep architectures. However it is not useful for investigating how these strategies are influenced by random initialization, as the features learned from multiple initializations look similar. If it was possible for us to visualize a variety of models at the same time, it would allow us to explore our hypothesis, and ascertain to what degree and how the set of pre-trained models (for different random seeds) is far from the set of models without pre-training. Do these two sets cover very different regions in parameter space? Are parameter trajectories getting stuck in many different apparent local minima?

Unfortunately, it is not possible to directly compare parameter values of two architectures, because many permutations of the same parameters give rise to the same model. However, one can take a functional approximation approach in which we compare the function (from input to output) represented by each network, rather than comparing the parameters. The function is the infinite ordered set of output values associated with all possible inputs, and it can be approximated with a finite number of inputs (preferably plausible ones). To visualize the trajectories followed during training, we use the following procedure. For a given model, we compute and concatenate all its outputs on the test set examples as one long vector summarizing where it stands in "function space". We get one such vector for each partially trained model (at each training iteration). This allows us to plot many learning trajectories, one for each initialization seed, with or without pre-training. Using a dimensionality reduction algorithm we then map these vectors to a two-dimensional space for visualization.[8] Figures 5 and 6 present the results using dimensionality reduction techniques that

8. Note that we can and do project the models with and without pre-training at the same time, so as to visualize them in the same space.

focus respectively on local[9] and global structure.[10] Each point is colored according to the training iteration, to help follow the trajectory movement.

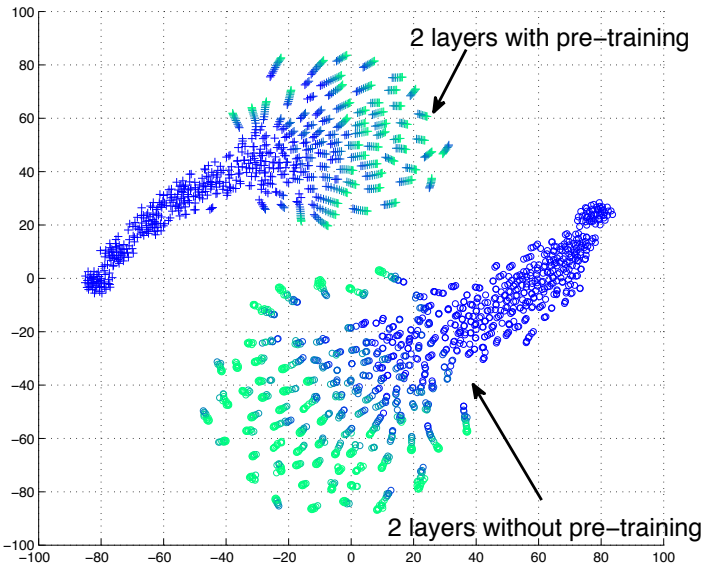


Figure 5: 2D visualizations with tSNE of the functions represented by 50 networks with and 50 networks without pre-training, as supervised training proceeds over MNIST. See Section 6.3 for an explanation. Color from dark blue to cyan and red indicates a progression in training iterations (training is longer without pre-training). The plot shows models with 2 hidden layers but results are similar with other depths.

What seems to come out of these visualizations is the following:

1. The pre-trained and not pre-trained models start and *stay* in different regions of function space.

2. From the visualization focusing on local structure (Figure 5) we see that all trajectories of a given type (with pre-training or without) initially move together. However, at some point (after about 7 epochs) the different trajectories (corresponding to different random seeds) diverge (slowing down into elongated jets) and never get back close to each other (this is more true for trajectories of networks without pre-training). This suggests that each trajectory moves into a different apparent local minimum.[11]

9. t-Distributed Stochastic Neighbor Embedding, or tSNE, by van der Maaten and Hinton (2008), with the default parameters available in the public implementation: `http://ict.ewi.tudelft.nl/~lvandermaaten/t-SNE.html`.
10. Isomap by Tenenbaum et al. (2000), with one connected component.
11. One may wonder if the divergence points correspond to a turning point in terms of overfitting. As shall be seen in Figure 8, the test error does not improve much after the 7th epoch, which reinforces this hypothesis.

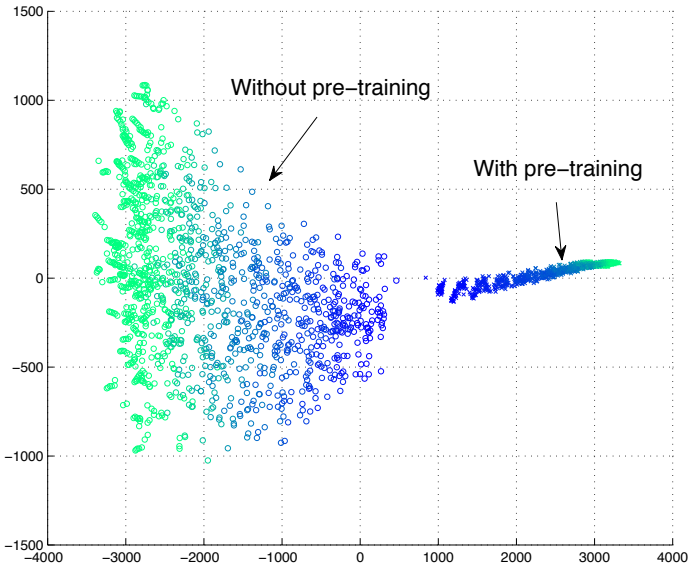


Figure 6: 2D visualization with ISOMAP of the functions represented by 50 networks with and 50 networks without pre-training, as supervised training proceeds over MNIST. See Section 6.3 for an explanation. Color from dark blue to cyan indicates a progression in training iterations (training is longer without pre-training). The plot shows models with 2 hidden layers but results are similar with other depths.

3. From the visualization focusing on global structure (Figure 6), we see the pre-trained models live in a disjoint and much smaller region of space than the not pre-trained models. In fact, from the standpoint of the functions found without pre-training, the pre-trained solutions look all the same, and their self-similarity increases (variance across seeds decreases) during training, while the opposite is observed without pre-training. This is consistent with the formalization of pre-training from Section 3, in which we described a theoretical justification for viewing unsupervised pre-training as a regularizer; there, the probabilities of pre-traininig parameters landing in a basin of attraction is small.

The visualizations of the training trajectories do seem to confirm our suspicions. It is difficult to guarantee that each trajectory actually does end up in a different local minimum (corresponding to a different function and not only to different parameters). However, all tests performed (visual inspection of trajectories in function space, but also estimation of second derivatives in the directions of all the estimated eigenvectors of the Jacobian not reported in details here) were consistent with that interpretation.

We have also analyzed models obtained at the end of training, to visualize the training criterion in the neighborhood of the parameter vector θ^* obtained. This is achieved by randomly sampling a direction v (from the stochastic gradient directions) and by plotting the training criterion around

θ^* in that direction, that is, at $\theta = \theta^* + \alpha v$, for $\alpha \in \{-2.5, -2.4, \ldots, -0.1, 0, 0.1, \ldots 2.4, 2.5\}$, and v normalized ($||v|| = 1$). This analysis is visualized in Figure 7. The error curves look close to quadratic and we seem to be near a local minimum in all directions investigated, as opposed to a saddle point or a plateau. A more definite answer could be given by computing the full Hessian eigenspectrum, which would be expensive. Figure 7 also suggests that the error landscape is a bit flatter in the case of unsupervised pre-training, and flatter for deeper architectures.

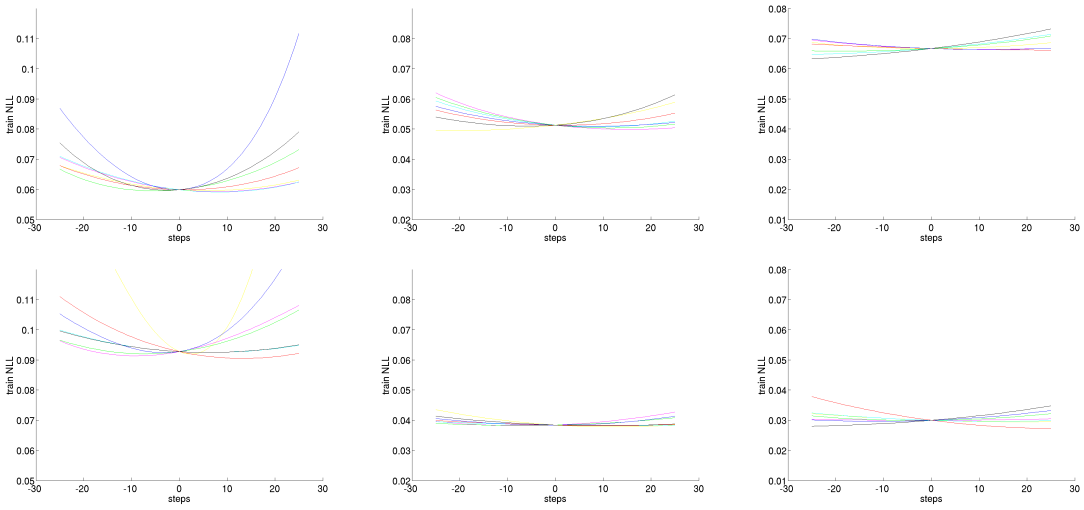


Figure 7: Training errors obtained on `Shapeset` when stepping in parameter space around a converged model in 7 random gradient directions (stepsize of 0.1). **Top**: no pre-training. **Bottom**: with unsupervised pre-training. **Left**: 1 hidden layer. **Middle**: 2 hidden layers. **Right**: 3 hidden layers. Compare also with Figure 8, where 1-layer networks with unsupervised pre-training obtain higher training errors.

6.4 Implications

The series of results presented so far show a picture that is consistent with our hypothesis. Better generalization that seems to be robust to random initializations is indeed achieved by pre-trained models, which indicates that unsupervised learning of $P(X)$ is helpful in learning $P(Y|X)$. The function space landscapes that we visualized point to the fact that there are many apparent local minima. The pre-trained models seem to end up in distinct regions of these error landscapes (and, implicitly, in different parts of the parameter space). This is both seen from the function space trajectories and from the fact that the visualizations of the learned features are qualitatively very different from those obtained by models without pre-training.

7. The Role of Unsupervised Pre-training

The observations so far in this paper confirm that starting the supervised optimization from pre-trained weights rather than from randomly initialized weights consistently yields better performing

classifiers on MNIST. To better understand where this advantage came from, it is important to realize that the *supervised objective being optimized is exactly the same in both cases*. The gradient-based optimization procedure is also the same. The only thing that differs is the starting point in parameter space: either picked at random or obtained after unsupervised pre-training (which also starts from a random initialization).

Deep architectures, since they are built from the composition of several layers of non-linearities, yield an error surface that is non-convex and hard to optimize, with the suspected presence of many local minima (as also shown by the above visualizations). A gradient-based optimization should thus end in the apparent local minimum of whatever *basin of attraction* we started from. From this perspective, the advantage of unsupervised pre-training could be that it puts us in a region of parameter space where basins of attraction run deeper than when picking starting parameters at random. The advantage would be due to a better **optimization**.

Now it might also be the case that unsupervised pre-training puts us in a region of parameter space in which training error is not necessarily better than when starting at random (or possibly worse), but which systematically yields better generalization (test error). Such behavior would be indicative of a **regularization** effect. Note that the two forms of explanation are *not necessarily mutually exclusive*.

Finally, a very simple explanation could be the most obvious one: namely the disparity in the magnitude of the weights (or more generally, the marginal distribution of the weights) at the start of the supervised training phase. We shall analyze (and rule out) this hypothesis first.

7.1 Experiment 1: Does Pre-training Provide a Better Conditioning Process for Supervised Learning?

Typically gradient descent training of the deep model is initialized with randomly assigned weights, small enough to be in the linear region of the parameter space (close to zero for most neural network and DBN models). It is reasonable to ask if the advantage imparted by having an initial unsupervised pre-training phase is simply due to the weights being larger and therefore somehow providing a better "conditioning" of the initial values for the optimization process; we wanted to rule out this possibility.

By conditioning, we mean the range and marginal distribution from which we draw initial weights. In other words, could we get the same performance advantage as unsupervised pre-training if we were still drawing the initial weights independently, but from a more suitable distribution than the uniform$[-1/\sqrt{k}, 1/\sqrt{k}]$? To verify this, we performed unsupervised pre-training, and computed marginal histograms for each layer's pre-trained weights and biases (one histogram per each layer's weights and biases). We then resampled new "initial" random weights and biases according to these histograms (independently for each parameter), and performed fine-tuning from there. The resulting parameters have the same marginal statistics as those obtained after unsupervised pre-training, but not the same joint distribution.

Two scenarios can be imagined. In the first, the initialization from marginals would lead to significantly better performance than the standard initialization (when no pre-training is used). This would mean that unsupervised pre-training does provide a better marginal conditioning of

the weights. In the second scenario, the marginals would lead to performance similar to or worse than that without pre-training.[12]

initialization.	Uniform	Histogram	Unsup.pre-tr.
1 layer	1.81 ± 0.07	1.94 ± 0.09	1.41 ± 0.07
2 layers	1.77 ± 0.10	1.69 ± 0.11	1.37 ± 0.09

Table 1: Effect of various initialization strategies on 1 and 2-layer architectures: independent uniform densities (one per parameter), independent densities from the marginals after unsupervised pre-training, or unsupervised pre-training (which samples the parameters in a highly dependent way so that they collaborate to make up good denoising auto-encoders.) Experiments on `MNIST`, numbers are mean and standard deviation of test errors (across different initialization seeds).

What we observe in Table 1 seems to fall within the first scenario. However, while initializing the weights to match the marginal distributions at the end of pre-training appears to slightly improve the generalization error on MNIST for 2 hidden layers, the difference is not significant and it is far from fully accounting for the discrepancy between the pre-trained and non-pre-trained results.

This experiment constitutes evidence against the preconditioning hypothesis, but does not exclude either the optimization hypothesis or the regularization hypothesis.

7.2 Experiment 2: The Effect of Pre-training on Training Error

The optimization and regularization hypotheses diverge on their prediction on how unsupervised pre-training should affect the training error: the former predicts that unsupervised pre-training should result in a lower training error, while the latter predicts the opposite. To ascertain the influence of these two possible explanatory factors, we looked at the test cost (Negative Log Likelihood on test data) obtained as a function of the training cost, along the trajectory followed in parameter space by the optimization procedure. Figure 8 shows 400 of these curves started from a point in parameter space obtained from random initialization, that is, without pre-training (blue), and 400 started from pre-trained parameters (red).

The experiments were performed for networks with 1, 2 and 3 hidden layers. As can be seen in Figure 8, while for 1 hidden layer, unsupervised pre-training reaches lower training cost than no pre-training, hinting towards a better optimization, this is not necessarily the case for the deeper networks. The remarkable observation is rather that, *at a same training cost level, the pre-trained models systematically yield a lower test cost* than the randomly initialized ones. The advantage appears to be one of *better generalization rather than merely a better optimization procedure*.

This brings us to the following result: unsupervised pre-training appears to have a similar effect to that of a good regularizer or a good "prior" on the parameters, even though no explicit regularization term is apparent in the cost being optimized. As we stated in the hypothesis, it might be reasoned that restricting the possible starting points in parameter space to those that minimize the unsupervised pre-training criterion (as with the SDAE), does in effect restrict the set of possible

12. We observed that the distribution of weights after unsupervised pre-training is fat-tailed. It is conceivable that sampling from such a distribution in order to initialize a deep architecture might actually *hurt* the performance of a deep architecture (compared to random initialization from a uniform distribution).

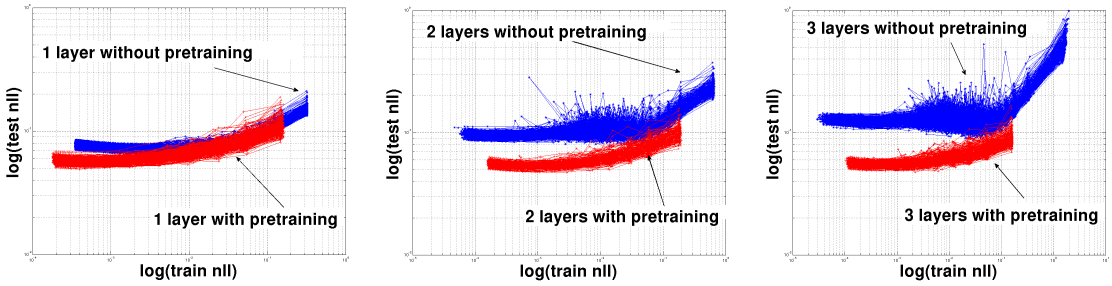


Figure 8: Evolution without pre-training (blue) and with pre-training (red) on MNIST of the log of the test NLL plotted against the log of the train NLL as training proceeds. Each of the 2×400 curves represents a different initialization. The errors are measured after each pass over the data. The rightmost points were measured after the first pass of gradient updates. Since training error tends to decrease during training, the trajectories run from right (high training error) to left (low training error). Trajectories moving up (as we go leftward) indicate a form of overfitting. All trajectories are plotted on top of each other.

final configurations for parameter values. Like regularizers in general, unsupervised pre-training (in this case, with denoising auto-encoders) might thus be seen as decreasing the variance and introducing a bias (towards parameter configurations suitable for performing denoising). Unlike ordinary regularizers, unsupervised pre-training does so in a data-dependent manner.

7.3 Experiment 3: The Influence of the Layer Size

Another signature characteristic of regularization is that the effectiveness of regularization increases as capacity (e.g., the number of hidden units) increases, effectively trading off one constraint on the model complexity for another. In this experiment we explore the relationship between the number of units per layer and the effectiveness of unsupervised pre-training. The hypothesis that unsupervised pre-training acts as a regularizer would suggest that we should see a trend of increasing effectiveness of unsupervised pre-training as the number of units per layer are increased.

We trained models on MNIST with and without pre-training using increasing layer sizes: 25, 50, 100, 200, 400, 800 units per layer. Results are shown in Figure 9. Qualitatively similar results were obtained on *Shapeset*. In the case of SDAE, we were expecting the denoising pre-training procedure to help classification performance most for large layers; this is because the denoising pre-training allows useful representations to be learned in the over-complete case, in which a layer is larger than its input (Vincent et al., 2008). What we observe is a more systematic effect: while unsupervised pre-training helps for larger layers and deeper networks, it also appears to hurt for too small networks.

Figure 9 also shows that DBNs behave qualitatively like SDAEs, in the sense that unsupervised pre-training architectures with smaller layers hurts performance. Experiments on InfiniteMNIST reveal results that are qualitatively the same. Such an experiment seemingly points to a re-verification of the regularization hypothesis. In this case, it would seem that unsupervised pre-training acts as an additional regularizer for both DBN and SDAE models—on top of the regularization provided by

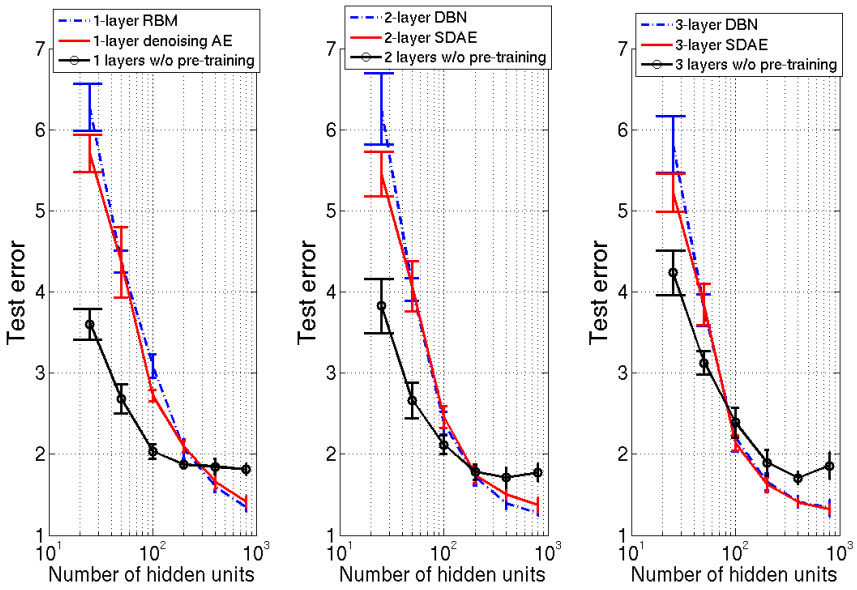


Figure 9: Effect of layer size on the changes brought by unsupervised pre-training, for networks with 1, 2 or 3 hidden layers. Experiments on MNIST. Error bars have a height of two standard deviations (over initialization seed). Pre-training hurts for smaller layer sizes and shallower networks, but it helps for all depths for larger networks.

the small size of the hidden layers. As the model size decreases from 800 hidden units, the generalization error increases, and it increases more with unsupervised pre-training presumably because of the extra regularization effect: small networks have a limited capacity already so further restricting it (or introducing an additional bias) can harm generalization. Such a result seems incompatible with a pure optimization effect. We also obtain the result that DBNs and SDAEs seem to have qualitatively similar effects as pre-training strategies.

The effect can be explained in terms of the role of unsupervised pre-training as promoting input transformations (in the hidden layers) that are useful at capturing the main variations in the input distribution $P(X)$. It may be that only a small subset of these variations are relevant for predicting the class label Y. When the hidden layers are small it is less likely that the transformations for predicting Y are included in the lot learned by unsupervised pre-training.

7.4 Experiment 4: Challenging the Optimization Hypothesis

Experiments 1–3 results are consistent with the regularization hypothesis and Experiments 2–3 would appear to directly support the regularization hypothesis over the alternative—that unsupervised pre-training aids in optimizing the deep model objective function.

In the literature there is some support for the optimization hypothesis. Bengio et al. (2007) constrained the top layer of a deep network to have 20 units and measured the training error of networks with and without pre-training. The idea was to prevent the networks from overfitting the training error simply with the top hidden layer, thus to make it clearer whether some optimization

effect (of the lower layers) was going on. The reported training and test errors were lower for pre-trained networks. One problem with the experimental paradigm used by Bengio et al. (2007) is their use of early stopping. This is problematic because, as previously mentioned, early stopping is itself a regularizer, and it can influence greatly the training error that is obtained. It is conceivable that if Bengio et al. (2007) had run the models to convergence, the results could have been different. We needed to verify this.

Figure 10 shows what happens without early stopping. The training error is still higher for pre-trained networks even though the generalization error is lower. This result now favors the regularization hypothesis against the optimization story. What may have happened is that early stopping prevented the networks without pre-training from moving too much towards their apparent local minimum.

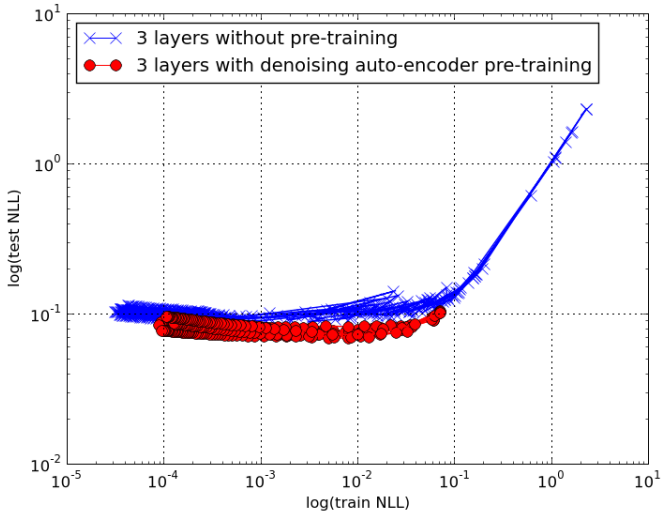


Figure 10: For `MNIST`, a plot of the log(train NLL) vs. log(test NLL) at each epoch of training. The top layer is constrained to 20 units.

7.5 Experiment 5: Comparing pre-training to L_1 and L_2 regularization

An alternative hypothesis would be that classical ways of regularizing could perhaps achieve the same effect as unsupervised pre-training. We investigated the effect of L_1 and L_2 regularization (i.e., adding a $||\theta||_1$ or $||\theta||_2^2$ term to the supervised objective function) in a network without pre-training. We found that while in the case of `MNIST` a small penalty can in principle help, the gain is nowhere near as large as it is with pre-training. For `InfiniteMNIST`, the optimal amount of L_1 and L_2 regularization is zero.[13]

13. Which is consistent with the classical view of regularization, in which its effect should diminish as we add more and more data.

This is not an entirely surprising finding: not all regularizers are created equal and these results are consistent with the literature on semi-supervised training that shows that unsupervised learning can be exploited as a particularly effective form of regularization.

7.6 Summary of Findings: Experiments 1-5

So far, the results obtained from the previous experiments point towards a pretty clear explanation of the effect of unsupervised pre-training: namely, that its effect is a regularization effect. We have seen that it is not simply sufficient to sample random weights with the same magnitude: the (data-dependent) unsupervised initialization is crucial. We have also observed that canonical regularizers (L_1/L_2 penalties on the weights) do not achieve the same level of performance.

The most compelling pieces of evidence in support of the regularization hypothesis are Figures 8 and 9. The alternative explanation—that unsupervised pre-training has an optimization effect—suggested by Bengio et al. (2007) doesn't seem to be supported by our experimental setup.

8. The Online Learning Setting

Our hypothesis included not only the statistical/phenomenological hypothesis that unsupervised pre-training acted as a regularizer, but also contains a mechanism for how such behavior arises both as a consequence of the dynamic nature of training—following a stochastic gradient through two phases of training and as a consequence of the non-convexity of the supervised objective function.

In our hypothesis, we posited that early examples induce changes in the magnitude of the weights that increase the amount of non-linearity of the network, which in turn decreases the number of regions accessible to the stochastic gradient descent procedure. This means that the early examples (be they pre-training examples or otherwise) determine the basin of attraction for the remainder of training; this also means that the early examples have a disproportionate influence on the configuration of parameters of the trained models.

One consequence to the hypothesized mechanism is that we would predict that in the online learning setting with unbounded or very large data sets, the behavior of unsupervised pre-training would diverge from the behavior of a canonical regularizer (L_1/L_2). This is because the effectiveness of a canonical regularizer **decreases** as the data set grows, whereas the effectiveness of unsupervised pre-training as a regularizer is **maintained** as the data set grows.

Note that stochastic gradient descent in online learning is a stochastic gradient descent optimization of the generalization error, so good online error in principle implies that we are optimizing well the generalization error. Indeed, each gradient $\frac{\partial L(x,y)}{\partial \theta}$ for example (x,y) (with $L(x,y)$ the supervised loss with input x and label y) sampled from the true generating distribution $P(x,y)$ is an unbiased Monte-Carlo estimator of the true gradient of generalization error, that is, $\sum_y \int_x \frac{\partial L(x,y)}{\partial \theta} P(x,y)dx$.

In this section we empirically challenge this aspect of the hypothesis and show that the evidence does indeed support our hypothesis over what is more typically expected from a regularizer.

8.1 Experiment 6: Effect of Pre-training with Very Large Data Sets

The results presented here are perhaps the most surprising findings of this paper. Figure 11 shows the online classification error (on the next block of examples, as a moving average) for 6 architectures that are trained on `InfiniteMNIST`: 1 and 3-layer DBNs, 1 and 3-layer SDAE, as well as 1 and 3-layer networks without pre-training.

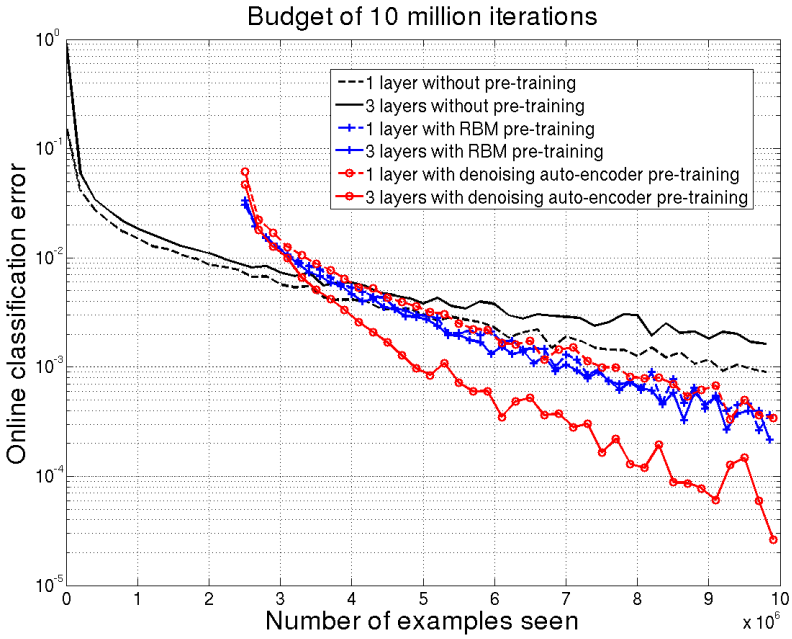


Figure 11: Comparison between 1 and 3-layer networks trained on `InfiniteMNIST`. Online classification error, computed as an average over a block of last 100,000 errors.

We can draw several observations from these experiments. First, 3-layer networks without pre-training are worse at generalization, compared to the 1-layer equivalent. This confirms the hypothesis that even in an online setting, optimization of deep networks is harder than shallow ones. Second, 3-layer SDAE models seem to generalize better than 3-layer DBNs. Finally and most importantly, the pre-training advantage does not vanish as the number of training examples increases, on the contrary.

Note that the number of hidden units of each model is a hyperparameter.[14] So theoretical results suggest that 1-layer networks without pre-training should in principle be able to represent the input distribution as capacity and data grow. Instead, without pre-training, the networks are not able to take advantage of the additional capacity, which again points towards the optimization explanation. It is clear, however, that **the starting point of the non-convex optimization matters**, even for networks that are seemingly "easier" to optimize (1-layer ones), which supports our hypothesis.

Another experiment that shows the effects of large-scale online stochastic non-convex optimization is shown in Figure 12. In the setting of `InfiniteMNIST`, we compute the error on the *training set*, in the same order that we presented the examples to the models. We observe several interesting results: first, note that both models are better at classifying more recently seen examples. This is a natural effect of stochastic gradient descent with a constant learning rate (which gives exponentially more weight to recent examples). Note also that examples at the beginning of training are essentially like test examples for both models, in terms of error. Finally, we observe that the pre-trained

14. This number was chosen individually for each model s.t. the error on the last 1 million examples is minimized. In practice, this meant 2000 units for 1-layer networks and 1000 units/layer for 3-layer networks.

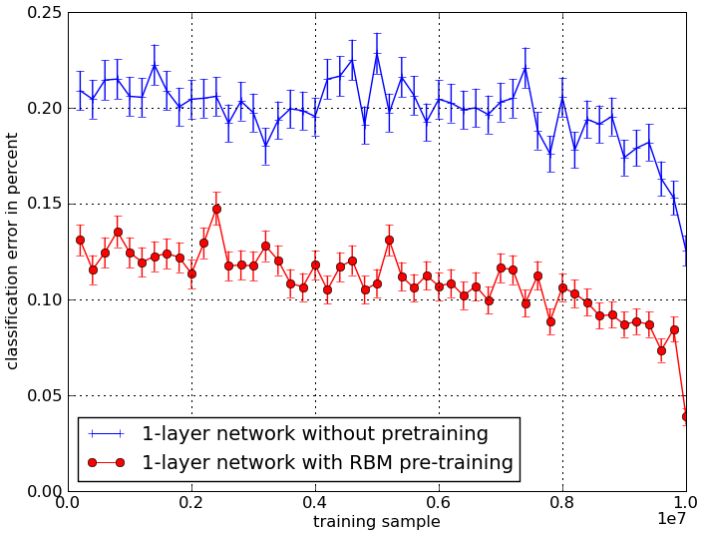


Figure 12: Error of 1-layer network with RBM pre-training and without, on the 10 million examples used for training it. The errors are calculated in the same order (from left to right, above) as the examples were presented during training. Each error bar corresponds to a block of consecutive training examples.

model is better across the board *on the training set*. This fits well with the optimization hypothesis, since it shows that unsupervised pre-training has an optimization effect.

What happens in this setting is that the training and generalization errors converge as the empirical distribution (defined by the training set) converges to the true data distribution. These results show that the effectiveness of unsupervised pre-training does not diminish with increasing data set sizes. This would be unexpected from a superficial understanding of unsupervised pre-training as a regularization method. However it is entirely consistent with our interpretation, stated in our hypothesis, of the role of unsupervised pre-training in the online setting with stochastic gradient descent training on a non-convex objective function.

8.2 Experiment 7: The Effect of Example Ordering

The hypothesized mechanism implies, due to the dynamics of learning—the increase in weight magnitude and non-linearity as training proceeds, as well as the dependence of the basin of attraction on early data—that, when training with stochastic gradient descent, we should see increased sensitivity to early examples. In the case of `InfiniteMNIST` we operate in an online stochastic optimization regime, where we try to find a local minimum of a highly non-convex objective function. It is then interesting to study to what extent the outcome of this optimization is influenced by the examples seen at different points during training, and whether the early examples have a stronger influence (which would not be the case with a convex objective).

To quantify the variance of the outcome with respect to training samples at different points during training, and to compare these variances for models with and without pre-training, we proceeded with the following experiment. Given a data set with 10 million examples, we vary (by resampling)

the first million examples (across 10 different random draws, sampling a different set of 1 million examples each time) and keep the other ones fixed. After training the (10) models, we measure the variance (across the 10 draws) of the *output* of the networks on a fixed test set (i.e., we measure the variance in function space). We then vary the next million examples in the same fashion, and so on, to see how much each of the ten parts of the training set influenced the final function.

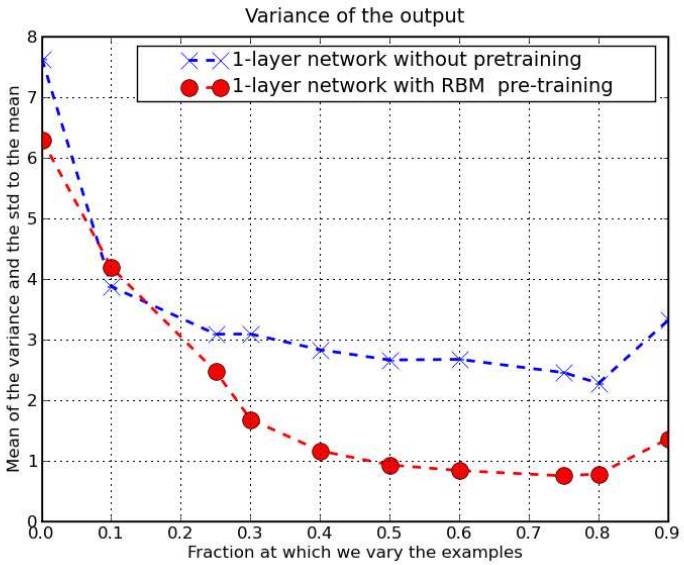


Figure 13: Variance of the output of a trained network with 1 layer. The variance is computed as a function of the point at which we vary the training samples. Note that the 0.25 mark corresponds to the start of pre-training.

Figure 13 shows the outcome of such an analysis. The samples at the beginning[15] do seem to influence the output of the networks more than the ones at the end. However, this variance is *lower* for the networks that have been pre-trained. In addition to that, one should note that the variance of pre-trained network at 0.25 (i.e., the variance of the output as a function of the first samples used for supervised training) is *lower* than the variance of the supervised network at 0.0. Such results imply that unsupervised pre-training can be seen as a sort of variance reduction technique, consistent with a regularization hypothesis. Finally, both networks are more influenced by the *last examples* used for optimization, which is simply due to the fact that we use stochastic gradient with a constant learning rate, where the most recent examples' gradient has a greater influence.

These results are consistent with what our hypothesis predicts: both the fact that early examples have greater influence (i.e., the variance is higher) and that pre-trained models seem to reduce this variance are in agreement with what we would have expected.

15. Which are *unsupervised* examples, for the red curve, until the 0.25 mark in Figure 13.

8.3 Experiment 8: Pre-training only k layers

From Figure 11 we can see that unsupervised pre-training makes quite a difference for 3 layers, on `InfiniteMNIST`. In Figure 14 we explore the link between depth and unsupervised pre-training in more detail. The setup is as follows: for both `MNIST` and `InfiniteMNIST` we pre-train only the bottom k layers and randomly initialize the top $n-k$ layers in the usual way. In this experiment, $n = 3$ and we vary k from 0 (which corresponds to a network with no pre-training) to $k = n$ (which corresponds to the normal pre-trained case).

For `MNIST`, we plot the log(train NLL) vs. log(test NLL) trajectories, where each point corresponds to a measurement after a certain number of epochs. The trajectories go roughly from the right to left and from top to bottom, corresponding to the lowering of the training and test errors. We can also see that models overfit from a certain point onwards.

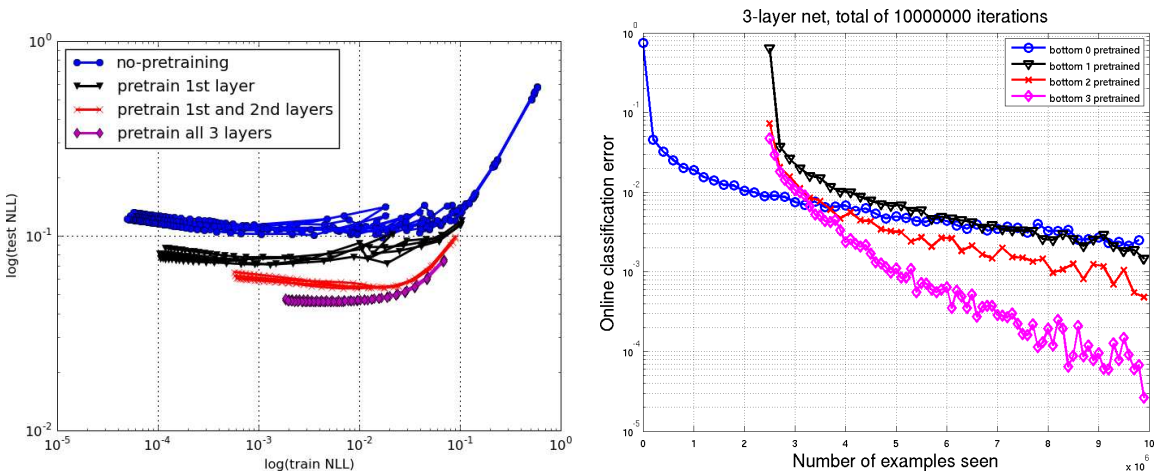


Figure 14: *On the left*: for `MNIST`, a plot of the log(train NLL) vs. log(test NLL) at each epoch of training. We pre-train the first layer, the first two layers and all three layers using RBMs and randomly initialize the other layers; we also compare with the network whose layers are all randomly initialized. *On the right*: `InfiniteMNIST`, the online classification error. We pre-train the first layer, the first two layers or all three layers using denoising auto-encoders and leave the rest of the network randomly initialized.

For `InfiniteMNIST`, we simply show the online error. The results are ambiguous w.r.t the difficulty of optimizing the lower layers versus the higher ones. We would have expected that the largest incremental benefit came from pre-training the first layer or first two layers. It is true for the first two layers, but not the first. As we pre-train more layers, the models become better at generalization. In the case of the finite `MNIST`, note how the final training error (after the same number of epochs) becomes *worse* with pre-training of more layers. This clearly brings additional support to the regularization explanation.

9. Discussion and Conclusions

We have shown that unsupervised pre-training adds robustness to a deep architecture. The same set of results also suggests that increasing the depth of an architecture that is not pre-trained increases the probability of finding poor apparent local minima. Pre-trained networks give consistently better generalization. Our visualizations point to the observations that pre-trained networks learn qualitatively different features (if networks are visualized in the weight space) compared to networks without pre-training. Moreover, the trajectories of networks with different initialization seeds seem to fall into many distinct apparent local minima, which are again different (and seemingly far apart) depending on whether we use pre-training or not.

We have shown that unsupervised pre-training is not simply a way of getting a good initial marginal distribution, and that it captures more intricate dependencies between parameters. One of our findings is that deep networks with unsupervised pre-training seem to exhibit some properties of a regularizer: with small enough layers, pre-trained deep architectures are systematically worse than randomly initialized deep architectures. Moreover, when the layers are big enough, the pre-trained models obtain worse training errors, but better generalization performance. Additionally, we have re-done an experiment which purportedly showed that unsupervised pre-training can be explained with an optimization hypothesis and observed a regularization effect instead. We also showed that classical regularization techniques (such as L_1/L_2 penalties on the network weights) cannot achieve the same performance as unsupervised pre-training, and that the effect of unsupervised pre-training does not go away with more training data, so if unsupervised pre-training is a regularizer, it is certainly of a rather different kind.

The two unsupervised pre-training strategies considered—denoising auto-encoders and Restricted Boltzmann Machines—seem to produce qualitatively similar observations. We have observed that, surprisingly, the pre-training advantage is present even in the case of really large training sets, pointing towards the conclusion that the starting point in the non-convex optimization problem is indeed quite important; a fact confirmed by our visualizations of filters at various levels in the network. Finally, the other important set of results show that unsupervised pre-training acts like a variance reduction technique, yet a network with pre-training has a lower training error on a very large data set, which supports an optimization interpretation of the effect of pre-training.

How do we make sense of all these results? The contradiction between what looks like regularization effects and what looks like optimization effects appears, on the surface, unresolved. Instead of sticking to these labels, we attempted to draw a hypothesis, described in Section 3 about the dynamics of learning in an architecture that is trained using two phases (unsupervised pre-training and supervised fine-tuning), which we believe to be consistent with all the above results.

This hypothesis suggests that there are consequences of the non-convexity of the supervised objective function, which we observed in various ways throughout our experiments. One of these consequences is that early examples have a big influence on the outcome of training and this is one of the reasons why in a large-scale setting the influence of unsupervised pre-training is still present. Throughout this paper, we have delved on the idea that the basin of attraction induced by the early examples (in conjunction with unsupervised pre-training) is, for all practical purposes, a basin from which supervised training does not escape.

This effect can be observed from the various visualizations and performance evaluations that we made. *Unsupervised pre-training,* **as a regularizer** *that only influences the starting point of supervised training, has an effect that, contrary to classical regularizers, does not disappear with*

more data (at least as far as we can see from our results). Basically, unsupervised pre-training favors hidden units that compute features of the input X that correspond to major factors of variation in the true $P(X)$. Assuming that some of these are near features useful at predicting variations in Y, unsupervised pre-training sets up the parameters near a solution of low predictive generalization error.

One of the main messages that our results imply is that the optimization of a non-convex objective function with stochastic gradient descent presents challenges for analysis, especially in a regime with large amounts of data. Our analysis so far shows that it is possible for networks that are trained in such a regime to be influenced more by early examples. This can pose problems in scenarios where we would like our networks to be able to capture more of the information in later examples, that is, when training from very large data sets and trying to capture a lot of information from them.

One interesting realization is that with a small training set, we do not usually put a lot of importance on minimizing the training error, because overfitting is a major issue; the training error is not a good way to distinguish between the generalization performance of two models. In that setting, unsupervised pre-training helps to find apparent local minima that have better generalization error. With a large training set, as we saw in Figure 12, the empirical and true distributions converge. In such a scenario, *finding a better apparent local minimum will matter and stronger (better) optimization strategies should have a significant impact on generalization when the training set is very large*. Note also that it would be interesting to extend our experimental techniques to the problem of training deep auto-encoders (with a bottleneck), where previous results (Hinton and Salakhutdinov, 2006) show that not only test error but also training error is greatly reduced by unsupervised pre-training, which is a strong indicator of an optimization effect. We hypothesize that the presence of the bottleneck is a crucial element that distinguishes the deep auto-encoders from the deep classifiers studied here.

In spite of months of CPU time on a cluster devoted to the experiments described here (which is orders of magnitude more than most previous work in this area), more could certainly be done to better understand these effects. Our original goal was to have well-controlled experiments with well understood data sets. It was not to advance a particular algorithm but rather to try to better understand a phenomenon that has been well documented elsewhere. Nonetheless, our results are limited by the data sets used and it is plausible that different conclusions could be drawn, should the same experiments be carried out on other data.

Our results suggest that optimization in deep networks is a complicated problem that is influenced in great part by the early examples during training. Future work should clarify this hypothesis. If it is true and we want our learners to capture really complicated distributions from very large training sets, it may mean that we should consider learning algorithms that reduce the effect of the early examples, allowing parameters to escape from the attractors in which current learning dynamics get stuck.

The observations reported here suggest more detailed explanations than those already discussed, which could be tested in future work. We hypothesize that the factors of variation present in the input distribution are disentangled more and more as we go from the input layer to higher-levels of the feature hierarchy. This is coherent with observations of increasing invariance to geometric transformations in DBNs trained on images (Goodfellow et al., 2009), as well as by visualizing the variations in input images generated by sampling from the model (Hinton, 2007; Susskind et al., 2008), or when considering the preferred input associated with different units at different depths (Lee et al.,

2009; Erhan et al., 2009). As a result, during early stages of learning, the upper layers (those that typically learn quickly) would have access to a more robust representation of the input and are less likely to be hindered by the entangling of factors variations present in the input. If this disentangling hypothesis is correct, it would help to explain how unsupervised pre-training can address the chicken-and-egg issue explained in Section 2: the lower layers of a supervised deep architecture need the upper layers to define what they should extract, and vice-versa. Instead, the lower layers can extract robust and disentangled representations of the factors of variation and the upper layers select and combine the appropriate factors (sometimes not all at the top hidden layer). Note that as factors of variation are disentangled, it could also happen that some of them are not propagated upward (before fine-tuning), because RBMs do not try to represent in their hidden layer input bits that are independent.

To further explain why smaller hidden layers yield worse performance with pre-training than without (Figure 9), one may hypothesize further that, for some data sets, the leading factors of variation present in $P(X)$ (presumably the only ones captured in a smaller layer) are less predictive of Y than random projections[16] can be, precisely because of the hypothesized disentangling effect. With enough hidden units, unsupervised pre-training may extract among the larger set of learned features some that are highly predictive of Y (more so than random projections). This additional hypothesis could be tested by measuring the mutual information between each hidden unit and the object categories (as done by Lee et al., 2009), as the number of hidden units is varied (like in Figure 9). It is expected that the unit with the most mutual information will be less informative with pre-training when the number of hidden units is too small, and more informative with pre-training when the number of hidden units is large enough.

Under the hypothesis that we have proposed in Section 3, the following result is unaccounted for: in Figure 8(a), training error is lower with pre-training when there is only one hidden layer, but worse with more layers. This may be explained by the following additional hypothesis. Although each layer extracts information about Y in some of its features, it is not guaranteed that all of that information is preserved when moving to higher layers. One may suspect this in particular for RBMs, which would not encode in their hidden layer any input bits that would be marginally independent of the others, because these bits would be explained by the visible biases: perfect disentangling of Y from the other factors of variation in X may yield marginally independent bits about Y. Although supervised fine-tuning should help to bubble up that information towards the output layer, it might be more difficult to do so for deeper networks, explaining the above-stated feature of Figure 8. Instead, in the case of a single hidden layer, less information about Y would have been dropped (if at all), making the job of the supervised output layer easier. This is consistent with earlier results (Larochelle et al., 2009) showing that for several data sets supervised fine-tuning significantly improves classification error, when the output layer only takes input from the top hidden layer. This hypothesis is also consistent with the observation made here (Figure 1) that unsupervised pre-training actually does not help (and can hurt) for too deep networks.

In addition to exploring the above hypotheses, future work should include an investigation of the connection between the results presented in this paper and by Hinton and Salakhutdinov (2006), where it seems to be hard to obtain a good training reconstruction error with deep auto-encoders (in an unsupervised setting) without performing pre-training. Other avenues for future work include the analysis and understanding of deep semi-supervised techniques where one does not separate

16. Meaning the random initialization of hidden layers.

between the pre-training phase and the supervised phase, such as work by Weston et al. (2008) and Larochelle and Bengio (2008). Such algorithms fall more squarely into the realm of semi-supervised methods. We expect that analyses similar to the ones we performed would be potentially harder, but perhaps revealing as well.

Many open questions remain towards understanding and improving deep architectures. Our conviction is that devising improved strategies for learning in deep architectures requires a more profound understanding of the difficulties that we face with them. This work helps with such understanding via extensive simulations and puts forward a hypothesis explaining the mechanisms behind unsupervised pre-training, which is well supported by our results.

Acknowledgments

This research was supported by funding from NSERC, MITACS, FQRNT, and the Canada Research Chairs. The authors also would like to thank the editor and reviewers, as well as Fernando Pereira for their helpful comments and suggestions.

References

Shun-ichi Amari, Noboru Murata, Klaus-Robert Müller, Michael Finke, and Howard Hua Yang. Asymptotic statistical theory of overtraining and cross-validation. *IEEE Transactions on Neural Networks*, 8(5):985–996, 1997.

Lalit Bahl, Peter Brown, Peter deSouza, and Robert Mercer. Maximum mutual information estimation of hidden markov parameters for speech recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 49–52, Tokyo, Japan, 1986.

Andrew E. Barron. Complexity regularization with application to artificial neural networks. In G. Roussas, editor, *Nonparametric Functional Estimation and Related Topics*, pages 561–576. Kluwer Academic Publishers, 1991.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 (NIPS'01)*, Cambridge, MA, 2002. MIT Press.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. Also published as a book. Now Publishers, 2009.

Yoshua Bengio and Olivier Delalleau. Justifying and generalizing contrastive divergence. *Neural Computation*, 21(6):1601–1621, June 2009.

Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*, pages 321–360. MIT Press, 2007.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. The curse of highly variable functions for local kernel machines. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18 (NIPS'05)*, pages 107–114. MIT Press, Cambridge, MA, 2006.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In Bernhard Schölkopf, John Platt, and Thomas Hoffman, editors, *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 153–160. MIT Press, 2007.

Marc H. Bornstein. *Sensitive periods in development : interdisciplinary perspectives / edited by Marc H. Bornstein*. Lawrence Erlbaum Associates, Hillsdale, N.J. :, 1987.

Olivier Chapelle, Jason Weston, and Bernhard Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15 (NIPS'02)*, pages 585–592, Cambridge, MA, 2003. MIT Press.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 160–167. ACM, 2008.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. Technical Report 1341, Université de Montréal, 2009.

Patrick Gallinari, Yann LeCun, Sylvie Thiria, and Francoise Fogelman-Soulie. Memoires associatives distribuees. In *Proceedings of COGNITIVA 87*, Paris, La Villette, 1987.

Ian Goodfellow, Quoc Le, Andrew Saxe, and Andrew Ng. Measuring invariances in deep networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 646–654. 2009.

Raia Hadsell, Ayse Erkan, Pierre Sermanet, Marco Scoffier, Urs Muller, and Yann LeCun. Deep belief net learning in a long-range vision system for autonomous off-road driving. In *Proc. Intelligent Robots and Systems (IROS'08)*, pages 628–633, 2008.

Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th annual ACM Symposium on Theory of Computing*, pages 6–20, Berkeley, California, 1986. ACM Press.

Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

Geoffrey E. Hinton. To recognize shapes, first learn to generate images. In Paul Cisek, Trevor Drew, and John Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*. Elsevier, 2007.

Geoffrey E. Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

Goeffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 536–543. ACM, 2008.

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 *Int. Conf. Mach. Learn.*, pages 473–480, 2007.

Hugo Larochelle, Yoshua Bengio, Jerome Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *The Journal of Machine Learning Research*, 10:1–40, January 2009.

Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled hybrids of generative and discriminative models. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'06)*, pages 87–94, Washington, DC, USA, 2006. IEEE Computer Society.

Yann LeCun. *Modèles connexionistes de l'apprentissage*. PhD thesis, Université de Paris VI, 1987.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Honglak Lee, Chaitanya Ekanadham, and Andrew Ng. Sparse deep belief net model for visual area V2. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 873–880. MIT Press, Cambridge, MA, 2008.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Léon Bottou and Michael Littman, editors, *Proceedings of the Twenty-sixth International Conference on Machine Learning (ICML'09)*. ACM, Montreal (Qc), Canada, 2009.

Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. In Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors, *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA., 2007.

Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In Léon Bottou and Michael Littman, editors, *Proceedings of the 26th International Conference on Machine Learning*, pages 737–744, Montreal, June 2009. Omnipress.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 (NIPS'01)*, pages 841–848, 2002.

Simon Osindero and Geoffrey E. Hinton. Modeling image patches with a directed hierarchy of markov random field. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 1121–1128, Cambridge, MA, 2008. MIT Press.

Dan Povey and Philip C. Woodland. Minimum phone error and i-smoothing for improved discriminative training. In *Acoustics, Speech, and Signal Processing, 2002. Proceedings. (ICASSP '02). IEEE International Conference on*, volume 1, pages I–105–I–108 vol.1, 2002.

Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 1137–1144. MIT Press, 2007.

Marc'Aurelio Ranzato, Y-Lan Boureau, and Yann LeCun. Sparse feature learning for deep belief networks. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 1185–1192, Cambridge, MA, 2008. MIT Press.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Using deep belief nets to learn covariance kernels for Gaussian processes. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 1249–1256, Cambridge, MA, 2008. MIT Press.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Semantic hashing. In *Proceedings of the 2007 Workshop on Information Retrieval and applications of Graphical Models (SIGIR 2007)*, Amsterdam, 2007. Elsevier.

Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey E. Hinton. Restricted Boltzmann machines for collaborative filtering. In Zoubin Ghahramani, editor, *Proceedings of the Twenty-fourth International Conference on Machine Learning (ICML'07)*, pages 791–798, New York, NY, USA, 2007. ACM.

Sebastian H. Seung. Learning continuous attractors in recurrent networks. In M.I. Jordan, M.J. Kearns, and S.A. Solla, editors, *Advances in Neural Information Processing Systems 10 (NIPS'97)*, pages 654–660. MIT Press, 1998.

Jonas Sjöberg and Lennart Ljung. Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, 62(6):1391–1407, 1995.

Joshua M. Susskind, Geoffrey E., Javier R. Movellan, and Adam K. Anderson. Generating facial expressions with deep belief nets. In V. Kordic, editor, *Affective Computing, Emotion Modelling, Synthesis and Recognition*, pages 421–440. ARS Publishers, 2008.

Joshua Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.

Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In Andrew McCallum and Sam Roweis, editors, *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 1096–1103. Omnipress, 2008.

Max Welling, Michal Rosen-Zvi, and Geoffrey E. Hinton. Exponential family harmoniums with an application to information retrieval. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17 (NIPS'04)*, pages 1481–1488, Cambridge, MA, 2005. MIT Press.

Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1168–1175, New York, NY, USA, 2008. ACM.

Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.

Long Zhu, Yuanhao Chen, and Alan Yuille. Unsupervised learning of probabilistic grammar-markov models for object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):114–128, 2009.

Journal of Machine Learning Research 11 (2010) 661-664 Submitted 8/09; Revised 1/10; Published 2/10

Error-Correcting Ouput Codes Library

Sergio Escalera SERGIO@MAIA.UB.ES
Oriol Pujol ORIOL@MAIA.UB.ES
Petia Radeva PETIA.IVANOVA@UB.EDU
Computer Vision Center
Edifici O, Campus UAB, 08193, Bellaterra, Barcelona, Spain

Editor: Cheng Soon Ong

Abstract

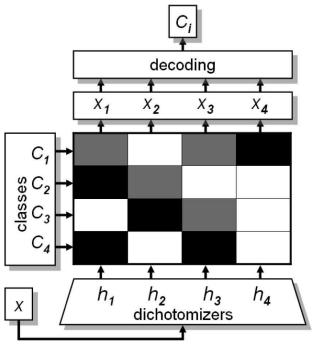
In this paper, we present an open source Error-Correcting Output Codes (ECOC) library. The ECOC framework is a powerful tool to deal with multi-class categorization problems. This library contains both state-of-the-art coding (one-versus-one, one-versus-all, dense random, sparse random, DECOC, forest-ECOC, and ECOC-ONE) and decoding designs (hamming, euclidean, inverse hamming, laplacian, β-density, attenuated, loss-based, probabilistic kernel-based, and loss-weighted) with the parameters defined by the authors, as well as the option to include your own coding, decoding, and base classifier.

Keywords: error-correcting output codes, multi-class classification, coding, decoding, open source, matlab, octave

1. Error-Correcting Output Codes

The Error-Correcting Output Codes (ECOC) framework (Dietterich and Bakiri, 1995) is a simple but powerful framework to deal with the multi-class categorization problem based on the embedding of binary classifiers. Given a set of N_c classes, the basis of the ECOC framework consists of designing a codeword for each of the classes. These codewords encode the membership information of each class for a given binary problem. Arranging the codewords as rows of a matrix, we obtain a "coding matrix" M_c, where $M_c \in \{-1,0,1\}^{N_c \times n}$, being n the length of the codewords codifying each class. From the point of view of learning, M_c is constructed by considering n binary problems, each one corresponding to a column of the matrix M_c. Each of these binary problems (or dichotomizers) splits the set of classes in two partitions (coded by +1 or -1 in M_c according to their class set membership, or 0 if the class is not considered by the current binary problem). Then, at the decoding step, applying the n trained binary classifiers, a code is obtained for each data point in the test set. This code is compared to the base codewords of each class defined in the matrix M_c, and the data point is assigned to the class with the "closest" codeword. Several decoding strategies have been proposed in literature. The reader is referred to Escalera et al. (2008) for a more detailed review. An example of an ECOC design is described in Fig. 1.

The ECOC designs are independent of the base classifier applied. They involve error-correcting properties (Dietterich and Bakiri, 1995) and have shown to be able to reduce the bias and variance produced by the learning algorithm (Kong and Dietterich, 1995). Because of these reasons, ECOCs have been widely used to deal with multi-class categorization problems.



ECOC coding design for a 4-class problem. White, black, and grey positions corresponds to the symbols +1, -1, and 0, respectively. Once the four binary problems are learnt, at the decoding step a new test sample X is tested by the n classifiers. Then, the new codeword $x = \{x_1,..,x_n\}$ is compared with the class codewords $\{C_1,..C_4\}$, classifying the new sample by the class c_i which codeword minimizes the decoding measure.

Figure 1: ECOC design example.

2. Library Algorithms

The ECOCs library is a Matlab/Octave code under the open source GPL license (gpl) with the implementation of the state-of-the-art coding and decoding ECOC designs. A main function defines the multi-class data, coding, decoding, and base classifier. A list of parameters are also included in order to tune the different strategies. In addition to the implemented coding and decoding designs, which are described in the following section, the user can include his own coding, decoding, and base classifier as defined in the user guide.

2.1 Implemented Coding Designs

The ECOC designs of the ECOC library cover the state-of-the-art of coding strategies, mainly divided in two main groups: problem-independent approaches, which do not take into account the distribution of the data to define the coding matrix, and the problem-dependent designs, where information of the particular domain is used to guide the coding design.

2.1.1 PROBLEM-INDEPENDENT ECOC DESIGNS

• One-versus-all (Rifkin and Klautau, 2004): N_c dichotomizers are learnt for N_c classes, where each one splits one class from the rest of classes.

• One-versus-one (Nilsson, 1965): $n = N_c(N_c-1)/2$ dichotomizers are learnt for N_c classes, splitting each possible pair of classes.

• Dense Random (Allwein et al., 2002): $n = 10 \cdot \log N_c$ dichotomizers are suggested to be learnt for N_c classes, where $P(-1) = 1 - P(+1)$, being $P(-1)$ and $P(+1)$ the probability of the symbols -1 and +1 to appear, respectively. Then, from a set of defined random matrices, the one which maximizes a decoding measure among all possible rows of M_c is selected.

• Sparse Random (Escalera et al., 2009): $n = 15 \cdot \log N_c$ dichotomizers are suggested to be learnt for N_c classes, where $P(0) = 1 - P(-1) - P(+1)$, defining a set of random matrices M_c and selecting the one which maximizes a decoding measure among all possible rows of M_c.

2.1.2 PROBLEM-DEPENDENT ECOC DESIGNS

• DECOC (Pujol et al., 2006): problem-dependent design that uses $n = N_c - 1$ dichotomizers. The partitions of the problem are learnt by means of a binary tree structure using exhaustive search or a $SFFS$ criterion. Finally, each internal node of the tree is embedded as a column in M_c.

• Forest-ECOC (Escalera et al., 2007): problem-dependent design that uses $n = (N_c - 1) \cdot T$ dichotomizers, where T stands for the number of binary tree structures to be embedded. This approach extends the variability of the classifiers of the DECOC design by including extra dichotomizers.

• ECOC-ONE (Pujol et al., 2008): problem-dependent design that uses $n = 2 \cdot N_c$ suggested dichotomizers. A validation sub-set is used to extend any initial matrix M_c and to increase its generalization by including new dichotomizers that focus on difficult to split classes.

2.2 Implemented Decoding Designs

The software comes with a complete set of ECOC decoding strategies. The notation used refers to that used in (Escalera et al., 2008):

• Hamming decoding: $HD(x, y_i) = \sum_{j=1}^{n} (1 - \text{sign}(x^j \cdot y_i^j))/2$, being x a test codeword and y_i a codeword from M_c corresponding to class C_i.

• Inverse Hamming decoding: $IHD(x, y_i) = \max(\Delta^{-1} D^T)$, where $\Delta(i_1, i_2) = HD(y_{i_1}, y_{i_2})$, and D is the vector of Hamming decoding values of the test codeword x for each of the base codewords y_i.

• Euclidean decoding: $ED(x, y_i) = \sqrt{\sum_{j=1}^{n} (x^j - y_i^j)^2}$.

• Attenuated Euclidean decoding: $AED(x, y_i) = \sqrt{\sum_{j=1}^{n} \mid y_i^j \mid\mid x^j \mid (x^j - y_i^j)^2}$.

• Loss-based decoding: $LB(\rho, y_i) = \sum_{j=1}^{n} L(y_i^j \cdot f^j(\rho))$, where ρ is a test sample, L is a loss-function, and f is a real-valued function $f : \mathcal{R}^n \rightarrow \mathcal{R}$.

• Probabilistic-based decoding:
$PD(y_i, x) = -log\left(\prod_{j \in [1,..,n]: M_c(i,j) \neq 0} P(x^j = M_c(i, j) | f^j) + K\right)$, where K is a constant factor that collects the probability mass dispersed on the invalid codes, and the probability $P(x^j = M_c(i, j) | f^j)$ is estimated by means of $P(x^j = y_i^j | f^j) = \frac{1}{1 + \mathbf{e}^{y_i^j (\upsilon^j f^j + \omega^j)}}$, where vectors υ and ω are obtained by solving an optimization problem (Passerini et al., 2004).

• Laplacian decoding: $LAP(x, y_i) = \frac{\alpha_i + 1}{\alpha_i + \beta_i + K}$, where α_i is the number of matched positions between x and y_i, β_i is the number of miss-matches without considering the positions coded by 0, and K is an integer value that codifies the number of classes considered by the classifier.

• Pessimistic β-Density Distribution decoding: accuracy s_i : $\int_{\nu_i - s_i}^{\nu_i} \psi_i(\nu, \alpha_i, \beta_i) d\nu = \frac{1}{3}$, where $\psi_i(\nu, \alpha_i, \beta_i) = \frac{1}{K} \nu^{\alpha_i} (1 - \nu)^{\beta_i}$, ψ_i is the β-Density Distribution between a codeword x and a class codeword y_i for class c_i, and $\nu \in \mathcal{R} : [0, 1]$.

• Loss-Weighted decoding: $LW(\rho, i) = \sum_{j=1}^{n} M_W(i, j) L(y_i^j \cdot f(\rho, j))$, where $M_W(i, j) = \frac{H(i,j)}{\sum_{j=1}^{n} H(i,j)}$, $H(i, j) = \frac{1}{m_i} \sum_{k=1}^{m_i} \varphi(h^j(\rho_k^i), i, j), \varphi(x^j, i, j) = \begin{cases} 1, & \text{if } x^j = y_i^j, \\ 0, & \text{otherwise.} \end{cases}$, m_i is the number of training samples from class C_i, and ρ_k^i is the kth sample from class C_i.

3. Implementation Details

The ECOCs Library comes with detailed documentation. A user guide describes the usage of the software. All the strategies and parameters used in the functions and files are described in detail. The user guide also presents examples of variable setting and execution, including a demo file.

About the computational complexity, the training and testing time depends on the data size, coding and decoding algorithms, as well as the base classifier used in the ECOC design.

Acknowledgments

This work has been supported in part by projects TIN2009-14404-C02 and CONSOLIDER-INGENIO CSD 2007-00018.

References

URL `http://www.gnu.org/licences/`.

E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2002.

T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–282, 1995.

S. Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and Forest-ECOC: A novel framework to detect and classify objects in clutter scenes. *Pattern Recognition Letters*, 28(13):1759–1768, 2007.

S. Escalera, O. Pujol, and P. Radeva. On the decoding process in ternary error-correcting output codes. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 99, 2008.

S. Escalera, O. Pujol, and P. Radeva. Separability of ternary codes for sparse designs of error-correcting output codes. *Pattern Recognition Letters*, 30:285–297, 2009.

E. B. Kong and T. G. Dietterich. Error-correcting output coding corrects bias and variance. *International Conference of Machine Learning*, pages 313–321, 1995.

N. J. Nilsson. *Learning Machines*. McGraw-Hill, 1965.

A. Passerini, M. Pontil, and P. Frasconi. New results on error correcting output codes of kernel machines. *IEEE Transactions on Neural Networks*, 15(1):45–54, 2004.

O. Pujol, P. Radeva, , and J. Vitrià. Discriminant ECOC: A heuristic method for application dependent design of error correcting output codes. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 28:1001–1007, 2006.

O. Pujol, S. Escalera, and P. Radeva. An incremental node embedding technique for error-correcting output codes. *Pattern Recognition*, 4:713–725, 2008.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

Journal of Machine Learning Research 11 (2010) 665-685 Submitted 6/08; Revised 10/09; Published 2/10

Second-Order Bilinear Discriminant Analysis

Christoforos Christoforou CCHRISTOFOROU@RKILEADERS.COM
R.K.I Leaders Limited
Agias Triados 26A
7100, Aradippou, Cyprus

Robert Haralick HARALICK@GC.CUNY.EDU
Department of Computer Science
Graduate Center, City University of New York
New York, NY 10011, USA

Paul Sajda PSAJDA@COLUMBIA.EDU
Department of Biomedical Engineering
Columbia University
New York, NY 10027, USA

Lucas C. Parra PARRA@ENGR.CCNY.CUNY.EDU
Department of Biomedical Engineering
City College, City University of New York
New York, NY 10031, USA

Editor: Mikio Braun

Abstract

Traditional analysis methods for single-trial classification of electro-encephalography (EEG) focus on two types of paradigms: phase-locked methods, in which the amplitude of the signal is used as the feature for classification, that is, event related potentials; and second-order methods, in which the feature of interest is the power of the signal, that is, event related (de)synchronization. The process of deciding which paradigm to use is *ad hoc* and is driven by assumptions regarding the underlying neural generators. Here we propose a method that provides an unified framework for the analysis of EEG, combining first and second-order spatial and temporal features based on a bilinear model. Evaluation of the proposed method on simulated data shows that the technique outperforms state-of-the art techniques for single-trial classification for a broad range of signal-to-noise ratios. Evaluations on human EEG—including one benchmark data set from the Brain Computer Interface (BCI) competition—show statistically significant gains in classification accuracy, with a reduction in overall classification error from 26%-28% to 19%.

Keywords: regularization, classification, bilinear decomposition, neural signals, brain computer interface

1. Introduction

The work presented in this paper is motivated by the analysis of functional brain imaging signals recorded via electroencephalography (EEG). EEG is measured across time and typically at multiple scalp locations, providing a spatio-temporal data set of the underlying neural activity. In addition, these measurements are often taken over multiple repetitions or trials, where trials may differ in the type of stimulus presented, the task given to the subject, or the subject's response. Analysis of these

signals is often expressed as a single-trial classification problem. The goal for the classifier is to determine from the EEG data which stimulus was presented or how the subject responded. Many of these classification techniques were originally developed in the context of Brain Computer Interfaces (BCI) but are now more widely used to interpret activity associated with neural processing.

In the case of BCI algorithms (Wolpaw et al., 2002; Birbaumer et al., 1999; Blankertz et al., 2002, 2003) the aim is to decode brain activity on a single-trial basis in order to provide a direct control pathway between a user's intentions and a computer. Such an interface could provide "locked in patients" a more direct and natural control over a neuroprosthesis or other computer applications (Birbaumer et al., 1999). Furthermore, by providing an additional communication channel for healthy individuals, BCI systems can be used to increase productivity and efficiency in high-throughput tasks (Gerson et al., 2006; Parra et al., 2008).

Single-trial discriminant analysis has also been used as a research tool to study the neural correlates of behavior. By extracting activity that differs maximally between two experimental conditions, the typically low signal-to-noise ratio of EEG can be overcome. The resulting discriminant components can be used to identify the spatial origin and time course of stimulus/response specific activity, while the improved SNR can be leveraged to correlate variability of neural activity across trials to behavioral variability and behavioral performance (Philiastides et al., 2006; Gerson et al., 2006; Philiastides and Sajda, 2006) In essence, discriminant analysis adds to the existing set of multi-variate statistical tools commonly used in neuroscience research (ANOVA, Hoteling T^2, Wilks' Λ test, etc.).

1.1 Traditional EEG Analysis

In EEG the signal-to-noise ratio (SNR) of individual channels is low, often at, or below -20dB. To overcome this limitation, all analysis methods perform some form of averaging, either across repeated trials, across time, or across electrodes. Traditional EEG analysis averages signals across many repeated trials for each individual electrode. Typical in this case is to average the measured potentials following stimulus presentation, thereby canceling uncorrelated noise that is not reproducible from one trial to the next. This averaged activity, called an event related potential (ERP), captures activity that is time-locked to the stimulus presentation but cancels induced oscillatory activity that is not locked in phase to the timing of the stimulus. Alternatively, many studies compute the oscillatory activity in specific frequency bands by filtering and squaring the signal prior to averaging. Induced changes in oscillatory activity are termed event related synchronization or desynchronization (ERS/ERD) Pfurtscheller and da Silva (1999).

Surprisingly, discriminant analysis methods developed thus far by the machine learning community have followed this dichotomy: First order methods in which the amplitude of the EEG signal is considered to be the feature of interest in classification—corresponding to ERP—and second-order methods in which the power of the feature is considered to be of importance for classification—corresponding to ERS/ERD. First order methods include temporal filtering and thresholding (Birbaumer et al., 1999), Fisher linear discriminants (Parra et al., 2005; Blankertz et al., 2002), hierarchical linear classifiers (Gerson et al., 2006) and bilinear discriminant analysis (Dyrholm et al., 2007; Tomioka and Aihara, 2007). Second-order methods include logistic regression with a quadratic term (Tomioka et al., 2007) and the well known common spatial patterns method (CSP) (Ramoser et al., 2000) and its variants: common spatio-spectral patterns (CSSP) (Lemm et al., 2005), and common sparse spectral spatial patterns (CSSSP) (Dornhege et al., 2006).

In the past, the process for choosing features for classification has been *ad hoc* and driven primarily by prior knowledge and/or assumptions regarding the underlying neurophysiology and task. From a machine-learning point of view, it seems limiting to commit *a priori* to only one type of feature. Instead, it would be desirable for the analysis method to extract the relevant neurophysiological activity *de novo* with minimal prior expectations.

In this paper we present a new framework that combines both first and second-order features in the analysis of EEG. Through a bilinear formulation, the method can simultaneously identify spatial linear components as well as temporal modulation of activity. These spatio-temporal components are identified such that their first and second-order statistics are maximally different between two conditions. Further, through the bilinear formulation, the method exploits the spatio-temporal nature of the EEG signals and provides a reduced parametrization of the high dimensional data space. We show that a broad set of state-of-the-art EEG analysis methods can be characterized as special cases under this bilinear framework. Simulated EEG data is then used to evaluate performance of the different methods under varying signal strengths. We conclude the paper with a performance comparison on human EEG. In all instances the performance of the present method is comparable or superior to the existing state-of-the-art.

2. Second-Order Bilinear Discriminant Analysis

To introduce the new method we start by formally defining the classification problem in EEG. We then present the bilinear model, discuss interpretation in the context of EEG, and establish a link to current analysis methods. The section concludes with the optimization criterion and regularization approaches. As the title of this section suggests, we termed our method Second-Order Bilinear Discriminant Analysis (SOBDA).

2.1 Problem Setting

Suppose that we are given examples of brain activity as a set of trials $\{\mathbf{X}_n, y_n\}_{n=1}^N, \mathbf{X}_n \in \mathbb{R}^{D\times T}, y_n \in \{-1,1\}$, where for each example n the matrix $\mathbf{X}_n$ corresponds to the EEG signal with D channels and T time samples and y_n indicates the class to which this example corresponds. The class label may indicated one of two conditions, that is, imagined right or left hand movement, stimulus or non-stimulus control conditions, etc. Given these examples the task is then to predict the class label y for a new trial with data $\mathbf{X}$.

2.2 Second-order Bilinear Model

To solve this problem we propose the following discriminant function

$$f(\mathbf{X};\theta) = C\,\mathrm{Trace}\left(\mathbf{U}^\top\mathbf{X}\mathbf{V}\right) + (1-C)\,\mathrm{Trace}\left(\Lambda\mathbf{A}^\top\mathbf{X}\mathbf{B}\mathbf{B}^\top\mathbf{X}^\top\mathbf{A}\right) + w_o\,, \tag{1}$$

where the parameters are $\theta = \{\mathbf{U}\in\mathbb{R}^{D\times R}, \mathbf{V}\in\mathbb{R}^{T\times R}, \mathbf{A}\in\mathbb{R}^{D\times K}\,\mathbf{B}\in\mathbb{R}^{T\times T'}, w_o\in\mathbb{R}, \Lambda\in diag(K)|\ \lambda_{ii}\in\{-1,+1\}, C\in[0,1]$. Some of these parameters may be specified using prior knowledge as will be discussed later. The scalars R, K and T' are chosen by the user and denote the rank of matrix $\mathbf{U}$,$\mathbf{V}$ $\mathbf{A}$ and $\mathbf{B}$. Typically we use $T' = T$. The goal will be to use the N examples to optimize these parameters such that the discriminant function takes on positive values for examples with $y_n = +1$ and negative values for $y_n = -1$. To accomplish this we will use a standard probabilistic

formalism—logistic regression—which will permit us to incorporate regularization criteria as prior probabilities on the parameter as will be explained in Sections 2.6 and 2.8.

2.3 Interpretation and Rationale of the Model

The discriminant criterion defined in (1) is the sum of a linear and a quadratic term, each combining bilinear components of the EEG signal. The first term can be interpreted as a spatio-temporal projection of the signal that captures the first-order statistics of the signal. Specifically, the columns $\mathbf{u}_r$ of $\mathbf{U}$ represent R linear projections in space (rows of $\mathbf{X}$). Similarly, each of the R columns of $\mathbf{v}_k$ in matrix $\mathbf{V}$ represent linear projections in time (columns of $\mathbf{X}$). By re-writing the term as:

$$\text{Trace}(\mathbf{U}^\top \mathbf{X}\mathbf{V}) = \text{Trace}(\mathbf{V}\mathbf{U}^\top \mathbf{X}) = \text{Trace}(\mathbf{W}^\top \mathbf{X}),$$

where we defined, $\mathbf{W} = \mathbf{U}\mathbf{V}^\top$, it is easy to see that the bilinear projection is a linear combination of elements of $\mathbf{X}$ with a rank-R constraint on $\mathbf{W}$. This expression is linear in $\mathbf{X}$ and thus captures directly the amplitude of the signal. In particular, the polarity of the signal (positive evoked response versus negative evoked response) will contribute to discrimination if it is consistent across trials. This term, therefore, captures phase-locked event related potential in the EEG signal. This bilinear projection reduces the number of model parameters of $\mathbf{W}$ from $D \times T$ dimensions to $R \times (D+T)$ which is a significant dimensionality reduction that alleviates the problem of over-fitting in parameters estimation given the small training set size. This projection assumes that the generators of class-dependent variances in the data have a low-rank contribution to each data matrix $\mathbf{X}$. This holds true in EEG data, where an electrical current source which is spatially static in the brain will give a rank-one contribution to the spatio-temporal $\mathbf{X}$ (Dyrholm and Parra, 2006).

The second term of Equation (1) is the power of spatially and temporally weighted signals and thus captures the second-order statistics of the signal. As before, each column of matrix $\mathbf{A}$ and $\mathbf{B}$ represent components that project the data in space and time respectively. Depending on the structure one enforces in matrix $\mathbf{B}$, different interpretations of the model can be achieved. In the general case where no structure on $\mathbf{B}$ is assumed, the model captures a linear combination of the elements of a rank-T' second-order matrix of the signal $\mathbf{X}\mathbf{B}(\mathbf{X}\mathbf{B})^\top$. In the case where Toeplitz structure is enforced on $\mathbf{B}$ (see Section 2.7), then $\mathbf{B}$ defines a temporal filter on the signal and the model captures powers of the filtered signal. Further, by allowing $\mathbf{B}$ to be learned from the data, we may be able to identify new frequency bands that have so far not been identified in novel experimental paradigms. The spatial weights $\mathbf{A}$ together with the Trace operation ensure that the power is measured, not in individual electrodes, but in some component space that may reflect activity distributed across several electrodes. The diagonal matrix Λ partitions the K spatial components (i.e., K columns of $\mathbf{A}$) into those that contribute power positively and those that contribute power negatively to the total sum. Since each column of $\mathbf{A}$ measures the power from different sources, then by multiplying the expression with Λ we capture the difference in power between different spatial components. As motivation consider the task of distinguishing between imagined left versus right hand movements. It is known that imagining a movement of the left hand reduces oscillatory activity over the motor cortex of the right hemisphere, while an imagined right-hand movement reduces oscillations over the left motor cortex. Each of these cortical areas will be captured by a different spatial distribution in the EEG. If we limit the columns of $\mathbf{A}$ to two, then these columns may capture the power of oscillatory activity over the right and left motor cortex respectively. One would like one of these two terms to contribute positively providing evidence of the observation belonging to the first class,

while the second should contribute negatively, supporting the observations coming from the second class. This can be achieved with the proper choice of Λ. Finally, the parameter C defines a convex combination of the first-order term and the second-order term. $C = 1$ indicates that the discriminant activity is dominated by the first-order features; $C = 0$ indicates that the activity is dominated by second-order features, and any value in between denotes the importance of one component versus the other.

2.4 Incorporating Prior Knowledge into the Model

Realizing that the parameters of the SOBDA model have a physical meaning (i.e., $\mathbf{u}_r$ and $\mathbf{a}_r$ map the sensor signal to a current-source space, $\mathbf{v}_r$ are temporal weight on a source signal and $\mathbf{b}_r$ can be arranged to represent a temporal filter) it becomes intuitive for the experimenter to incorporate prior knowledge of an experimental setup in the model. If the signal of interest is known to be in a specific frequency band, one can fix matrix $\mathbf{B}$ to capture only the desired frequency band. For example, $\mathbf{B}$ can be fixed to a Toeplitz matrix with coefficients corresponding to an 8Hz-12Hz band-pass filter, then the second-order term is able to extract power in the alpha-band which is known to be modulated during motor related tasks. It is often the case that experimenters have a hypothesis about the temporal profile of the signal of interest, for example the P300 signal or the N170 are known EEG responses with a positive peak at 300ms or negative peak at 170ms and are associated with surprise or processing of faces respectively. In such a scenario the experimenter can fix the temporal profile parameter $\mathbf{V}$ to emphasize time samples around the expected location of the peak activity and optimize over the rest of the parameters. The model also provides the ability to integrate information from fMRI studies. fMRI has high spatial resolution and can provide locations within the brain that may be known to participate in the processing during a particular experimental paradigm. This location information can be incorporated into the present model by fixing the spatial parameters $\mathbf{u}_r$ and $\mathbf{a}$ to reflect a localized source (often approximated as a current dipole). The remaining temporal parameters of the model can then be optimized.

2.5 SOBDA as a Generalized EEG Analysis Framework

The present model provides a generic framework that encompasses a number of popular EEG analysis techniques. The following list identifies some of the algorithms and how they relate to the model used in the SOBDA framework:

- Set $C = 1, R = 1$ and choose temporal component $\mathbf{v}$ to select a time window of interest (i.e., set $v_j = 1$ if j is inside the window of interest,$v_j = 0$ otherwise). Learn the spatial filters $\mathbf{u}$. This exactly corresponds to averaging over time and classifying in the sensor space as in Parra et al. (2002, 2005)

- Set $C = 1$ and select some $R > 1$ and choose the component vectors $\mathbf{v}_r$ to select multiple time windows of interest as in 1. Learn for each temporal window the corresponding spatial vector $\mathbf{u}_r$ from examples separately and then combine these components by learning a linear combination of the elements. This corresponds to the multiple window hierarchical classifier as in Gerson et al. (2006) and Parra et al. (2008)

- Set $C = 1, R = D$ while constraining $\mathbf{U}$ to be a diagonal matrix and select, separately for each channel, the time window $\mathbf{v}_r$ which is most discriminative. Then train the diagonal terms of

$\mathbf{U}$ resulting in a latency dependent spatial filter (Luo and Sajda, 2006a). Alternatively, in the first step, use feature selection to find the right set of time windows $\mathbf{v}_r$ simultaneously for all channels (Luo and Sajda, 2006b).

- Set $C = 1$,$R = 1$ and learn the spatial and temporal components $\mathbf{u}, \mathbf{v}$ simultaneously. This reduces to the rank-one bilinear discriminant as in Dyrholm and Parra (2006)

- Select $C = 1$ and some $R > 1$ and learn all columns of the spatial and temporal projection matrix $\mathbf{U}$ and $\mathbf{V}$ simultaneously. This results in the *Bilinear Discriminant Component Analysis (BDCA)* (Dyrholm et al., 2007).

- Set $C = 0$, $K = 2$ and fix $\mathbf{B}$ to a Toeplitz structure encoding a specific frequency band and set the diagonal of Λ to be $[1 - 1]$. Then learn the spatial component $\mathbf{A}$. This then reduces to the logistic regression with a quadratic term (Tomioka et al., 2007) which is related to the Common Spatial Patters (CSP) algorithm of Ramoser et al. (2000).

- Define $\hat{\mathbf{X}}$ to be the concatenation of $\mathbf{X}$ with itself delayed in time by τ samples, where τ is specified by the user, fix $\mathbf{B}$ to a Toeplitz structure, $C = 0$, and $\mathbf{A} \in \mathbb{R}^{2D \times 2}$, learn the matrix $\mathbf{A}$. This configuration can be related to the Common Spatio-Spectral Pattern algorithm of Lemm et al. (2005).

2.6 Logistic Regression

To optimize the model parameters $\mathbf{U}, \mathbf{V}, \mathbf{A}$ and $\mathbf{B}$ we use a Logistic Regression (LR) formalism. The probabilistic formalism is particularly convenient when imposing additional statistical properties on the coefficients such as smoothness or sparseness. In addition, in our experience, linear LR performs well in strongly overlapping high-dimensional data-sets and is insensitive to outliers, the later being of particular concern when including quadratic features.

Under the Logistic Regression model the probability that a trial belongs to class y after seeing data $\mathbf{X}$ is given by the class posterior probability

$$P(y|\mathbf{X};\theta) = \frac{1}{1 + e^{-yf(\mathbf{X};\theta)}}\,.$$

With this definition, the discriminant criterion given by the log-odds ratio of the posterior class probability

$$\log \frac{P(y=+1|\mathbf{X})}{P(y=-1|\mathbf{X})} = f(\mathbf{X};\theta)\,,$$

is simply the discriminant function which we chose to define in (1) as a sum of linear and quadratic terms. The Likelihood of observing the N examples under this model is then given by

$$L(\theta) = -\sum_{n=1}^{N} \log(1 + e^{-y_n f(\mathbf{X}_n;\theta)})\,. \tag{2}$$

Training consists of maximizing this likelihood using a gradient assent algorithm. Analytic gradi-

ents of the log likelihood (2) with respect to the various parameters are given by:

$$\begin{aligned}
\frac{\partial L(\theta)}{\partial \mathbf{u}_r} &= C\sum_{n=1}^{N} y_n \pi_n \mathbf{X}_n \mathbf{v}_r\,, \\
\frac{\partial L(\theta)}{\partial \mathbf{v}_r} &= C\sum_{n=1}^{N} y_n \pi_n \mathbf{u}_r \mathbf{X}_n\,, \\
\frac{\partial L(\theta)}{\partial \mathbf{a}_r} &= 2(1-C)\lambda_r \sum_{n=1}^{N} y_n \pi_n \mathbf{X}_n \mathbf{B}\mathbf{B}^\top \mathbf{X}_n^\top \mathbf{a}_r\,, \qquad (3) \\
\frac{\partial L(\theta)}{\partial \mathbf{b}_t} &= 2(1-C)\sum_{n=1}^{N} y_n \pi_n \mathbf{X}^\top \mathbf{A}\Lambda\mathbf{A}^\top \mathbf{X}\mathbf{b}_t\,, \qquad (4)
\end{aligned}$$

where we define

$$\pi_n = 1 - P(y_n|\mathbf{X}_n) = \frac{e^{-y_n f(\mathbf{X}_n;\theta)}}{1+e^{-y_n f(\mathbf{X}_n;\theta)}}\,,$$

and $\mathbf{u}_i, \mathbf{v}_i, \mathbf{a}_i$ and $\mathbf{b}_i$ correspond to the i_{th} columns of $\mathbf{U}, \mathbf{V}, \mathbf{A}$ and $\mathbf{B}$ respectively.

2.7 Enforcing Structure on B

If matrix $\mathbf{B}$ is constrained to have a circular Toeplitz structure then it can be represented as $\mathbf{B} = \mathbf{F}^{-1}\mathbf{D}\mathbf{F}$, where $\mathbf{F}$ denotes the orthonormal Fourier matrix with $\mathbf{F}^H = \mathbf{F}^{-1}$, and $\mathbf{D}$ is a diagonal complex-valued matrix of Fourier coefficients. In such a case we can re-write Equations (3) and (4) as

$$\begin{aligned}
\frac{\partial L(\theta)}{\partial \mathbf{a}_r} &= 2(1-C)\sum_{n=1}^{N} y_n \pi_n \mathbf{X}_n \mathbf{F}^{\mathrm{H}} \mathbf{D}\mathbf{D}^{\mathrm{H}} \mathbf{F}\mathbf{X}_n^\top \mathbf{a}_r\,. \\
\frac{\partial L(\theta)}{\partial d_i} &= 2(1-C)\sum_{n=1}^{N} y_n \pi_n \left(\mathbf{F}\mathbf{X}_n^\top \mathbf{A}\Lambda\mathbf{A}^\top \mathbf{X}_n \mathbf{F}^{\mathrm{H}}\right)_{ii} d_i\,.
\end{aligned}$$

and the parameters are now optimized with respect to Fourier coefficients $d_i = (\mathbf{D})_{i,i}$. An iterative gradient descent optimization procedure can be used to solve the minimization above.

This way of modeling $\mathbf{B}$ opens up a new perspective on the capabilities of the model. These last two equations are equally applicable for any choice of orthonormal basis $\mathbf{F}$. For example, the columns of $\mathbf{F}$ can represent a set of wavelet basis vectors. We note that a wavelet basis can be thought of as time-frequency representation of the signal; hence, proper selection of a wavelet basis allows for the method to not only capture the stationary power of the signal, but also the local changes in power within the T samples of matrix $\mathbf{X}$.

2.8 Regularization

Due to the high dimensional space in which the model lies and the limited samples available during training (typically in the order of 100), a maximum likelihood estimate of the parameters will over-train the data and have poor generalization performance. To ensure good generalization performance additional regularization criteria are required. The probabilistic formulation of Logistic Regression can incorporate regularization terms as prior probabilities resulting in maximum a posteriori (MAP) estimates.

We choose Gaussian process priors (Rasmussen and Williams, 2005) on the various parameters of the model and ensure smoothness by choosing the proper covariance matrices. Spatial and temporal smoothness is typically a valid assumption in EEG (Penny et al., 2005). Specifically, the spatial components of the model (i.e., columns of $\mathbf{U}$, and $\mathbf{A}$) follow a normal distribution with $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_u)$, $\mathbf{a}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_a)$ where the covariance matrices $\mathbf{K}_u$ and $\mathbf{K}_a$ define the degree and form of the smoothness of $\mathbf{u}$ and $\mathbf{a}$. This is done through choice of covariance function: Let r be a spatial or temporal measure in context of $\mathbf{X}$. For instance r is a measure of spatial distance between data acquisition sensors, or a measure of time difference between two samples in the data. Then a covariance function $k(r)$ expresses the degree of correlation between any two points with that given distance. For example, a class of covariance functions that has been suggested for modeling smoothness in physical processes, the Matérn class, is given by:

$$\mathrm{k}_{\text{Matérn}}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} r}{l} \right)^{\nu} \mathrm{B}\left(\frac{\sqrt{2\nu} r}{1} \right) ,$$

where l is a length-scale parameter, and ν is a shape parameter. Parameter l can be roughly though of as the distance within which points are significantly correlated (Rasmussen and Williams, 2005). The parameter ν defines the degree of ripple. The covariance matrix $\mathbf{K}$ is then built by evaluating the covariance function

$$(\mathbf{K})_{ij} = \sigma^2 \mathrm{k}_{\text{Matérn}}(r_{ij})$$

where $r_{i,j}$ denotes the physical distance of sensor-i from sensor-j, and σ^2 defines the overall scale parameter. Similarly, the Gaussian prior can be used on the columns of the temporal matrix $\mathbf{V}$ (i.e., $mv \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_v)$). The Matérn function was preferred because it allows for a low parametrization of the covariance matrix (two parameters define the entire covariance), but also because of the physical and intuitive interpretation of its parameters. Specifically the parameter l is associated with the physical concept of distance between measurements (either in space or time). This understanding of the parameters is useful since it allows for an educated search strategy in setting the proper values for these parameters.

Regularizing logistic regression amounts to minimizing the negative log-likelihood plus the negative log-priors, which can be written as:

$$\arg \min_{\mathbf{U},\mathbf{V},\mathbf{A},\mathbf{B},w_o} -L(\theta) + \frac{1}{2} \left(\sum_{r=1}^{R} \mathbf{u}_r^\top \mathbf{K}_u^{-1} \mathbf{u}_r + \mathbf{v}_r^\top \mathbf{K}_v^{-1} \mathbf{v}_r + \sum_{k=1}^{K} \mathbf{a}_k^\top \mathbf{K}_a^{-1} \mathbf{a}_k + \sum_{t=1}^{T'} \mathbf{b}_t^\top \mathbf{K}_t^{-1} \mathbf{b}_t \right) , \qquad (5)$$

where we ignored constants that have no effect in the optimization. The covariances of these priors are given by $\mathbf{K}_u, \mathbf{K}_a \in \mathbb{R}^{D \times D}$ and $\mathbf{K}_v, \mathbf{K}_b \in \mathbb{R}^{T \times T}$ and control the smoothness of the parameter space. In the case of the spectral regularization we use the identity matrix for the covariance, $\mathbf{K}_b = \sigma^2 \mathbf{I}$, since the smoothness assumption does not necessarily hold in the spectral domain.

Following Rasmussen and Williams (2005) the shape parameter was chosen to be $\nu = 100$ for the spatial components and $\nu = 2.5$ for the temporal components. Reasonable choices for the length-scale parameter l may be 25ms, 50ms or 100ms and in space 1cm, 2cm, and 3cm. Cross-validation was used to select among these choices. The overall scale parameters σ were chose to be the same for space and time components, but allowed to take on separate values for the first and second order component. We used a line-search procedure in combination with cross-validation to select appropriate values for σ.

2.9 Optimization

Optimization (5) is achieved using a coordinate decent type algorithm (Nielsen, 2005) with parameters $\mathbf{U}, \mathbf{V}$ and $\mathbf{A}, \mathbf{B}$ optimized separately. We obtain analytic expressions for both the gradient and the Hessian of the function, however, in the optimization only the gradient information is used.[1] We first optimize the parameters $\mathbf{U}$ and $\mathbf{V}$, then optimize parameters $\mathbf{A}$ and $\mathbf{B}$ and finally perform a line search to determine the value of C.

Given that the optimization function is non-convex, the gradient decent method only finds local minima. In fact, the performance of SOBDA is particularly sensitive to the starting conditions of the spectral parameter $\mathbf{d}$ (parameter $\mathbf{d}$ enters the model when enforcing a Toeplitz structure on $\mathbf{B}$, see section 2.7.), while it is quite robust to the choice of initial conditions for the remaining parameters $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{A}$. A common technique in global optimization is to use parameter seeding and multiple runs of the optimization procedure. For most parameters it was sufficient to try a few random initial starting points. However, for the spectral parameter we found it important to initialize to a frequency band that was expected to carry useful information, for example, 8Hz-30Hz. Note that the present learning task falls into the class of bi-convex optimization problems for which efficient algorithms have been developed (Floudas, 1997).

3. Results

We evaluated our algorithm on 3300 simulated data sets as well as 6 real EEG recordings, including a data set used in the Brain Computer Interface Competitions II (Blankertz et al., 2004). The simulation aims to quantify the algorithm's performance on a broad spectrum of conditions and various noise levels, as well as to compare the extracted spatial, temporal and frequency components with ground truth. The evaluation on real data set compares the cross-validation performance of the proposed method with three popular methods used in EEG analysis and BCI. Results show that our method outperformed these methods, decreasing the overall classification error rates from 26%-28% to 19%. For the data set of the BCI competition we also report performance results on the independent test set and compare to the previous results.

The three methods we will compare with are Bilinear Discriminant Component Analysis (BDCA) (Dyrholm et al., 2007), Common Spatial Patterns (CSP) (Ramoser et al., 2000), and Matrix Logistic Regression (MLR) (Tomioka et al., 2007). For the evaluation on the 6 real EEG data sets, we further compare our method to the trace norm regularized Matrix Logistic Regression (sMLR) (Tomioka and Aihara, 2007). These may be considered current state-of-the art methods in EEG single-trial analysis. In our evaluation we use a rank one approximation for the BDCA as in Dyrholm et al. (2007). We implemented CSP following the description of Ramoser et al. (2000). We used two spatial patterns (SP) and employ a logistic regression classifier on the resulting SP. In the case of MLR we use the rank-2 approximation as described in the corresponding paper (Tomioka et al., 2007). For sMLR we used the implementation provide in Tomioka and Aihara (2007). Since CSP,MLR and sMLR require the data to be band-pass filtered to the frequency of interest, data sets where filtered in the range of 8Hz-30Hz for these two methods. For our algorithm we use rank-1 for the first-order parameters $\mathbf{U}$ and $\mathbf{V}$ with $R = 1$. For the spatial parameter $\mathbf{A}$ we set $K = 2$ allowing for two spatial patterns, while we enforce a Toeplitz structure on $\mathbf{B}$. We initialize the parameters

1. We discard the Hessian information because of its computational cost and the non-convexity of the optimization function. The Hessian of a non-convex function would need to be approximated by a positive definite matrix in each iteration.

$\mathbf{U}, \mathbf{V}$ and $\mathbf{A}$ by a random assignment. While we initialize the matrix $\mathbf{B}$ to encode a band-pass filter in the range of $8Hz - 30Hz$ as in the case of CSP, MLR, sMLR. As discussed in Section 2.7, enforcing a Toeplitz structure on $\mathbf{B}$ implies a representation of $\mathbf{B}$ in the form $\mathbf{B} = \mathbf{F}^{-1}\mathbf{D}\mathbf{F}$, where $\mathbf{F}$ denotes the orthonormal Fourier matrix with $\mathbf{F}^{\mathrm{H}} = \mathbf{F}^{-1}$, and $\mathbf{D}$ is a diagonal complex-valued matrix of Fourier coefficients. In our implementation, we optimize the coefficients of the matrix $\mathbf{D}$ instead of $\mathbf{B}$ directly.

3.1 Simulated EEG Data

Simulated data for a two-class problem was generated using standard EEG simulation software (GmbH, 2006). This software can generate electrode measurements under the assumption of dipolar current sources in the brain. We used 3 dipoles at three different locations, with one dipole used to generate evoked response activity, one dipole to generate induced oscillatory activity, and one dipole to generate unrelated noise/interference. The first dipole's component simulates a P300 evoked response potential (ERP) signal. We used a half-sinusoid lasting 125ms with the peak positioned at 300ms after trial-onset and a trial-to-trial Gaussian temporal jitter with standard deviation of 10ms. The second dipole's component simulates ERS/ERD in the frequency band of 8Hz to 30Hz. A variable signal in this frequency band was generated by bandpass filtering an uncorrelated Gaussian process. The third dipole was used to generate noise in the source space representing brain activity that is not related to the evoked/induced activity. Electric potentials at $D = 31$ electrode locations were generated corresponding to 500ms of EEG signal sampled at 100Hz ($T = 50$ samples). In addition to this rank-one noise we added noise to each sensor representing other sources of noise (muscle activity, skin potentials, inductive noise, amplifier noise, etc.). All noise sources were white. Trials belonging to the first class ($y_n = +1$) contained the ERP and ERD/ERS source signals scaled appropriately to achieve a specified SNR for each data set. The second class was generated by only including the noise with no ERP or ERD/ERS activity. A data set is specified by indicating the SNR for the ERP component and the SNR for the ERD/ERS component. A total of 500 trials for each class were generated for each classification problem. The SNR of the ERP component is in the range of -33dB to -13dB, and in the range of -22dB to -10dB for the oscillatory component. This is a very broad range in terms of SNR. We note that -20dB translates to the signal being 10 times smaller than the noise. ERP signals are known to be as low as $-20dB$ so this evaluation captures some extreme cases of SNR. We generated 35 data sets for each combination of SNR resulting to a total of 3300 data sets.

3.2 Performance Results on Simulated Data

The simulation results are summarized in Figure 1. The top two rows show the performance of each of the methods as a function of the SNR. The contours of the classification performance for each method as a function of the SNR of the first-order and the second-order components are shown. It is clear that BDCA performance is only affected by the noise in the linear term while CSP and MLR performance only changes as a function of the second-order component's SNR. SOBDA however, uses both first and second-order terms, hence performs well in data sets where at least one of the components has reasonable SNR. This finding confirms that SOBDA performs well in a broader range of SNRs than the other three competitive methods. The third row in 1 shows the difference in classification performance between SOBDA vs (BDCA,CSP,MLR).

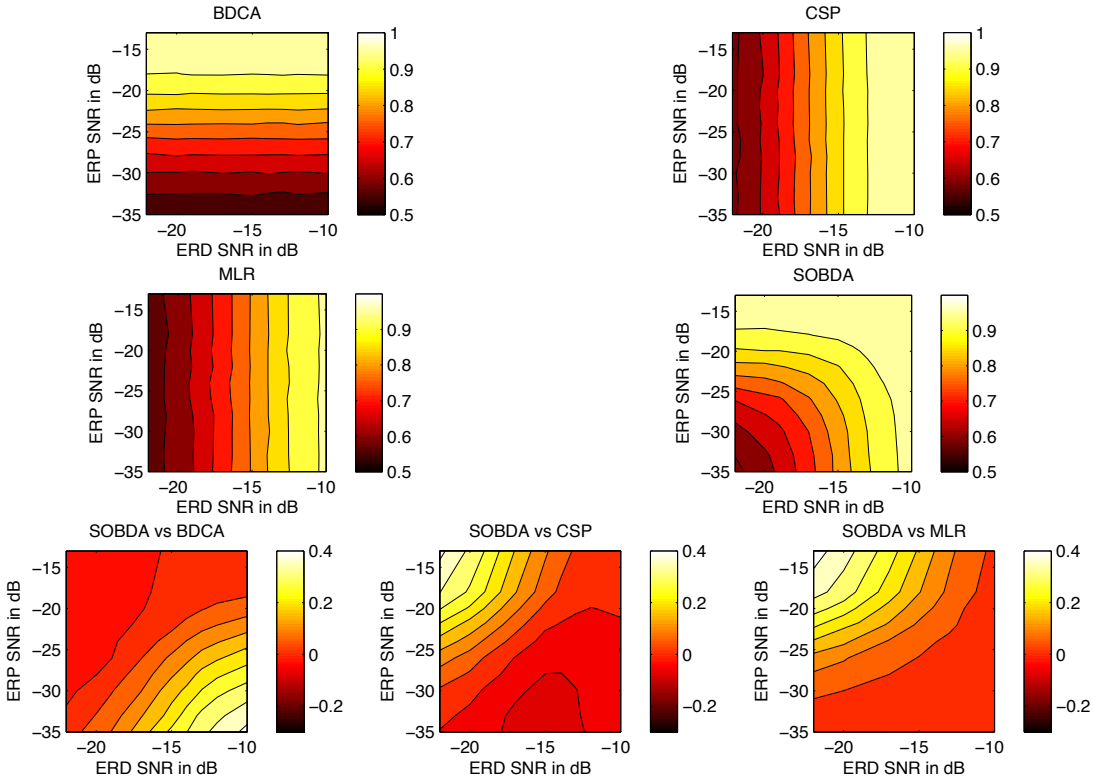


Figure 1: Performance results on simulated data. *Second and third row:* Probability of correct classification (Pc) as a function of the component's SNR. SOBDA equi-performance contours span larger area in the SNR space than any of the other three algorithms. *Third row:* Difference in Pc performance between SOBDA and each of the three methods as a function of components SNR.

As a decomposition method, SOBDA extracts spatial, temporal and frequency components. The advantage of simulated data is that we can now compare the extracted information to ground truth. The component recovered for one of the data sets at $-22dB$ and $-15dB$ is shown in figure 2. The first row shows the extracted temporal component $\mathbf{U}$ and the frequency component $\mathbf{d}$.[2] We can see that the method extracted a temporal component with a peak at 300ms which is exactly the signal used in the simulation data design. Similarly, the frequency band extracted shows a higher amplitude in the range of 8Hz-30Hz which is the band used to generate the oscillatory component. The spatial components extracted and the corresponding dipole used in the model generation are shown in rows two and three in the figure. It is clear that the topography of the extracted components is similar for the first and second-order components. The last column of the figure captures the second-order oscillatory component and the dipole of the rank one noise. Visual inspection allows one to give neurological interpretations to the extracted components. Further, the results can be used as input to

2. $\mathbf{d}$ the vector of diagonal elements of matrix $\mathbf{D}$, such that $\mathbf{B} = \mathbf{F}^{\mathrm{H}}\mathbf{D}\mathbf{F}$

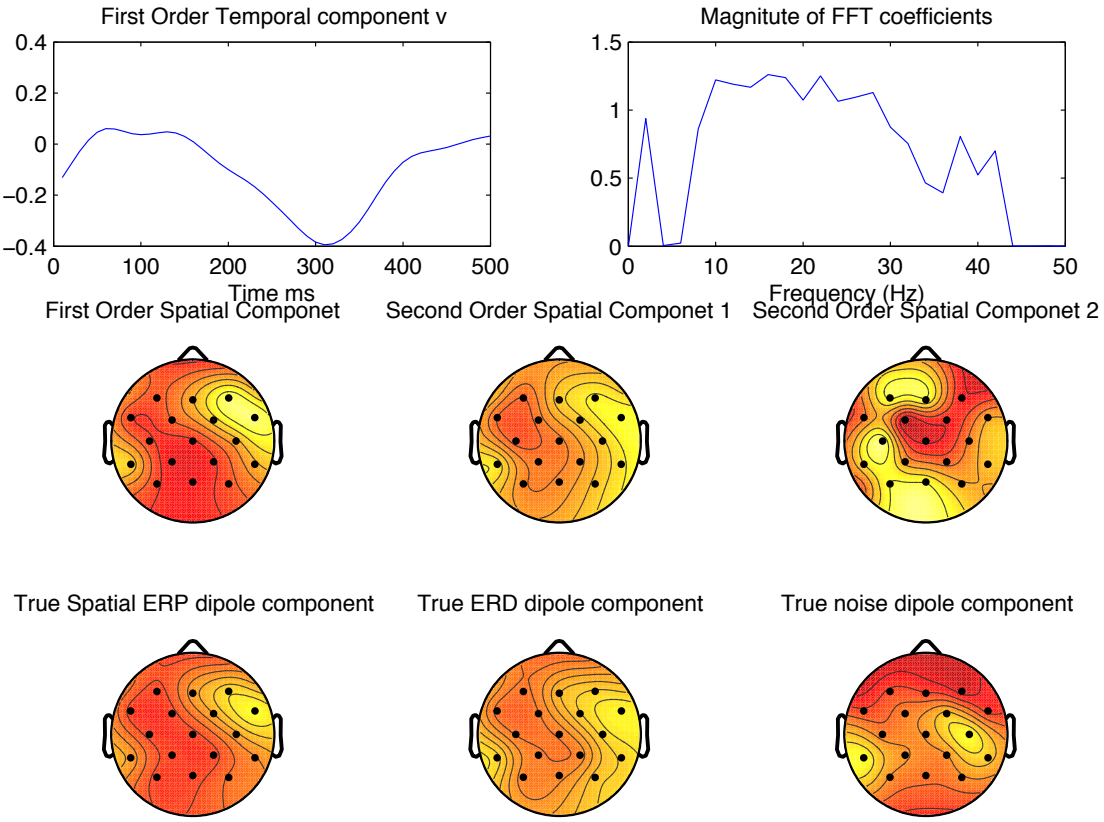


Figure 2: Extracted components on simulated data set with first-order SNR at $-22dB$ and second-order SNR at $-15dB$. *Top row:* Extracted temporal weight of linear term (left) and frequency weights of quadratic term (right). *Center row:* Extracted spatial weights. *Bottom row:* Distribution of electric potentials corresponding to the three dipoles used during stimulus generation.

a source localization algorithm, or as a guide to reduce the number of electrodes in a brain computer interface.

3.2.1 GENERIC INITIALIZATION OF FREQUENCY COMPONENT

In the evaluation presented above, we initialized the matrix $\mathbf{B}$ to encode a band-pass in the range of 8Hz - 30Hz as it was the case for CSP and MLR. In this section we demonstrate the ability of the proposed SOBDA in cases where no initialization information is available. Specifically, we evaluated the SOBDA algorithm on a simulated data set using the process described above, but this time we initialize matrix B to a high-pass filter with cut of frequency at 1 Hz. High pass filtering is a standard preprocessing steps in EEG that removes the DC power. Figure 3 shows the temporal and frequency component obtained from SOBDA. As it is evident from the figure, the resulting frequency component has higher weights for frequencies in the band 8Hz-30Hz, which is the band used to generate the power component in the simulated data. Thus the proposed method is able optimize the frequency band even in cases where we use a generic initialization of the matrix $\mathbf{B}$.

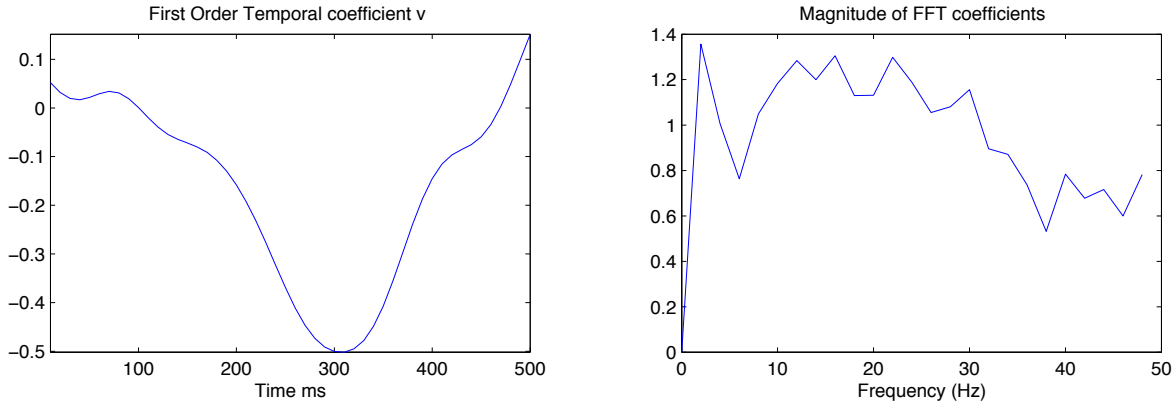


Figure 3: Discriminant coefficients on simulated data set with first-order SNR at $-22dB$ and second-order SNR at $-15dB$. The Fourier coefficients were initialized to a high-pass filter with cut off frequency at 1 Hz *Left figure:* Extracted temporal weight of linear term.*Right figure:* Magnitude of the Fourier coefficients in $\mathbf{D}$, such that $\mathbf{B} = \mathbf{F}^{\mathrm{H}}\mathbf{D}\mathbf{F}$.

3.3 Human Subject EEG

To evaluate the performance of the proposed method on real data we first applied the algorithm to an EEG data set that was made available through The BCI Competition 2003 (Blankertz et al., 2004, Data Set IV). EEG was recorded on 28 channels for a single subject performing "self-paced key typing", that is, pressing with the index and little fingers corresponding keys in a self-chosen order and timing. Key-presses occurred at an average speed of 1 key per second. Trial matrices were extracted by epoching the data starting 630ms before each key-press. A total of 416 epochs were recorded, each of length 500ms. For the competition, the first 316 epochs were used for classifier training, while the remaining 100 epochs were used as a test set. Data was recorded at 1000Hz with a pass-band between 0.05 and 200Hz, then down sampled to 100Hz sampling rate.

For this experiment, the matrix $\mathbf{B}$ was fixed to a Toeplitz structure that encodes a 10Hz-33Hz bandpass filter and only the parameters $\mathbf{U}, \mathbf{V}, \mathbf{A}$ and w_0 were trained. The number of columns of $\mathbf{U}$ and $\mathbf{V}$ were set to $R = 1$ and the number of columns for $\mathbf{A}$ was set to $K = 2$. The selection of these parameters is motivated by the task at hand. Specifically, we are looking for one ERP components associated with the *readiness* potential that is, the slow increase in amplitude before an actual hand movement. In the case of the second-order term involving the parameter $\mathbf{A}$ we set $K = 2$ because we are interested in finding the modulation of oscillatory activity associated with the different movements of the movements of the hands. Hands and fingers are represented in somato-sensory cortex covering different areas and will hence modulate activity in distinct spatial profiles. In order to detect the power difference of these two components we set, $\Lambda = [1,0;0,-1]$, in agreement with the original approach of Wolpaw et al. (2002).

The temporal filter was selected based on prior knowledge of the relevant frequency band. This demonstrates the flexibility of our approach to either incorporate prior knowledge when available or extract it from data otherwise. Regularization parameters where chosen via a five fold-cross validation procedure as described in Section 2.8.

Benchmark performance was measured on the test set which had not been used during either training or cross-validation. The number of misclassified trials in the test set was 13 which places

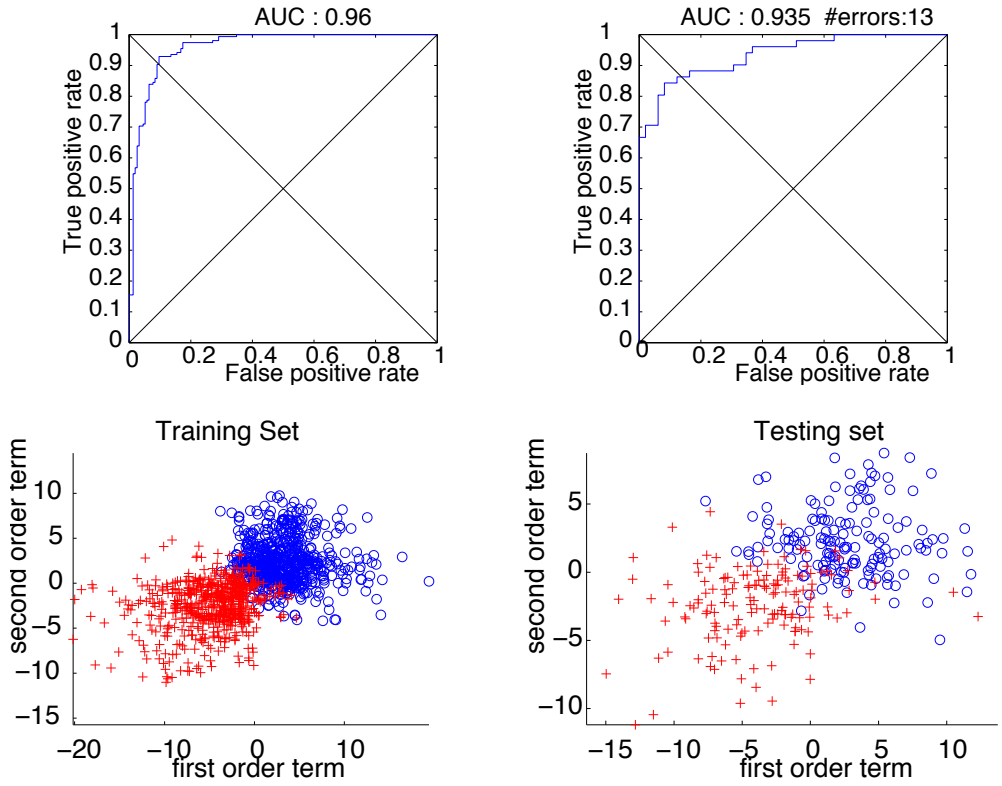


Figure 4: Results on human EEG for BCI. *Top row:* Cross-validation performance shown as ROC curve with area under the curve of 0.96 for the benchmark data set (left) and 0.93 for the independent test set (right). There were a total of 13 errors on unseen data, which is less than any of the results previously reported. *Bottom row:* Scatter plot of the first-order term vs second-order term of the model, on the training and testing set for the benchmark data set ('+' left key, and 'o' right key). It is clear that the two types of features contain independent information that can help improve the classification performance.

our method in a new first place ranking, based on the results of the competition (Blankertz et al., 2004). The receiver-operator characteristic curve (ROC) for cross-validation and for the independent test set are shown in Figure 4. The Figure also shows the contribution of the linear and quadratic terms for every trial for the two types of key-presses.

To further validate our method we performed our own EEG recordings asking subjects now to respond with the left and right index fingers. We obtain five more data sets with the same number of electrodes. For each data set and each algorithm we performed 20 repetitions of a five-fold cross-validation procedure. Each repetition uses a different partitioning of the data. For the cross-validation evaluation of these data sets, we initialized (but did not fix) matrix $\mathbf{B}$ to a Toeplitz structure that encodes a 10Hz-33Hz bandpass filter and trained over all parameters $\mathbf{U}, \mathbf{V}, \mathbf{A}, \mathbf{B}$ and w_0.[3] The number of columns of $\mathbf{U}$ and $\mathbf{V}$ were set to 1, where two columns were used for $\mathbf{A}$. This corresponds to the parameter configuration of $R = 1, K = 2$ and $T' = T$.

3. We remind the reader that in the actual implementation we optimize the Fourier coefficients $\mathbf{D}$ instead of matrix $\mathbf{B}$

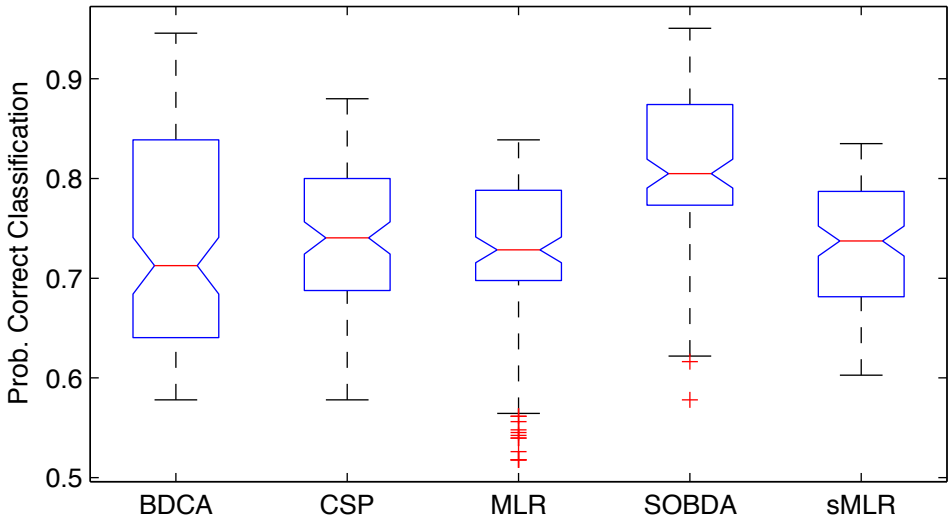


Figure 5: Estimate of the spread of the probability of correct identification from multiple cross-validation repetitions. Lines show lower quartile, median, and upper quartile values for each of the methods on all data sets. + symbols represent outliers.

Figure 5 shows performance distribution across these bootstrap repetitions using a standard boxplot. The mean performance and standard deviation of each data set and algorithm are summarized in table 1. The reduction in the overall classification error is from 26%-28% to 19%. In the mean, SOBDA outperforms competitive methods in five out of the six data sets, while achieving a comparable performance on data set 2. The performance optained with SOBDA is comparable to performance gains that may be obtained by combining existing first and second order methods (e.g., CSP and BDCA—data not shown).

Figure 6 shows the extracted components for 3 of the 6 data sets. We note that in all three cases the extracted components follow the general shape of the pre-motor or readiness potential (a.k.a. Bereitschafts potential) which known to precede a voluntary muscle movement. In addition, for two of the data sets, the frequency weightings suggest that alpha band activity also provides discriminant information for this task. This finding is consistent with the changes in the μ rhythm—that is, alpha-band activity localized over the motor cortex and associated with motor planning and execution. This demonstrates the ability of our method to learn first and second-order features that are consistent with, and can be linked to existing knowledge of the underlying neuronal signal generators.

Experiment	BDCA	CSP	MLR	SOBDA	sMLR
1	0.84 ± 0.011	0.8 ± 0.017	0.82 ± 0.011	$\mathbf{0.88 \pm 0.013}$	0.78 ± 0.0089
2	0.69 ± 0.037	$\mathbf{0.84 \pm 0.017}$	0.77 ± 0.028	0.83 ± 0.021	0.82 ± 0.012
3	$\mathbf{0.63 \pm 0.018}$	0.62 ± 0.016	0.55 ± 0.02	$\mathbf{0.63 \pm 0.017}$	0.62 ± 0.015
4	0.72 ± 0.021	0.78 ± 0.015	0.77 ± 0.015	$\mathbf{0.79 \pm 0.018}$	0.76 ± 0.021
5	0.64 ± 0.018	0.7 ± 0.022	0.7 ± 0.011	$\mathbf{0.78 \pm 0.013}$	0.73 ± 0.0097
6	0.93 ± 0.01	0.7 ± 0.016	0.72 ± 0.01	$\mathbf{0.94 \pm 0.0089}$	0.68 ± 0.0056
Mean	0.7412	0.7388	0.7213	0.8068	0.7316

Table 1: Probability of correct identification for the six EEG data sets obtained by each of the four methods. The last row indicates the percentage of decrease in the classification error achieved by SOBDA compared to each one of the methods. $\pm$ range indicates one standard deviation for results of multiple cross-validation repetitions.

4. Rank-Selection

In our results, we selected the rank for the parameters $\mathbf{U}$ and $\mathbf{V}$ to be one (i.e., $R = 1$) and the rank for the parameter $\mathbf{A}$ to be two (i.e., $K = 2$). The selection of these parameters was motivated in Section 3.3. Specifically, in the current experimental paradigm, we are looking for one ERP components associated with the *readiness* potential, that is, the slow increase in amplitude before an actual hand movement. The search for a single component suggests setting $R = 1$, one spatio-temporal component. In the case of the second-order term involving the parameter $\mathbf{A}$ we set the $K = 2$ because we are interested in finding two components corresponding to the two different spatial profiles of the two classes. To validate our selection for these parameters, we preformed repeated cross-validation evaluation of our algorithm for different configurations of the parameters R and K. The parameter R was tested for values $\{1,2,3,4\}$ while parameter K was tested for $\{2,4\}$. The results of this evaluation are summarized in Figure 7. The Figure 7.a shows the mean cross-validation performance of the SOBDA algorithm across all real-EEG data sets for all configurations of the parameters R and K. It is evident from this figure that configuration $R = 1, K = 2$ corresponds to the best selection for these parameters on average for these data sets. The Figure 7.b shows the cross-validation performance of the SOBDA algorithm for each data set separately and for all configuration of the parameters R and K. The cross-validation procedure can be used to determine or validate the configuration of parameters R and K in cases were no prior knowledge is available about the signal of interest.

5. Conclusion

In this paper we presented a new method called Second-Order Bilinear Discriminant Analysis (SOBDA) for analyzing EEG signals on a single-trial basis. The method combines linear and quadratic features thus encompassing and extending a number of existing EEG analysis methods. We evaluated the SOBDA algorithm in both simulated and real human EEG data sets. We show a reduction in the classification error on human EEG when comparing our method to the state-of-the-art. The results on simulated data characterize the operational range of these algorithms in terms of SNR and shows that the proposed algorithm operates well where other methods fail. The parametrization

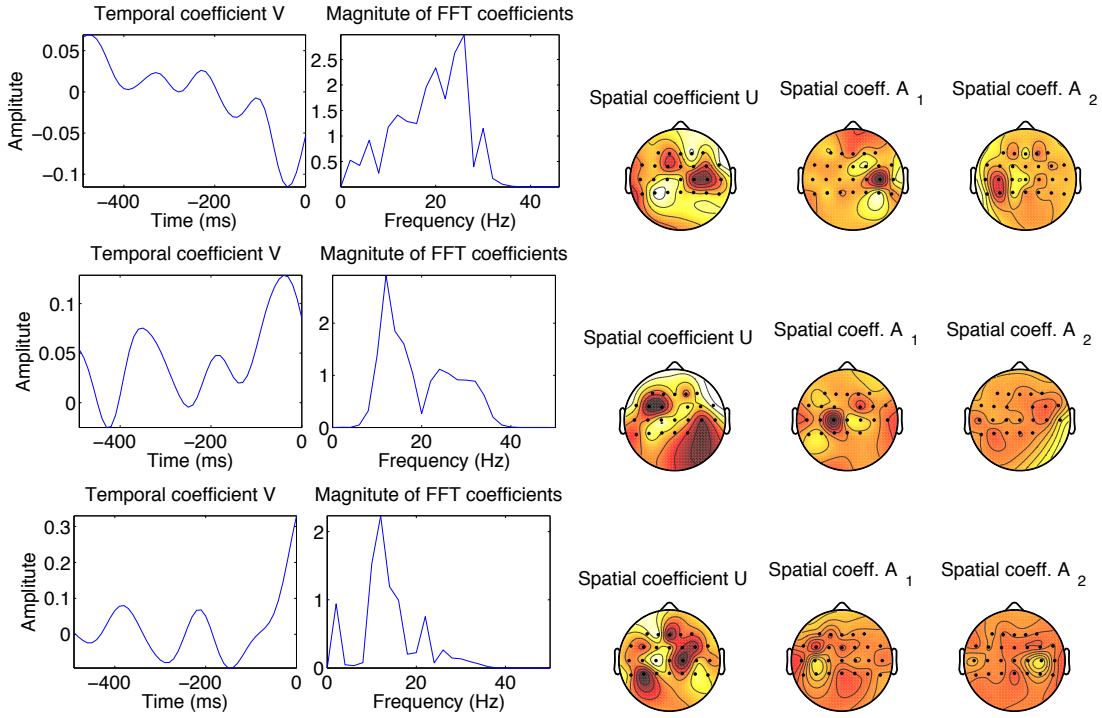


Figure 6: Extracted components in EEG for data sets 6, 4, and 3. *Left:* Temporal weights of linear component (first column) and and frequency weights of quadratic component (second column). *Right:* Spatial weights of linear component (third column) and two spatial weights for second-order spatial components (fourth and fifth column).

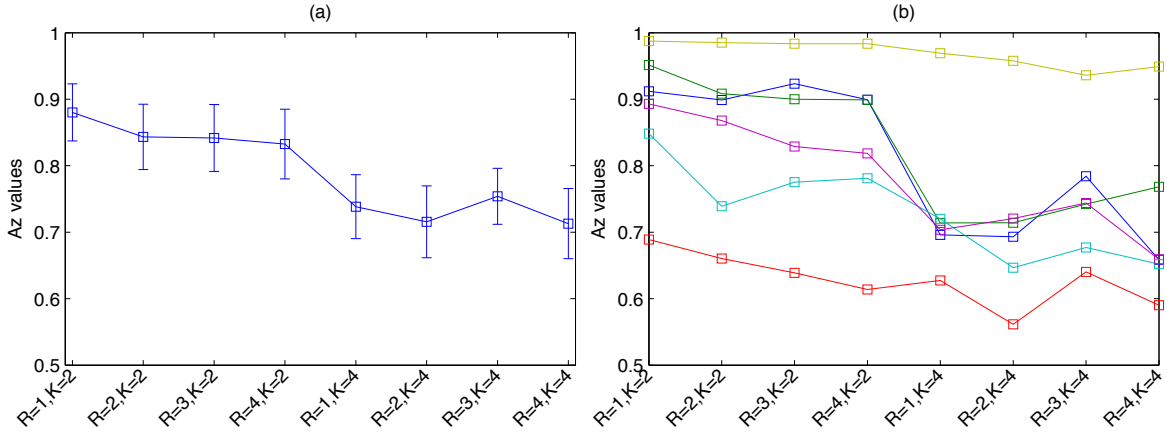


Figure 7: Cross-validation performance of SOBDA on the six real-EEG data sets used in the evaluation, at various configuration of the parameters R and K. (a) The mean cross-validation performance across data sets at various configuration of the parameters R and K.(b) Cross-validation performance for each of the data sets at various configuration of the parameters R and K.

of the discriminant criterion is intuitive, allowing one to incorporate prior knowledge as well as to derive spatial, temporal, and spectral information about the underlying neurological activity.

6. Derivations

In this section we derive the analytic gradient formulas of the negative log-likelihood function defined in (2). In general the gradient with respect to any of the variables can be expressed as:

$$\begin{aligned}\frac{\partial L(\theta)}{\partial \theta} &= -\sum_{n=1}^{N} \frac{\partial \log(1+e^{-y_n f(\mathbf{X}_n;\theta)})}{\partial \theta} \\ &= -\sum_{n=1}^{N} \frac{1}{1+e^{-y_n f(\mathbf{X};\theta)}} \frac{\partial\{1+e^{-y_n f(\mathbf{X}_n;\theta)}\}}{\partial \theta} \\ &= \sum_{n=1}^{N} y_n \frac{e^{-y_n f(\mathbf{X};\theta)}}{1+e^{-y_n f(\mathbf{X};\theta)}} \frac{\partial f(\mathbf{X}_n;\theta)}{\partial \theta},\end{aligned}$$

Now one has to take the specific derivatives with respect to each of the variables in θ is:

The gradient with respect to $\mathbf{u}_r$, the rth column of $\mathbf{U}$.

$$\begin{aligned}\frac{\partial\{f(\mathbf{X}_n;\theta)+w0\}}{\partial \mathbf{u}_r} &= C\frac{\partial\{\mathrm{Trace}\,\mathbf{U}^\top \mathbf{X}_n \mathbf{V}\}}{\partial \mathbf{u}_r} \\ &= C\frac{\partial\{\sum_{r'=1}^{R} \mathbf{u}_{r'}^\top \mathbf{X}_n \mathbf{v}_{r'}\}}{\partial \mathbf{u}_r} \\ &= C\mathbf{X}_n \mathbf{v}_r\,.\end{aligned}$$

The gradient with respect to $\mathbf{v}_r$, the *rth* column of $\mathbf{V}$ is:

$$\begin{aligned}\frac{\partial\{f(\mathbf{X}_n;\theta)+w0\}}{\partial \mathbf{v}_r} &= C\frac{\partial\{\mathrm{Trace}\,\mathbf{U}^\top \mathbf{X}_n \mathbf{V}\}}{\partial \mathbf{v}_r} \\ &= C\frac{\partial\{\sum_{r'=1}^{R} \mathbf{u}_{r'}^\top \mathbf{X}_n \mathbf{v}_{r'}\}}{\partial \mathbf{v}_r} \\ &= C\mathbf{u}_r^\top \mathbf{X}_n\,.\end{aligned}$$

The gradient with respect to $\mathbf{a}_r$, the *rth* column of $\mathbf{A}$ is:

$$\begin{aligned}\frac{\partial\{f(\mathbf{X}_n;\theta)+w0\}}{\partial \mathbf{a}_r} &= (1-C)\frac{\partial\{\mathrm{Trace}\,\mathbf{A}^\top(\mathbf{X}_n\mathbf{B})(\mathbf{X}_n\mathbf{B})^\top\mathbf{A}\}}{\partial \mathbf{a}_r} \\ &= (1-C)\frac{\partial\{\sum_{r'=1}^{K} \lambda_{r'}\mathbf{a}_{r'}^\top(\mathbf{X}_n\mathbf{B})(\mathbf{X}_n\mathbf{B})^\top\mathbf{a}_{r'}\}}{\partial \mathbf{a}_r} \\ &= 2(1-C)\lambda_r(\mathbf{X}_n\mathbf{B})(\mathbf{X}_n\mathbf{B})^\top\mathbf{a}_r\,,\end{aligned}$$

The gradient with respect to $\mathbf{b}_r$, the rth column of $\mathbf{B}$ is:

$$\begin{aligned}
\frac{\partial\{f(\mathbf{X}_n;\theta)+w0\}}{\partial \mathbf{b}_r} &= (1-C)\frac{\partial\{\text{Trace}\,\Lambda\mathbf{A}^\top(\mathbf{X}_n\mathbf{B})(\mathbf{X}_n\mathbf{B})^\top\mathbf{A}\}}{\partial \mathbf{b}_r} \\
&= (1-C)\frac{\partial\{\text{Trace}\,\mathbf{B}^\top\mathbf{X}_n^\top\mathbf{A}\Lambda\mathbf{A}^\top\mathbf{X}_n\mathbf{B}\}}{\partial \mathbf{b}_r} \\
&= (1-C)\frac{\partial\{\sum_{r'=1}^{K}\mathbf{b}_{r'}^\top\mathbf{X}_n^\top\mathbf{A}^\top\Lambda\mathbf{A}^\top\mathbf{X}_n\mathbf{b}_{r'}\}}{\partial \mathbf{b}_r} \\
&= 2(1-C)(\mathbf{X}_n^\top\mathbf{A}\Lambda\mathbf{A}^\top\mathbf{X}_n)\mathbf{b}_r\,.
\end{aligned}$$

Acknowledgments

This work was funded by DARPA (contract #:NBCHCD80029).

References

N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kubler, J. Perelmouter, E. Taub, and H. Flor. A spelling device for the paralysed. *Nature*, 398(6725):297–8, Mar FebruaryMay 1999.

B. Blankertz, G. Curio, and K. Müller. Classifying single trial EEG: Towards brain computer interfacing. In T. G. Diettrich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*. MIT Press, 2002., 2002.

B. Blankertz, G. Dornhege, C. Schfer, R. Krepki, J. Kohlmorgen, K. Müller, V. Kunzmann, F. Losch, and G. Curio. Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial EEG analysis. *IEEE Trans. Neural Sys. Rehab. Eng.*, 11(2): 127–131, 2003.

B. Blankertz, K.-R. Müller, G. Curio, T.M. Vaughan, G. Schalk, J.R. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schroder, and N. Birbaumer. The bci competition 2003: progress and perspectives in detection and discrimination of EEG single trials. *Biomedical Engineering, IEEE Transactions on*, 51(6):1044–1051, 2004.

G. Dornhege, Blankertz B, and K.R. Krauledat M. Losch F. Curio G.Müller. Combined optimization of spatial and temporal filters for improving brain-computer interfacing. *IEEE Trans. Biomed. Eng. 2006*, 2006.

M. Dyrholm and L.C. Parra. Smooth bilinear classification of EEG. In *In Proc. 28th Annu. Int Conf. IEEE Engineering in Medicine and Biology Society*, 2006.

M. Dyrholm, C. Christoforou, and L.C. Parra. Bilinear discriminant component analysis. *J. Mach. Learn. Res.*, 8:1097–1111, 2007. ISSN 1533-7928.

C.A. Floudas. Deterministic global optimization in design, control, and computational chemistry. *In Proceedings: Large Scale Optimization with Applications. Part III: Optimal Design and Control, (L.T. Biegler, A. Conn, L. Coleman, and F. Santosa, Editors)*, pages 129–184, 1997.

A.D. Gerson, L.C. Parra, and P. Sajda. Cortically-coupled computer vision for rapid image search. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14:174–179, June 2006.

MEGIS Software GmbH. BESA. `http://www.besa.de/products/besa/`, 2006.

S. Lemm, B. Blankertz, G. Curio, and K. Müller. Spatio-spectral filters for improving the classification of single trial EEG. *IEEE Trans Biomed Eng., 52(9):1541–8, 2005.*, 2005.

A. Luo and P. Sajda. Learning discrimination trajectories in EEG sensor space: Application to inferring task difficulty. *J. Neural Eng.*, 3:L1–L6, 2006a.

A. Luo and P. Sajda. Using single-trial EEG to estimate the timing of target onset during rapid serial visual presentation. In *Proc. Engineering in Medicine and Biology Society(EMBC2006)*, 2006b.

H. B. Nielsen. IMMOPTIBOX. General optimization software available at `http://www.imm.dtu.dk/ hbn/immoptibox/`, 2005.

L. Parra, C. Alvino, A. Tang, B. Pearlmutter, N. Young, A. Osman, and P. Sajda. Linear spatial integration for single-trial detection in encephalogra phy. *Neuroimage*, 17:223–230, 2002.

L.C. Parra, C.D. Spence, A.D Gerson, and P. Sajda. Recipes for the linear analysis of EEG. *Neuroimage*, 28(2):326–341, November 2005. ISSN 1053-8119.

L.C. Parra, C. Christoforou, A.D. Gerson, M. Dyrholm, A. Luo, M. Wagner, M.G. Philiastides, and P. Sajda. Spatiotemporal linear decoding of brain state: Application to performance augmentation in high-throughput tasks. *IEEE, Signal Processing Magazine*, January 2008.

W. D. Penny, N. J. Trujillo-Barreto, and K. J. Friston. Bayesian fMRI time series analysis with spatial priors. *NeuroImage*, 24:350362, 2005.

G. Pfurtscheller and F. H. Lopes da Silva. Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clin Neurophysiol*, 110(11):1842–1857, November 1999. ISSN 1388-2457.

M.G. Philiastides and P. Sajda. Temporal characterization of the neural correlates of perceptual decision making in the human brain. *Cerebral Cortex*, 16(4), April 2006.

M.G. Philiastides, R. Ratcliff, and P. Sajda. Neural representation of task difficulty and decision making during perceptual categorization: A timing diagram. *Journal of Neuroscience*, 26(35): 8965–8975, August 2006.

H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Trans. Rehab. Eng.*, 8:441–446, December 2000. URL `citeseer.ist.psu.edu/ramoser98optimal.html`.

C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

R. Tomioka and K. Aihara. Classifying matrices with a spectral regularization. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 895–902, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: http://www.ibis.t.u-tokyo.ac.jp/ryotat/lrds.

R. Tomioka, K. Aihara, and K. Müller. Logistic regression for single trial EEG classification. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1377–1384. MIT Press, Cambridge, MA, 2007.

J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain-computer interfaces for communication and control. *Clin Neurophysiol*, 113(6):767–791, June 2002. ISSN 1388-2457.

Journal of Machine Learning Research 11 (2010) 687-712 Submitted 6/09; Revised 11/09; Published 2/10

On the Rate of Convergence of the Bagged Nearest Neighbor Estimate

Gérard Biau GERARD.BIAU@UPMC.FR
LSTA & LPMA
Université Pierre et Marie Curie – Paris VI
Boîte 158, 175 rue du Chevaleret
75013 Paris, France

Frédéric Cérou FREDERIC.CEROU@INRIA.FR
INRIA Rennes Bretagne Atlantique
Aspi project-team
Campus de Beaulieu
35042 Rennes Cedex, France

Arnaud Guyader ARNAUD.GUYADER@UHB.FR
Université Rennes II – Haute Bretagne
Campus Villejean
Place du Recteur Henri Le Moal, CS 24307
35043 Rennes Cedex, France

Editor: Gábor Lugosi

Abstract

Bagging is a simple way to combine estimates in order to improve their performance. This method, suggested by Breiman in 1996, proceeds by resampling from the original data set, constructing a predictor from each subsample, and decide by combining. By bagging an n-sample, the crude nearest neighbor regression estimate is turned into a consistent weighted nearest neighbor regression estimate, which is amenable to statistical analysis. Letting the resampling size k_n grows appropriately with n, it is shown that this estimate may achieve optimal rate of convergence, independently from the fact that resampling is done with or without replacement. Since the estimate with the optimal rate of convergence depends on the unknown distribution of the observations, adaptation results by data-splitting are presented.

Keywords: bagging, resampling, nearest neighbor estimate, rates of convergence

1. Introduction

Ensemble methods are popular machine learning algorithms which train multiple learners and combine their predictions. The success of ensemble algorithms on many benchmark data sets has raised considerable interest in understanding why such methods succeed and identifying circumstances in which they can be expected to produce good results. It is now well known that the generalization ability of an ensemble can be significantly better than that of a single predictor, and ensemble learning has therefore been a hot topic during the past years. For a comprehensive review of the domain, we refer the reader to Dietterich (2000) and the references therein.

1.1 Bagging

One of the first and simplest ways to combine predictors in order to improve their performance is bagging (**b**ootstrap **agg**regat**ing**), suggested by Breiman (1996). This ensemble method proceeds by generating subsamples from the original data set, constructing a predictor from each resample, and decide by combining. It is one of the most effective computationally intensive procedures to improve on unstable estimates or classifiers, especially for large, high dimensional data set problems where finding a good model in one step is impossible because of the complexity and scale of the problem. Bagging has attracted much attention and is frequently applied, although its statistical mechanisms are not yet fully understood and are still under active investigation. Recent theoretical contributions to bagging and related methodologies include those of Friedman and Hall (2000), Bühlmann and Yu (2002), Hall and Samworth (2005), Buja and Stuetzle (2006) and Biau and Devroye (2008).

It turns out that Breiman's bagging principle has a simple application in the context of nearest neighbor methods. Nearest neighbor predictors are one of the oldest approaches to regression and classification (Fix and Hodges, 1951, 1952; Cover and Hart, 1967; Cover, 1968a,b; Györfi, 1978; Venkatesh et al., 1992; Psaltis et al., 1994). A major attraction of nearest neighbor procedures is their simplicity. For implementation, they require only a measure of distance in the sample space, along with samples of training data, hence their popularity as a starting point for refinement, improvement and adaptation to new settings (see for example Devroye et al., 1996, Chap. 19). Before we formalize the link between bagging and nearest neighbors, some definitions are in order. Throughout the paper, we suppose that we are given a sample $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ of i.i.d. $\mathbb{R}^d \times \mathbb{R}$-valued random variables with the same distribution as a generic pair $(\mathbf{X}, Y)$ satisfying $\mathbb{E}Y^2 < \infty$. The space $\mathbb{R}^d$ is equipped with the standard Euclidean metric. For fixed $\mathbf{x} \in \mathbb{R}^d$, our mission is to estimate the regression function $r(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ using the data $\mathcal{D}_n$. With this respect, we say that a regression function estimate $r_n(\mathbf{x})$ is consistent if $\mathbb{E}[r_n(\mathbf{X}) - r(\mathbf{X})]^2 \to 0$ as $n \to \infty$. It is universally consistent if this property is true for all distributions of $(\mathbf{X}, Y)$ with $\mathbb{E}Y^2 < \infty$.

1.2 Bagging and Nearest Neighbors

Recall that the 1-nearest neighbor (1-NN) regression estimate sets $r_n(\mathbf{x}) = Y_{(1)}(\mathbf{x})$ where $Y_{(1)}(\mathbf{x})$ is the observation of the feature vector $\mathbf{X}_{(1)}(\mathbf{x})$ whose Euclidean distance to $\mathbf{x}$ is minimal among all $\mathbf{X}_1, \ldots, \mathbf{X}_n$. Ties are broken in favor of smallest indices. It is clearly not, in general, a consistent estimate (Devroye et al., 1996, Chap. 5). However, by bagging, one may turn the 1-NN estimate into a consistent one, provided that the size of the resamples is sufficiently small.

We proceed as follows, via a randomized basic regression estimate r_{k_n} in which $1 \leq k_n \leq n$ is a parameter. The elementary predictor r_{k_n} is the 1-NN rule for a random subsample $\mathcal{S}_n$ drawn with (or without) replacement from $\{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$, with $\mathrm{Card}(\mathcal{S}_n) = k_n$. We apply bagging, that is, we repeat the random sampling an infinite number of times, and take the average of the individual outcomes. Thus, the bagged regression estimate $r_n^\star$ is defined by

$$r_n^\star(\mathbf{x}) = \mathbb{E}^\star\left[r_{k_n}(\mathbf{x})\right],$$

where $\mathbb{E}^\star$ denotes expectation with respect to the resampling distribution, conditionally on the data set $\mathcal{D}_n$.

The following result, proved in Biau and Devroye (2008), shows that for an appropriate choice of k_n, the bagged version of the 1-NN regression estimate is universally consistent:

Theorem 1 *If $k_n \to \infty$ and $k_n/n \to 0$, then $r_n^\star$ is universally consistent.*

In this theorem, the fact that resampling is done with or without replacement is irrelevant. Thus, by bagging, one may turn the crude 1-NN procedure into a consistent one, provided that the size of the resamples is sufficiently small. To understand the statistical forces driving Theorem 1, recall that if we let $V_1 \geq V_2 \geq \ldots \geq V_n \geq 0$ denote deterministic weights that sum to one, then the regression estimate

$$\sum_{i=1}^{n} V_i Y_{(i)}(\mathbf{x}),$$

with $(\mathbf{X}_{(1)}(\mathbf{x}), Y_{(1)}(\mathbf{x})), \ldots, (\mathbf{X}_{(n)}(\mathbf{x}), Y_{(n)}(\mathbf{x}))$ the reordering of the data such that

$$\|\mathbf{x} - \mathbf{X}_{(1)}(\mathbf{x})\| \leq \ldots \leq \|\mathbf{x} - \mathbf{X}_{(n)}(\mathbf{x})\|$$

is called a weighted nearest neighbor regression estimate. It is known to be universally consistent provided $V_1 \to 0$ and $\sum_{i>\varepsilon n} V_i \to 0$ for all $\varepsilon > 0$ as $n \to \infty$, see Stone (1977) and Devroye (1981) and Devroye et al. (1996, Problems 11.7, 11.8). The crux to prove Theorem 1 is to observe that $r_n^\star$ is in fact a weighted nearest neighbor estimate with

$$V_i = \mathbb{P}(i\text{-th nearest neighbor of } \mathbf{x} \text{ is the 1-NN in a random selection}).$$

Then, a moment's thought shows that for the "with replacement" sampling

$$V_i = \left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1 - \frac{i}{n}\right)^{k_n},$$

whereas for sampling "without replacement", V_i is hypergeometric:

$$V_i = \begin{cases} \dfrac{\binom{n-i}{k_n - 1}}{\binom{n}{k_n}}, & i \leq n - k_n + 1 \\ 0, & i > n - k_n + 1. \end{cases}$$

The core of the proof of Theorem 1 is then to show that, in both cases, the weights V_i satisfy the conditions $V_1 \to 0$ and $\sum_{i>\varepsilon n} V_i \to 0$ for all $\varepsilon > 0$ as $n \to \infty$. These weights have been independently exhibited by Steele (2009), who also shows on practical examples that substantial reductions in prediction error are possible by bagging the 1-NN estimate. Note also that this new expression for the 1-NN bagged estimate makes any Monte-Carlo approach unnecessary to evaluate the estimate. Indeed, up to now, this predictor was implemented by Monte-Carlo, that is, by repeating the random sampling T times, and taking the average of the individual outcomes. Formally, if $Z_t = r_{k_n}(\mathbf{x})$ is the prediction in the t-th round of bagging, the bagged regression estimate was approximately evaluated as

$$r_n^\star(\mathbf{x}) \approx \frac{1}{T} \sum_{t=1}^{T} Z_t,$$

where $Z_1, \ldots, Z_T$ are the outcomes in the individual rounds. Clearly, writing the 1-NN bagged estimate as an (exact) weighted nearest neighbor predictor makes such calculations useless.

On the other hand, the fact that the bagged 1-NN estimate reduces to a weighted nearest neighbor estimate may seem at first sight somehow disappointing. Indeed, we get the ordinary k_n-NN rule back by the choice

$$V_i = \begin{cases} 1/k_n & \text{if } i \leq k_n \\ 0 & \text{otherwise,} \end{cases}$$

and, with an appropriate choice of the sequence (k_n), this regression estimate is known to have optimal asymptotic properties (see Györfi et al., 2002, Chap. 6, and the references therein). Thus, the question is: Why would one care about the bagged nearest neighbor rule then? The answer is twofold. First, bagging the 1-NN is a very popular technique for regression and classification in the machine learning community, and most—if not all—empirical studies report practical improvements over the traditional k_n-NN method. Secondly (and most importantly), analysing 1-NN bagging is part of a larger project trying to understand the driving forces behind the random forests estimates, which were defined in Breiman (2001). In short, random forests are some of the most successful ensemble methods that exhibit performance on the level of boosting and support vector machines. These learning procedures typically involve a resampling step, which may be interpreted as a particular 1-NN bagged procedure based on the so-called "layered nearest neighbor" proximities (Lin and Jeon, 2006; Biau and Devroye, 2008).

Thus, in the present paper, we go one step further in bagging investigation and study the rate of convergence of $\mathbb{E}[r_n^\star(\mathbf{X}) - r(\mathbf{X})]^2$ to 0 as $n \to \infty$. We will start our analysis by stating a comprehensive theorem on the rate of convergence of general weighted nearest neighbor estimates (Section 2.1). Then, this result will be particularized to 1-NN bagging, by distinguishing the "with replacement" (Section 2.2) and the "without replacement" (Section 2.3) cases. For the sake of clarity, technical proofs are postponed to Section 3.

Throughout the document, we will be interested in rate of convergence results for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$ such that $\mathbf{X}$ has compact support with diameter 2ρ, the regression function r is Lipschitz with constant C and, for all $\mathbf{x} \in \mathbb{R}^d$, $\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$ (the symbol $\mathbb{V}$ denotes variance). It is known (see, for example Ibragimov and Khasminskii, 1980, 1981, 1982) that for the class $\mathcal{F}$, the sequence $(n^{-\frac{2}{d+2}})$ is the optimal minimax rate of convergence. In particular,

$$\liminf_{n\to\infty} \inf_{r_n} \sup_{(\mathbf{X},Y)\in\mathcal{F}} \frac{\mathbb{E}[r_n(\mathbf{X}) - r(\mathbf{X})]^2}{\left((\rho C)^d \sigma^2\right)^{\frac{2}{d+2}} n^{-\frac{2}{d+2}}} \geq \Lambda$$

for some positive constant Λ independent of C, ρ and σ^2. Here the infimum is taken over all estimates r_n, that is, over all square integrable measurable functions of the data. As a striking result, we prove in Sections 2.2 and 2.3 that, irrespectively of the resampling type, for $d \geq 3$ and a suitable choice of the sequence (k_n), the estimate $r_n^\star$ is of optimum rate for the class $\mathcal{F}$, that is

$$\limsup_{n\to\infty} \sup_{(\mathbf{X},Y)\in\mathcal{F}} \frac{\mathbb{E}[r_n^\star(\mathbf{X}) - r(\mathbf{X})]^2}{\left((\rho C)^d \sigma^2\right)^{\frac{2}{d+2}} n^{-\frac{2}{d+2}}} \leq \Lambda$$

for some positive Λ independent of C, ρ and σ^2. Since the parameter k_n of the estimate with the optimal rate of convergence depends on the unknown distribution of $(\mathbf{X}, Y)$, especially on the smoothness of the regression function, we present in Section 2.4 adaptive (i.e., data-dependent) choices of k_n which preserve the minimax optimality of the estimate.

We wish to emphasize that all the results are obtained by letting the resampling size k_n grows with n in such a manner that $k_n \to \infty$ and $k_n/n \to 0$. These results are of interest because the majority of bagging experiments employ relatively large resample sizes. In fact, much of the evidence *against* the performance of bagged nearest neighbor methods is for full sample size resamples (see the discussion in Breiman, 1996, paragraph 6.4), except the notable results of Hall and Samworth (2005) and Steele (2009), who also report encouraging numerical results in the context of regression and classification.

2. Rates of Convergence

As an appetizer, we start our analysis of the 1-NN bagged regression estimate from a larger point of view, by offering a general theorem on the rate of convergence of weighted nearest neighbor estimates, that is, estimates of the form

$$r_n(\mathbf{x}) = \sum_{i=1}^{n} V_i Y_{(i)}(\mathbf{x})$$

with nonnegative weights satisfying the constraints $\sum_{i=1}^{n} V_i = 1$ and $V_1 \geq V_2 \geq \ldots \geq V_n \geq 0$.

2.1 Weighted Nearest Neighbor Estimates

Let us first recall various topological definitions that will be used in the paper. We first define the well-known notion of covering numbers which characterize the massiveness of a set (Kolmogorov and Tihomirov, 1961). As put forward in Kulkarni and Posner (1995), these quantities play a key role in the context of nearest neighbor analysis. Let $\mathcal{B}(\mathbf{x}, \varepsilon)$ denote the open Euclidean ball in $\mathbb{R}^d$ centered at $\mathbf{x}$ of radius ε.

Definition 2 *Let $\mathcal{A}$ be a subset of $\mathbb{R}^d$. The ε-covering number $\mathcal{N}(\varepsilon)$ $[= \mathcal{N}(\varepsilon, \mathcal{A})]$ is defined as the smallest number of open balls of radius ε that cover the set $\mathcal{A}$. That is*

$$\mathcal{N}(\varepsilon) = \inf \left\{ r \geq 1 : \exists \mathbf{x}_1, \ldots, \mathbf{x}_r \in \mathbb{R}^d \text{ such that } \mathcal{A} \subset \bigcup_{i=1}^{r} \mathcal{B}(\mathbf{x}_i, \varepsilon) \right\}.$$

A set $\mathcal{A} \subset \mathbb{R}^d$ is bounded if and only if $\mathcal{N}(\varepsilon) < \infty$ for all $\varepsilon > 0$. Note that as a function of ε, $\mathcal{N}(\varepsilon)$ is nonincreasing, piecewise-constant and right-continuous. The following discrete function, called the metric covering radius, can be interpreted as a pseudo-inverse of the function $\mathcal{N}(\varepsilon)$:

Definition 3 *The metric covering radius $\mathcal{N}^{-1}(r)$ $[= \mathcal{N}^{-1}(r, A)]$ is defined as the smallest radius such that there exist r balls of this radius which cover the set $\mathcal{A}$. That is*

$$\mathcal{N}^{-1}(r) = \inf \left\{ \varepsilon > 0 : \exists \mathbf{x}_1, \ldots, \mathbf{x}_r \in \mathbb{R}^d \text{ such that } \mathcal{A} \subset \bigcup_{i=1}^{r} \mathcal{B}(\mathbf{x}_i, \varepsilon) \right\}.$$

We note that $\mathcal{N}^{-1}(r)$ is a nonincreasing discrete function of r.

Throughout the paper, we will denote by μ the distribution of $\mathbf{X}$, which will be assumed to be a bounded random variable. Recall that the support $\mathcal{S}(\mu)$ of μ is defined as the collection of all $\mathbf{x}$ with $\mu(\mathcal{B}(\mathbf{x}, \varepsilon)) > 0$ for all $\varepsilon > 0$. Letting $\rho = \mathcal{N}^{-1}(1, \mathcal{S}(\mu))$, we observe that 2ρ is an upper bound on the diameter of $\mathcal{S}(\mu)$. We are now in a position to state the main result of this subsection. We let the symbol $\lfloor . \rfloor$ denote the integer part function.

Theorem 4 *Let $r_n(\mathbf{x}) = \sum_{i=1}^{n} V_i Y_{(i)}(\mathbf{x})$ be a weighted nearest neighbor estimate of $r(\mathbf{x})$. Suppose that $\mathbf{X}$ is bounded, and set $\rho = \mathcal{N}^{-1}(1, \mathcal{S}(\mu))$. Suppose in addition that, for all $\mathbf{x}$ and $\mathbf{x}' \in \mathbb{R}^d$,*

$$\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$$

and

$$\left| r(\mathbf{x}) - r(\mathbf{x}') \right| \leq C \|\mathbf{x} - \mathbf{x}'\|,$$

for some positive constants σ^2 and C. Then

(*i*) *If* $d = 1$,

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \sigma^2 \sum_{i=1}^{n} V_i^2 + 16\rho^2 C^2 \sum_{i=1}^{n} V_i \frac{i}{n}.$$

(*ii*) *If* $d = 2$,

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \sigma^2 \sum_{i=1}^{n} V_i^2 + 8\rho^2 C^2 \sum_{i=1}^{n} V_i \frac{i}{n} \left[1 + \ln\left(\frac{n}{i}\right)\right].$$

(*iii*) *If* $d \geq 3$,

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \sigma^2 \sum_{i=1}^{n} V_i^2 + \frac{8\rho^2 C^2}{1 - 2/d} \sum_{i=1}^{n} V_i \left\lfloor \frac{n}{i} \right\rfloor^{-2/d}.$$

Proof Setting

$$\tilde{r}_n(\mathbf{x}) = \sum_{i=1}^{n} V_i \, r(\mathbf{X}_{(i)}(\mathbf{x})),$$

the proof of Theorem 4 will rely on the variance/bias decomposition

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 = \mathbb{E}\left[r_n(\mathbf{X}) - \tilde{r}_n(\mathbf{X})\right]^2 + \mathbb{E}\left[\tilde{r}_n(\mathbf{X}) - r(\mathbf{X})\right]^2. \tag{1}$$

The first term is easily bounded by noting that, for all $\mathbf{x} \in \mathbb{R}^d$,

$$\begin{aligned}
\mathbb{E}\left[r_n(\mathbf{x}) - \tilde{r}_n(\mathbf{x})\right]^2 &= \mathbb{E}\left[\sum_{i=1}^{n} V_i \left(Y_{(i)}(\mathbf{x}) - r(\mathbf{X}_{(i)}(\mathbf{x}))\right)\right]^2 \\
&= \mathbb{E}\left[\sum_{i=1}^{n} V_i^2 \left(Y_{(i)}(\mathbf{x}) - r(\mathbf{X}_{(i)}(\mathbf{x}))\right)^2\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{n} V_i^2 \sigma^2 \left(\mathbf{X}_{(i)}(\mathbf{x})\right)\right] \\
&\leq \sigma^2 \sum_{i=1}^{n} V_i^2. \qquad (2)
\end{aligned}$$

To analyse the bias term in (1), we will need the following result, which bounds the convergence rate of the expected i-th nearest neighbor squared distance in terms of the metric covering radii of the support of the distribution μ of $\mathbf{X}$. Proposition 5 is a generalization of Theorem 1, page 1032 in Kulkarni and Posner (1995), which only reports results for the rate of convergence of *the* nearest neighbor. Therefore, this result is interesting by itself.

Proposition 5 *Suppose that* $\mathbf{X}$ *is bounded*. *Then*

$$\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 \leq \frac{8i}{n} \sum_{j=1}^{\lfloor n/i \rfloor} \left[\mathcal{N}^{-1}\left(j, \mathcal{S}(\mu)\right)\right]^2.$$

For any bounded set $\mathcal{A}$ in the Euclidean d-space, the covering radius satisfies $\mathcal{N}^{-1}(r, \mathcal{A}) \leq \mathcal{N}^{-1}(1, \mathcal{A}) r^{-1/d}$ (see Kolmogorov and Tihomirov, 1961). Hence the following corollary:

Corollary 6 *Suppose that* $\mathbf{X}$ *is bounded, and set* $\rho = \mathcal{N}^{-1}(1, \mathcal{S}(\mu))$. *Then*

(*i*) *If* $d = 1$,

$$\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 \leq \frac{16\rho^2 i}{n}.$$

(*ii*) *If* $d = 2$,

$$\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 \leq \frac{8\rho^2 i}{n}\left[1 + \ln\left(\frac{n}{i}\right)\right].$$

(*iii*) *If* $d \geq 3$,

$$\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 \leq \frac{8\rho^2 \lfloor n/i \rfloor^{-\frac{2}{d}}}{1 - 2/d}.$$

Thus, to prove Theorem 4, it suffices to note from (1) and (2) that

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \sigma^2 \sum_{i=1}^{n} V_i^2 + \mathbb{E}\left[\tilde{r}_n(\mathbf{X}) - r(\mathbf{X})\right]^2.$$

Next,

$$\begin{aligned}
\mathbb{E}\left[\tilde{r}_n(\mathbf{x}) - r(\mathbf{x})\right]^2 &= \mathbb{E}\left[\sum_{i=1}^{n} V_i \left(r(\mathbf{X}_{(i)}(\mathbf{x})) - r(\mathbf{x})\right)\right]^2 \\
&\leq \mathbb{E}\left[\sum_{i=1}^{n} V_i \left|r(\mathbf{X}_{(i)}(\mathbf{x})) - r(\mathbf{x})\right|\right]^2 \\
&\leq C^2\, \mathbb{E}\left[\sum_{i=1}^{n} V_i \left\|\mathbf{X}_{(i)}(\mathbf{x}) - \mathbf{x}\right\|\right]^2 \\
&\leq C^2 \left[\sum_{i=1}^{n} V_i\, \mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{x}) - \mathbf{x}\|^2\right] \\
&\qquad \text{(by Jensen's inequality).}
\end{aligned}$$

Therefore, integrating with respect to the distribution of $\mathbf{X}$, we obtain

$$\mathbb{E}\left[\tilde{r}_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq C^2 \left[\sum_{i=1}^{n} V_i\, \mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2\right],$$

and the conclusion follows by applying Corollary 6. ■

Theorem 4 offers a general result, which can be made more precise according to the weights definition. Taking for example

$$V_i = \begin{cases} 1/k_n & \text{if } i \leq k_n \\ 0 & \text{otherwise,} \end{cases}$$

we get the ordinary k_n-NN rule back. Here,

$$\sum_{i=1}^{n} V_i^2 = \frac{1}{k_n}$$

and

$$\begin{aligned}\sum_{i=1}^{n} V_i \left\lfloor \frac{n}{i} \right\rfloor^{-2/d} &= \frac{1}{k_n} \sum_{i=1}^{k_n} \left\lfloor \frac{n}{i} \right\rfloor^{-2/d} \\ &\leq \frac{1}{k_n} \sum_{i=1}^{k_n} \left\lfloor \frac{n}{k_n} \right\rfloor^{-2/d} \\ &= \left\lfloor \frac{n}{k_n} \right\rfloor^{-2/d} \\ &\leq \xi \left(\frac{n}{k_n} \right)^{-2/d}\end{aligned}$$

for some positive ξ. Therefore, in this context, according to Theorem 4, for $d \geq 3$, there exists a sequence (k_n) with $k_n \propto n^{\frac{2}{d+2}}$ such that

$$\mathbb{E}\left[r_n(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \Lambda \left(\frac{(\rho C)^d \sigma^2}{n} \right)^{\frac{2}{d+2}},$$

for some positive constant Λ independent of ρ, C and σ^2. This is exactly Theorem 6.2, page 93 of Györfi et al. (2002), which states that the standard nearest neighbor estimate is of optimum rate for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$ such that $\mathbf{X}$ has compact support with covering radius ρ, the regression function r is Lipschitz with constant C and, for all $\mathbf{x} \in \mathbb{R}^d$, $\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$ (note however that the ordinary k_n-NN predictor is *not* optimal for higher smoothness, see Problem 6.2 in Györfi et al., 2002).

The adaptation of Theorem 4 to the 1-NN bagged regression estimate needs more careful attention. This will be the topic of the next two Sections.

2.2 Bagging with Replacement

This bagging-type is sometimes called moon-bagging, standing for **m o**ut **o**f **n b**ootstrap **agg**regat**ing**. As seen in the introduction, in this case, the weighted nearest neighbor regression estimate takes the form

$$r_n^\star(\mathbf{x}) = \sum_{i=1}^{n} V_i Y_{(i)}(\mathbf{x}),$$

where

$$V_i = \left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1 - \frac{i}{n}\right)^{k_n}.$$

From now on, $\Gamma(t)$ will denote the Gamma function, that is,

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} \mathrm{d}x, \quad t > 0.$$

In order to make full use of Theorem 4, we first need a careful control of the term $\sum_{i=1}^n V_i^2$. This is done in the next proposition.

Proposition 7 *For $i = 1, \ldots, n$, let*

$$V_i = \left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1 - \frac{i}{n}\right)^{k_n}.$$

Then

$$\sum_{i=1}^{n} V_i^2 \leq \frac{2k_n}{n}\left(1 + \frac{1}{n}\right)^{2k_n}.$$

The message of Proposition 7 is that, when resampling is done with replacement, the variance term of the bagged NN estimate is $\mathrm{O}(k_n/n)$. Let us now turn to the bias term analysis.

Proposition 8 *For $i = 1, \ldots, n$, let*

$$V_i = \left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1 - \frac{i}{n}\right)^{k_n}.$$

Then

(*i*) *If $d = 1$,*

$$\sum_{i=1}^{n} V_i \frac{i}{n} \leq \frac{2}{k_n}\left(1 + \frac{1}{n}\right)^{k_n}.$$

(*ii*) *If $d = 2$,*

$$\sum_{i=1}^{n} V_i \frac{i}{n}\left[1 + \ln\left(\frac{n}{i}\right)\right] \leq \frac{2}{k_n}\left(1 + \frac{1}{n}\right)^{k_n}\left[1 + \ln(k_n + 1)\right].$$

(*iii*) *If $d \geq 3$,*

$$\sum_{i=1}^{n} V_i \left\lfloor \frac{n}{i} \right\rfloor^{-2/d} \leq \frac{1}{n^{k_n}} + \alpha_d \left(1 + \frac{1}{n}\right)^{k_n} {k_n}^{-\frac{2}{d}},$$

where

$$\alpha_d = 2\Gamma\left(\frac{d-2}{d}\right)\Gamma\left(\frac{d+2}{d}\right).$$

The take-home message here is that, for $d \geq 3$, the squared bias is $\mathrm{O}(k_n^{-2/d})$. Finally, putting all the pieces together, we obtain

Theorem 9 *Suppose that $\mathbf{X}$ is bounded, and set $\rho = \mathcal{N}^{-1}(1, \mathcal{S}(\mu))$. Suppose in addition that, for all $\mathbf{x}$ and $\mathbf{x}' \in \mathbb{R}^d$,*

$$\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$$

and

$$\left|r(\mathbf{x}) - r(\mathbf{x}')\right| \leq C\|\mathbf{x} - \mathbf{x}'\|,$$

for some positive constants σ^2 and C. Then

(*i*) *If $d = 1$,*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \frac{2\sigma^2 k_n}{n}\left(1 + \frac{1}{n}\right)^{2k_n} + \frac{32\rho^2 C^2}{k_n}\left(1 + \frac{1}{n}\right)^{k_n}.$$

(*ii*) *If* $d = 2$,

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \frac{2\sigma^2 k_n}{n}\left(1+\frac{1}{n}\right)^{2k_n} + \frac{16\rho^2 C^2}{k_n}\left(1+\frac{1}{n}\right)^{k_n}\left[1+\ln(k_n+1)\right].$$

(*iii*) *If* $d \geq 3$,

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \frac{2\sigma^2 k_n}{n}\left(1+\frac{1}{n}\right)^{2k_n} + \frac{8\rho^2 C^2}{1-2/d}\left[\frac{1}{n^{k_n}} + \alpha_d\left(1+\frac{1}{n}\right)^{k_n} k_n^{-\frac{2}{d}}\right],$$

where

$$\alpha_d = 2\Gamma\left(\frac{d-2}{d}\right)\Gamma\left(\frac{d+2}{d}\right).$$

By balancing the terms in Theorem 9, we are led to the following corollary:

Corollary 10 *Under the assumptions of Theorem 9,*

(*i*) *If* $d = 1$, *there exists a sequence* (k_n) *such that* $k_n \to \infty$, $k_n/n \to 0$, *and*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \Lambda \frac{\rho C \sigma}{\sqrt{n}},$$

for some positive constant Λ *independent of* ρ, C *and* σ^2.

(*ii*) *If* $d = 2$, *there exists a sequence* (k_n) *such that* $k_n \to \infty$, $k_n/n \to 0$, *and*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq (\Lambda + o(1))\,\rho C \sigma \sqrt{\frac{\ln n}{n}},$$

for some positive constant Λ *independent of* ρ, C *and* σ^2.

(*iii*) *If* $d \geq 3$, *there exists a sequence* (k_n) *with* $k_n \propto n^{\frac{d}{d+2}}$ *such that*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq \Lambda\left(\frac{(\rho C)^d \sigma^2}{n}\right)^{\frac{2}{d+2}},$$

for some positive constant Λ *independent of* ρ, C *and* σ^2.

Two important remarks are in order.

1. First, we note that, for $d \geq 3$ and a suitable choice of k_n, the bagged 1-NN estimate achieves both the minimax $n^{-2/(d+2)}$ rate and the optimal order of magnitude $((\rho C)^d \sigma^2)^{2/(d+2)}$ in the constant, for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$ such that $\mathbf{X}$ has compact support with covering radius ρ, the regression function r is Lipschitz with constant C and, for

all $\mathbf{x} \in \mathbb{R}^d$, $\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$. Seconds, the bound is valid for finite sample sizes, so that we are in fact able to approach the minimax lower bound not only asymptotically but even for finite sample sizes. On the other hand, the estimate with the optimal rate of convergence depends on the unknown distribution of $(\mathbf{X}, Y)$, and especially on the covering radius ρ and the smoothness of the regression function measured by the constant C. It is to correct this situation that we present adaptation results in Section 2.4.

2. For $d = 1$, the obtained rate is not optimal, whereas it is optimal up to a log term for $d = 2$. This low-dimensional phenomenon is also known to hold for the traditional k_n-NN regression estimate, which does not achieve the optimal rates in dimensions 1 and 2 (see Problems 6.1 and 6.7 in Györfi et al., 2002, Chap. 3).

2.3 Bagging Without Replacement

We briefly analyse in this subsection the rate of convergence of the bagged 1-NN regression estimate, assuming this time that, at each step, the k_n observations are distinctly chosen at random within the sample set $\mathcal{D}_n$. This alternative aggregation scheme is called subagging (for **sub**sample **agg**regat**ing**) in Bühlmann and Yu (2002). We know that, in this case, the weighted nearest neighbor regression estimate takes the form

$$r_n^\star(\mathbf{x}) = \sum_{i=1}^n V_i Y_{(i)}(\mathbf{x}),$$

where

$$V_i = \begin{cases} \dfrac{\binom{n-i}{k_n-1}}{\binom{n}{k_n}}, & i \leq n - k_n + 1 \\ 0, & i > n - k_n + 1. \end{cases}$$

Due to the fact that there is no repetition in the sampling process, the analysis turns out to be simpler. To prove Theorem 13 below, we start again by a control of the variance term $\sum_{i=1}^n V_i^2$.

Proposition 11 *For $i = 1, \ldots, n$, let*

$$V_i = \begin{cases} \dfrac{\binom{n-i}{k_n-1}}{\binom{n}{k_n}}, & i \leq n - k_n + 1 \\ 0, & i > n - k_n + 1. \end{cases}$$

Then

$$\sum_{i=1}^n V_i^2 \leq \frac{k_n}{n} \frac{1}{(1 - k_n/n + 1/n)^2}.$$

Thus, as for bagging with replacement, the variance term of the without replacement bagged 1-NN estimate is $O(k_n/n)$. The bias term may be treated by resorting to Theorem 4, via complex calculations due to the complicate form of the bagging weights. However, a much simpler route may be followed. Recall that

$$\tilde{r}_n^\star(\mathbf{x}) = \sum_{i=1}^n V_i \, r(\mathbf{X}_{(i)}(\mathbf{x})),$$

and observe that

$$\tilde{r}_n^\star(\mathbf{x}) = \mathbb{E}^\star\left[r(\mathbf{X}_{(1)}^\star(\mathbf{x}))\right],$$

where $\mathbf{X}_{(1)}^\star(\mathbf{x})$ is the nearest neighbor of $\mathbf{x}$ in a random subsample $\mathcal{S}_n$ drawn without replacement from $\{(\mathbf{X}_1,Y_1),\ldots,(\mathbf{X}_n,Y_n)\}$ with $\mathrm{Card}(\mathcal{S}_n)=k_n$, and $\mathbb{E}^\star$ denotes expectation with respect to the resampling distribution, conditionally on the data set $\mathcal{D}_n$. This is the basic ingredient for the proof of the next proposition.

Proposition 12 *Suppose that* $\mathbf{X}$ *is bounded, and set* $\rho = \mathcal{N}^{-1}(1,\mathcal{S}(\mu))$. *Suppose in addition that, for all* $\mathbf{x}$ *and* $\mathbf{x}' \in \mathbb{R}^d$,

$$\left|r(\mathbf{x}) - r(\mathbf{x}')\right| \le C\|\mathbf{x}-\mathbf{x}'\|,$$

for some positive constant C. *Then*

(*i*) *If* $d=1$,

$$\mathbb{E}\left[\tilde{r}_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{16\rho^2C^2}{k_n}.$$

(*ii*) *If* $d=2$,

$$\mathbb{E}\left[\tilde{r}_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{8\rho^2C^2}{k_n}(1+\ln k_n).$$

(*iii*) *If* $d\ge 3$,

$$\mathbb{E}\left[\tilde{r}_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{8\rho^2C^2}{1-2/d}\,{k_n}^{-\frac{2}{d}}.$$

Thus, for $d\ge 3$, $\mathbb{E}\left[\tilde{r}_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 = \mathrm{O}(k_n^{-2/d})$. Combining Proposition 11 and Proposition 12 leads to the desired theorem:

Theorem 13 *Suppose that* $\mathbf{X}$ *is bounded, and set* $\rho = \mathcal{N}^{-1}(1,\mathcal{S}(\mu))$. *Suppose in addition that, for all* $\mathbf{x}$ *and* $\mathbf{x}' \in \mathbb{R}^d$,

$$\sigma^2(\mathbf{x}) = \mathbb{V}[Y\,|\,\mathbf{X}=\mathbf{x}] \le \sigma^2$$

and

$$\left|r(\mathbf{x}) - r(\mathbf{x}')\right| \le C\|\mathbf{x}-\mathbf{x}'\|,$$

for some positive constants σ^2 *and* C. *Then*

(*i*) *If* $d=1$,

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{k_n}{n}\,\frac{\sigma^2}{(1-k_n/n+1/n)^2} + \frac{16\rho^2C^2}{k_n}.$$

(*ii*) *If* $d=2$,

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{k_n}{n}\,\frac{\sigma^2}{(1-k_n/n+1/n)^2} + \frac{8\rho^2C^2}{k_n}(1+\ln k_n).$$

(*iii*) *If* $d\ge 3$,

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \le \frac{k_n}{n}\,\frac{\sigma^2}{(1-k_n/n+1/n)^2} + \frac{8\rho^2C^2}{1-2/d}\,k_n^{-\frac{2}{d}}.$$

By balancing the variance and bias terms, we obtain the following useful corollary:

Corollary 14 *Under the assumptions of Theorem 13,*

(*i*) *If $d = 1$, there exists a sequence (k_n) such that $k_n \to \infty$, $k_n/n \to 0$, and*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq (\Lambda + o(1))\,\frac{\rho C \sigma}{\sqrt{n}},$$

for some positive constant Λ independent of ρ, C and σ^2.

(*ii*) *If $d = 2$, there exists a sequence (k_n) such that $k_n \to \infty$, $k_n/n \to 0$, and*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq (\Lambda + o(1))\,\rho C \sigma \sqrt{\frac{\ln n}{n}},$$

for some positive constant Λ independent of ρ, C and σ^2.

(*iii*) *If $d \geq 3$, there exists a sequence (k_n) with $k_n \propto n^{\frac{d}{d+2}}$ such that*

$$\mathbb{E}\left[r_n^\star(\mathbf{X}) - r(\mathbf{X})\right]^2 \leq (\Lambda + o(1)) \left(\frac{(\rho C)^d \sigma^2}{n}\right)^{\frac{2}{d+2}},$$

for some positive constant Λ independent of ρ, C and σ^2.

As in bagging with replacement, Corollary 14 expresses the fact that, for $d \geq 3$, the without replacement bagged 1-NN estimate asymptotically achieves both the minimax $n^{-2/(d+2)}$ rate of convergence and the optimal order of magnitude $((\rho C)^d \sigma^2)^{d/(d+2)}$ in the constant, for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$.

2.4 Adaptation

In the previous subsections, the parameter k_n of the estimate with the optimal rate of convergence for the class $\mathcal{F}$ depends on the unknown distribution of $(\mathbf{X}, Y)$, especially on the smoothness of the regression function measured by the Lipschitz constant C. In this subsection, we present a data-dependent way of choosing the resampling size k_n and show that, for bounded Y, the estimate with parameter chosen in such an adaptive way achieves the optimal rate of convergence (irrespectively of the resampling type). To this aim, we split the sample $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ in two parts of size $\lfloor n/2 \rfloor$ and $n - \lfloor n/2 \rfloor$, respectively (assuming $n \geq 2$). The first half is denoted by $\mathcal{D}_n^\ell$ (learning set) and is used to construct the bagged 1-NN estimate $r^\star_{\lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell) = r^\star_{k, \lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell)$ (for the sake of clarity, we make the dependence of the estimate upon k explicit). The second half of the sample, denoted by $\mathcal{D}_n^t$ (testing set), is used to choose k by picking $\hat{k}_n \in K = \{1, \ldots, \lfloor n/2 \rfloor\}$ to minimize the empirical risk

$$\frac{1}{n - \lfloor n/2 \rfloor} \sum_{i = \lfloor n/2 \rfloor + 1}^{n} \left(Y_i - r^\star_{k, \lfloor n/2 \rfloor}(\mathbf{X}_i)\right)^2.$$

Define the estimate

$$r_n^\star(\mathbf{x}) = r^\star_{\hat{k}_n, \lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell),$$

and note that $r_n^\star$ depends on the entire data $\mathcal{D}_n$. If $|Y| \leq L < \infty$ almost surely, a straightforward adaptation of Theorem 7.1 in Györfi et al. (2002) shows that, for any $\delta > 0$,

$$\mathbb{E}[r_n^\star(\mathbf{X}) - r(\mathbf{X})]^2 \leq (1+\delta) \inf_{k \in K} \mathbb{E}[r_{k,\lfloor n/2 \rfloor}^\star(\mathbf{X}) - r(\mathbf{X})]^2 + \Xi \frac{\ln n}{n},$$

for some positive constant Ξ depending only on L, d and δ. Immediately from Corollary 10 and Corollary 14 we can conclude:

Theorem 15 *Suppose that $|Y| \leq L$ almost surely, and let $r_n^\star$ be the bagged 1-NN estimate with $k \in K = \{1, \ldots, \lfloor n/2 \rfloor\}$ chosen by data-splitting, irrespectively of the resampling type. Then $(\ln n)^{(d+2)/(2d)} n^{-1/2} \leq \rho C$ together with $d \geq 3$ implies, for $n \geq 2$,*

$$\mathbb{E}[r_n^\star(\mathbf{X}) - r(\mathbf{X})]^2 \leq (\Lambda + o(1)) \left(\frac{(\rho C)^d}{n} \right)^{\frac{2}{d+2}},$$

for some positive constant Λ which depends only on L and d.

Thus, the expected error of the estimate obtained via data-splitting is bounded from above up to a constant by the corresponding minimax lower bound for the class $\mathcal{F}$ of regression functions, with the optimal dependence in C and ρ.

3. Proofs

Proofs of the main results are gathered in this section.

3.1 Proof of Proposition 5

All the covering and metric numbers we use in this proof are pertaining to the bounded set $\mathcal{S}(\mu)$. Therefore, to lighten notation a bit, we set $\mathcal{N}(\varepsilon) = \mathcal{N}(\varepsilon, \mathcal{S}(\mu))$ and $\mathcal{N}^{-1}(r) = \mathcal{N}^{-1}(r, \mathcal{S}(\mu))$.

Let $\mathbf{X}'$ be a random variable distributed as and independent of $\mathbf{X}$, and let, for $\varepsilon > 0$,

$$F_{\mathbf{X}}(\varepsilon) = \mathbb{P}\left(\|\mathbf{X} - \mathbf{X}'\| \leq \varepsilon \,|\, \mathbf{X} \right)$$

be the conditional cumulative distribution function of the Euclidean distance between $\mathbf{X}$ and $\mathbf{X}'$. Set finally

$$D_{(i)}(\mathbf{X}) = \|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|.$$

Clearly,

$$\begin{aligned} \mathbb{P}\left(D_{(i)}^2(\mathbf{X}) > \varepsilon \right) &= \mathbb{E}\left[\mathbb{P}\left(D_{(i)}(\mathbf{X}) > \sqrt{\varepsilon} \,|\, \mathbf{X} \right) \right] \\ &= \mathbb{E}\left[\sum_{j=0}^{i-1} \binom{n}{j} \left[F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right) \right]^j \left[1 - F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right) \right]^{n-j} \right]. \end{aligned} \tag{3}$$

Take $\mathcal{B}_1, \ldots, \mathcal{B}_{\mathcal{N}(\sqrt{\varepsilon}/2)}$ a $\sqrt{\varepsilon}/2$-covering of $\mathcal{S}(\mu)$, and define an $\mathcal{N}(\sqrt{\varepsilon}/2)$-partition of $\mathcal{S}(\mu)$ as follows. For each $\ell = 1, \ldots, \mathcal{N}(\sqrt{\varepsilon}/2)$, let

$$\mathcal{P}_\ell = \mathcal{B}_\ell - \bigcup_{j=1}^{\ell-1} \mathcal{B}_j.$$

Then $\mathcal{P}_\ell \subset \mathcal{B}_\ell$ and

$$\bigcup_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} \mathcal{B}_\ell = \bigcup_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} \mathcal{P}_\ell,$$

with $\mathcal{P}_i \cap \mathcal{P}_m = \emptyset$. Also,

$$\sum_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} \mu(\mathcal{P}_\ell) = 1.$$

Thus, letting $p_\ell = \mu(\mathcal{P}_\ell)$, we may write

$$\begin{aligned} F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right) &\geq \mathbb{P}(\exists\, \ell = 1, \ldots, \mathcal{N}(\sqrt{\varepsilon}/2) : \mathbf{X} \in \mathcal{P}_\ell \text{ and } \mathbf{X}' \in \mathcal{P}_\ell \,|\, \mathbf{X}) \\ &= \mathbb{E}\left[\sum_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} \mathbf{1}_{[\mathbf{X} \in \mathcal{P}_\ell]} \mathbf{1}_{[\mathbf{X}' \in \mathcal{P}_\ell]} \middle| \mathbf{X} \right] \\ &= \sum_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} p_\ell \mathbf{1}_{[\mathbf{X} \in \mathcal{P}_\ell]}. \end{aligned}$$

As a by-product, we remark that, for all $\varepsilon > 0$, $F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right) > 0$ almost surely. Moreover

$$\mathbb{E}\left[\frac{1}{F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)}\right] \leq \mathbb{E}\left[\frac{1}{\sum_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} p_\ell \mathbf{1}_{[\mathbf{X} \in \mathcal{P}_\ell]}}\right] = \mathbb{E}\left[\sum_{\ell=1}^{\mathcal{N}(\sqrt{\varepsilon}/2)} \frac{1}{p_\ell} \mathbf{1}_{[\mathbf{X} \in \mathcal{P}_\ell]}\right],$$

leading to

$$\mathbb{E}\left[\frac{1}{F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)}\right] \leq \mathcal{N}\left(\frac{\sqrt{\varepsilon}}{2}\right). \tag{4}$$

Consequently, combining inequalities (3), (4) and technical Lemma 16, we obtain

$$\begin{aligned} \mathbb{P}\left(D^2_{(i)}(\mathbf{X}) > \varepsilon\right) &= \mathbb{E}\left[\frac{1}{F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)} \sum_{j=0}^{i-1} \binom{n}{j} \left[F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)\right]^{j+1} \left[1 - F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)\right]^{n-j}\right] \\ &\leq \frac{i}{n+1} \mathbb{E}\left[\frac{1}{F_{\mathbf{X}}\left(\sqrt{\varepsilon}\right)}\right] \\ &\leq \frac{i}{n} \mathcal{N}\left(\frac{\sqrt{\varepsilon}}{2}\right). \end{aligned}$$

Thus, since $\mathbb{P}(D^2_{(i)}(\mathbf{X}) > \varepsilon) = 0$ for $\varepsilon > 4[\mathcal{N}^{-1}(1)]^2$, we obtain

$$\begin{aligned} \mathbb{E}\left[D^2_{(i)}(\mathbf{X})\right] &= \int_0^\infty \mathbb{P}(D^2_{(i)}(\mathbf{X}) > \varepsilon) \mathrm{d}\varepsilon \\ &= \int_0^{4[\mathcal{N}^{-1}(1)]^2} \mathbb{P}(D^2_{(i)}(\mathbf{X}) > \varepsilon) \mathrm{d}\varepsilon \\ &\leq 4\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 + \frac{i}{n} \int_{4[\mathcal{N}^{-1}(\lfloor n/i \rfloor)]^2}^{4[\mathcal{N}^{-1}(1)]^2} \mathcal{N}(\sqrt{\varepsilon}/2) \mathrm{d}\varepsilon. \end{aligned}$$

Since $\mathcal{N}(\sqrt{\varepsilon}) = j$ for $\mathcal{N}^{-1}(j) \leq \sqrt{\varepsilon} < \mathcal{N}^{-1}(j-1)$, we get

$$\begin{aligned}
\mathbb{E}\left[D^2_{(i)}(\mathbf{X})\right] &\leq 4\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 + \frac{4i}{n}\int_{[\mathcal{N}^{-1}(\lfloor n/i \rfloor)]^2}^{[\mathcal{N}^{-1}(1)]^2} \mathcal{N}(\sqrt{\varepsilon})\mathrm{d}\varepsilon \\
&\leq 4\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 + \frac{4i}{n}\sum_{j=2}^{\lfloor n/i \rfloor}\int_{[\mathcal{N}^{-1}(j)]^2}^{[\mathcal{N}^{-1}(j-1)]^2} j\,\mathrm{d}\varepsilon \\
&= 4\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 \\
&\quad + \frac{4i}{n}\left[2\left[\mathcal{N}^{-1}(1)\right]^2 - \left\lfloor \frac{n}{i} \right\rfloor\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 + \sum_{j=2}^{\lfloor n/i \rfloor - 1}\left[\mathcal{N}^{-1}(j)\right]^2\right] \\
&\leq \frac{8i}{n}\left[\mathcal{N}^{-1}(1)\right]^2 + \frac{4i}{n}\left[\mathcal{N}^{-1}\left(\left\lfloor \frac{n}{i} \right\rfloor\right)\right]^2 + \frac{4i}{n}\sum_{j=2}^{\lfloor n/i \rfloor - 1}\left[\mathcal{N}^{-1}(j)\right]^2,
\end{aligned}$$

where the last statement follows from the inequality

$$-\frac{4i}{n}\left\lfloor \frac{n}{i} \right\rfloor + 4 \leq \frac{4i}{n}.$$

In conclusion, we are led to

$$\mathbb{E}\left[D^2_{(i)}(\mathbf{X})\right] \leq \frac{8i}{n}\sum_{j=1}^{\lfloor n/i \rfloor}\left[\mathcal{N}^{-1}(j)\right]^2,$$

as desired.

3.2 Proof of Corollary 6

For any bounded set $\mathcal{A}$ in the Euclidean d-space, the covering radius satisfies $\mathcal{N}^{-1}(r,\mathcal{A}) \leq \mathcal{N}^{-1}(1,\mathcal{A})r^{-1/d}$ (see Kolmogorov and Tihomirov, 1961). Consequently, using Proposition 5, we obtain

(*i*) For $d = 1$,

$$\begin{aligned}
\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 &\leq \frac{8\rho^2 i}{n}\sum_{j=1}^{\lfloor n/i \rfloor} j^{-2} \\
&\leq \frac{8\rho^2 i}{n}\left[1 + \int_1^{\lfloor n/i \rfloor} x^{-2}\mathrm{d}x\right] \\
&\leq \frac{16\rho^2 i}{n}.
\end{aligned}$$

(*ii*) For $d = 2$,

$$\begin{aligned}
\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 &\leq \frac{8\rho^2 i}{n}\sum_{j=1}^{\lfloor n/i \rfloor} j^{-1} \\
&\leq \frac{8\rho^2 i}{n}\left[1 + \int_1^{\lfloor n/i \rfloor} x^{-1}\mathrm{d}x\right] \\
&\leq \frac{8\rho^2 i}{n}\left[1 + \ln\left(\frac{n}{i}\right)\right].
\end{aligned}$$

(*iii*) For $d \geq 3$,

$$\begin{aligned}\mathbb{E}\|\mathbf{X}_{(i)}(\mathbf{X}) - \mathbf{X}\|^2 &\leq \frac{8\rho^2 i}{n} \sum_{j=1}^{\lfloor n/i \rfloor} j^{-\frac{2}{d}} \\ &\leq \frac{8\rho^2 i}{n} \int_0^{\lfloor n/i \rfloor} x^{-\frac{2}{d}} \mathrm{d}x \\ &= \frac{8\rho^2 \lfloor n/i \rfloor^{-\frac{2}{d}}}{1-2/d}.\end{aligned}$$

In the last statement, we used the inequality $i/n \leq 1/\lfloor n/i \rfloor$.

3.3 Proof of Proposition 7

An easy calculation shows that

$$\begin{aligned}\sum_{i=1}^n V_i^2 &= \sum_{i=1}^n \left[\left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1 - \frac{i}{n}\right)^{k_n} \right]^2 \\ &= 2 \sum_{i=0}^{n-1} \left(1 - \frac{i}{n}\right)^{k_n} \left[\left(1 - \frac{i}{n}\right)^{k_n} - \left(1 - \frac{i+1}{n}\right)^{k_n} \right] - 1.\end{aligned}$$

Let the map $f : \mathbb{R} \to \mathbb{R}$ be defined by $f(x) = (1-x)^{k_n}$. Then, by the mean value theorem,

$$0 \leq \left(1 - \frac{i}{n}\right)^{k_n} - \left(1 - \frac{i+1}{n}\right)^{k_n} \leq -\frac{1}{n} f'\left(\frac{i}{n}\right) = \frac{k_n}{n} \left(1 - \frac{i}{n}\right)^{k_n - 1}.$$

Thus,

$$\sum_{i=1}^n V_i^2 \leq \frac{2k_n}{n} \sum_{i=0}^{n-1} \left(1 - \frac{i}{n}\right)^{2k_n - 1} - 1.$$

In addition, let the map $g : \mathbb{R} \to \mathbb{R}$ be defined by $g(x) = (1-x)^{2k_n - 1}$. Observing that

$$\int_0^1 g(x) \mathrm{d}x = \frac{1}{2k_n},$$

we obtain

$$\begin{aligned}\sum_{i=1}^n V_i^2 &\leq 2k_n \left[\frac{1}{n} \sum_{i=0}^{n-1} g\left(\frac{i}{n}\right) - \int_0^1 g(x) \mathrm{d}x \right] \\ &= 2k_n \sum_{i=0}^{n-1} \int_{i/n}^{(i+1)/n} \left[g\left(\frac{i}{n}\right) - g(x) \right] \mathrm{d}x.\end{aligned}$$

Invoking again the mean value theorem, we may write, for all $x \in [i/n, (i+1)/n]$,

$$0 \leq g\left(\frac{i}{n}\right) - g(x) \leq -\frac{1}{n} g'\left(\frac{i}{n}\right).$$

Therefore,

$$\sum_{i=1}^{n} V_i^2 \leq \frac{2k_n}{n^2} \sum_{i=0}^{n-1} \left[-g'\left(\frac{i}{n}\right) \right].$$

Clearly,

$$\frac{1}{n} \sum_{i=0}^{n-1} \left[-g'\left(\frac{i}{n}\right) \right] \leq -\int_{-1/n}^{1-1/n} g'(x)\mathrm{d}x = g\left(-\frac{1}{n}\right) - g\left(1-\frac{1}{n}\right).$$

Putting all the pieces together, we finally obtain

$$\begin{aligned}\sum_{i=1}^{n} V_i^2 &\leq \frac{2k_n}{n} \left[\left(1+\frac{1}{n}\right)^{2k_n-1} - \left(\frac{1}{n}\right)^{2k_n-1} \right] \\ &\leq \frac{2k_n}{n} \left(1+\frac{1}{n}\right)^{2k_n}.\end{aligned}$$

This concludes the proof of the proposition.

3.4 Proof of Proposition 8

We distinguish between the cases $d = 1$, $d = 2$ and $d \geq 3$.

(i) If $d = 1$, for $i = 1, \ldots, n$, by definition of the V_i's,

$$\sum_{i=1}^{n} V_i \frac{i}{n} = \sum_{i=1}^{n} \left[\left(1 - \frac{i-1}{n}\right)^{k_n} - \left(1-\frac{i}{n}\right)^{k_n} \right] \frac{i}{n}.$$

Thus

$$\begin{aligned}\sum_{i=1}^{n} V_i \frac{i}{n} &= \sum_{i=1}^{n} \left[\left(1 - \frac{i}{n} + \frac{1}{n}\right)^{k_n} - \left(1-\frac{i}{n}\right)^{k_n} \right] \frac{i}{n} \\ &= \sum_{i=1}^{n} \left[\sum_{j=1}^{k_n} \binom{k_n}{j} \frac{1}{n^j} \left(1-\frac{i}{n}\right)^{k_n-j} \right] \frac{i}{n} \\ &= \sum_{j=1}^{k_n} \binom{k_n}{j} \frac{1}{n^{j-1}} \left[\frac{1}{n} \sum_{i=1}^{n} \frac{i}{n} \left(1-\frac{i}{n}\right)^{k_n-j} \right].\end{aligned}$$

For all $j = 1, \ldots, k_n$, we use the inequality

$$\frac{1}{n} \sum_{i=1}^{n} \frac{i}{n} \left(1-\frac{i}{n}\right)^{k_n-j} \leq 2 \int_0^1 x(1-x)^{k_n-j}\mathrm{d}x,$$

which is clearly true for $j = k_n$, without the factor 2 in front of the integral. For $j < k_n$, it is illustrated in Figure 1, where we have plotted the function $f(x) = x(1-x)^{k_n-j}$. The factor 2 is necessary because f is not monotonic on $[0,1]$.

Consequently,

$$\sum_{i=1}^{n} V_i \frac{i}{n} \leq 2 \sum_{j=1}^{k_n} \binom{k_n}{j} \frac{1}{n^{j-1}} \int_0^1 x(1-x)^{k_n-j}\mathrm{d}x.$$

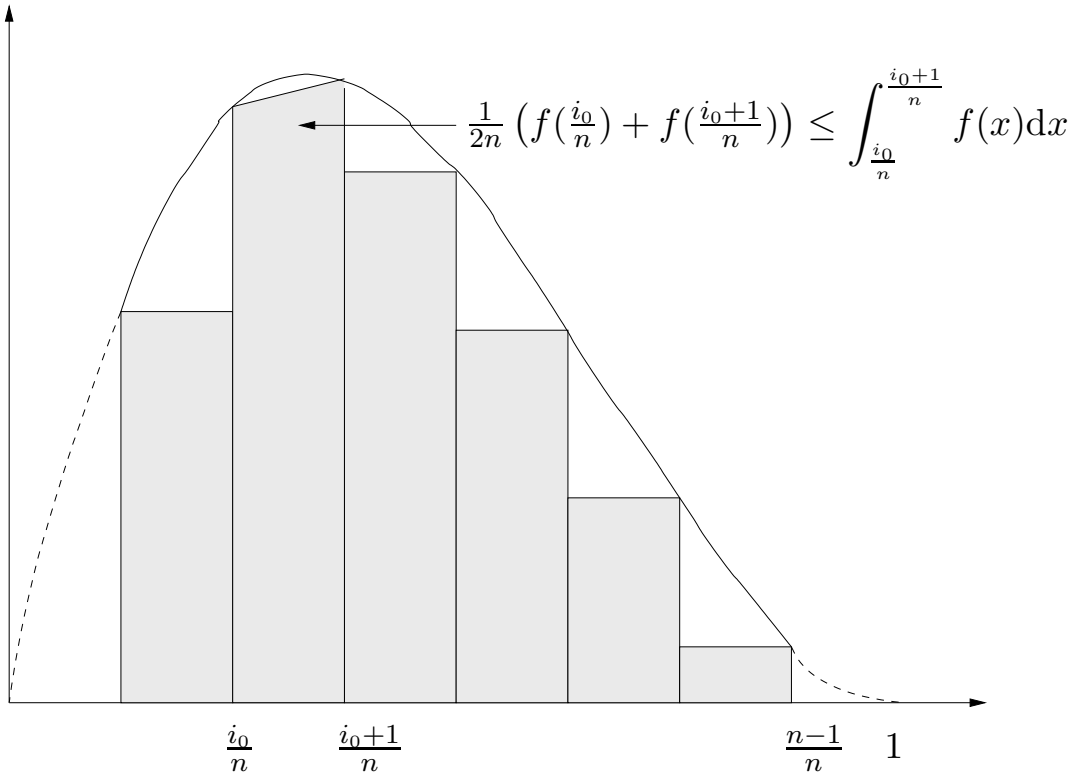


Figure 1: Illustration of $\frac{1}{n}\sum_{i=1}^{n}\frac{i}{n}\left(1-\frac{i}{n}\right)^{k_n-j} \le 2\int_0^1 x(1-x)^{k_n-j}\mathrm{d}x$.

Recalling the general formula

$$\int_0^1 x^{p-1}(1-x)^{q-1}\mathrm{d}x = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}, \quad p,q>0, \tag{5}$$

we obtain

$$\begin{aligned}
\sum_{i=1}^n V_i \frac{i}{n} &\leq 2\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\frac{\Gamma(2)\Gamma(k_n-j+1)}{\Gamma(k_n-j+3)} \\
&= 2\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\frac{1}{(k_n-j+1)(k_n-j+2)} \\
&= 2\sum_{j=1}^{k_n}\binom{k_n}{j-1}\frac{1}{n^{j-1}}\frac{1}{j(k_n-j+2)} \\
&= 2\sum_{j=0}^{k_n-1}\binom{k_n}{j}\frac{1}{n^{j}}\frac{1}{(j+1)(k_n-j+1)}.
\end{aligned}$$

Observing finally that $(j+1)(k_n-j+1)\geq k_n$ for all $j=0,\ldots,k_n-1$, we conclude

$$\sum_{i=1}^n V_i \frac{i}{n} \leq \frac{2}{k_n}\sum_{j=0}^{k_n-1}\binom{k_n}{j}\frac{1}{n^j} \leq \frac{2}{k_n}\left(1+\frac{1}{n}\right)^{k_n}.$$

(ii) For $d=2$, a reasoning similar to the one reported in statement (i) above can be followed, to show that

$$\begin{aligned}
\sum_{i=1}^n V_i \frac{i}{n}\left[1+\ln\binom{n}{i}\right] \leq 2\Bigg[&\frac{1}{k_n}\left(1+\frac{1}{n}\right)^{k_n} \\
&-\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\int_0^1 x(1-x)^{k_n-j}\ln x\,\mathrm{d}x\Bigg].
\end{aligned} \tag{6}$$

Denoting by H_n the n-th harmonic number, that is,

$$H_n = 1+\frac{1}{2}+\ldots+\frac{1}{n},$$

we have, for all $m\geq 0$ (see for example Gradshteyn and Ryzhik, 2007, Formula (4.253.1)),

$$-\int_0^1 x(1-x)^m\ln x\,\mathrm{d}x = \frac{H_{m+2}-1}{(m+1)(m+2)}.$$

Thus we may write

$$\begin{aligned}
&-\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\int_0^1 x(1-x)^{k_n-j}\ln x\,\mathrm{d}x \\
&\quad=\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\frac{H_{k_n-j+2}-1}{(k_n-j+1)(k_n-j+2)} \\
&\quad=\sum_{j=0}^{k_n-1}\binom{k_n}{j}\frac{1}{n^{j}}\frac{H_{k_n-j+1}-1}{(j+1)(k_n-j+1)}.
\end{aligned}$$

For all $j=0,\ldots,k_n-1$, we have $(j+1)(k_n-j+1)\geq k_n$, as well as

$$\begin{aligned} H_{k_n-j+1}-1 &= \frac{1}{2}+\ldots+\frac{1}{k_n-j+1} \\ &\leq \int_1^{k_n-j+1}\frac{\mathrm{d}x}{x} \\ &= \ln(k_n-j+1) \\ &\leq \ln(k_n+1). \end{aligned}$$

Therefore,

$$\begin{aligned} &-\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\int_0^1 x(1-x)^{k_n-j}\ln x\,\mathrm{d}x \\ &\leq \frac{\ln(k_n+1)}{k_n}\sum_{j=0}^{k_n-1}\binom{k_n}{j}\frac{1}{n^j} \\ &\leq \frac{\ln(k_n+1)}{k_n}\left(1+\frac{1}{n}\right)^{k_n}. \end{aligned} \tag{7}$$

Combining inequalities (6) and (7) leads to the desired result.

(*iii*) For $d\geq 3$, we note that for all $i=1,\ldots,n-1$,

$$\left\lfloor\frac{n}{i}\right\rfloor^{-\frac{2}{d}}\leq\left(\frac{i/n}{1-i/n}\right)^{\frac{2}{d}},$$

and set consequently

$$S_n=\frac{1}{n^{k_n}}+\sum_{i=1}^{n-1}\left[\left(1-\frac{i-1}{n}\right)^{k_n}-\left(1-\frac{i}{n}\right)^{k_n}\right]\left(\frac{i/n}{1-i/n}\right)^{\frac{2}{d}}.$$

We obtain

$$\begin{aligned} S_n &= \frac{1}{n^{k_n}}+\sum_{i=1}^{n-1}\left[\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^j}\left(1-\frac{i}{n}\right)^{k_n-j}\right]\left(\frac{i/n}{1-i/n}\right)^{\frac{2}{d}} \\ &= \frac{1}{n^{k_n}}+\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\left[\frac{1}{n}\sum_{i=1}^{n-1}\left(1-\frac{i}{n}\right)^{k_n-j-\frac{2}{d}}\left(\frac{i}{n}\right)^{\frac{2}{d}}\right] \\ &\leq \frac{1}{n^{k_n}}+2\sum_{j=1}^{k_n}\binom{k_n}{j}\frac{1}{n^{j-1}}\int_0^1 x^{\frac{2}{d}}(1-x)^{k_n-j-\frac{2}{d}}\,\mathrm{d}x. \end{aligned}$$

Applying formula (5) again, together with the identity

$$\Gamma\left(p+\frac{d-2}{d}\right)=\Gamma\left(\frac{d-2}{d}\right)\prod_{\ell=1}^{p}\left(\ell-\frac{2}{d}\right),$$

we obtain

$$\begin{aligned}
S_n &\leq \frac{1}{n^{k_n}} + \alpha_d \sum_{j=1}^{k_n} \binom{k_n}{j} \frac{1}{n^{j-1}} \frac{1}{(k_n - j + 1)} \prod_{\ell=1}^{k_n - j} \left(1 - \frac{2}{d\ell}\right) \\
&\quad \text{(with } \alpha_d = 2\Gamma((d-2)/d)\,\Gamma((d+2)/d)\text{)} \\
&= \frac{1}{n^{k_n}} + \alpha_d \sum_{j=1}^{k_n} \frac{k_n!}{j!(k_n - j + 1)!} \frac{1}{n^{j-1}} \prod_{\ell=1}^{k_n - j} \left(1 - \frac{2}{d\ell}\right) \\
&= \frac{1}{n^{k_n}} + \alpha_d \sum_{j=1}^{k_n} \frac{1}{n^{j-1}} \binom{k_n}{j-1} \frac{1}{j} \prod_{\ell=1}^{k_n - j} \left(1 - \frac{2}{d\ell}\right) \\
&= \frac{1}{n^{k_n}} + \alpha_d \sum_{j=0}^{k_n - 1} \frac{1}{n^j} \binom{k_n}{j} \frac{1}{j+1} \prod_{\ell=1}^{k_n - j - 1} \left(1 - \frac{2}{d\ell}\right).
\end{aligned}$$

Thus, by technical Lemma 17,

$$\begin{aligned}
S_n &\leq \frac{1}{n^{k_n}} + \alpha_d \sum_{j=0}^{k_n - 1} \binom{k_n}{j} \frac{k_n^{-\frac{2}{d}}}{n^j} \\
&\leq \frac{1}{n^{k_n}} + \alpha_d \left(1 + \frac{1}{n}\right)^{k_n} k_n^{-\frac{2}{d}}.
\end{aligned}$$

This concludes the proof of Proposition 8.

3.5 Proof of Proposition 11

We have, for $i = 1, \ldots, n - k_n + 1$,

$$\begin{aligned}
V_i &= \frac{\binom{n-i}{k_n - 1}}{\binom{n}{k_n}} \\
&= \frac{k_n}{n - k_n + 1} \prod_{j=0}^{k_n - 2} \left(1 - \frac{i}{n-j}\right) \\
&\leq \frac{k_n}{n - k_n + 1} \prod_{j=0}^{k_n - 2} \left(1 - \frac{i}{n}\right) \\
&= \frac{k_n}{n - k_n + 1} \left(1 - \frac{i}{n}\right)^{k_n - 1}.
\end{aligned}$$

This yields

$$\begin{aligned}
\sum_{i=1}^{n} V_i^2 &\leq \frac{k_n^2}{(n - k_n + 1)^2} \sum_{i=1}^{n - k_n + 1} \left(1 - \frac{i}{n}\right)^{2(k_n - 1)} \\
&\leq \frac{k_n^2 n}{(n - k_n + 1)^2} \frac{1}{n} \sum_{i=1}^{n} \left(1 - \frac{i}{n}\right)^{2(k_n - 1)}.
\end{aligned}$$

Observing finally that

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}\left(1-\frac{i}{n}\right)^{2(k_n-1)} &\le \int_0^1 (1-x)^{2(k_n-1)}\mathrm{d}x \\ &= \frac{1}{2k_n-1},\end{aligned}$$

we conclude that

$$\sum_{i=1}^{n} V_i^2 \le \frac{k_n^2 n}{(2k_n-1)(n-k_n+1)^2} \le \frac{k_n}{n}\,\frac{1}{(1-k_n/n+1/n)^2}.$$

3.6 Proof of Proposition 12

Recall that

$$\tilde{r}_n^\star(\mathbf{x}) = \sum_{i=1}^{n} V_i\, r(\mathbf{X}_{(i)}(\mathbf{x})),$$

and observe that

$$\tilde{r}_n^\star(\mathbf{x}) = \mathbb{E}^\star\left[r(\mathbf{X}_{(1)}^\star(\mathbf{x}))\right],$$

where $\mathbf{X}_{(1)}^\star(\mathbf{x})$ is the nearest neighbor of $\mathbf{x}$ in a random subsample $\mathcal{S}_n$ drawn without replacement from $\{(\mathbf{X}_1,Y_1),\ldots,(\mathbf{X}_n,Y_n)\}$ with $\mathrm{Card}(\mathcal{S}_n)=k_n$, and $\mathbb{E}^\star$ denotes expectation with respect to the resampling distribution, conditionally on the data set $\mathcal{D}_n$. Consequently, by Jensen's inequality,

$$\begin{aligned}\mathbb{E}\left[\tilde{r}_n^\star(\mathbf{x})-r(\mathbf{x})\right]^2 &= \mathbb{E}\left[\mathbb{E}^\star\left[r\left(\mathbf{X}_{(1)}^\star(\mathbf{x})\right)\mid\mathcal{D}_n\right]-r(\mathbf{x})\right]^2 \\ &= \mathbb{E}\left[\mathbb{E}^\star\left[r\left(\mathbf{X}_{(1)}^\star(\mathbf{x})\right)-r(\mathbf{x})\,|\,\mathcal{D}_n\right]\right]^2 \\ &\le \mathbb{E}\left[\mathbb{E}^\star\left[\left(r\left(\mathbf{X}_{(1)}^\star(\mathbf{x})\right)-r(\mathbf{x})\right)^2\mid\mathcal{D}_n\right]\right] \\ &= \mathbb{E}\left[r\left(\mathbf{X}_{(1)}^\star(\mathbf{x})\right)-r(\mathbf{x})\right]^2 \\ &\le C^2\,\mathbb{E}\|\mathbf{X}_{(1)}^\star(\mathbf{x})-\mathbf{x}\|^2.\end{aligned}$$

Since $\mathrm{Card}(\mathcal{S}_n)=k_n$, we conclude by applying Corollary 6, with $i=1$ and replacing n by k_n.

3.7 Two Technical Lemmas

Lemma 16 *For $j=0,\ldots,n-1$, let the map $\varphi_{n,j}(p)$ be defined by*

$$\varphi_{n,j}(p) = \binom{n}{j} p^{j+1}(1-p)^{n-j}, \quad 0\le p\le 1.$$

Then, for all $i=1,\ldots,n$,

$$\sup_{0\le p\le 1}\sum_{j=0}^{i-1}\varphi_{n,j}(p) \le \frac{i}{n+1}.$$

Proof Each map $\varphi_{n,j}$ is nonnegative, continuously increasing on the interval $[0,(j+1)/(n+1)]$ and decreasing on $[(j+1)/(n+1),1]$. Consequently, the supremum of the continuous function $\sum_{j=0}^{i-1}\varphi_{n,j}(p)$ is achieved at some point $p_\star$ of the interval $[1/(n+1),i/(n+1)]$. That is,

$$\begin{aligned}\sup_{0\le p\le 1}\sum_{j=0}^{i-1}\varphi_{n,j}(p) &= \sum_{j=0}^{i-1}\varphi_{n,j}(p_\star)\\ &= p_\star\sum_{j=0}^{i-1}\binom{n}{j}p_\star^j(1-p_\star)^{n-j}\\ &\le p_\star\sum_{j=0}^{n}\binom{n}{j}p_\star^j(1-p_\star)^{n-j}\\ &= p_\star \le \frac{i}{n+1}.\end{aligned}$$

■

Lemma 17 *For each $d\ge 3$, each $k_n\ge 1$, and $j=0,\dots,k_n-1$, we have*

$$\frac{1}{j+1}\prod_{\ell=1}^{k_n-j-1}\left(1-\frac{2}{d\ell}\right)\le k_n^{-\frac{2}{d}}.$$

Proof First, since $0\le 1-x\le e^{-x}$ for all $x\in[0,1]$,

$$\prod_{\ell=1}^{k_n-j-1}\left(1-\frac{2}{d\ell}\right)\le\exp\left(-\frac{2}{d}\sum_{\ell=1}^{k_n-j-1}\frac{1}{\ell}\right).$$

Thus, using $1+1/2+\ldots+1/p\ge\ln(p+1)$, we deduce

$$\prod_{\ell=1}^{k_n-j-1}\left(1-\frac{2}{d\ell}\right)\le(k_n-j)^{-\frac{2}{d}}.$$

To conclude, we use the fact that, for $j=0,\dots,k_n-1$,

$$\frac{1}{j+1}(k_n-j)^{-\frac{2}{d}}\le k_n^{-\frac{2}{d}}.$$

To see this, note that the inequality may be written under the equivalent form

$$\left(1-\frac{j}{k_n}\right)^{-\frac{2}{d}}\le 1+j=1+k_n\cdot\frac{j}{k_n}.$$

The result can easily be deduced from a comparison between the maps $\varphi:x\mapsto(1-x)^{-2/d}$ and $\psi:x\mapsto 1+k_nx$ on the interval $[0,1-1/k_n]$. Just note that $\varphi(0)=\psi(0)$, $\varphi(1-1/k_n)=k_n^{2/d}\le k_n=\psi(1-1/k_n)$ since $d\ge 3$, and φ is convex while ψ is affine. ■

Acknowledgments

The authors are greatly indebted to Luc Devroye for valuable comments and insightful suggestions on the first draft of the paper. They also thank two referees and the Acting Editor for their careful reading of the paper and thoughtful critical comments.

References

G. Biau and L. Devroye. *On the Layered Nearest Neighbour Estimate, the Bagged Nearest Neighbour Estimate and the Random Forest Method in Regression and Classification*. Technical Report, Université Pierre et Marie Curie, Paris, 2008. URL `http://www.lsta.upmc.fr/BIAU/bd4.pdf`.

L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

P. Bühlmann and B. Yu. Analyzing bagging. *The Annals of Statistics*, 30:927–961, 2002.

A. Buja and W. Stuetzle. Observations on bagging. *Statistica Sinica*, 16:323–352, 2006.

T.M. Cover. Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, 14:50–55, 1968a.

T.M. Cover. Rates of convergence for nearest neighbor procedures. In *Proceedings of the Hawaii International Conference on Systems Sciences*, pages 413–415, Honolulu, 1968b.

T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

L. Devroye. On the almost everywhere convergence of nonparametric regression function estimates. *The Annals of Statistics*, 9:1310–1319, 1981.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, 1996.

T.G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *First International Workshop on Multiple Classifier Systems*, Lecture Notes in Computer Science, pages 1–15, New York, 2000. Springer-Verlag.

E. Fix and J.L. Hodges. *Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties*. Technical Report 4, Project Number 21-49-004, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

E. Fix and J.L. Hodges. *Discriminatory analysis: Small sample performance*. Technical Report 11, Project Number 21-49-004, USAF School of Aviation Medicine, Randolph Field, Texas, 1952.

J.H. Friedman and P. Hall. On bagging and nonlinear estimation. *Journal of Statistical Planning and Inference*, 137:669–683, 2000.

I.S. Gradshteyn and I.M. Ryzhik. *Table of Integrals, Series, and Products*. Academic Press, New York, 2007.

L. Györfi. On the rate of convergence of nearest neighbor rules. *IEEE Transactions on Information Theory*, 29:509–512, 1978.

L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer-Verlag, New York, 2002.

P. Hall and R.J. Samworth. Properties of bagged nearest neighbour classifiers. *Journal of the Royal Statistical Society B*, 67:363–379, 2005.

I.A. Ibragimov and R.Z. Khasminskii. On nonparametric estimation of regression. *Doklady Akademii Nauk SSSR*, 252:780–784, 1980.

I.A. Ibragimov and R.Z. Khasminskii. *Statistical Estimation: Asymptotic Theory*. Springer-Verlag, New York, 1981.

I.A. Ibragimov and R.Z. Khasminskii. On the bounds for quality of nonparametric regression function estimation. *Theory of Probability and its Applications*, 27:81–94, 1982.

A.N. Kolmogorov and V.M. Tihomirov. ε-entropy and ε-capacity of sets in functional spaces. *American Mathematical Society Translations*, 17:277–364, 1961.

S.R. Kulkarni and S.E. Posner. Rates of convergence of nearest neighbor estimation under arbitrary sampling. *IEEE Transactions on Information Theory*, 41:1028–1039, 1995.

Y. Lin and Y. Jeon. Random forests and adaptive nearest neighbours. *Journal of the American Statistical Association*, 101:578–590, 2006.

D. Psaltis, R.R. Snapp, and S.S. Venkatesh. On the finite sample performance of the nearest neighbor classifier. *IEEE Transactions on Information Theory*, 40:820–837, 1994.

B.M. Steele. Exact bootstrap k-nearest neighbor learners. *Machine Learning*, 74:235–255, 2009.

C.J. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 5:595–645, 1977.

S.S. Venkatesh, R.R. Snapp, and D. Psaltis. Bellman strikes again! The growth rate of sample complexity with dimension for the nearest neighbor classifier. In J. Kittler and F. Roli, editors, *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 93–102, Pittsburgh, 1992.

Journal of Machine Learning Research 11 (2010) 713-741 Submitted 7/08; Revised 4/09; Published 2/10

A Fast Hybrid Algorithm for Large-Scale ℓ_1-Regularized Logistic Regression

Jianing Shi JS2615@COLUMBIA.EDU
Department of Biomedical Engineering
Columbia University
New York, NY 10025, USA

Wotao Yin WOTAO.YIN@RICE.DU
Department of Computational and Applied Mathematics
Rice University
Houston, TX 77005, USA

Stanley Osher SJO@MATH.UCLA.EDU
Department of Mathematics
University of California Los Angeles
Los Angeles, CA 90095, USA

Paul Sajda PS629@COLUMBIA.EDU
Department of Biomedical Engineering
Columbia University
New York, NY 10025, USA

Editor: Saharon Rosset

Abstract

ℓ_1-regularized logistic regression, also known as sparse logistic regression, is widely used in machine learning, computer vision, data mining, bioinformatics and neural signal processing. The use of ℓ_1 regularization attributes attractive properties to the classifier, such as feature selection, robustness to noise, and as a result, classifier generality in the context of supervised learning. When a sparse logistic regression problem has large-scale data in high dimensions, it is computationally expensive to minimize the non-differentiable ℓ_1-norm in the objective function. Motivated by recent work (Koh et al., 2007; Hale et al., 2008), we propose a novel hybrid algorithm based on combining two types of optimization iterations: one being very fast and memory friendly while the other being slower but more accurate. Called hybrid iterative shrinkage (HIS), the resulting algorithm is comprised of a fixed point continuation phase and an interior point phase. The first phase is based completely on memory efficient operations such as matrix-vector multiplications, while the second phase is based on a truncated Newton's method. Furthermore, we show that various optimization techniques, including line search and continuation, can significantly accelerate convergence. The algorithm has global convergence at a geometric rate (a Q-linear rate in optimization terminology). We present a numerical comparison with several existing algorithms, including an analysis using benchmark data from the UCI machine learning repository, and show our algorithm is the most computationally efficient without loss of accuracy.

Keywords: logistic regression, ℓ_1 regularization, fixed point continuation, supervised learning, large scale

1. Introduction

Logistic regression is an important linear classifier in machine learning and has been widely used in computer vision (Bishop, 2007), bioinformatics (Tsuruoka et al., 2007), gene classification (Liao and Chin, 2007), and neural signal processing (Parra et al., 2005; Gerson et al., 2005; Philiastides and Sajda, 2006). ℓ_1-regularized logistic regression or so-called sparse logistic regression (Tibshirani, 1996), where the weight vector of the classifier has a small number of nonzero values, has been shown to have attractive properties such as feature selection and robustness to noise. For supervised learning with many features but limited training samples, overfitting to the training data can be a problem in the absence of proper regularization (Vapnik, 1982, 1988). To reduce overfitting and obtain a robust classifier, one must find a sparse solution.

Minimizing or limiting the ℓ_1-norm of an unknown variable (the weight vector in logistic regression) has long been recognized as a practical avenue for obtaining a sparse solution. The use of ℓ_1 minimization is based on the assumption that the classifier parameters have, *a priori*, a Laplace distribution, and can be implemented using maximum-a-posteriori (MAP). The ℓ_2-norm is a result of penalizing the mean of a Gaussian prior, while a ℓ_1-norm models a Laplace prior, a distribution with heavier tails, and penalizes on its median. Such an assumption attributes important properties to ℓ_1-regularized logistic regression in that it tolerates outliers and, therefore, is robust to irrelevant features and noise in the data. Since the solution is sparse, the nonzero components in the solution correspond to useful features for classification; therefore, ℓ_1 minimization also performs feature selection (Littlestone, 1988; Ng, 1998), an important task for data mining and biomedical data analysis.

1.1 Logistic Regression

The basic form of logistic regression seeks a hyperplane that separates data belonging to two classes. The inputs are a set of training data $X = [x_1, \cdots, x_m]^\top \in \mathbb{R}^{m \times n}$, where each row of X is a sample and samples of either class are assumed to be independently identically distributed, and class labels $b \in \mathbb{R}^m$ are of $-1/+1$ elements. A linear classifier is a hyperplane $\{x : w^\top x + v = 0\}$, where $w \in \mathbb{R}^n$ is a set of weights and $v \in \mathbb{R}$ the intercept. The conditional probability for the classifier label b based on the data, according to the logistic model, takes the following form,

$$p(b_i|x_i) = \frac{\exp\big((w^T x_i + v)b_i\big)}{1 + \exp\big((w^T x_i + v)b_i\big)}, \quad i = 1, ..., m.$$

The average logistic loss function can be derived from the empirical logistic loss, computed from the negative log-likelihood of the logistic model associated with all the samples, divided by number of samples m,

$$l_{\text{avg}}(w, v) = \frac{1}{m} \sum_{i=1}^{m} \theta\big((w^T x_i + v)b_i\big),$$

where θ is the logistic transfer function: $\theta(z) := \log(1 + \exp(-z))$. The classifier parameters w and v can be determined by minimizing the average logistic loss function,

$$\arg\min_{w,v} \; l_{\text{avg}}(w, v).$$

Such an optimization can also be interpreted as a MAP estimate for classifier weights w and intercept v.

1.2 ℓ_1-Regularized Logistic Regression

The so-called *sparse logistic regression* has emerged as a popular linear decoder in the field of machine learning, adding the ℓ_1-penalty on the weights w:

$$\arg\min_{w,v} \; l_{\text{avg}}(w,v) + \lambda \|w\|_1, \tag{1}$$

where λ is a regularization parameter. It is well-known that ℓ_1 minimization tends to give sparse solutions. The ℓ_1 regularization results in logarithmic sample complexity bounds (number of training samples required to learn a function), making it an effective learner even under an exponential number of irrelevant features (Ng, 1998, 2004). Furthermore, ℓ_1 regularization also has appealing asymptotic sample-consistency for feature selection (Zhao and Yu, 2007).

Signals arising in the natural world tend to be sparse (Parra et al., 2001). Sparsity also arises in signals represented in a certain basis, such as the wavelet transform, the Krylov subspace, etc. Exploiting sparsity in a signal is therefore a natural constraint to employ in algorithm development. An exact form of sparsity can be sought using the ℓ_0 regularization, which explicitly penalizes the number of nonzero components,

$$\arg\min_{w,v} \; l_{\text{avg}}(w,v) + \lambda \|w\|_0. \tag{2}$$

Although theoretically attractive, problem (2) is in general NP-hard (Natarajan, 1995), requiring an exhaustive search. Due to this computational complexity, ℓ_1 regularization has become a popular alternative, and is subtly different than ℓ_0 regularization, in that the ℓ_1-norm penalizes large coefficients/parameters more than small ones.

The idea of adopting the ℓ_1 regularization for seeking sparse solutions to optimization problems has a long history. As early as the 1970's, Claerbout and Muir first proposed to use ℓ_1 to deconvolve seismic traces (Claerbout and Muir, 1973), where a sparse reflection function was sought from bandlimited data (Taylor et al., 1979). In the 1980's, Donoho et al. quantified the ability of ℓ_1 to recover sparse reflectivity functions (Donoho and Stark, 1989; Donoho and Logan, 1992). After the 1990s', there was a dramatic rise of applications using the sparsity-promoting property of the ℓ_1-norm. Sparse model selection was proposed in statistics using LASSO (Tibshirani, 1996), wherein the proposed soft thresholding is related to wavelet thresholding (Donoho et al., 1995). Basis pursuit, which aims to extract sparse signal representation from overcomplete dictionaries, also underwent great development during this time (Donoho and Stark, 1989; Donoho and Logan, 1992; Chen et al., 1998; Donoho and Huo, 2001; Donoho and Elad, 2003; Donoho, 2006). In recent years, minimization of the ℓ_1-norm has appeared as a key element in the emerging field of compressive sensing (Candés et al., 2006; Candés and Tao, 2006; Figueiredo et al., 2007; Hale et al., 2008). ℓ_1 minimization also has far reaching impact on various applications such as portfolio optimization (Lobo et al., 2007), sparse principle component analysis (d'Aspremont et al., 2005; Zou et al., 2006), sparse interconnect wiring design (Vandenberghe et al., 1997, 1998), sparse control system design (Hassibi et al., 1999), and optimization of well-connected sparse graphs (Ghosh and Boyd, 2006). Research on total variation based image processing also shows that minimizing the ℓ_1-norm of the intensity gradient can effectively remove random noise (Rudin et al., 1992). In the realm of machine learning, ℓ_1 regularization exists in various forms of classifiers, including ℓ_1-regularized logistic regression (Tibshirani, 1996), ℓ_1-regularized probit regression (Figueiredo and Jain, 2001; Figueiredo, 2003), ℓ_1-regularized support vector machines (Zhu et al., 2004), and ℓ_1-regularized multinomial logistic regression (Krishnapuram et al., 2005).

1.3 Existing Algorithms for ℓ_1-Regularized Logistic Regression

The ℓ_1-regularized logistic regression problem (1) is a convex and non-differentiable problem. A solution always exists but can be non-unique. These characteristics postulate some difficulties in solving the problem. Generic methods for nondifferentiable convex optimization, such as the ellipsoid method and various sub-gradient methods (Shor, 1985; Polyak, 1987), are not designed to handle instances of (1) with data of very large scale. There has been very active development on numerical algorithms for solving the ℓ_1-regularized logistic regression, including LASSO (Tibshirani, 1996), Gl1ce (Lokhorst, 1999), Grafting (Perkins and Theiler, 2003), GenLASSO (Roth, 2004), and SCGIS (Goodman, 2004). The IRLS-LARS (iteratively reweighted least squares least angle regression) algorithm uses a quadratic approximation for the average logistic loss function, which is consequently solved by the LARS (least angle regression) method (Efron et al., 2004; Lee et al., 2006). The BBR (Bayesian logistic regression) algorithm, described in Eyheramendy et al. (2003), Madigan et al. (2005), and Genkin et al. (2007), uses a cyclic coordinate descent method for the Bayesian logistic regression. Glmpath, a solver for ℓ_1-regularized generalized linear models using path following methods, can also handle the logistic regression problem (Park and Hastie, 2007). MOSEK is a general purpose primal-dual interior point solver, which can solve the ℓ_1-regularized logistic regression by formulating the dual problem, or treating it as a geometric program (Boyd et al., 2007). SMLR, algorithms for various sparse linear classifiers, can also solve sparse logistic regression (Krishnapuram et al., 2005). Recently, Koh, Kim, and Boyd proposed an interior-point method (Koh et al., 2007) for solving (1). Their algorithm takes truncated Newton steps and uses preconditioned conjugated gradient iterations. This interior-point solver is efficient and provides a highly accurate solution. The truncated Newton method has fast convergence, but forming and solving the underlying Newton systems require excessive amounts of memory for large-scale problems, making solving such large-scale problems prohibitive. A comparison of several of these different algorithms can be found in Schmidt et al. (2007).

1.4 Our Hybrid Algorithm

In this paper, we propose a hybrid algorithm that is comprised of two phases: the first phase is based on a new algorithm called iterative shrinkage, inspired by a fixed point continuation (FPC) (Hale et al., 2008), which is computationally fast and memory friendly; the second phase is a customized interior point method, devised by Koh et al. (2007).

Figure 1 shows a diagram of our hybrid algorithm, termed Hybrid Iterative Shrinkage (HIS) algorithm. Our algorithm requires less memory and, on mid/large-scale problems, runs faster than the interior point method. The iterative shrinkage phase only performs matrix-vector multiplications in size of X, as well as a very simple *shrinkage* operation (see (6) below), and therefore requires minimal memory consumption. By extending the results in Hale et al. (2008), we prove Q-linear convergence and show that the signs of w_{opt} (hence, the indices of nonzero elements) are obtained in a finite number of steps, typically much earlier than convergence. Based on the latter result, we propose a hybrid algorithm that is even faster and results in highly accurate solutions. Specifically, our algorithm predicts the sign changes in future shrinkage iterations, and when the signs of w^k are likely to be stable, switches to the interior point method and operates on a reduced problem that is much smaller than the original. The interior point method achieves high accuracy in the solution, making our hybrid algorithm equally accurate, as will be shown in the Section 4.

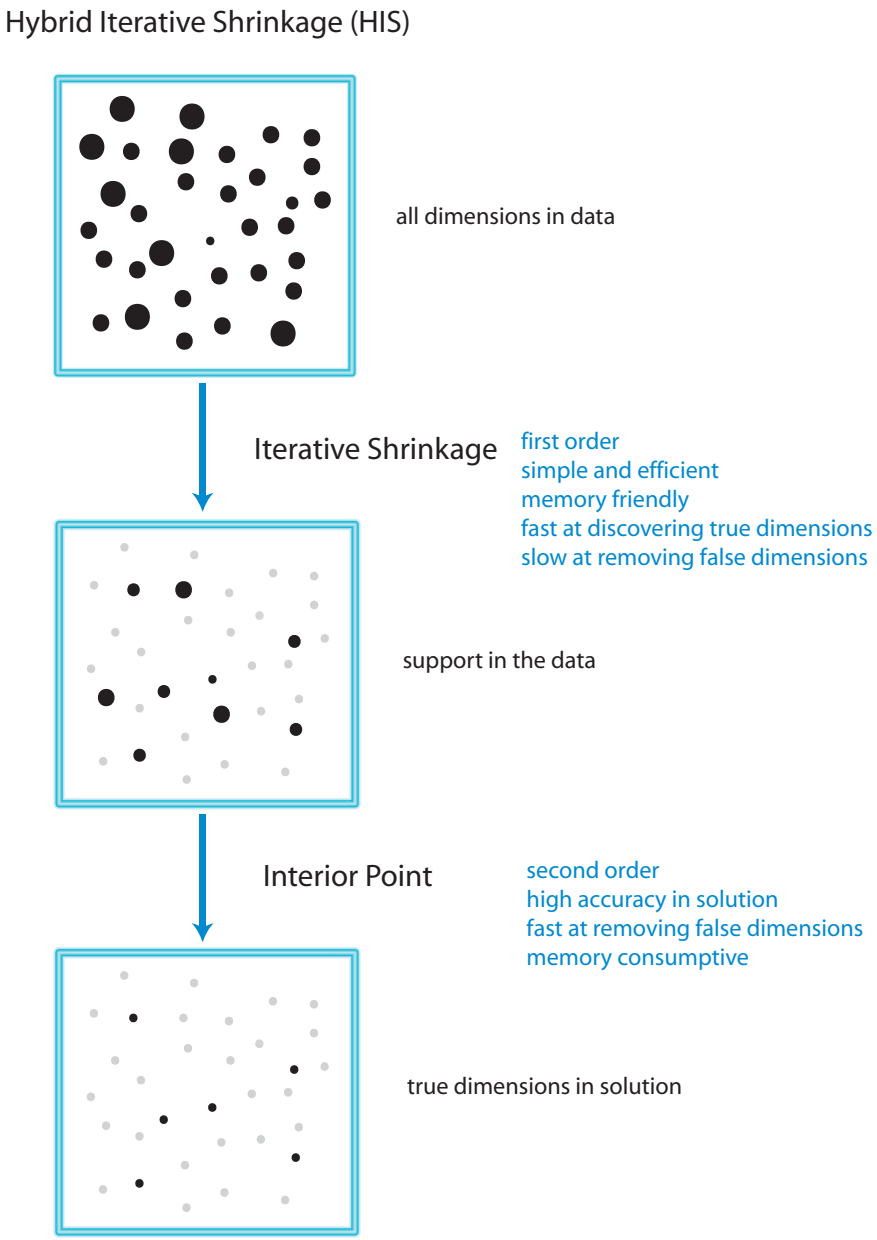


Figure 1: A diagram of our proposed hybrid iterative shrinkage (HIS) algorithm. The HIS algorithm is comprised of two phases: the iterative shrinkage phase and the interior point phase. The iterative shrinkage is inspired by a fixed point continuation method (Hale et al., 2008), which is computationally fast and memory friendly. The interior point method is based on a second-order truncated Newton method, devised by Koh et al. (2007). Our hybrid approach takes advantage of different computational strengths of the two methods and uses them for optimal algorithm acceleration while attaining high accuracy. Black dots indicate the nonzero dimensions, gray dots indicate dimensions that are eliminated, and the size of the dots show the error that each dimension contributes to the final solution. Note that the final solution is sparse with an overall small error.

There are several novel aspects of our hybrid approach. The rationale of the hybrid approach is based on the observation that the iterative shrinkage phase reduces the algorithm to gradient projection after a finite number of iterations, which will be described in Section 3.1. We build on this observation a hybrid approach to take advantage of the two phases of the computation using two types of numerical methods. In the first phase, inspired by the FPC by Hale et al. (2008), we customize the iterative shrinkage algorithm for the sparse logistic regression, whose objective function is not quadratic. In particular, the step length in the iterative shrinkage algorithm is not constant, unlike the compressive sensing problem. Therefore, we resort to a line search strategy to avoid computing the Hessian matrix (required for finding the step length for stability). In addition, the ℓ_1 regularization is only applied to the w component and not v in sparse logistic regression. This change in the model requires a different shrinkage step, as well as a careful treatment in the line search strategy.

The remainder of the paper is organized as follows. In Section 2, we present the iterative shrinkage algorithm for sparse logistic regression, and prove its global convergence and Q-linear convergence. In Section 3, we provide the rationale for the hybrid approach, together with a description of the hybrid algorithm. Numerical results are presented in Section 4. We conclude the paper in Section 5.

2. Sparse Logistic Regression using Iterative Shrinkage

The iterative shrinkage algorithm used in the first phase is inspired by a fixed point continuation algorithm by Hale et al. (2008).

2.1 Notation

For simplicity, we define $\|\cdot\| := \|\cdot\|_2$, as the Euclidean norm. The *support* of $x \in \mathbb{R}^n$ is denoted by $\mathrm{supp}(x) := \{i : x_i \neq 0\}$. We use g to denote the gradient of f, that is, $g(x) = \nabla f(x), \forall x$. For any index set $I \subseteq \{1,\ldots,n\}$ (later, we will use index sets E and L), $|I|$ is the cardinality of I and x_I is defined as the sub-vector of x of length $|I|$, consisting only of components x_i, $i \in I$. Similarly, for any vector-value mapping h, $h_I(x)$ denotes the sub-vector of $h(x)$ consisting of $h_i(x)$, $i \in I$.

To express the subdifferential of $\|\cdot\|_1$ we will use the signum function and multi-function (i.e., set-valued mapping). The signum function of $t \in \mathbb{R}$ is

$$\mathrm{sgn}(t) := \begin{cases} +1 & t > 0, \\ 0 & t = 0, \\ -1 & t < 0; \end{cases}$$

while the signum multi-function of $t \in \mathbb{R}$ is

$$\mathrm{SGN}(t) := \partial|t| = \begin{cases} \{+1\} & t > 0, \\ [-1,1] & t = 0, \\ \{-1\} & t < 0, \end{cases}$$

which is also the subdifferential of $|t|$.

For $x \in \mathbb{R}^n$, we define $\mathrm{sgn}(x) \in \mathbb{R}^n$ and $\mathrm{SGN}(x) \subset \mathbb{R}^n$ component-wise as $(\mathrm{sgn}(x))_i := \mathrm{sgn}(x_i)$ and $(\mathrm{SGN}(x))_i := \mathrm{SGN}(x_i)$, $i = 1,2,\cdots,n$, respectively. Furthermore, vector operators such as $|x|$

and $\max\{x,y\}$ are defined to operate component-wise, analogous with the definitions of sgn and SGN above. For $x,y \in \mathbb{R}^n$, let $x \odot y \in \mathbb{R}^n$ denote the component-wise product of x and y, that is, $(x \odot y)_i = x_i y_i$. Finally, we let X^* denote the set of all optimal solutions of problem (3).

2.2 Review of Fixed Point Continuation for ℓ_1-minimization

A fixed-point continuation algorithm was proposed in Hale et al. (2008) as a fast algorithm for large-scale ℓ_1-regularized convex optimization problems. The authors considered the following large-scale ℓ_1-regularized minimization problem,

$$\min_{x\in\mathbb{R}^n} f(x) + \lambda\|x\|_1, \tag{3}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable and convex, but not necessarily strictly convex, and $\lambda > 0$. They devised a fixed-point iterative algorithm and proved its global convergence, including finite convergence for some quantities, and a Q-linear (Quotient-linear) convergence rate without assuming strict convexity of f or solution uniqueness. Numerically they demonstrated Q-linear convergence in the quadratic case $f(x) = \|Ax-b\|_2^2$, where A is completely dense, and applied their algorithm to ℓ_1-regularized compressed sensing problems. As we will adopt this algorithm for solving our problem (1), we review some important and useful results here and develop some new insights in the context of ℓ_1-regularized logistic regression.

The rationale for FPC is based on the idea of operator splitting. It is well-known in convex analysis that minimizing a function in the form of $\phi(x) = \phi_1(x) + \phi_2(x)$, where both ϕ_1 and ϕ_2 are convex, is equivalent to finding a zero of the subdifferential $\partial\phi(x)$, that is, seeking x satisfying $\mathbf{0} \in T_1(x) + T_2(x)$ for $T_1 := \partial\phi_1$ and $T_2 := \partial\phi_2$. We say $(I+\tau T_1)$ is invertible if $y = x+\tau T_1(x)$ has a unique solution x for any given y. For $\tau > 0$, if $(I+\tau T_1)$ is invertible and T_2 is single-valued, then

$$\begin{aligned} \mathbf{0} \in T_1(x) + T_2(x) &\iff \mathbf{0} \in (x+\tau T_1(x)) - (x-\tau T_2(x)) \\ &\iff (I-\tau T_2)x \in (I+\tau T_1)x \\ &\iff x = (I+\tau T_1)^{-1}(I-\tau T_2)x. \end{aligned} \tag{4}$$

This gives rise to the forward-backward splitting algorithm in the form of a fixed-point iteration,

$$x^{k+1} := (I+\tau T_1)^{-1}(I-\tau T_2)x^k. \tag{5}$$

Applying (4) to problem (1), where $\phi_1(x) := \lambda\|x\|_1$ and $\phi_2(x) := f(x)$, the authors of Hale et al. (2008) obtained the following optimality condition of x^*:

$$x^* \in X^* \iff \mathbf{0} \in g(x^*) + \lambda \mathrm{SGN}(x^*) \Longleftrightarrow x^* = (I+\tau T_1)^{-1}(I-\tau T_2)x^*,$$

where $T_2(\cdot) = g(\cdot)$, the gradient of $f(\cdot)$, and $(I+\tau T_1)^{-1}(\cdot)$ is the shrinkage operator. Therefore, the fixed-point iteration (5) for solving (3) becomes

$$x^{k+1} = s \circ h(x^k),$$

which is a composition of two mappings s and h from $\mathbb{R}^n$ to $\mathbb{R}^n$.

The gradient descent operator is defined as

$$h(\cdot) := I(\cdot) - \tau\nabla f(\cdot).$$

The shrinkage operator, on the other hand, can be written as

$$s(\cdot) = \mathrm{sgn}(\cdot) \odot \max\{|\cdot| - \nu, 0\}, \tag{6}$$

where $\nu = \lambda\tau$. Shrinkage is also referred to soft-thresholding in the language of wavelet analysis:

$$(s(y))_i = \begin{cases} y_i - \nu, & y_i > \nu, \\ 0, & y_i \in [-\nu, \nu], \\ y_i + \nu, & y_i < -\nu. \end{cases}$$

In each iteration, the gradient descent step h reduces $f(x)$ by moving along the negative gradient direction of $f(x^k)$ and the shrinkage step s reduces the ℓ_1-norm by "shrinking" the magnitude of each nonzero component in the input vector.

2.3 Iterative Shrinkage for Sparse Logistic Regression

Recall that in the sparse logistic regression problem (1), the ℓ_1 regularization is only applied to w, not to v. Therefore, we propose a slightly different fixed point iteration. For simplicity of notation, we define column vectors $u = (w; v) \in \mathbb{R}^{n+1}$ and $c_i = (a_i; b_i) \in \mathbb{R}^{n+1}$, where $a_i = bx_i$, for $i = 1, 2, ..., m$. This reduces (1) to

$$\min_u \; l_{\mathrm{avg}}(u) + \lambda \|u_{1:n}\|_1,$$

where $l_{\mathrm{avg}} = \frac{1}{m}\sum_{i=1}^m \theta(c_i^T u)$, and θ denotes the logistic transfer function $\theta(z) = \log(1 + \exp(-z))$.

The gradient and Hessian of l_{avg} with respect to u is given by

$$\begin{aligned} g(u) &\equiv \nabla l_{\mathrm{avg}}(u) = \frac{1}{m}\sum_{i=1}^m \theta'(c_i^\top u) c_i, \\ H(u) &\equiv \nabla^2 l_{\mathrm{avg}}(u) = \frac{1}{m}\sum_{i=1}^m \theta''(c_i^\top u) c_i c_i^\top, \end{aligned}$$

where $\theta'(z) = -(1 + e^z)^{-1}$ and $\theta''(z) = (2 + e^{-z} + e^z)^{-1}$. To guarantee convergence, we require the step length be bounded by $2(\max_u \lambda_{\max} H(u))^{-1}$.

The iterative shrinkage algorithm for sparse logistic regression is

$$\begin{aligned} u^{k+1} &= s \circ h_{1:n}(u^k), \text{ for } w \text{ component}, \\ u^{k+1} &= h_{n+1}(u^k), \text{ for } v \text{ component}, \end{aligned} \tag{7}$$

which is a composition of two mappings h and s from $\mathbb{R}^n$ to $\mathbb{R}^n$, where the gradient operator is

$$h(\cdot) = \cdot - \tau g(\cdot) = \cdot - \tau \nabla l_{\mathrm{avg}}(\cdot).$$

While the authors in Hale et al. (2008) use a constant step length satisfying

$$0 < \tau < 2/\lambda_{\max}\{H_{EE}(u) : u \in \Omega\},$$

we employ line search to avoid the expensive calculation of maximum eigenvalues. We will present the convergence of the iterative shrinkage algorithm in Section 2.4. The details of the line search algorithm will be discussed in Section 2.5.

Algorithm 1 Fixed-Point Continuation Algorithm

Require: $A = [c_1^\top; c_2^\top; \cdots; c_m^\top] \in \mathbb{R}^{m\times(n+1)}$, $u = (w; v) \in \mathbb{R}^{n+1}$, $f(u) = m^{-1}\phi(Au)$, task: $\min_u l_{\text{avg}}(u) + \lambda\|u\|_1$
  Initialize u^0
  while "not converge" **do**
    Armijo-like line search algorithm (Algorithm 2)
    $k = k + 1$
  end while

2.4 Convergence

Global convergence and finite convergence on certain quantities were proven in Hale et al. (2008) when the following conditions are met: (i) the optimal solution set X^* is non-empty, (ii) $f \in C^2$ and its Hessian $H = \nabla^2 f$ is positive semi-definite in $\Omega = \{x : \|x - x^*\| \le \rho\} \subset \mathbb{R}^n$ for $\rho > 0$, and (iii) the maximum eigenvalue of H is bounded on Ω by a constant $\hat{\lambda}_{\max}$ and the step length τ is uniformly less than $2/\hat{\lambda}_{\max}$. These conditions are sufficient for the forward operator $h(\cdot)$ to be non-expansive.

Assumption 1 *Assume problem* (1) *has an optimal solution set $X^* \neq \emptyset$, and there exists a set*

$$\Omega = \{x : \|x - x^*\| \le \rho\} \subset \mathbb{R}^n$$

for some $x^* \in X^*$ and $\rho > 0$ such that $f \in C^2(\Omega)$, $H(x) := \nabla^2 f(x) \succeq 0$ for $x \in \Omega$ and

$$\hat{\lambda}_{\max} := \sup_{x\in\Omega} \lambda_{\max}(H(x)) < \infty.$$

For simplicity for the analysis, we choose a constant step length τ in the fixed-point iterations (7)*: $x^{k+1} = s(x^k - \tau g(x^k))$, where $\nu = \tau\lambda$, and*

$$\tau \in \left(0, 2/\hat{\lambda}_{\max}\right),$$

which guarantees that $h(\cdot) = I(\cdot) - \tau g(\cdot)$ is non-expansive in Ω.

Theorem 1 *Under Assumption, the sequence $\{u^k\}$ generated by the fixed-point iterations* (7) *applied to problem* (1) *from any starting point $x^0 \in \Omega$ converges to some $u^* \in U^* \cap \Omega$. In addition, for all but finitely many iterations, we have*

$$u_i^k = u_i^* = 0, \qquad \forall i \in L = \{i : |g_i^*| < \lambda, 1 \le i \le n\}, \tag{8}$$

$$\text{sgn}(h_i(u^k)) = \text{sgn}(h_i(u^*)) = -\frac{1}{\lambda} g_i^*, \qquad \forall i \in E = \{i : |g_i^*| = \lambda, 1 \le i \le n\}, \tag{9}$$

where as long as

$$\omega := \min\{\nu(1 - \frac{|g_i^*|}{\lambda}) : i \in L\} > 0.$$

The numbers of iterations not satisfying (8) *and* (9) *do not exceed $\|u^0 - u^*\|^2/\omega^2$ and $\|u^0 - u^*\|^2/\nu^2$, respectively.*

Proof We sketch the proof here. First, the iteration (7) is shown to be non-expansive in ℓ_2, that is, $\|u^k - u^*\|$ does not increase in k with the assumption on the step length τ. Specifically, in Assumption, the step length τ is chosen small enough to guarantee that $\|h(u^k) - h(u^*)\| \leq \|u^k - u^*\|$ (in practice, τ is determined, for example, by line search.) On the other hand, through a component-wise analysis, one can show that no matter what τ is, the shrinkage operator $s(\cdot)$ is always non-expansive, that is, $\|s(h_{1:n}(u^k)) - s(h_{1:n}(u^*))\| \leq \|h_{1:n}(u^k) - h_{1:n}(u^*)\|$. Therefore, from the definition of u^{k+1} in (7), we have

$$\|u^{k+1} - u^*\| \leq \|u^k - u^*\|, \tag{10}$$

using the fact that u^* is optimal if and only if u^* is a fixed point with respect to (7). However, this non-expansiveness of (7) does not directly give convergence.

Next, $\{u^k\}$ is shown to have a limit point $\bar{x}$, that is, a subsequence converging to $\bar{u}$, due to the compactness of Ω and (10). (7) can be proven to converge globally to $\bar{u}$. To show this, we first get

$$\|[s \circ h_{1:n}(\bar{u}); h_{n+1}(\bar{u})] - [s \circ h_{1:n}(u^*); h_{n+1}(u^*)]\| = \|\bar{u} - u^*\|,$$

from the fact that $\bar{u}$ is a limit point, and then use this equation to show that $\bar{u} = [s \circ h_{1:n}(\bar{u}); h_{n+1}(\bar{u})]$, that is, $\bar{u}$ is a fixed point with respect to (7), and thus an optimal solution. Repeating the first step above we have $\|u^{k+1} - \bar{u}\| \leq \|u^k - \bar{u}\|$, which extends $\bar{u}$ from being the limit of a subsequence to one of the entire sequence.

Finally, to obtain the finite convergence result, we need to take a closer look at the shrinkage operator $s(\cdot)$. When (8) does not hold for some iteration k at component i, we have $|u_i^{k+1} - u_i^*|^2 \leq |u_i^k - u_i^*|^2 - \omega^2$, and for (9), we have $|u_i^{k+1} - u_i^*|^2 \leq |u_i^k - u_i^*|^2 - \nu^2$. Obviously, there can be only a finite number of iterations k in which either (8) or (9) does not hold, and such numbers do not exceed $\|u^0 - u^*\|^2/\omega^2$ and $\|u^0 - u^*\|^2/\nu^2$, respectively. ■

A linear convergence result with a certain convergence rate can also be obtained. As long as $H_{EE}(x^*) := [H_{i,j}(x^*)]_{i,j \in E}$ has full rank *or* $f(x)$ is convex quadratic in x, the sequence $\{x^k\}$ converges to x^* *R*-linearly, and $\{\|x^k\|_1 + \mu f(x^k)\}$ converges to $\|x^*\|_1 + \mu f(x^*)$ *Q*-linearly. Furthermore, if $H_{EE}(x^*)$ has the full rank, then *R*-linear convergence can be strengthened to *Q*-linear convergence by using the fact that the minimal eigenvalue of H_{EE} at x^* is strictly greater than 0.

2.5 Line Search

An important element of the iterative shrinkage algorithm is the step length τ at each iteration. To ensure the stability of the algorithm, we require that the step length satisfy

$$0 < \tau < 2/\lambda_{\max}\{H_{EE}(u) : u \in \Omega\}.$$

In compressive sensing, where the smooth part of the objective function is quadratic, the step length is constant. In sparse logistic regression, however, the Hessian matrix changes at each iteration. If one has to dynamically compute the step length at each iteration, this requires an expensive computation for the Hessian matrix. Therefore, we resort to an "Armijo-like" line search algorithm to avoid such a computational burden. For large-scale problems, a line search method, if used appropriately, can save tremendous CPU time and memory. Convergence of the Armijo-like line search is not proven in our paper, however heuristic results are obtained through numerical experiments.

Algorithm 2 Armijo-like Line Search Algorithm

Compute heuristic step length α_0
Gradient step: $u^{k-} = u^k - \alpha_0 \nabla l_{\text{avg}}(u^k)$
Shrinkage step: $u^{k+} = s_{1:n}(u^{k-}, \lambda\alpha_0)$
Obtain search direction: $p^k = u^{k+} - u^k$
while "j $<$ max line search attempts" **do**
  if Armijo-like condition is met **then**
    Accept line search step, update $u^{k+1} = u^k + \alpha_j p^k$
  else
    Keep backtracking $\alpha_j = \mu\alpha_{j-1}$
  end if
  $j = j+1$
end while

Let's denote the objective function for the ℓ_1-regularized logistic regression as $\phi(u)$ for convenience:

$$\phi(u) = l_{\text{avg}}(u) + \lambda\|u_{1:n}\|_1,$$

where $l_{\text{avg}}(u) = \frac{1}{m}\sum_{i=1}^{m} \theta(c_i^T u)$ and θ is the logistic transfer function. A line search method, at each iteration, computes the step length α_k and the search direction p^k:

$$u^{k+1} = u^k + \alpha_k p^k.$$

The search direction will be described in Eqn. (12). For our sparse logistic regression, a sequence of step length candidates are identified, and a decision is made to accept one when certain conditions are satisfied. We compute a heuristic step length and gradually decrease it until a sufficient decrease condition is met.

Let's define the heuristic step length as α_0. Ideally the choice of step length α_0, would be a global minimizer of the smooth part of the objective function,

$$\varphi(\alpha) = l_{\text{avg}}(u^k + \alpha p^k), \;\; \alpha > 0,$$

which is too expensive to evaluate, unlike the quadratic case in compressive sensing. Therefore, an inexact line search strategy is usually performed in practice to identify a step length that achieves sufficient decrease in $\varphi(\alpha)$ at minimal cost. Motivated by a similar approach in GPSR (Figueiredo et al., 2007), we compute the heuristic step length through a minimizer of the quadratic approximation for $\varphi(\alpha)$,

$$l_{\text{avg}}(u^k - \alpha\nabla l_{\text{avg}}(u^k)) \approx l_{\text{avg}}(u^k) - \alpha\nabla l_{\text{avg}}(u^k)^T \nabla l_{\text{avg}}(u^k) + 0.5\alpha^2 \nabla l_{\text{avg}}(u^k)^T H(u^k) \nabla l_{\text{avg}}(u^k).$$

Differentiating the right-hand side with respect to α and setting the derivative to zero, we obtain

$$\alpha_0 = \frac{\nabla l_{\text{avg}}(\bar{u}^k)^T \nabla l_{\text{avg}}(\bar{u}^k)}{\nabla l_{\text{avg}}(\bar{u}^k)^T H(\bar{u}^k) \nabla l_{\text{avg}}(\bar{u}^k)}, \tag{11}$$

where $\bar{u}_i^k = 0$, if $u_i = 0$ or $|g_i| < \lambda$ and $\bar{u}^k = u^k$, otherwise. From (11) and the strict positiveness of θ'', we can see that the denominator is strictly positive as long as the gradient is nonzero. Computationally a very useful trick is not to compute the Hessian matrix directly, since we only use the vector-matrix product between the gradient vector $l_{\text{avg}}(\bar{u}^k)$ and the Hessian matrix $H(\bar{u}^k)$.

Based on the heuristic step length α_0, we can obtain the search direction p^k, which is a combination of the gradient descent step and the shrinkage step:

$$\begin{aligned} u^{k-} &= u^k - \alpha_0 \nabla l_{\text{avg}}(u^k), \\ u^{k+} &= s_{1:n}(u^{k-}, \lambda\alpha_0), \\ p^k &= u^{k+} - u^k. \end{aligned} \tag{12}$$

It is easy to verify that $s_\nu(y)$ is the solution to the non-smooth unconstrained minimization problem $\min \frac{1}{2}\|x-y\|_2^2 + \lambda\|x\|_1$. This minimization problem is equivalent to the following smooth constrained optimization problem,

$$\min \frac{1}{2}\|x-y\|_2^2 + \nu z, \text{ subject to } (x,z) \in \Omega := \{(x,z) \mid \|x\|_1 \le z\},$$

whose optimality condition is

$$(s(x,\nu)-x)^T(y-s(x,\nu)+\nu(z-\|s(x,\nu)\|_1) \ge 0,$$

for all $x \in \mathbb{R}^n$, $(y,z) \in \Omega$ and $\nu > 0$. Once we substitute $u-\tau g$ for x, u for y, $\|u_{1:n}\|_1$ for z and set $\nu = \lambda\tau$, the optimality condition becomes

$$(s_{1:n}(u-\tau g, \lambda\tau) - (u-\tau g))^T (u - s_{1:n}(u-\tau g, \lambda\tau)) + \lambda\tau(\|u_{1:n}\|_1 - \|s_{1:n}(u-\tau g, \lambda\tau)\|_1) \ge 0.$$

Using the fact $u^+ = s_{1:n}(u-\tau g, \lambda\tau)$, $p = u^+ - u$, we get

$$g^T p + \lambda(\|u^+_{1:n}\|_1 - \|u_{1:n}\|_1) \le -p^T p/\tau,$$

which means

$$\nabla l_{\text{avg}}(u^k)^T p^k + \lambda\|u^{k+}_{1:n}\|_1 - \lambda\|u^k_{1:n}\|_1 \le 0.$$

We then geometrically backtrack the step lengths, letting $\alpha_j = \alpha_0,\ \mu\alpha_0,\ \mu^2\alpha_0, \ldots$, until the following Armijo-like condition is satisfied:

$$\phi(u^k + \alpha_j p^k) \le C_k + \alpha_j \Delta_k.$$

Notice that the Armijo-like condition for line search stipulates that the step length α_j in the search direction p^k should produce a sufficient decrease of the objective function $\phi(u)$. C_k is a reference value with respect to the previous objective values, while the decrease in the objective function is described as

$$\Delta_k := \nabla l_{\text{avg}}(u^k)^T p^k + \lambda\|u^{k+}_{1:n}\|_1 - \lambda\|u^k_{1:n}\|_1 \le 0.$$

There are two types of Armijo-like conditions depending on the choice of C_k. One can choose $C_k = \phi(u^k)$, which makes the line search monotone. One can also derive a non-monotone line search, where C_k is a convex combination of the previous value C_{k-1} and the function value $\phi(u^k)$. We refer interested readers to Wen et al. (2009) for more details.

Figure 2 illustrates the computational speedup using the line search. The top panel shows the evolution of the objective function as a function of iterations. Tested on the benchmark data from the UCI repository, we see that our algorithm results in a speedup of 40 (6000 iterations without line search vs. 150 iterations with line search). The bottom panel shows the step length used in the algorithm. In the absence of the line search, we require that the step length satisfy $\tau < 2/\hat{\lambda}_{\max}$. For the Armijo-like line search, we illustrate both the heuristic step length α_0 (solid black curve) and the actual step length after backtracking (dashed red curve). Red asterisk labels the transition points on the continuation path, a concept we will discuss in the next section. Note that the step lengths can be on the order of 100 times larger for line search vs. no line search.

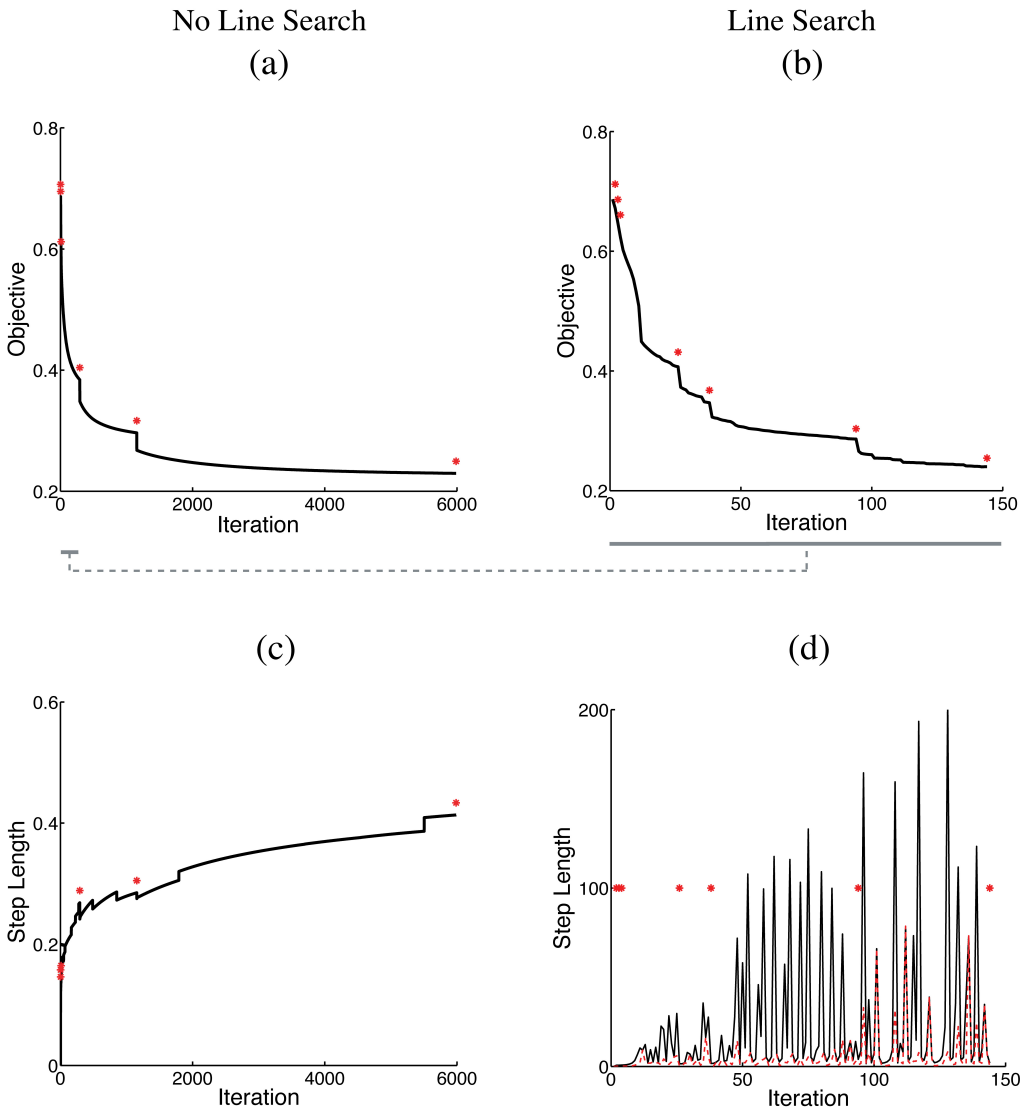


Figure 2: Illustration of the Armijo-like line search, comparing the iterative shrinkage algorithm with (right column) and without (left column) line search. (a) The objective function of the iterative shrinkage algorithm without line search, attaining convergence after 6000 iterations. (b) The objective of the iterative shrinkage algorithm with line search, converging at around 150 iterations. The gray bars under the "iteration" axes highlight the difference between the number of iterations—the gray bar in (a) represents the same number of iterations as the gray bar in (b). (c) The step length without line search is bounded by $2/\hat{\lambda}_{\max}$ to ensure convergence. (d) The step length used in the Armijo-like line search, (solid black curve) heuristic step length α_0 (Eqn. 11), (dashed red curve) actual time step after backtracking. The transition point on the continuation path is indicated in (red asterisk). Data used in this numerical experiment are the ionosphere data from the UCI machine learning data repository (http://archive.ics.uci.edu/ml/datasets/Ionosphere). Parameters used are $utol = 0.001$, $gtol = 0.01$, $\lambda_0 = 0.1$, $\lambda = 0.001$.

2.6 Continuation Path

A continuation strategy is adopted in our algorithm, by designing a regularization path similar to that is used in Hale et al. (2008),

$$\lambda_0 > \lambda_1 > ... > \lambda_{L-1} = \bar{\lambda}.$$

This idea is closely related to the homotopy algorithm in statistics, and has been successfully applied to the ℓ_1-regularized quadratic case, where the fidelity term is $f(x) = \|Ax-b\|_2^2$. The rationale of using such a continuation strategy is due to a fast rate of convergence for large λ. Therefore, by taking advantage of different convergence rate for a family of regularization parameter λ, if stopped appropriately, we can speed up the convergence rate of the full path. An intriguing discussion regarding the convergence rate of fixed-point algorithm with λ and ω, the spectral properties of Hessian, was presented in Hale et al. (2008). In the case of the logistic regression, we have decided to use the geometric progression for the continuation path. We define

$$\lambda_i = \lambda_0 \beta^{i-1}, \text{for } i = 0, ..., L-1,$$

where λ_0 can be calculated based on the ultimate $\bar{\lambda}$ we are interested in and the continuation path length L, that is, $\lambda_0 = \bar{\lambda}/\beta^{L-1}$.

As mentioned earlier, the goal of a continuation strategy is to construct a path with different rate of convergence, with which we can speed up the whole algorithm. The solution obtained from a previous subpath associated with λ_{i-1} is used as the initial condition for the next subpath for λ_i. Note that we design the path length L and the geometric progression rate β in such a way that the initial regularization λ_0 is fairly large, leading to a sparse solution for the initial path. Therefore, the initial condition for the whole path, considering the sparsity in solution, is a zero vector.

Another design issue regarding such a continuation strategy is we stop each subpath according to some criteria, in an endeavor to approximate the solution in the next λ as fast as possible. This means that a strong convergence is not required in subpath's except for the final one, and we can vary the stopping criteria to "tighten" such a convergence as we proceed. The following two stopping criteria are used:

$$\frac{\|u^{k+1} - u^k\|}{\max(\|u^k\|, 1)} < utol_i,$$
$$\frac{\|\nabla l_{\text{avg}}(u^k)\|_\infty}{\lambda_i} - 1 < gtol.$$

The first stopping criterion requires that relative change in u be small, while the second one is related to the optimality condition, defined in Eqn. (13). Theoretically, we would like to vary $utol_i$ to attain a seamless Q-linear convergence path. Such a choice seems to be problem dependent, and probably even data dependent in practice. It remains an important, yet difficult research topic to study the properties of different continuation strategies. We have chosen to use a geometric progression for the tolerance value, $utol_i = utol_0 * \gamma^{i-1}$, with $utol_0 = utol/\gamma^{L-1}$. In our numerical simulation, we use $utol = 10^{-4}$ and $gtol = 0.2$.

Figure 3 shows the continuation path using fixed $utol$ and a varying $utol$ following geometric progression. When we use a fixed $utol$ to ensure strong convergence each for λ along the path, the solver spends a lot of time evolving slowly. One can see in (a) that the objective function shows a

fairly flat reduction at earlier stages of the path. Clearly by relaxing the convergence at earlier stages of the path, we can accelerate the computation, shown in (b). The choice of *utol* and *gtol* seems to be data dependent in our experience, and the result we show in (b) might be suboptimal. Further optimization of the continuation path can potentially accelerate the computation even more, which remains an open question for future research.

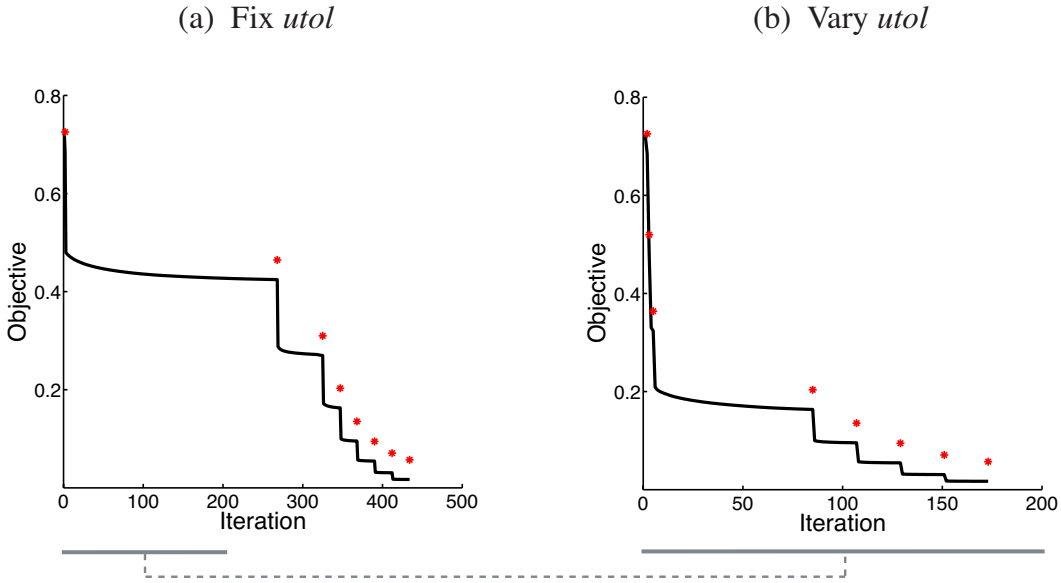


Figure 3: Illustration of the continuation strategy (a) using a fixed $utol = 0.0001$ is used for the stopping criterion, (b) using a varying *utol* according to geometric progression. Note that a stronger convergence is not necessary in earlier stages on the continuation path. By using a varying *utol*, especially tightening *utol* as we move along the path, we can accelerate the fixed point continuation algorithm. Shown is the objective value (black curve) as a function of iteration, where the transition point on the regularization path is labeled in (red asterisk). Data used in this experiment has 10000 dimension and 100 samples. A continuation path of length 8, starting from 0.128 and ending at 0.001.

3. Hybrid Iterative Shrinkage (HIS) Algorithm

In this section we describe a hybrid approach called HIS, which uses the iterative shrinkage algorithm described previously to enforce sparsity and identify the support in the data, followed by a subspace optimization via an interior point method.

3.1 Why A Hybrid Approach?

The hybrid approach is based on an interesting observation for the iterative shrinkage algorithm, regarding some finite convergence properties. The optimality condition for $\min f(x) + \lambda\|x\|_1$ is the following

$$g(x) + \lambda \mathrm{SGN}(x) \in \mathbf{0}, \tag{13}$$

which requires that $|g_i| \leq \lambda$, for $i = 1, ..., n$. We define two index sets

$$L := \{i : |g_i^*| < \lambda\} \text{ and } E := \{i : |g_i^*| = \lambda\},$$

where $g^* = g(u^*)$ is constant for all $u^* \in X^*$ and $|g_i^*| \leq \lambda$ for all i. Hence, $L \cap E = \emptyset$ and $L \cup E = \{1, \ldots, n\}$. The following holds true for all but a finite number of k:

$$\begin{aligned} u_i^k = u_i^* = 0, &\qquad \forall i \in L, \\ \operatorname{sgn}(h_i(u^k)) = \operatorname{sgn}(h_i(u^*)) = -\frac{1}{\lambda} g_i^*, &\qquad \forall i \in E. \end{aligned}$$

Assume that the underlying problem is nondegenerate, then L and E equal the sets of zero and nonzero components in x^*. According to the above finite convergence result, the iterative shrinkage algorithm obtains L and E, and thus the optimal support and signs of the optimal nonzero components, in a finite number of steps.

Corollary 2 *Under Assumption 1, after a finite number of iterations, the fixed-point iteration* (7) *reduces to gradient projection iterations for minimizing* $\phi(u_E)$ *over a constraint set* O_E*, where*

$$\phi(u_E) := -(g_E^*)^\top u_E + f((u_E; \mathbf{0})), \text{ and}$$

$$O_E = \{u_E \in \mathbb{R}^{|E|} : -\operatorname{sgn}(g_E^*) \odot u_E \geq 0\}.$$

Specifically, we have $u^{k+1} = (u_E^{k+1}; \mathbf{0})$ *in which*

$$u_E^{k+1} := P_{O_E}\left(u_E^k - \tau \nabla \phi(u_E^k)\right),$$

where P_{O_E} *is the orthogonal projection onto* O_E*, and* $\nabla \phi(u_E) = -g_E^* + g_E((u_E; \mathbf{0}))$.

This corollary, see Corollary 4.6 in Hale et al. (2008), can be directly applied to sparse logistic regression. The fixed point continuation reduces to the gradient projection after a finite number of iterations. The proof of this corollary is in general true for the $u_{1:n}$, that is, the w component in our problem.

Corollary 2 implies an important fact: there are two phases in the fixed point continuation algorithm. In the first phase, the number of nonzero elements in the x evolve rapidly, until after a finite number of iterations, when the support (non-zero elements in a vector) is found. Precisely, it means that for all $k > K$, the nonzero entries in u^k include all true nonzero entries in u^* with the matched signs. However, unless k is large, u^k typically also has extra nonzeros. At this point, the fixed point continuation reduces to the gradient projection, starting the second phase of the algorithm. In the second phase, the zero elements in the vector stay unaltered, while the magnitude of the nonzero elements (support) keeps evolving.

The above observation is a general statement for any f that is convex. Recall the quadratic case, where $f = \|y - Ax\|_2^2$, the second phase is very fast in terms of convergence rate. This is due to the quadratic function, and in an application to compressive sensing, the fixed point continuation algorithm alone has resulted in super-fast performance for large-scale problems (Hale et al., 2008). In the case of sparse logistic regression, we have a non-strictly convex f, the average logistic regression. This results in a fairly slow convergence rate when the algorithm reaches the second phase. In view of the continuation strategy we have, this greatly affects the speed of the last subpath, with the

regularization parameter $\bar{\lambda}$ of interest. In some sense, we have designed a continuation path that is super-fast until it reaches the second phase of the final subpath. This is not surprising given that the fixed point continuation algorithm is based on gradient descent and shrinkage operator. We envision that by switching to a Newton's method, we can accelerate the second phase.

Based on this intuition, we are now in a position to describe a hybrid algorithm: a fixed point continuation plus an interior point truncated Newton method. For the latter part we resort to the customized interior point in Koh et al. (2007). We modified the source code of the *l1logreg* software (written in C), and built an interface to our MATLAB code. This hybrid approach, based on our observation of the two phases, enables us to attain a good balance of speed and accuracy.

3.2 Interior Point Phase

The second phase of our HIS algorithm used an interior point method developed by Koh et al. (2007). We directly used a well-developed software package *l1logreg*[1] and modified the source code to build an interface to MATLAB. We review some key points for the interior point method here.

In Koh et al. (2007), the authors overcome the difficulty of non-differentiability of the objective function by transforming the original problem into an equivalent one with linear inequality constraints,

$$\begin{array}{ll} \min & \frac{1}{m}\sum_{i=1}^{m} l_{\text{avg}}(w^T a_i + v b_i) + \lambda \mathbf{1}^T u \\ \text{s.t.} & -u_i \leq w_i \leq u_i, \;\; i = 1, ..., n. \end{array}$$

A logarithmic barrier function, smooth and convex, is further constructed for the bound constraints,

$$\rho(w,u) = -\sum_{i=1}^{n} \log(u_i + w_i) - \sum_{i=1}^{n} \log(u_i - w_i),$$

defined on the domain $\{(w,u) \in \mathbb{R}^n \times \mathbb{R}^n | |w_i| < u_i, i = 1, ..., n\}$. The following optimization problem can be obtained by augmenting the logarithmic barrier,

$$\psi_t(v,w,u) = t l_{\text{avg}}(v,w) + t\lambda \mathbf{1}^T u + \rho(w,u),$$

where $t > 0$. The resulting objective function is smooth, strictly convex and bounded below, and therefore has a unique minimizer $(v^*(t), w^*(t), u^*(t))$. This defines a curve in $\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n$, parameterized by t, called the central path. The optimal solution is also shown to be dual feasible. In addition, $(v^*(t), w^*(t))$ is $2n/t$-suboptimal.

As a primal interior-point method, the authors computed a sequence of points on the central path, for an increasing sequence of values of t, and minimized $\psi_t(v,w,u)$ for each t using a truncated Newton's method. The interior point method was customized by the authors in several ways: 1) the dual feasible point and the associated duality gap was computed in a cheap fashion, 2) the central path parameter t was updated to achieve a robust convergence when combined with the preconditioned conjugate gradient (PCG) algorithm, 3) an option for solving the Newton's system was given for problems of different scales, where small and medium dense problems were solved by direct methods (Cholesky factorization), while large problems were solved using iterative methods (conjugate gradients). Interested readers are referred to Koh et al. (2007) for more details.

1. Software can be downloaded at `http://www.stanford.edu/~boyd/papers/l1_logistic_reg.html`.

3.3 The Hybrid Algorithm

The hybrid algorithm leverages the computational strengths of both the iterative shrinkage solver and the interior point solver.

Algorithm 3 Hybrid Iterative Shrinkage (**HIS**) Algorithm

Require: $A = [c_1^\top; c_2^\top; \cdots; c_m^\top] \in \mathbb{R}^{m \times n+1}$, $u = (w; v) \in \mathbb{R}^{n+1}$, $f(u) = m^{-1}\phi(Au)$
  task: $\min_u l_{\text{avg}}(u) + \lambda\|u\|_1$
  Initialize u^0
  PHASE 1 : ITERATIVE SHRINKAGE
  Select λ_0 and $utol_0$
  while "not converge" **do**
    if "the last continuation path", $i == (L-1)$ and "transition condition" **then**
      "transit into PHASE 2"
    else
      Update $\lambda_i = \lambda_{i-1}\beta$, $utol_i = utol_{i-1}\gamma$
      Compute heuristic step length α_0
      Gradient descent step: $u^{k-} = u^k - \alpha_0 \nabla l_{\text{avg}}(u^k)$
      Shrinkage step: $u^{k+} = s_{1:n}(u^{k-}, \lambda\alpha_0)$
      Obtain line search direction: $p^k = u^{k+} - u^k$
      while "j $<$ max line search attempts" **do**
        if Armijo-like condition is met **then**
          Accept line search step, update $u^{k+1} = u^k + \alpha_j p^k$
        else
          Keep backtracking $\alpha_j = \mu\alpha_{j-1}$
        end if
        $j = j + 1$
      end while
    end if
  end while
  PHASE 2 : INTERIOR POINT
  Initialize $\tilde{w} = w_{nonzero}$ get subproblem $\min \psi_t(v, \tilde{w}, u)$
  while "not converged" $\eta > \varepsilon$ **do**
    Solve the Newton system : $\nabla^2\psi_t^k(v, \tilde{w}, u)[\Delta v, \Delta\tilde{w}, \Delta u] = -\nabla\psi_t^k(v, \tilde{w}, u)$
    Backtracking line search : find the smallest integer $j \geq 0$ that satisfies
    $\psi_t^k(v + \alpha_j\Delta v, \tilde{w} + \alpha_j\Delta\tilde{w}, u + \alpha_j\Delta u) \leq \psi_t^k(v, \tilde{w}, u) + c\alpha_j\nabla\psi_t^k(v, \tilde{w}, u)^T[\Delta v, \Delta\tilde{w}, \Delta u]$
    Update $\psi_t^{k+1}(v, \tilde{w}, u) = \psi_t^k(v, \tilde{w}, u) + \alpha_j(\Delta v, \Delta\tilde{w}, \Delta u)$
    Check dual feasibility
    Evaluate duality gap η
    $k = k + 1$
  end while

In the first phase, we use the iterative shrinkage solver, due to its computational efficiency and memory friendliness. It is especially beneficial to have a memory friendly solver for the initial phase when one is dealing with large-scale data sets. Recall that we use a continuation strategy for the iterative shrinkage phase, where a sequence of λ's is used along a regularization path. In the

last subpath where λ is the desired one, we transit to the interior point when the true support of the vector is found. The corollary in Section 3.1 states that iterative shrinkage recovers the true support in a finite number of steps. In addition, iterative shrinkage obtains all true nonzero components long before the true support is obtained. Therefore, as long as the iterative shrinkage seems to stagnate, which can be observed when the objective function evolves very slowly, it is highly likely that all true nonzero components are obtained. This indicates that the algorithm is ready for switching to the interior point.

In practice, we require the following transition condition,

$$\frac{\|u^{k+1} - u^k\|}{\max(\|u^k\|, 1)} < utol_t,$$

and extract the nonzero components in w as the input to the interior point solver. By doing so, we reduce the problem to a subproblem where the dimension is much smaller, and solve the subproblem using the interior point method.

The resulting hybrid algorithm achieves high computational speed while attaining the same numerical accuracy as the interior point method, as demonstrated with empirical results in the next section.

4. Numerical Results

In this section we present numerical results, on a variety of data sets, to demonstrate the benefits of our hybrid framework in terms of computational efficiency and accuracy.

4.1 Benchmark

We carried out a numerical comparison of the HIS algorithm with several existing algorithms in literature for ℓ_1-regularized logistic regression. Inspired by a comparison study on this topic by Schmidt et al. (2007),[2] we compared our algorithm with 10 algorithms, including a generalized version of Gauss-Seidel, Shooting, Grafting, Sub-Gradient, epsL1, Log-Barrier, Log-Norm, SmoothL1, EM, ProjectionL1 and Interior-Point method. In the numerical study, we replaced the interior point solver by the one written by Koh et al. (2007). Benchmark data were taken from the publicly available UCI machine learning repository.[3] We used 10 data sets of small to median size (internetad1, arrhythmia, glass, horsecolic, indiandiabetes, internetad2, ionosphere, madelon, pageblock, spambase, spectheart, wine).

All of the methods were run until the same convergence criteria was met, where appropriate, for instance the step length, change in function value, negative directional derivative, optimality condition, convergence tolerance is less than 10^{-6}. We treated each algorithm solver as a black box and evaluated both the computation time and the sparsity (measured by cardinality of solution). We set an upper limit of 250 iterations, meaning we stop the solver when the number of iteration exceeds 250. Since different algorithm has different speed for each iterate (usually a Newton step is more expensive than a gradient descent step), we think the computation time is a more appropriate evaluation criterion than number of iterations. The ability of the algorithm to find a sparse solution, measured by the cardinality, was also evaluated in this process.

2. Source code is available at `http://www.cs.wisc.edu/~gfung/GeneralL1`.
3. UCI machine learning repository is at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Figure 4 shows the benchmark result using data from the UCI machine learning repository. All numerical results shown are averaged over a regularization path. The parameters for the regularization path are calculated according to each data set, where the maximal regularization parameter is calculated as follows:

$$\lambda_{max} = \frac{1}{m}\left\| \frac{m_-}{m} \sum_{b_i=+1} a_i + \frac{m_+}{m} \sum_{b_i=-1} a_i \right\|_\infty, \tag{14}$$

where m_- is the number of training samples with label -1 and m_+ is the number of training samples with label $+1$ (Koh et al., 2007). λ_{max} is an upper bound for the useful range of regularization parameter. When $\lambda \geq \lambda_{max}$, the cardinality of the solution will be zero. In this case, we test a regularization path of length 10, that is, $\lambda_{max}, 0.9\lambda_{max}, 0.8\lambda_{max}$ ... $0.1\lambda_{max}$. Among all the numerical solvers, our HIS algorithm is the most efficient. HIS achieves comparable cardinality in the solution, compared to the interior point solver.

We also evaluated the accuracy of the solution by looking at the classification performance using Kfold cross-validation. Table 1 summaries the accuracy of the solution using the HIS algorithm, compared to the interior point (IP) algorithm. Clearly, HIS algorithm achieves comparable accuracy compared to IP, an algorithm that is recognized for its high accuracy.

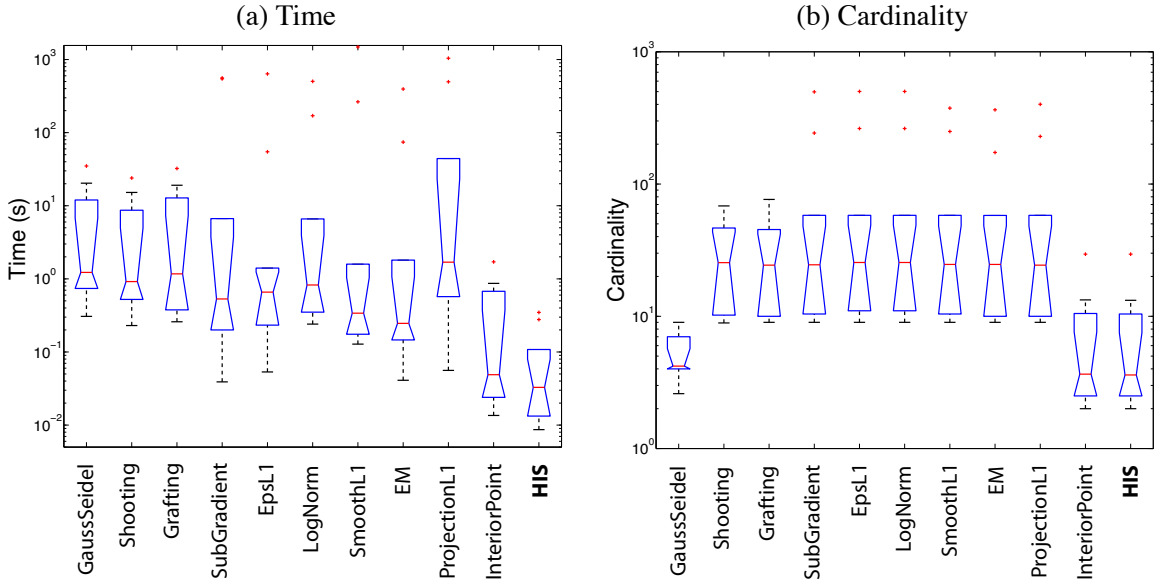


Figure 4: Comparison of our hybrid iterative shrinkage (HIS) method with several other existing methods in literature. Benchmark data were taken from the UCI machine learning repository, including 10 publicly available data sets. (a) Distribution of computation time across 10 data sets, (b) Distribution of cardinality for the solution across 10 data sets, averaged over a regularization path.

4.2 Scaling Result

Numerical experiments were carried out to study how our algorithm scales with the problem size. For the sake of generality, we used simulated data whose dimension ranges from 64 to 131072. The

Accuracy Comparison
($Az \in [0.5, 1.0]$)

dataname	accuracy(HIS)	accuarcy(IP)
arrhythmia	0.7363	0.7363
glass	0.6102	0.6102
horsecolic	0.5252	0.5252
ionosphere	0.5756	0.5756
madelon	0.6254	0.6254
spectheart	0.5350	0.5350
wine	0.6102	0.6102
internetad	0.8486	0.8486

Table 1: Comparison of solution accuracy for our hybrid iterative shrinkage (HIS) algorithm and the interior point (IP) algorithm. Accuracy of the solution was measured by Az value, resulted from Kfold cross-validation, where Kfold is 10. A regularization path of varying λ were computed to determine the maximum generalized Az value. The data sets were taken from the UCI machine learning repository.

data is drawn from a Normal distribution, where the mean of the distribution is shifted by a small amount for each class (0.1 for samples with label 1, and -0.1 for samples with label -1). The number of samples is the same for both classes and chosen to be smaller than the dimension of the data. Experiments for each dimension were carried out on 100 different sets of random data. We compared the mean and variances of the computation time, and compared our HIS algorithm to the IP algorithm.

Table 2 summarizes the computational speed for the HIS algorithm and the IP algorithm. It is noteworthy that the HIS algorithm improves the efficiency of computation, while maintaining comparable accuracy to the IP algorithm. Figure 5 plots the computation result as a function of dimension for better illustration. In (a) one can clearly see the speedup we gain from the HIS algorithm (red), compared to the IP algorithm (blue). We also show the solution quality in (b), where the weights we get from both solvers, is comparable.

4.3 Regularization Parameter

In general, the regularization parameter λ affects the number of iterations to converge for any solver. As λ becomes smaller, the cardinality of the solution increases, and the computation time needed for convergence also increases. Therefore when one seeks a solution with less sparsity (small λ), it is more computationally expensive.

In practice, when one carries out classification on a set of data, the optimal regularization parameter is often unknown. Speaking of optimality, we refer to a regularization parameter that results in the best classification result evaluated using Kfold cross-validation. One would run the algorithm along a regularization path, $\lambda_{max}, ..., \lambda_{min}$, where λ_{max} is computed by Eqn. (14) and where λ_{min} is supplied by the user.

Figure 6 shows the evolution of solution along the regularization path, using a small data set (ionosphere) from the UCI machine learning repository. This explores sparsity of different degrees

Speed Comparison
(in *second*)

dimension	mean(HIS)	std(HIS)	mean(IP)	std(IP)
64	0.0026	0.00069	0.0043	0.00057
128	0.0025	0.00058	0.0049	0.00037
256	0.0026	0.00075	0.0078	0.00052
512	0.0024	0.00059	0.018	0.0017
1024	0.0023	0.00056	0.029	0.0023
2048	0.0026	0.00064	0.054	0.0026
4096	0.0028	0.00057	0.098	0.0050
8192	0.0030	0.00059	0.19	0.0076
16384	0.0033	0.00055	0.40	0.018
32768	0.0038	0.00055	0.89	0.037
65536	0.0049	0.00054	2.01	0.096
131072	0.0077	0.00056	4.49	0.24

Table 2: Speed comparison of the HIS algorithm with the IP algorithm, based on simulated random benchmark data. Shown here is the computation speed as a function of dimension. Data used here are generated by sampling from two Gaussian distributions. Note that in the simulation, the continuation path used in the iterative shrinkage may or may not be optimal, which means that the speed profile for the HIS algorithm can be essentially accelerated even more.

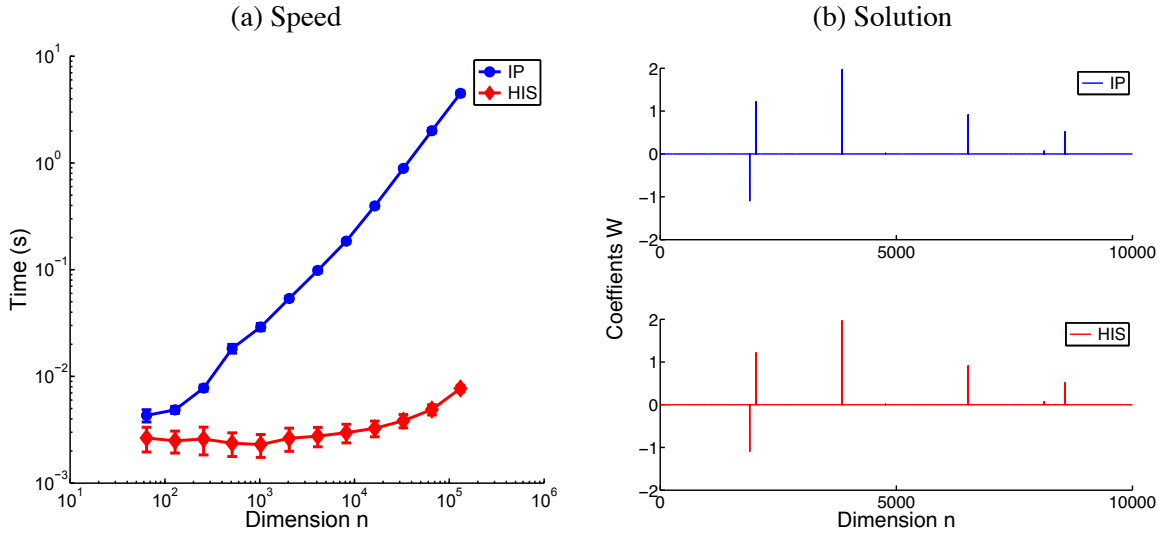


Figure 5: Comparison for the random benchmark data, between the HIS algorithm and the IP algorithm. (a) Speed profile for these two approaches: (blue curve) shows the speed profile for the IP algorithm, and (red curve) shows the speed profile for the HIS algorithm as a function of the data dimension. (b) An example of the solutions using the IP algorithm (blue) and the HIS algorithm (red).

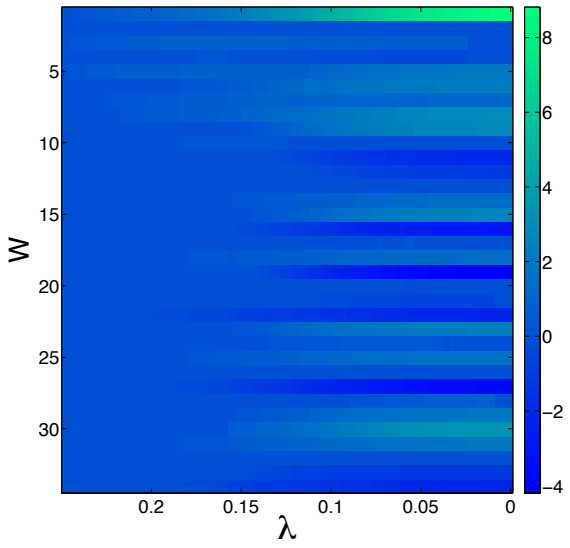


Figure 6: Solution w evolves along a regularization path, following a geometric progression from 10^{-1} to 10^{-4}. Data is ionosphere from UCI machine learning repository. As the λ becomes smaller, the cardinality of the solution goes up.

in the solution, and one can determine the optimal sparsity for the data. This is an attractive property of this model, where one can search in the feature space the most informative features about discrimination.

We illustrate the effect of the regularization parameter using real data of large scale. The data concerns a two alternative force choice task for face versus car discrimination. We used a spiking neuron model of primary visual cortex to map the input into cortical space, and decoded the resulting spike trains using sparse logistic regression (Shi et al., 2009). The data has 40960 dimensions and 360 samples for each of the two classes. Kfold cross-validation was used to evaluate the classification performance, where the number of Kfolds is 10 in our simulation.

The speedup of the HIS algorithm compared to the IP algorithm is shown in Figure 7(a), where blue indicates the computation time of the IP algorithm, and red shows the HIS algorithm. The HIS algorithm results in a significant speedup over the IP algorithm, without loss of accuracy. Note that there is an issue of model selection when we apply sparse logistic regression model to the data, in a sense there exists an optimal level of sparsity that achieves the best classification result. We ran the model with a sequence of regularization parameters, which resulted in classification result (evaluated by Az value from Kfold cross-validation). Figure 7(b) illustrates the classification result as a function of the cardinality of the solution. One can see the bell shape in the curve, which provides a route to select the optimal sparsity for the solution.

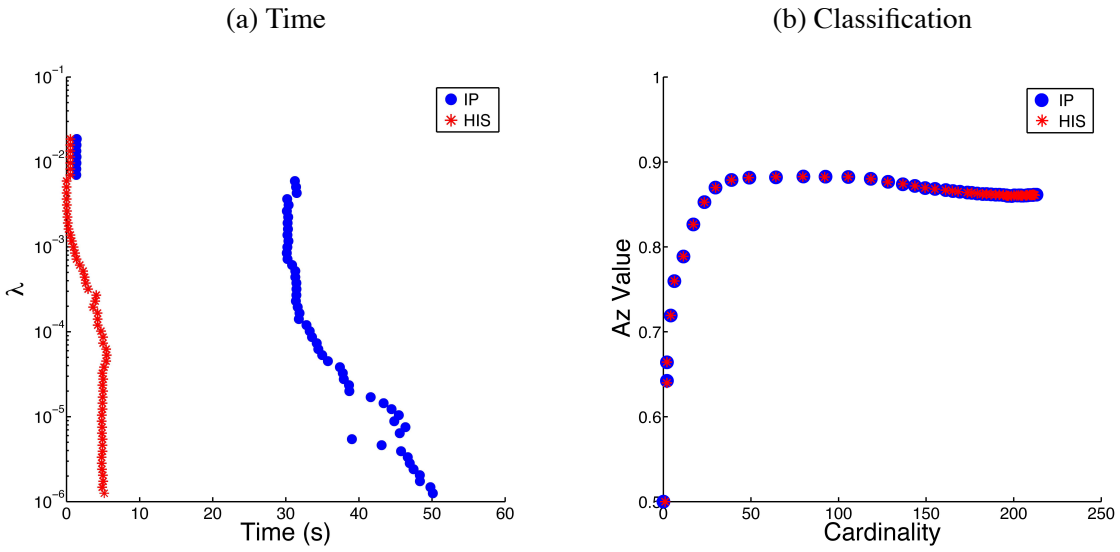


Figure 7: An example using real data of large scale, $n = 40960$, $m = 360$. (a) Computation time along such a regularization path, where the smaller λ requires more computation time. Note that the simulation is carried out for each λ separately. (b) Classification performance derived from ROC analysis based on Kfold cross-validation. Data used in this simulation are neural data for a visual discrimination task (Shi et al., 2009).

4.4 Data Sets with Large Dimensions and Samples

We applied the HIS algorithm to some examples of real-world data that have both large dimensions n and samples m. In this case, we considered text classification using the binary rcv1 data[4] (Lewis et al., 2004), and real-sim data.[5]

We ran the simulation on an Apple Mac Pro with two 3 GHz Quad-Core Intel processors, and 8 GB of memory. The timing of the simulation was calculated within the Matlab interface. All the operations were optimized for sparse matrix computation. Table 3 summarizes the numerical results. For both examples of text classification, we observed a speedup using the HIS algorithm while attaining the same numerical accuracy, compared with the IP algorithm. The regularization parameter does affect the computational efficiency, as we have observed in the previous section.

5. Conclusion

We have presented in this paper a computationally efficient algorithm for the ℓ_1-regularized logistic regression, also called the sparse logistic regression. The sparse logistic regression is a widely used model for binary classification in supervised learning. The ℓ_1 regularization leads to sparsity in the solution, making it a robust classifier for data whose dimensions are larger than the number of

4. Binary rcv1 data is available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#rcv1.binary`.
5. Real-sim data is available at `http://www.cs.umass.edu/~mccallum/code-data.html`.

Text Classification Application
(in *second*)

	rcv1 $n = 20242$ $m = 47236$ nonzero = 1498952		real-sim $n = 72309$ $m = 20958$ nonzero = 3709083	
λ	Time(HIS)	Time(IP)	Time(HIS)	Time(IP)
10^{-1}	0.11	1.93	0.62	6.61
10^{-2}	0.27	1.93	0.62	6.61
10^{-3}	2.08	8.20	5.50	18.45
10^{-4}	5.80	8.66	13.12	19.36

Table 3: Illustration of performance on text classification, where both the dimensions n and samples m are large-scale. We compare the computational efficiency of the HIS and IP algorithms. In both cases, the solution accuracy is the same.

samples. Sparsity also provides an attractive avenue for feature selection, useful for various data mining tasks.

Solving the large-scale sparse logistic regression usually requires expensive computational resources, depending on the specific solver, memory and/or CPU time. The interior point method is so far the most efficient solver in the literature, but requires expensive memory consumption. We have presented the HIS algorithm, which couples a fast shrinkage method and a slower but more accurate interior point method. The iterative shrinkage algorithm has global convergence with a Q-linear rate. Various techniques such as line search and continuation strategy are used to accelerate the computation. The shrinkage solver only involves the gradient descent and the shrinkage operator, both of which are first-order. Based solely on efficient memory operations such as matrix-vector multiplication, the shrinkage solver serves as the first phase for the algorithm. This reduces the problem to a subspace whose dimension is smaller than the original problem. The HIS algorithm then transits into the second phase, using a more accurate interior point solver. We numerically compare the HIS algorithm with other popular algorithms in the literature, using benchmark data from the UCI machine learning repository. We show that the HIS algorithm is the most computationally efficient, while maintaining high accuracy. The HIS algorithm also scales very well with dimension of the problem, making it attractive for solving large-scale problems.

There are several ways to extend the HIS algorithm. One is to extend it beyond binary classification, allowing for multiple classes (Krishnapuram and Hartemink, 2005). The other is to further improve the regularization path. When applying the HIS algorithm, one will usually explore a range of sparsity by constructing a regularization path (λ_{max}, λ_1, ..., λ_{min}). Usually the smaller the λ, the more expensive it is to employ the shrinkage algorithm. One can accelerate the computation using the Bregman regularization, inspired by Yin et al. (2008). The Bregman iterative algorithm essentially boosts the solution by solving a sequence of optimizations, resulting in a different regularization path. Bregman has also been shown to improve solution quality in the presence of noise (Burger et al., 2006; Shi and Osher, 2008; Osher et al., 2010). We will discuss such a regularization path in a future paper.

Acknowledgments

We thank Mads Dyrholm (Columbia University) for fruitful discussions. We appreciate the anonymous reviewers, who have helped improve the quality of our paper. Jianing Shi and Paul Sajda's work was supported by grants from NGA (HM1582-07-1-2002) and NIH (EY015520). Wotao Yin's work was supported by NSF CAREER Award DMS-0748839 and ONR Grant N00014-08-1-1101. Stanley Osher's work was supported by NSF grants DMS-0312222, and ACI-0321917 and NIH G54 RR021813.

References

C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.

S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

M. Burger, G. Gilboa, S. Osher, and J. Xu. Nonlinear inverse scale space methods. *Communications in Mathematical Sciences*, 4:179–212, 2006.

E.J. Candés and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inform. Theory*, 52(2):5406–5425, 2006.

E.J. Candés, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Scientific Computing*, 20:33–61, 1998.

J.F. Claerbout and F. Muir. Robust modeling with erratic data. *Geophysics*, 38(5):826–844, 1973.

A. d'Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. In *Advances in Neural Information Processing Systems*, pages 41–48. MIT Press, 2005.

D.L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006.

D.L. Donoho and M. Elad. Optimally sparse representations in general nonorthogonal dictionaries by ℓ_1 minimization. *Proc. Nat'l Academy of Science*, 100(5):2197–2202, 2003.

D.L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE. Trans. Inform. Theory*, 48(9):2845–2862, 2001.

D.L. Donoho and B.F. Logan. Signal recovery and the large sieve. *SIAM J. Appl. Math.*, 52(2): 577–591, 1992.

D.L. Donoho and P.B. Stark. Uncertainty principle and signal recovery. *SIAM J. Appl. Math.*, 49(3): 906–931, 1989.

D.L. Donoho, I. Johnstone, G. Kerkyacharian, and D. Picard. Wavelet shrinkage: Asymptopia? *J. Roy. Stat. Soc. B*, 57(2):301–337, 1995.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

S. Eyheramendy, A. Genkin, W. Ju, D. Lewis, and D. Madigan. Sparse bayesian classifiers for text categorization. Technical report, J. Intelligence Community Research and Development, 2003.

M. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25:1150–1159, 2003.

M. Figueiredo and A. Jain. Bayesian learning of sparse classifiers. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 35–41, 2001.

M. Figueiredo, R. Nowak, and S. Wright. Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. *IEEE J. Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, 1(4):586–598, 2007.

A. Genkin, D.D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

A.D. Gerson, L.C. Parra, and P. Sajda. Cortical origins of response time variability during rapid discrimination of visual objects. *Neuroimage*, 28(2):342–353, 2005.

A. Ghosh and S. Boyd. Growing well-connected graphs. In *45th IEEE Conference on Decision and Control*, pages 6605–6611, 2006.

J. Goodman. Exponential priors for maximum entropy models. In *Proceedings of the Annual Meetings of the Association for Computational Linguistics*, 2004.

E. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: methodology and convergence. *SIAM J. Optimization*, 19(3):1107–1130, 2008.

A. Hassibi, J. How, and S. Boyd. Low-authority controller design via convex optimization. *AIAA Journal of Guidance, Control and Dynamics*, 22(6):862–872, 1999.

K. Koh, S.-J. Kim, and S. Boyd. An interior-point method for large-scale ℓ_1-regularized logistic regression. *J. Machine Learning Research*, 8:1519–1555, 2007.

B. Krishnapuram and A. Hartemink. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(6):957–968, 2005.

B. Krishnapuram, L. Carin, and M. Figueiredo. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(6): 957–968, 2005.

S. Lee, H. Lee, P. Abbeel, and A. Ng. Efficient ℓ_1-regularized logistic regression. In *21th National Conference on Artificial Intelligence (AAAI)*, 2006.

D.D. Lewis, Y. Yang, T.G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *J. Machine Learning Research*, 5:361–397, 2004.

J.G. Liao and K.V. Chin. Logistic regression for disease classification using microarray data: model selection in a large p and small n case. *Bioinformatics*, 23(15):1945–51, 2007.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

M. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):376–394, 2007.

J. Lokhorst. The lasso and generalised linear models. Technical report, Honors Project, Department of Statistics, University of Adelaide, South Australia, Australia, 1999.

D. Madigan, A. Genkin, D. Lewis, and D Fradkin. Bayesian multinomial logistic regression for author identification. In *Maxent Conference*, pages 509–516, 2005.

B.K. Natarajan. Sparse approximate solutions to linear system. *SIAM J. Computing*, 24(2):227–234, 1995.

A. Ng. Feature selection, ℓ_1 vs ℓ_2 regularization, and rotational invariance. In *International Conference on Machine Learning (ICML)*, pages 78–85. ACM Press, New York, 2004.

A. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *International Conference on Machine Learning (ICML)*, pages 404–412, 1998.

S. Osher, Y. Mao, B. Dong, and W. Yin. Fast linearized bregman iteration for compressive sensing and sparse denoising. *Communications in Mathematical Sciences*, 8(1):93–111, 2010.

M.Y. Park and T. Hastie. ℓ_1 regularized path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69:659–677, 2007.

L.C. Parra, C. Spence, and P. Sajda. Higher-order statistical properties arising from the non-stationarity of natural signals. In *Advances in Neural Information Processing Systems*, volume 13, pages 786–792, 2001.

L.C. Parra, C.D. Spence, A.D. Gerson, and P. Sajda. Recipes for the linear analysis of EEG. *Neuroimage*, 28(2):326–341, 2005.

S. Perkins and J. Theiler. Online feature selection using grafting. In *Proceedings of the Twenty-First International Conference on Machine Learning (ICML)*, pages 592–599. ACM Press, 2003.

M.G. Philiastides and P. Sajda. Temporal characterization of the neural correlates of perceptual decision making in the human brain. *Cereb Cortex*, 16(4):509–518, 2006.

B. Polyak. *Introduction to Optimization*. Optimization Software, 1987.

V. Roth. The generalized lasso. *IEEE Tran. Neural Networks*, 15(1):16–28, 2004.

L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: a comparative study and two new approaches. In *European Conference on Machine Learning (ECML)*, pages 286–297, 2007.

J. Shi and S. Osher. A nonlinear inverse scale space method for a convex multiplicative noise model. *SIAM J. Imaging Sciences*, 1(3):294–321, 2008.

J. Shi, J. Wielaard, R.T. Smith, and P. Sajda. Perceptual decision making investigated via sparse decoding of a spiking neuron model of V1. In *4th International IEEE/EMBS Conference on Neural Engineering*, pages 558–561, 2009.

N.Z. Shor. *Minimization Methods for Non-differentiable functions*. Springer Series in Computational Mathematics. Springer, 1985.

H.L. Taylor, S.C. Banks, and J.F. McCoy. Deconvolution with the ℓ_1 norm. *Geophysics*, 44(1): 39–52, 1979.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. B*, 58(1):267–288, 1996.

Y. Tsuruoka, J. McNaught, J. Tsujii, and S. Ananiadou. Learning string similarity measures for gene/protein name dictionary look-up using logistic regression. *Bioinformatics*, 23(20):2768–74, 2007.

L. Vandenberghe, S. Boyd, and A. El Gamal. Optimal wire and transistor sizing for circuits with non-tree topology. In *IEEE/ACM International Conference on Computer Aided Design*, pages 252–259, 1997.

L. Vandenberghe, S. Boyd, and A. El Gamal. Optimizing dominant time constant in RC circuits. *IEEE Trans. Computer-Aided Design*, 17(2):110–125, 1998.

V. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, 1982.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1988.

Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation. Technical report, Rice University CAAM TR09-01, 2009.

W. Yin, S. Osher, J. Darbon, and D. Goldfarb. Bregman iterative algorithm for ℓ_1-minimization with applications to compressed sensing. *SIAM J. Imaging Science*, 1(1):143–168, 2008.

P. Zhao and B. Yu. On model selection consistency of lasso. *J. Machine Learning Research*, 7: 2541–2567, 2007.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems*, volume 16, pages 49–56. MIT Press, 2004.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principle component analysis. *J. Computational and Graphical Statistics*, 15(2):262–286, 2006.

Journal of Machine Learning Research 11 (2010) 743-746 Submitted 11/09; Published 2/10

PyBrain

Tom Schaul TOM@IDSIA.CH
Justin Bayer JUSTIN@IDSIA.CH
Daan Wierstra DAAN@IDSIA.CH
Yi Sun YI@IDSIA.CH
IDSIA, University of Lugano
Manno-Lugano, 6900, Switzerland

Martin Felder FELDER@IN.TUM.DE
Frank Sehnke SEHNKE@IN.TUM.DE
Thomas Rückstieß RUECKSTI@IN.TUM.DE
Jürgen Schmidhuber* JUERGEN@IDSIA.CH
Technische Universität München
Garching D-86748, Germany

Editor: Soeren Sonnenburg

Abstract

PyBrain is a versatile machine learning library for Python. Its goal is to provide flexible, easy-to-use yet still powerful algorithms for machine learning tasks, including a variety of predefined environments and benchmarks to test and compare algorithms. Implemented algorithms include Long Short-Term Memory (LSTM), policy gradient methods, (multidimensional) recurrent neural networks and deep belief networks.

Keywords: Python, neural networks, reinforcement learning, optimization

1. Introduction

PyBrain is a machine learning library written in Python designed to facilitate both the application of and research on premier learning algorithms such as LSTM (Hochreiter and Schmidhuber, 1997), deep belief networks, and policy gradient algorithms. Emphasizing both sequential and non-sequential data and tasks, PyBrain implements many recent learning algorithms and architectures ranging from areas such as supervised learning and reinforcement learning to direct search / optimization and evolutionary methods.

PyBrain is implemented in Python, with the scientific library SciPy being its only strict dependency. As is typical for programming in Python/SciPy, development time is greatly reduced as compared to languages such as Java/C++, at the cost of lower speed. PyBrain embodies a compositional setup, which means that it is designed to be able to connect various types of architectures and algorithms.

PyBrain goes beyond existing Python libraries in breadth in that it provides a toolbox for supervised, unsupervised and reinforcement learning as well as black-box and multi-objective optimization. In addition to standard algorithms (some of which, to the best of our knowledge, are not available as Python implementations elsewhere) for application-oriented users, it contains ref-

∗. Also at IDSIA, University of Lugano, Galleria 2, Manno-Lugano, 6900, Switzerland.

erence implementations of a number of algorithms at the bleeding edge of research. Furthermore, it sets itself apart by its flexibility for composing custom neural networks architectures, ranging from (multi-dimensional) recurrent networks to restricted Boltzmann machines or convolutional networks.

2. Library Overview

The library includes different types of training algorithms, trainable architectural components, specialized data sets and standardized benchmark tasks/environments. The available algorithms generally function both in sequential and non-sequential settings, and appropriate data handling tools have been developed for special applications, ranging from reinforcement learning to handwriting recognition applications. Implemented algorithms and methods come with unit tests in order to assure correctness and soundness.

In the following, we will provide a short overview of the different features of the library.

Supervised Learning Training algorithms include classical gradient-based methods and extensions both for non-sequential and sequential data. PyBrain also features Gaussian processes, the evolino algorithm and an SVM wrapper.

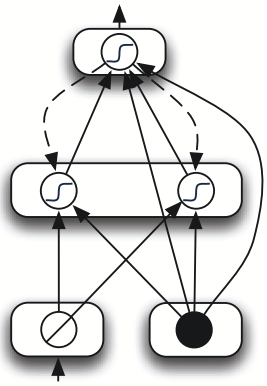
Black-Box Optimization / Evolutionary Methods Various black-box optimization algorithms have been implemented. In addition to traditional evolution strategies, covariance matrix adaptation, co-evolitionary and genetic algorithms (including NSGA-II for multi-objective optimization), we have included recent new algorithms (not available in other libraries) such as fitness expectation maximization, natural evolution strategies and policy gradients with parameter-based exploration.

Reinforcement Learning The reinforcement learning algorithms of PyBrain encompass basic methods such as Q-learning, SARSA and REINFORCE, but also natural actor-critic, neural-fitted Q-iteration, recurrent policy gradients, state-dependent exploration and reward-weighted regression.

Architectures Available architectures include standard feedforward neural networks, recurrent neural networks and LSTM, bi- and multidimensional recurrent neural networks and deep belief networks. The library puts emphasis on, but is not limited to, (recurrent) neural network structures arbitrarily structured as a directed acyclic graph of modules and connections.

Compositionality The basic structure of the library enables a compositional approach to machine learning. Different algorithms and architectures can be connected and composed, and then used and trained as desired. For example, arbitrarily structured recurrent neural network graphs can be trained using various different algorithms such as black-box search, policy gradients and supervised learning. For example, both an LSTM architecture and a deep belief network, though constituting wildly different architectures, can be trained using the same gradient implementation (e.g., RPROP).

Tasks and Benchmarks For black-box and multi-objective optimization, standard benchmarks are included in the library. For reinforcement learning settings, there are classical (PO)MDP mazes, (non-Markov) (double) pole balancing and various ODE environments based on physics simulations. In order to make data processing as easy as possible, PyBrain features functionality for constructing, serializing and deserializing data sets to files or over network connections.



```
# Load Data Set.
ds = SequentialDataSet.loadFromFile('parity.mat')

# Build a recurrent Network.
net = buildNetwork(1, 2, 1, bias=True,
                   hiddenclass=TanhLayer,
                   outclass=TanhLayer,
                   recurrent=True)
recCon = FullConnection(net['out'], net['hidden0'])
net.addRecurrentConnection(recCon)
net.sortModules()

# Create a trainer for backprop and train the net.
trainer = BackpropTrainer(net, ds, learningrate=0.05)
trainer.trainEpochs(1000)
```

Figure 1: Code example for solving the parity problem with a recurrent neural network.

Speed Optimization The development cycle of Python/SciPy is short, especially compared to languages such as Java/C/C++ and especially in scientific / machine learning settings. However, this comes at a cost-reduced speed. In order to partially mitigate this potential problem, we have implemented many parts of the library in C/C++ using SWIG as a bridge. Parallel implementations in both Python and C++ exist for crucial bottleneck elements of the library. The resulting speedup approaches an equivalent C++ implementation within an order of magnitude, and comes even closer for large architectures.

3. An Illustrative Example

In order to provide a useful example to a novice user, we construct a recurrent network that is able to solve the parity problem. The network is given a sequence of binary inputs and the corresponding objective is to determine whether the network has been provided an even or an odd number of 1s so far. The implemented topology and the corresponding code are shown in Figure 1.

We first construct a feed forward network consisting of a single linear input cell, a layer of two hidden cells and one output cell. The network also needs a bias, which is connected to the hidden and output layer. The layers are fully connected with one another.

The network is first created using a 'convenience' function. A recurrent connection from the output unit to the hidden layer is crucial to learn the problem and thus added afterwards.

It should be stressed that the clarity and shortness of this code example is representative for many algorithms and architectures used in actual problems. Constructing networks for classification, control or function approximation often requires even fewer lines of code.

4. Concluding Remarks

In PyBrain we emphasize simplicity, compositionality and the ability to combine various architectures and algorithm types. We believe this to be advantageous in the light of recent developments in sequence processing, deep belief networks, policy gradients and visual processing. Pybrain is easy to use, and is well-documented, both in code and in documents and tutorials explaining the use of the basic capabilities of the library. Among machine learning libraries written in Python,

PyBrain stands out for its combination of breadth, versatility and maturity of development. In order to further demonstrate its practical applicability to scientific research, we could emphasize that PyBrain, so far, has already been used in fourteen scientific peer-reviewed publications, for example, Sehnke et al. (2008), Rückstieß et al. (2008), Schaul and Schmidhuber (2009), Sun et al. (2009) and Wierstra et al. (2008).

5. Availability and Requirements

As of the time of writing, PyBrain has reached version 0.3 and both the entire source code and the documentation are available from the website, `www.pybrain.org`, under the BSD license. The available platforms are Mac OS X, Linux and Microsoft Windows. The only strict dependency is SciPy (`www.scipy.org`), highly recommended are matplotlib (required for most example scripts) and setuptools. The documentation includes a quickstart tutorial, installation instructions, tutorials on advanced topics, and an extensive API reference. Since PyBrain is under active development, we encourage researchers to contribute their work and provide guidelines for how they can do so.

Acknowledgments

This research was funded in part by SNF grants 200021-111968/1, 200021-113364/1 and 200020-116674/1, the Excellence Cluster Cognition For Technical Systems (CoTeSys) of the German Research Foundation (DFG) and the EU FP6 project #033287.

References

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997. ISSN 0899-7667.

Thomas Rückstieß, Martin Felder, and Jürgen Schmidhuber. State-dependent exploration for policy gradient methods. In *Proceedings of the European Conference on Machine Learning (ECML)*, 2008.

Tom Schaul and Jürgen Schmidhuber. Scalable neural networks for board games. In *International Conference on Artificial Neural Networks (ICANN)*, 2009.

Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Policy gradients with parameter-based exploration for control. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 2008.

Yi Sun, Daan Wierstra, Tom Schaul, and Jürgen Schmidhuber. Efficient natural evolution strategies. In *Genetic and Evolutionary Computation Conference (GECCO)*, 2009.

Daan Wierstra, Tom Schaul, Jan Peters, and Jürgen Schmidhuber. Fitness expectation maximization. In *Lecture Notes in Computer Science, Parallel Problem Solving from Nature - PPSN X*, pages 337–346. Springer-Verlag, 2008.

Journal of Machine Learning Research 11 (2010) 747-788 Submitted 12/08; Revised 9/09; Published 2/10

Maximum Relative Margin and Data-Dependent Regularization

Pannagadatta K. Shivaswamy PKS2103@CS.COLUMBIA.EDU
Tony Jebara JEBARA@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
New York NY 10027 USA

Editor: John Shawe-Taylor

Abstract

Leading classification methods such as support vector machines (SVMs) and their counterparts achieve strong generalization performance by maximizing the margin of separation between data classes. While the maximum margin approach has achieved promising performance, this article identifies its sensitivity to affine transformations of the data and to directions with large data spread. Maximum margin solutions may be misled by the spread of data and preferentially separate classes along large spread directions. This article corrects these weaknesses by measuring margin not in the absolute sense but rather only relative to the spread of data in any projection direction. Maximum relative margin corresponds to a data-dependent regularization on the classification function while maximum absolute margin corresponds to an ℓ_2 norm constraint on the classification function. Interestingly, the proposed improvements only require simple extensions to existing maximum margin formulations and preserve the computational efficiency of SVMs. Through the maximization of relative margin, surprising performance gains are achieved on real-world problems such as digit, text classification and on several other benchmark data sets. In addition, risk bounds are derived for the new formulation based on Rademacher averages.

Keywords: support vector machines, kernel methods, large margin, Rademacher complexity

1. Introduction

In classification problems, the aim is to learn a classifier that generalizes well on future data from a limited number of training examples. Support vector machines (SVMs) and maximum margin classifiers (Vapnik, 1995; Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004) have been a particularly successful approach both in theory and in practice. Given a labeled training set, these return a predictor that accurately labels previously unseen test examples. For simple binary classification in Euclidean spaces, this predictor is a function $f : \mathbb{R}^m \to \{\pm 1\}$ estimated from observed training data $(\mathbf{x}_i, y_i)_{i=1}^n$ consisting of inputs $\mathbf{x}_i \in \mathbb{R}^m$ and outputs $y_i \in \{\pm 1\}$. A linear function[1] $f(\mathbf{x}) := \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$ where $\mathbf{w} \in \mathbb{R}^m, b \in \mathbb{R}$ serves as the decision rule throughout this article. The parameters of the hyperplane $(\mathbf{w}, b)$ are estimated by maximizing the margin (e.g., the distance between the hyperplanes defined by $\mathbf{w}^\top \mathbf{x} + b = 1$ and $\mathbf{w}^\top \mathbf{x} + b = -1$) while minimizing a weighted upper bound on the misclassification rate on training data (via so-called slack variables). In practice, the margin is maximized by minimizing $\frac{1}{2}\mathbf{w}^\top \mathbf{w}$ plus an upper bound on the misclassification rate.

1. In this article the dot product $\mathbf{w}^\top \mathbf{x}$ is used with the understanding that it can be replaced with a generalized inner product or by using a kernel for generic objects.

While maximum margin classification works well in practice, its solution can easily be perturbed by an (invertible) affine or scaling transformation of the input space. For instance, by transforming all training and testing inputs by an invertible linear transformation, the SVM solution and its resulting classification performance can be significantly varied. This is worrisome since an adversary could directly exploit this shortcoming and transform the data to drive performance down; a synthetic example showing this effect will be presented in Section 5. Moreover, this phenomenon is not limited to an explicit adversarial setting; it can naturally occur in many real world classification problems, especially in high dimensions. This article will explore such shortcomings in maximum margin solutions (or equivalently, SVMs in the context of this article) which exclusively measure margin by the points near the classification boundary regardless of how spread the remaining data is away from the separating hyperplane. An alternative approach will be followed based on controlling the spread while maximizing the margin. This helps overcome this bias and produces a formulation that is affine invariant. The key is to recover a large margin solution while normalizing the margin by the spread of the data. Thus, margin is measured in a *relative* sense rather than in the absolute sense. In addition, theoretical results using Rademacher averages support this intuition. The resulting classifier will be referred to as the relative margin machine (RMM) and was first introduced by Shivaswamy and Jebara (2009a) with this longer article serving to provide more details, more thorough empirical evaluation and more theoretical support.

Traditionally, controlling spread has been an important theme in classification problems. For instance, classical linear discriminant analysis (LDA) (Duda et al., 2000) finds projections of the data so that the inter-class separation is large while within-class scatter is small. However, the spread (or scatter in this context) is estimated by LDA using only simple first and the second order statistics of the data. While this is appropriate if class-conditional densities are Gaussian, second-order statistics are inappropriate for many real-world data sets and thus, the classification performance of LDA is typically weaker than that of SVMs. The estimation of spread should not make second-order assumptions about the data and should be tied to the margin criterion (Vapnik, 1995). A similar line of reasoning has been proposed to perform feature selection. Weston et al. (2000) showed that second order tests and filtering methods on features perform poorly compared to wrapper methods on SVMs which more reliably remove features that have low discriminative value. In this prior work, a feature's contribution to margin is compared to its effect on the radius of the data by computing bounding hyper-spheres rather than simple second-order statistics. Unfortunately, there, only axis-aligned feature selection was considered. Similarly, ellipsoidal kernel machines (Shivaswamy and Jebara, 2007) were proposed to normalize data in feature space by estimating bounding hyper-ellipsoids while avoiding second-order assumptions. Similarly, the radius-margin bound has been used as a criterion to tune the hyper-parameters of the SVM (Keerthi, 2002). Another criterion based jointly on ideas from the SVM method as well as Linear Discriminant Analysis has been studied by Zhang et al. (2005). This technique involves first solving the SVM and then solving an LDA problem based on the support vectors that were obtained. While these previous methods showed performance improvements, they relied on multiple-step locally optimal algorithms for interleaving spread information with margin estimation.

To overcome the limitations of local non-convex optimization schemes, the formulations derived here will remain convex, will be efficiently solvable and will admit helpful generalization bounds. A similar method to the RMM was described by Haffner (2001), yet that approach started from a different overall motivation. In contrast, this article starts with a novel intuition, produces a novel algorithm and provides novel empirical and theoretical support. Another interesting contact point

is the second order perceptron framework (Cesa-Bianchi et al., 2005) which parallels some of the intuitions underlying the RMM. In an on-line setting, the second order perceptron maintains both a decision rule and a covariance matrix to whiten the data. The mistake bounds it inherits were shown to be better than those of the classical perceptron algorithm. Alternatively, one may consider distributions over classifier solutions which provide a different estimate than the maximum margin setting and have also shown empirical improvements over SVMs (Jaakkola et al., 1999; Herbrich et al., 2001). In recent papers, Dredze et al. (2008) and Crammer et al. (2009a) consider a distribution on the perceptron hyperplane. These distribution assumptions permit update rules that resemble whitening of the data, thus alleviating adversarial affine transformations and producing changes to the basic maximum margin formulation that are similar in spirit to those the RMM provides. In addition, recently, a new batch algorithm called the Gaussian margin machine (GMM) (Crammer et al., 2009b) has been proposed. The GMM maintains a Gaussian distribution over weight vectors for binary classification and seeks the least informative distribution that correctly classifies training data. While the GMM is well motivated from a PAC-Bayesian perspective, the optimization problem itself is expensive involving a log-determinant optimization.

Another alternative route for improving SVM performance includes the use of additional examples. For instance, unlabeled or test examples may be available in semi-supervised or transductive formulations of the SVM (Joachims, 1999; Belkin et al., 2005). Alternatively, additional data that does not belong to any of the classification classes of interest may be available as in the so-called Universum approach (Weston et al., 2006; Sinz et al., 2008). In principle, these methods also change the way margin is measured and the way regularization is applied to the learning problem. While additional data can be helpful in overcoming limitations for many classifiers, this article will be interested in only the simple binary classification setting. The argument is that, without any additional assumptions beyond the simple classification problem, maximizing margin in the absolute sense may be suboptimal and that maximizing relative margin is a promising alternative.

Further, large margin methods have been successfully applied to a variety of tasks such as parsing (Collins and Roark, 2004; Taskar et al., 2004), matrix factorization (Srebro et al., 2005), structured prediction (Tsochantaridis et al., 2005), etc.; in fact, the RMM approach could be readily adapted to such problems. For instance, RMM has been successfully extended to structured prediction problems (Shivaswamy and Jebara, 2009b).

The organization of this article is as follows. Motivation from various perspectives are given in Section 2. The relative margin machine formulation is detailed in Section 3 and several variants and implementations are proposed. Generalization bounds for the various function classes are studied in Section 4. Experimental results are provided in Section 5. Finally, conclusions are presented in Section 6. Some proofs and otherwise standard results are provided in the Appendix.

1.1 Notation

Throughout this article, boldface letters indicate vectors/matrices. For two vectors $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{u} \leq \mathbf{v}$ indicates that $u_i \leq v_i$ for all i from 1 to m. $\mathbf{1}$, $\mathbf{0}$ and $\mathbf{I}$ denote the vectors of all ones, all zeros and the identity matrix respectively; $\mathbf{0}$ also denotes a matrix of all zeros in some contexts. The dimensionality of vectors and matrices should be clear from the context.

2. Motivation

This section provides three different (an intuitive, a probabilistic and an affine transformation based) motivations for maximizing the margin relative to the data spread.

2.1 Intuitive Motivation with a Two Dimensional Example

Consider the simple two dimensional data set in Figure 1 where the goal is to separate the two classes of points: triangles and squares. The figure depicts three scaled versions of the two dimensional problem to illustrate potential problems with the large margin solution.

In the topmost plot in the left column of Figure 1, two possible linear decision boundaries separating the classes are shown. The red (or dark shade) solution is the SVM estimate while the green (or light shade) solution is the proposed maximum relative margin alternative. Clearly, the SVM solution achieves the largest margin possible while separating both classes, yet is this necessarily the best solution?

Next, consider the same set of points after a scaling transformation in the second and the third row of Figure 1. Note that all these three problems correspond to the same discrimination problem up to a scaling factor. With progressive scaling, the SVM increasingly deviates from the maximum relative margin solution (green), clearly indicating that the SVM decision boundary is sensitive to affine transformations of the data. Essentially, the SVM produces a family of different solutions as a result of the scaling. This sensitivity to scaling and affine transformations is worrisome. If the SVM solution and its generalization accuracy vary with scaling, an adversary may exploit such scaling to ensure that the SVM performs poorly. Meanwhile, an algorithm producing the maximum relative margin (green) decision boundary could remain resilient to adversarial scaling.

In the previous example, a direction with a small spread in the data produced a good and affine-invariant discriminator which maximized relative margin. Unlike the maximum margin solution, this solution accounts for the spread of the data in various directions. This permits it to recover a solution which has a large margin relative to the spread in that direction. Such a solution would otherwise be overlooked by a maximum margin criterion. A small margin in a correspondingly smaller spread of the data might be better than a large absolute margin with correspondingly larger data spread. This particular weakness in large margin estimation has only received limited attention in previous work.

It is helpful to consider the generative model for the above motivating example. Therein, each class was generated from a one dimensional line distribution with the two classes on two parallel lines. In this case, the maximum relative margin (green) decision boundary should obtain zero test error even if it is estimated from a finite number of examples. However, for finite training data, the SVM solution will make errors and will do so increasingly as the data is scaled further. While it is possible to anticipate these problems and choose kernels or nonlinear mappings to correct for them in advance, this is not necessarily practical. The right mapping or kernel is never provided in advance in realistic settings. Instead, one has to estimate kernels and nonlinear mappings, a difficult endeavor which can often exacerbate the learning problem. Similarly, simple data preprocessing (affine whitening to make the data set zero-mean and unit-covariance or scaling to place the data into a zero-one box) can also fail, possibly because of estimation problems in recovering the correct transformation (this will be shown in real-world experiments).

The above arguments show that large margin on its own is not enough; it is also necessary to control the spread of the data after projection. Therefore, maximum margin should be traded-off or

balanced with the goal of simultaneously minimizing the spread of the projected data, for instance, by bounding the spread $|\mathbf{w}^\top \mathbf{x} + b|$. This will allow the linear classifier to recover large margin solutions not in the absolute sense but rather *relative to* the spread of the data in that projection direction.

In the case of a kernel such as the RBF kernel, the points are first mapped to a space so that all the input examples are unit vectors (i.e., $\langle \phi(\mathbf{x}), \phi(\mathbf{x}) \rangle = 1$). Note that the intuitive motivation proposed here still applies in such cases. No matter how they are mapped initially, a large margin solution still projects these points to the real line where the margin of separation is maximized. However, the spread of the projection can still vary significantly among the different projection directions. Given the above motivation, it is important to achieve a large margin relative to the spread of the projections even in such situations. Furthermore, experiments will support this intuition with dramatic improvements on many real problems and with a variety of kernels (including radial basis function and polynomial kernels).

2.2 Probabilistic Motivation

In this subsection, an informal motivation is provided to illustrate why maximizing relative margin may be helpful. Suppose $(\mathbf{x}_i, y_i)_{i=1}^n$ are drawn independently and identically (*iid*) from a distribution $\mathcal{D}$. A classifier $\mathbf{w} \in \mathbb{R}^m$ is sought which will produce low error on future unseen examples according to the decision rule $\hat{y} = \text{sign}(\mathbf{w}^\top \mathbf{x})$. An alternative criterion is that the classifier should produce a large value of η according to the following expression:

$$\Pr_{(\mathbf{x},y)\sim\mathcal{D}} \left[y\mathbf{w}^\top \mathbf{x} \geq 0 \right] \geq \eta,$$

where $\mathbf{w} \in \mathbb{R}^m$ is the classifier. One way to ensure the above constraint is by requiring that the following inequality hold:

$$\mathbf{E}_{\mathcal{D}}[y\mathbf{w}^\top \mathbf{x}] \geq \sqrt{\frac{\eta}{1-\eta}} \sqrt{\mathbf{V}_{\mathcal{D}}[y\mathbf{w}^\top \mathbf{x}]}. \tag{1}$$

A proof of the above claim for a general distribution can be found in Shivaswamy et al. (2006). In fact, Gaussian margin machines (Crammer et al., 2009b) start with a similar motivation but assume a Gaussian distribution on the classifier.

According to (1), achieving a low probability of error requires the projections to have a large mean and a small variance. The mean and variance for the true distribution $\mathcal{D}$ may be unavailable, however, the empirical counterparts of these quantities are available and known to be concentrated. The above inequality is used as a loose motivation. Instead of precisely finding low variance and high mean projections, this paper implements this intuition by trading off between large margin and small projections of the data while correctly classifying most of the examples with a hinge loss.

2.3 Motivation From an Affine Invariance Perspective

Another motivation for maximum relative margin can be made by reformulating the classification problem altogether. Instead of learning a classifier from data, consider learning an affine transformation on data such that an a priori *fixed* classifier performs well. The data will be mapped by an affine transformation such that it is separated with large margin while it also produces a small radius. Recall that maximum margin classification and SVMs are motivated by generalization bounds based on Vapnik-Chervonenkis complexity arguments. These generalization bounds depend on the

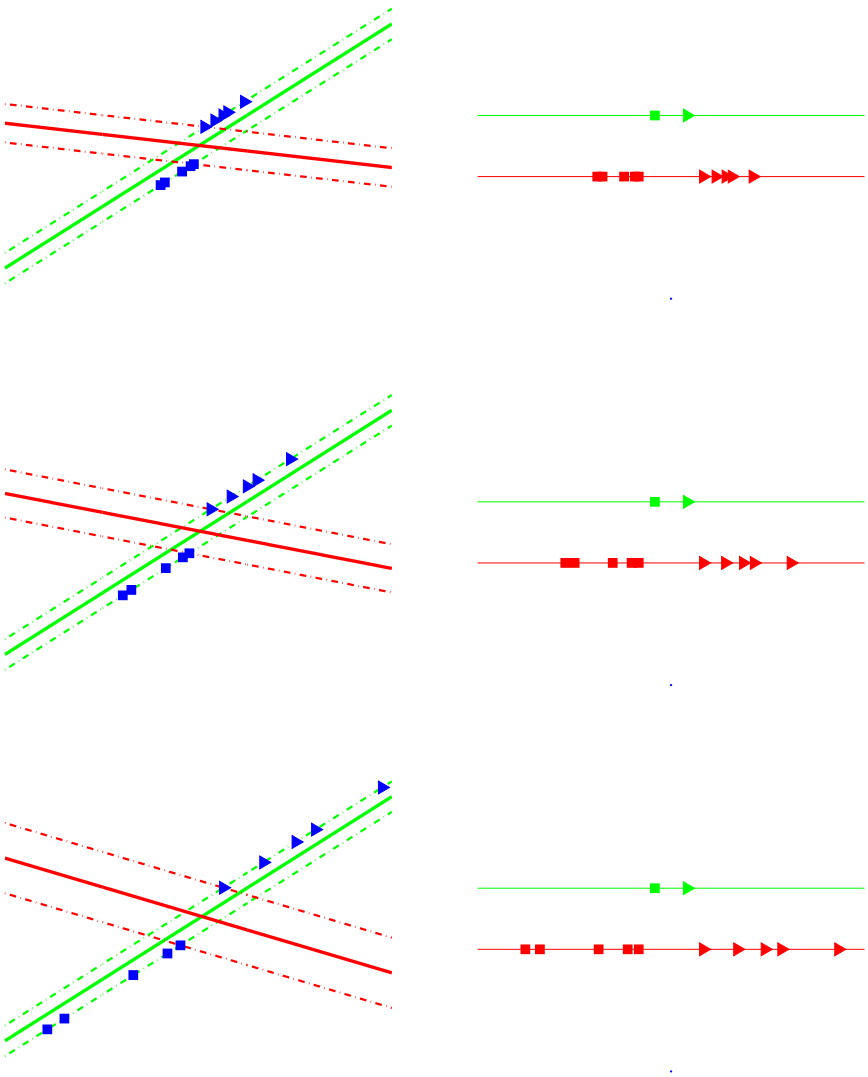


Figure 1: Left: As the data is scaled, the maximum margin SVM solution (red or dark shade) deviates from the maximum relative margin solution (green or light shade). Three different scaling scenarios are shown. Right: The projections of the examples (that is $\mathbf{w}^\top \mathbf{x} + b$) on the real line for the SVM solution (red or dark shade) and the proposed classifier (green or light shade) under each scaling scenario. These projections have been drawn on separated axes for clarity. The absolute margins for the maximum margin solution (red) are 1.24, 1.51 and 2.08 from top to bottom. For the maximum relative margin solution (green) the absolute margin is merely 0.71. However, the relative margin (the ratio of absolute margin to the spread of the projections) is 41%, 28%, and 21% for the maximum margin solution (red) and 100% for the relative margin solution (green). The scale of all axes is kept locked to permit direct visual comparison.

ratio of the margin to the radius of the data (Vapnik, 1995). Similarly, Rademacher generalization bounds (Shawe-Taylor and Cristianini, 2004) also consider the ratio of the trace of the kernel matrix to the margin. Here the radius of the data refers to an R such that $||\mathbf{x}|| \leq R$ for all $\mathbf{x}$ drawn from a distribution.

Instead of learning a classification rule, the optimization problem considered in this section will recover an affine transformation which achieves a large margin from a *fixed* decision rule while also achieving small radius. Assume the classification hyperplane is given a priori via the decision boundary $\mathbf{w}_0^\top \mathbf{x} + b_0 = 0$ with the two supporting margin hyperplanes $\mathbf{w}_0^\top \mathbf{x} + b_0 = \pm\rho$. Here, $\mathbf{w}_0 \in \mathbb{R}^m$ can be an arbitrary unit vector and b_0 is an arbitrary scalar. Consider the problem of mapping all the training points (by an affine transformation $\mathbf{x} \rightarrow \mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{A} \in \mathbb{R}^{m \times m}, \mathbf{b} \in \mathbb{R}^m$) so that the mapped points (i.e., $\mathbf{A}\mathbf{x}_i + \mathbf{b}$) satisfy the classification constraints $\mathbf{w}_0^\top \mathbf{x} + b_0 = \pm\rho$ while producing small radius, $\sqrt{R}$. The choice of $\mathbf{w}_0$ and b_0 is arbitrary since the affine transformation can completely compensate for it. For brevity, denote by $\tilde{\mathbf{A}} = [\mathbf{A}\ \mathbf{b}]$ and $\tilde{\mathbf{x}} = [\mathbf{x}^\top\ 1]^\top$. With this notation, the affine transformation learning problem is formalized by the following optimization:

$$\begin{aligned} \min_{\tilde{\mathbf{A}}, R, \rho} \quad & -\rho + ER & (2) \\ & y_i(\mathbf{w}_0^\top \tilde{\mathbf{A}} \tilde{\mathbf{x}}_i + b_0) \geq \rho, & \forall 1 \leq i \leq n \\ & \frac{1}{2}(\tilde{\mathbf{A}}\tilde{\mathbf{x}}_i)^\top (\tilde{\mathbf{A}}\tilde{\mathbf{x}}_i) \leq R & \forall 1 \leq i \leq n. \end{aligned}$$

The parameter E trades off between the radius of the affine transformed data and the margin[2] that will be obtained. The following Lemma shows that this affine transformation learning problem is basically equivalent to learning a large margin solution with a small spread.

Lemma 1 *The solution* $\tilde{\mathbf{A}}^*$ *to* (2) *is a rank one matrix*.

Proof Consider the Lagrangian of the above problem with Lagrange multipliers $\boldsymbol{\alpha}, \boldsymbol{\lambda}, \geq \mathbf{0}$:

$$\begin{aligned} \mathcal{L}(\tilde{\mathbf{A}}, \rho, R, \boldsymbol{\alpha}, \boldsymbol{\lambda}) = -\rho + ER - \sum_{i=1}^n \alpha_i (y_i(\mathbf{w}_0^\top \tilde{\mathbf{A}}\tilde{\mathbf{x}}_i + b_0) - \rho) \\ + \sum_{i=1}^n \lambda_i (\frac{1}{2}(\tilde{\mathbf{A}}\tilde{\mathbf{x}}_i)^\top (\tilde{\mathbf{A}}\tilde{\mathbf{x}}_i) - R). \end{aligned}$$

Differentiating the above Lagrangian with respect to $\mathbf{A}$ gives the following expression:

$$\frac{\partial \mathcal{L}(\tilde{\mathbf{A}}, \rho, R, \alpha, \lambda)}{\partial \tilde{\mathbf{A}}} = -\sum_{i=1}^n \alpha_i y_i \mathbf{w}_0 \tilde{\mathbf{x}}_i^\top + \tilde{\mathbf{A}} \sum_{i=1}^n \lambda_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top. \quad (3)$$

From (3), at optimum,

$$\tilde{\mathbf{A}}^* \sum_{i=1}^n \lambda_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top = -\sum_{i=1}^n \alpha_i y_i \mathbf{w}_0 \tilde{\mathbf{x}}_i^\top.$$

It is therefore clear that $\tilde{\mathbf{A}}^*$ can always be chosen to have rank one since the right hand side of the expression is just an outer product of two vectors. ■

2. For brevity, the so-called slack variables have been intentionally omitted since the proof holds in any case.

Lemma 1 gives further intuition on why one should limit the spread of the recovered classifier. Learning a transformation matrix $\tilde{\mathbf{A}}$ so as to maximize the margin while minimizing the radius given an a priori hyperplane $(\mathbf{w}_0, b_0)$ is no different from learning a classification hyperplane $(\mathbf{w}, b)$ with a large margin as well as a small spread. This is because the rank of the affine transformation $\tilde{\mathbf{A}}^*$ is one; thus, $\tilde{\mathbf{A}}^*$ merely maps all the points $\tilde{\mathbf{x}}_i$ onto a line achieving a certain margin ρ but also limiting the output or spread. This means that finding an affine transformation which achieves a large margin and small radius is equivalent to finding a $\mathbf{w}$ and b with a large margin and with projections constrained to remain close to the origin. Thus, the affine transformation learning problem complements the intuitive arguments in Section 2.1 and also suggests that the learning algorithm should bound the spread of the data.

3. From Absolute Margin to Relative Margin

This section will provide an upgrade path from the maximum margin classifier (or SVM) to a maximum relative margin formulation. Given independent identically distributed examples $(\mathbf{x}_i, y_i)_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \{\pm 1\}$ are drawn from $\Pr(\mathbf{x}, y)$, the support vector machine primal formulation is as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t. } & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \ \ \xi_i \geq 0 \qquad\qquad \forall 1 \leq i \leq n. \end{aligned} \tag{4}$$

The above is an easily solvable quadratic program (QP) and maximizes the margin by minimizing $\|\mathbf{w}\|^2$. Since real data is seldom separable, slack variables (ξ_i) are used to relax the hard classification constraints. Thus, the above formulation maximizes the margin while minimizing an upper bound on the number of classification errors. The trade-off between the two quantities is controlled by the parameter C. Equivalently, the following dual of the formulation (4) can be solved:

$$\begin{aligned} \max_{\alpha} \ & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \\ \text{s.t. } & \sum_{i=1}^{n}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \qquad\qquad \forall 1 \leq i \leq n. \end{aligned} \tag{5}$$

Lemma 2 *The formulation in* (5) *is invariant to a rotation of the inputs*.

Proof Replace each $\mathbf{x}_i$ with $\mathbf{A}\mathbf{x}_i$ where $\mathbf{A}$ is a rotation matrix such that $\mathbf{A} \in \mathbb{R}^{m\times m}$ and $\mathbf{A}^\top\mathbf{A} = \mathbf{I}$. It is clear that the dual remains the same. ■

However, the dual is not the same if $\mathbf{A}$ is more general than a rotation matrix, for instance, if it is an arbitrary affine transformation.

The above classification framework can also handle non-linear classification readily by making use of Mercer kernels. A kernel function $k : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ replaces the dot products $\mathbf{x}_i^\top \mathbf{x}_j$ in (5). The kernel function k is such that $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, where $\phi : \mathbb{R}^m \to \mathcal{H}$ is a mapping to a Hilbert space. Thus, solving the SVM dual formulation (5) with a kernel function can give a

non-linear solution in the input space. In the rest of this article, $\mathbf{K} \in \mathbb{R}^{n \times n}$ denotes the Gram matrix whose individual entries are given by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. When applying Lemma 2 on a kernel defined feature space, the affine transformation is on $\phi(\mathbf{x}_i)$ and not on $\mathbf{x}_i$.

3.1 The Whitened SVM

One way of limiting sensitivity to affine transformations while recovering a large margin solution is to whiten the data with the covariance matrix prior to estimating the SVM solution. This may also reduce the bias towards regions of large data spread as discussed in Section 2. Denote by

$$\mathbf{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top - \frac{1}{n^2}\sum_{i=1}^{n} \mathbf{x}_i \sum_{j=1}^{n} \mathbf{x}_j^\top, \text{ and } \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i,$$

the sample covariance and sample mean, respectively. Now, consider the following formulation called $\mathbf{\Sigma}$-SVM:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \ & \frac{1-D}{2}\|\mathbf{w}\|^2 + \frac{D}{2}\|\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t. } & y_i(\mathbf{w}^\top(\mathbf{x}_i - \boldsymbol{\mu}) + b) \geq 1 - \xi_i,\ \xi_i \geq 0 \qquad \forall 1 \leq i \leq n \end{aligned} \tag{6}$$

where $0 \leq D \leq 1$ is an additional parameter that trades off between the two regularization terms. When $D = 0$, (6) gives back the usual SVM primal (although on translated data). The dual of (6) can be shown to be:

$$\begin{aligned} \max_{\alpha} \ & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\alpha_i y_i (\mathbf{x}_i - \boldsymbol{\mu})^\top ((1-D)\mathbf{I} + D\mathbf{\Sigma})^{-1} \sum_{j=1}^{n}\alpha_j y_j (\mathbf{x}_j - \boldsymbol{\mu}) \\ \text{s.t. } & \sum_{i=1}^{n}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \qquad \forall 1 \leq i \leq n. \end{aligned} \tag{7}$$

It is easy to see that the above formulation (7) is translation invariant and tends to an affine invariant solution when D tends to one. However, there are some problems with this formulation. First, the whitening process only considers second order statistics of the input data which may be inappropriate for non-Gaussian data sets. Furthermore, there are computational difficulties associated with whitening. Consider the following term:

$$(\mathbf{x}_i - \boldsymbol{\mu})^\top ((1-D)\mathbf{I} + D\mathbf{\Sigma})^{-1} (\mathbf{x}_j - \boldsymbol{\mu}).$$

When $0 < D < 1$, it can be shown, by using the Woodbury matrix inversion formula, that the above term can be kernelized as

$$\begin{aligned} \hat{k}(\mathbf{x}_i, \mathbf{x}_j) = & \frac{1}{1-D}\left(k(\mathbf{x}_i, \mathbf{x}_j) - \frac{\mathbf{K}_i^\top \mathbf{1}}{n} - \frac{\mathbf{K}_j^\top \mathbf{1}}{n} + \frac{\mathbf{1}^\top \mathbf{K} \mathbf{1}}{n^2} \right) \\ & - \frac{1}{1-D}\left(\left(\mathbf{K}_i - \frac{\mathbf{K}\mathbf{1}}{n} \right)^\top \left(\frac{\mathbf{I}}{n} - \frac{\mathbf{1}\mathbf{1}^\top}{n^2} \right) \left[\frac{1-D}{D}\mathbf{I} + \mathbf{K}\left(\frac{\mathbf{I}}{n} - \frac{\mathbf{1}\mathbf{1}^\top}{n^2} \right) \right]^{-1} \left(\mathbf{K}_j - \frac{\mathbf{K}\mathbf{1}}{n} \right) \right), \end{aligned}$$

where $\mathbf{K}_i$ is the i^{th} column of $\mathbf{K}$. This implies that the $\mathbf{\Sigma}$-SVM can be solved merely by solving (5) after replacing the kernel with $\hat{k}(\mathbf{x}_i, \mathbf{x}_j)$ as defined above. Note that the above formula involves a matrix inversion of size n, making the kernel computation alone $O(n^3)$. Even performing whitening as a preprocessing step in the feature space would involve this matrix inversion which is often computationally prohibitive.

3.2 Relative Margin Machines

While the above $\mathbf{\Sigma}$-SVM does address some of the issues of data spread, it made second order assumptions to recover $\mathbf{\Sigma}$ and involved a cumbersome matrix inversion. A more direct and efficient approach to control the spread is possible and will be proposed next.

The SVM will be modified such that the projections on the training examples remain bounded. A parameter will also be introduced that helps trade off between large margin and small spread of the projection of the data. This formulation will initially be solved by a quadratically constrained quadratic program (QCQP) in this section. The dual of this formulation will also be of interest and yield further geometric intuitions.

Consider the following formulation called the relative margin machine (RMM):

$$\begin{aligned} \min_{\mathbf{w},b} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i & (8)\\ \text{s.t. } & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1-\xi_i, \ \ \xi_i \geq 0 & \forall 1 \leq i \leq n \\ & \frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 \leq \frac{B^2}{2} & \forall 1 \leq i \leq n. \end{aligned}$$

This formulation is similar to the SVM primal (4) except for the additional constraints $\frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 \leq \frac{B^2}{2}$. The formulation has one extra parameter B in addition to the SVM parameter C. When B is large enough, the above QCQP gives the same solution as the SVM. Also note that only settings of $B > 1$ are meaningful since a value of B less than one would prevent any training examples from clearing the margin, that is, none of the examples could satisfy $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$ otherwise. Let $\mathbf{w}_C$ and b_C be the solutions obtained by solving the SVM (4) for a particular value of C. It is clear, then, that $B > \max_i |\mathbf{w}_C^\top \mathbf{x}_i + b_C|$, makes the constraint on the second line in the formulation (8) inactive for each i and the solution obtained is the same as the SVM estimate. This gives an upper threshold for the parameter B so that the RMM solution is not trivially identical to the SVM solution.

As B is decreased, the RMM solution increasingly differs from the SVM solution. Specifically, with a smaller B, the RMM still finds a large margin solution but with a smaller projection of the training examples. By trying different B values (within the aforementioned thresholds), different large relative margin solutions are explored. It is helpful to next consider the dual of the RMM problem.

The Lagrangian of (8) is given by:

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \alpha, \lambda, \beta) = & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left(y_i(\mathbf{w}^\top \mathbf{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{n}\beta_i\xi_i \\ & + \sum_{i=1}^{n}\lambda_i\left(\frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 - \frac{1}{2}B^2\right), \end{aligned}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda} \geq 0$ are the Lagrange multipliers corresponding to the constraints. Differentiating with respect to the primal variables and equating to zero produces:

$$
\begin{aligned}
(\mathbf{I} + \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \mathbf{x}_i^\top)\mathbf{w} + b \sum_{i=1}^{n} \lambda_i \mathbf{x}_i &= \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \\
\frac{1}{\boldsymbol{\lambda}^\top \mathbf{1}} (\sum_{i=1}^{n} \alpha_i y_i - \sum_{i=1}^{n} \lambda_i \mathbf{w}^\top \mathbf{x}_i) &= b, \\
\alpha_i + \beta_i &= C \qquad \forall 1 \leq i \leq n.
\end{aligned}
$$

Denoting by

$$
\boldsymbol{\Sigma}_{\boldsymbol{\lambda}} = \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \mathbf{x}_i^\top - \frac{1}{\boldsymbol{\lambda}^\top \mathbf{1}} \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \sum_{j=1}^{n} \lambda_j \mathbf{x}_j^\top, \text{ and } \boldsymbol{\mu}_{\boldsymbol{\lambda}} = \frac{1}{\boldsymbol{\lambda}^\top \mathbf{1}} \sum_{i=1}^{n} \lambda_i \mathbf{x}_i,
$$

the dual of (8) can be shown to be:

$$
\begin{aligned}
&\max_{\boldsymbol{\alpha}, \boldsymbol{\lambda}} \ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\lambda}})^\top (\mathbf{I} + \boldsymbol{\Sigma}_{\boldsymbol{\lambda}})^{-1} \sum_{j=1}^{n} \alpha_j y_j (\mathbf{x}_j - \boldsymbol{\mu}_{\boldsymbol{\lambda}}) + \frac{1}{2} B^2 \sum_{i=1}^{n} \lambda_i \qquad (9) \\
&\text{s.t. } 0 \leq \alpha_i \leq C \quad \lambda_i \geq 0 \qquad \forall 1 \leq i \leq n.
\end{aligned}
$$

Moreover, the optimal $\mathbf{w}$ can be shown to be:

$$
\mathbf{w} = (\mathbf{I} + \boldsymbol{\Sigma}_{\boldsymbol{\lambda}})^{-1} \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\lambda}}).
$$

Note that the above formulation is translation invariant since $\boldsymbol{\mu}_{\boldsymbol{\lambda}}$ is subtracted from each $\mathbf{x}_i$. $\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$ corresponds to a shape matrix (which is potentially low rank) determined by $\mathbf{x}_i$'s that have non-zero λ_i. From the Karush-Kuhn-Tucker (KKT) conditions of (8) it is clear that $\lambda_i (\frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 - \frac{B^2}{2}) = 0$. Consequently $\lambda_i > 0$ implies $(\frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 - \frac{B^2}{2}) = 0$. Notice the similarity in the two dual formulations in (7) and (9); both formulations look similar except for the choice of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ which transform the inputs. The RMM in (9) whitens data with the matrix $(\mathbf{I} + \boldsymbol{\Sigma}_{\boldsymbol{\lambda}})$ while simultaneously solving an SVM-like classification problem. While this is similar in spirit to the $\boldsymbol{\Sigma}$-SVM, the matrix $(\mathbf{I} + \boldsymbol{\Sigma}_{\boldsymbol{\lambda}})$ is being estimated directly to optimize the margin with a small data spread. The $\boldsymbol{\Sigma}$-SVM only whitens data as a preprocessing independently of the margin and the labels. The $\boldsymbol{\Sigma}$-SVM is equivalent to the RMM only in the rare situation when all $\lambda_i = t$ for some t which makes the $\boldsymbol{\mu}_{\boldsymbol{\lambda}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$ in the RMM and $\boldsymbol{\Sigma}$-SVM identical up to a scaling factor.

In practice, the above formulation will not be solved since it is computationally impractical. Solving (9) requires semi-definite programming (SDP) which prevents the method from scaling beyond a few hundred data points. Instead, an equivalent optimization will be used which gives the same solution but only requires quadratic programming. This is achieved by simply replacing the constraint $\frac{1}{2}(\mathbf{w}^\top \mathbf{x}_i + b)^2 \leq \frac{1}{2} B^2$ with the two equivalent linear constraints: $(\mathbf{w}^\top \mathbf{x}_i + b) \leq B$ and $-(\mathbf{w}^\top \mathbf{x}_i + b) \leq B$. With these linear constraints replacing the quadratic constraint, the problem is now merely a QP. In the primal, the QP has $4n$ constraints (including $\boldsymbol{\xi} \geq 0$) instead of the $2n$ constraints in the SVM. Thus, the RMM's quadratic program has the same order of complexity as the SVM. In the next section, an efficient implementation of the RMM problem is presented.

3.3 Fast Implementation

Once the quadratic constraints have been replaced with linear constraints, the RMM is merely a quadratic program which admits many fast implementation schemes. It is now possible to adapt previous fast SVM algorithms in the literature to the RMM. In this section, the *SVM*^{light} (Joachims, 1998) approach will be adapted to the following RMM optimization problem

$$\begin{aligned} \min_{\mathbf{w},b} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i & (10)\\ \text{s.t. } & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1-\xi_i, \ \ \xi_i \geq 0 & \forall 1 \leq i \leq n \\ & \mathbf{w}^\top \mathbf{x}_i + b \leq B & \forall 1 \leq i \leq n \\ & -\mathbf{w}^\top \mathbf{x}_i - b \leq B & \forall 1 \leq i \leq n. \end{aligned}$$

The dual of (10) can be shown to be the following:

$$\begin{aligned} \max_{\boldsymbol{\alpha},\boldsymbol{\lambda},\boldsymbol{\lambda}^*} \ & -\frac{1}{2}(\boldsymbol{\alpha}\bullet\mathbf{y}-\boldsymbol{\lambda}+\boldsymbol{\lambda}^*)^\top \mathbf{K}(\boldsymbol{\alpha}\bullet\mathbf{y}-\boldsymbol{\lambda}+\boldsymbol{\lambda}^*) + \boldsymbol{\alpha}^\top\mathbf{1} - B\boldsymbol{\lambda}^\top\mathbf{1} - B\boldsymbol{\lambda}^{*\top}\mathbf{1} & (11)\\ \text{s.t. } & \boldsymbol{\alpha}^\top\mathbf{y} - \boldsymbol{\lambda}^\top\mathbf{1} + \boldsymbol{\lambda}^{*\top}\mathbf{1} = 0 \\ & 0 \leq \boldsymbol{\alpha} \leq C\mathbf{1} \\ & \boldsymbol{\lambda}, \boldsymbol{\lambda}^* \geq \mathbf{0}, \end{aligned}$$

where the operator $\bullet$ denotes the element-wise product of two vectors.

The QP in (11) is solved in an iterative way. In each step, only a subset of the dual variables are optimized. For instance, in a particular iteration, take q, r and s ($\tilde{q}$, $\tilde{r}$ and $\tilde{s}$) to be indices of the free (fixed) variables in $\boldsymbol{\alpha}$, $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}^*$ respectively (ensuring that $q \cup \tilde{q} = \{1,2,\ldots n\}$ and $q \cap \tilde{q} = \emptyset$ and proceeding similarly for the other two indices). The optimization over the free variables in that step can then be expressed as:

$$\begin{aligned} \max_{\boldsymbol{\alpha}_q,\boldsymbol{\lambda}_r,\boldsymbol{\lambda}_s^*} \ & -\frac{1}{2}\begin{bmatrix} \boldsymbol{\alpha}_q\bullet\mathbf{y}_q \\ \boldsymbol{\lambda}_r \\ \boldsymbol{\lambda}_s^* \end{bmatrix}^\top \begin{bmatrix} \mathbf{K}_{qq} & -\mathbf{K}_{qr} & \mathbf{K}_{qs} \\ -\mathbf{K}_{rq} & \mathbf{K}_{rr} & -\mathbf{K}_{rs} \\ \mathbf{K}_{sq} & -\mathbf{K}_{sr} & \mathbf{K}_{ss} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_q\bullet\mathbf{y}_q \\ \boldsymbol{\lambda}_r \\ \boldsymbol{\lambda}_s^* \end{bmatrix} & (12)\\ & -\frac{1}{2}\begin{bmatrix} \boldsymbol{\alpha}_q\bullet\mathbf{y}_q \\ \boldsymbol{\lambda}_r \\ \boldsymbol{\lambda}_s^* \end{bmatrix}^\top \begin{bmatrix} \mathbf{K}_{q\tilde{q}} & -\mathbf{K}_{q\tilde{r}} & \mathbf{K}_{q\tilde{s}} \\ -\mathbf{K}_{r\tilde{q}} & \mathbf{K}_{r\tilde{r}} & -\mathbf{K}_{r\tilde{s}} \\ \mathbf{K}_{s\tilde{q}} & -\mathbf{K}_{s\tilde{r}} & \mathbf{K}_{s\tilde{s}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\tilde{q}}\bullet\mathbf{y}_{\tilde{q}} \\ \boldsymbol{\lambda}_{\tilde{r}} \\ \boldsymbol{\lambda}_{\tilde{s}}^* \end{bmatrix} \\ & + \boldsymbol{\alpha}_q^\top\mathbf{1} - B\boldsymbol{\lambda}_r^\top\mathbf{1} - B\boldsymbol{\lambda}_s^{*\top}\mathbf{1} \\ \text{s.t. } & \boldsymbol{\alpha}_q^\top y_q - \boldsymbol{\lambda}_r^\top\mathbf{1} + \boldsymbol{\lambda}_s^{*\top}\mathbf{1} = -\boldsymbol{\alpha}_{\tilde{q}}^\top y_{\tilde{q}} + \boldsymbol{\lambda}_{\tilde{r}}^\top\mathbf{1} - \boldsymbol{\lambda}_{\tilde{s}}^{*\top}\mathbf{1}, \\ & \mathbf{0} \leq \boldsymbol{\alpha}_q \leq C\mathbf{1}, \\ & \boldsymbol{\lambda}_r, \ \boldsymbol{\lambda}_s^* \geq \mathbf{0}. \end{aligned}$$

While the first term in the above objective is quadratic in the free variables (over which it is optimized), the second term is merely linear. Essentially, the above is a working-set scheme which iteratively solves the QP over subsets of variables until some termination criteria are achieved. The following enumerates the termination criteria that will be used in this article. If $\boldsymbol{\alpha}, \boldsymbol{\lambda}, \boldsymbol{\lambda}^*$ and b are

the current solution (b is determined by the KKT conditions just as with SVMs), then:

$$\begin{aligned}
&\forall i \text{ s.t. } 0<\alpha_i<C: && b-\varepsilon \le y_i - (\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)) \le b+\varepsilon \\
&\forall i \text{ s.t. } \alpha_i = 0: && y_i(\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \ge 1-\varepsilon \\
&\forall i \text{ s.t. } \alpha_i = C: && y_i(\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \le 1+\varepsilon \\
&\forall i \text{ s.t. } \lambda_i > 0: && B-\varepsilon \le (\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \le B+\varepsilon \\
&\forall i \text{ s.t. } \lambda_i = 0: && (\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \le B-\varepsilon \\
&\forall i \text{ s.t. } \lambda_i^* > 0: && B-\varepsilon \le -(\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \le B+\varepsilon \\
&\forall i \text{ s.t. } \lambda_i^* = 0: && -(\sum_{j=1}^{n}(\alpha_j y_j - \lambda_j + \lambda_j^*)k(\mathbf{x}_i,\mathbf{x}_j)+b) \le B-\varepsilon.
\end{aligned}$$

In each step of the algorithm, a small sub-problem of the structure of (12) is solved. To select the free variables, these conditions are checked to find the worst violating variables both from the top of the violation list and from the bottom. The selected variables are optimized by solving (12) while keeping the other variables fixed. Since only a small QP is solved in each step, the cubic time scaling behavior is circumvented for improved efficiency. A few other book-keeping tricks have also been adapted from *SVM^{light}* to yield other minor improvements.

Denote by p the number of elements chosen in each step of the optimization (i.e., $p = |q| + |r| + |s|$). The QP in each step takes $O(p^3)$ and updating the prediction values to compute the KKT violations takes $O(nq)$ time. Sorting the output values to choose the most violated constraints takes $O(n\log(n))$ time. Thus, the total time taken in each iteration of the algorithm is $O(p^3 + n\log(n) + nq)$. Empirical running times are provided in Section 5 for a digit classification problem.

Many other fast SVM solvers could also be adapted to the RMM. Recent advances such as the cutting plane SVM algorithm (Joachims, 2006), Pegasos (Shalev-Shwartz et al., 2007) and so forth are also applicable and are deferred for future work.

3.4 Variants of the RMM

It is not always desirable to have a parameter in a formulation that would depend explicitly on the output from a previous computation as in (10). It is possible to overcome this issue via the following optimization problem:

$$\begin{aligned}
\min_{\mathbf{w},b,\xi,t\geq 1} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i + Dt \qquad (13)\\
\text{s.t. } & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1-\xi_i, \ \ \xi_i \geq 0 && \forall 1 \leq i \leq n,\\
& +(\mathbf{w}^\top \mathbf{x}_i + b) \leq t && \forall 1 \leq i \leq n,\\
& -(\mathbf{w}^\top \mathbf{x}_i + b) \leq t && \forall 1 \leq i \leq n.
\end{aligned}$$

Note that (13) has a parameter D instead of the parameter B in (10). The two optimization problems are equivalent in the sense that for every value of B in (10), it is possible to have a corresponding D such that both optimization problems give the same solution.

Further, in some situations, a hard constraint bounding the outputs as in (13) can be detrimental due to outliers. Thus, it might be required to have a relaxation on the bounding constraints as well. This motivates the following relaxed version of (13):

$$\begin{aligned}
\min_{\mathbf{w},b,\xi,t\geq 1} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i + D(t + \frac{\nu}{n}\sum_{i=1}^{n}(\tau_i + \tau_i^*)) \qquad (14)\\
\text{s.t. } & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1-\xi_i, \ \ \xi_i \geq 0 && \forall 1 \leq i \leq n,\\
& +(\mathbf{w}^\top \mathbf{x}_i + b) \leq t + \tau_i && \forall 1 \leq i \leq n,\\
& -(\mathbf{w}^\top \mathbf{x}_i + b) \leq t + \tau_i^* && \forall 1 \leq i \leq n.
\end{aligned}$$

In the above formulation, ν controls the fraction of outliers. It is not hard to derive the dual of the above to express it in kernelized form.

4. Risk Bounds

This section provides generalization guarantees for the classifiers of interest (the SVM, $\mathbf{\Sigma}$-SVM and RMM) which all produce decision[3] boundaries of the form $\mathbf{w}^\top \mathbf{x} = 0$ from a limited number of examples. In the SVM, the decision boundary is found by minimizing a combination of $\mathbf{w}^\top \mathbf{w}$ and an upper bound on the number of errors. This minimization is equivalent to choosing a function $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$ from a set of linear functions with bounded ℓ_2 norm. Therefore, with a suitable choice of E, the SVM solution chooses the function $g(\cdot)$ from the set $\{\mathbf{x} \rightarrow \mathbf{w}^\top \mathbf{x} | \frac{1}{2}\mathbf{w}^\top \mathbf{w} \leq E\}$.

By measuring the complexity of the function class being explored, it is possible to derive generalization guarantees and risk bounds. A natural measure of how complex a function class is the Rademacher complexity which has been fruitful in the derivation of generalization bounds. For SVMs, such results can be found in Shawe-Taylor and Cristianini (2004). This section continues in the same spirit and defines the function classes and their corresponding Rademacher complexities for slightly modified versions of the RMM as well as the $\mathbf{\Sigma}$-SVM. Furthermore, these will be used to provide generalization guarantees for both classifiers. The style and content of this section closely follows that of Shawe-Taylor and Cristianini (2004).

The function classes for the RMM and $\mathbf{\Sigma}$-SVM will depend on the data. Thus, these both entail so-called data-dependent regularization which is not quite as straightforward as the function classes explored by SVMs. In particular, the data involved in defining data-dependent function classes will

3. The bias term is suppressed in this section for brevity.

be treated differently and referred to as landmarks to distinguish them from the training data. Landmark data is used to define the function class while training data is used to select a specific function from the class. This distinction is important for the following theoretical derivations. However, in practical implementations, both the $\mathbf{\Sigma}$-SVM and the RMM may use the training data to both define the function class and to choose the best function within it. Thus, the distinction between landmark data and training data is merely a formality for deriving generalization bounds which require independent sets of examples for both stages. Ultimately, however, it will be possible to still provide generalization guarantees that are independent of the particular landmark examples. Details of this argument are provided in Section 4.6. For this section, however, it is assumed that, in parallel with the training data, a separate data set of landmarks is provided to define the function class for the RMM and the $\mathbf{\Sigma}$-SVM.

4.1 Function Class Definitions

Consider the training data set $(\mathbf{x}_i, y_i)_{i=1}^n$ with $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \{\pm 1\}$ which are drawn independently and identically distributed (*iid*) from an unknown underlying distribution $\mathbf{P}[(\mathbf{x}, y)]$ denoted as $\mathcal{D}$. The features of the training examples above are denoted by the set $\mathbf{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$.

Given a choice of the parameter E in the SVM (where E plays the role of the regularization parameter), the set of linear functions the SVM considers is:

Definition 3 $\mathcal{F}_E := \{\mathbf{x} \to \mathbf{w}^\top \mathbf{x} | \frac{1}{2}\mathbf{w}^\top \mathbf{w} \leq E\}$.

The RMM maximizes the margin while also limiting the spread of projections on the training data. It effectively considers the following function class:

Definition 4 $\mathcal{H}_{E,D}^S := \{\mathbf{x} \to \mathbf{w}^\top \mathbf{x} | \frac{\bar{D}}{2}\mathbf{w}^\top \mathbf{w} + \frac{D}{2}(\mathbf{w}^\top \mathbf{x}_i)^2 \leq E \ \forall 1 \leq i \leq n\}$.

Above, take $\bar{D} := 1 - D$ and $0 < D < 1$ trades off between large margin and small spread on the projections.[4] Since the above function class depends on the training examples, standard Rademacher analysis, which is straightforward for the SVM, is no longer applicable. Instead, define another function class for the RMM using a distinct set of landmark examples.

A set $\mathbf{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{n_v}\}$ drawn *iid* from the same distribution $\mathbf{P}[\mathbf{x}]$, denoted as $\mathcal{D}_x$, is used as the landmark examples. With these landmark examples, the modified RMM function class can be written as:

Definition 5 $\mathcal{H}_{E,D}^V := \{\mathbf{x} \to \mathbf{w}^\top \mathbf{x} | \frac{\bar{D}}{2}\mathbf{w}^\top \mathbf{w} + \frac{D}{2}(\mathbf{w}^\top \mathbf{v}_i)^2 \leq E \ \forall 1 \leq i \leq n_v\}$.

Finally, function classes that are relevant for the $\mathbf{\Sigma}$-SVM are considered. These limit the average projection rather than the maximum projection. The data-dependent function class is defined as below:

Definition 6 $\mathcal{G}_{E,D}^S := \{\mathbf{x} \to \mathbf{w}^\top \mathbf{x} | \frac{\bar{D}}{2}\mathbf{w}^\top \mathbf{w} + \frac{D}{2n}\sum_{i=1}^n (\mathbf{w}^\top \mathbf{x}_i)^2 \leq E\}$.

A different landmark set $\mathbf{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$, again drawn *iid* from $\mathcal{D}_x$, is used in defining the corresponding landmark function class:

Definition 7 $\mathcal{G}_{B,D}^U := \{\mathbf{x} \to \mathbf{w}^\top \mathbf{x} | \frac{\bar{D}}{2}\mathbf{w}^\top \mathbf{w} + \frac{D}{2n}\sum_{i=1}^n (\mathbf{w}^\top \mathbf{u}_i)^2 \leq B\}$.

Note that the parameter E is fixed in $\mathcal{H}_{E,D}^V$ but n_v may be different from n. In the case of $\mathcal{G}_{B,D}^U$, the number of landmarks is the same (n) as the number of training examples but the parameter B is used instead of E. These distinctions are intentional and will be clarified in subsequent sections.

4. Zero and one are excluded from the range of D to avoid degenerate cases.

4.2 Rademacher Complexity

In this section the Rademacher complexity of the aforementioned function classes are quantified by bounding the empirical Rademacher complexity. Rademacher complexity measures the richness of a class of real-valued functions with respect to a probability distribution (Bartlett and Mendelson, 2002; Shawe-Taylor and Cristianini, 2004; Bousquet et al., 2004).

Definition 8 *For a sample* $\mathbf{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ *generated by a distribution on* $\mathbf{x}$ *and a real-valued function class* $\mathcal{F}$ *with domain* $\mathbf{x}$*, the empirical Rademacher complexity*[5] *of* $\mathcal{F}$ *is*

$$\hat{R}(\mathcal{F}) := \mathbf{E}_\sigma \left[\sup_{f \in \mathcal{F}} \left| \frac{2}{n} \sum_{i=1}^{n} \sigma_i f(\mathbf{x}_i) \right| \right]$$

where $\sigma = \{\sigma_1, \ldots \sigma_n\}$ *are independent random variables that take values* $+1$ *or* -1 *with equal probability. Moreover, the Rademacher complexity of* $\mathcal{F}$ *is:* $R(\mathcal{F}) := \mathbf{E}_\mathbf{S} \left[\hat{R}(\mathcal{F}) \right]$.

A stepping stone for quantifying the true Rademacher complexity is obtained by considering its empirical counterpart.

4.3 Empirical Rademacher Complexity

In this subsection, upper bounds on the empirical Rademacher complexities are derived for the previously defined function classes. These bounds provide insights on the regularization properties of the function classes for the sample $\mathbf{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\}$.

Theorem 9 $\hat{R}(\mathcal{F}_E) \leq T_0 := \frac{2\sqrt{2E}}{n} \sqrt{tr(\mathbf{K})}$, *where* $tr(\mathbf{K})$ *is the trace of the Gram matrix of the elements in* $\mathbf{S}$.

Proof

$$\begin{aligned}
\hat{R}(\mathcal{F}_E) &= \mathbf{E}_\sigma \left[\sup_{f \in \mathcal{F}_E} \left| \frac{2}{n} \sum_{i=1}^{n} \sigma_i f(\mathbf{x}_i) \right| \right] = \frac{2}{n} \mathbf{E}_\sigma \left[\max_{||\mathbf{w}|| \leq \sqrt{2E}} \left| \mathbf{w}^\top \sum_{i=1}^{n} \sigma_i \mathbf{x}_i \right| \right] \\
&\leq \frac{2\sqrt{2E}}{n} \mathbf{E}_\sigma \left[\left| \left| \sum_{i=1}^{n} \sigma_i \mathbf{x}_i \right| \right| \right] = \frac{2\sqrt{2E}}{n} \mathbf{E}_\sigma \left[\left(\sum_{i=1}^{n} \sigma_i \mathbf{x}_i^\top \sum_{j=1}^{n} \sigma_j \mathbf{x}_j \right)^{\frac{1}{2}} \right] \\
&\leq \frac{2\sqrt{2E}}{n} \left(\mathbf{E}_\sigma \left[\sum_{i,j=1}^{n} \sigma_i \sigma_j \mathbf{x}_i^\top \mathbf{x}_j \right] \right)^{\frac{1}{2}} = \frac{2\sqrt{2E}}{n} \sqrt{\mathrm{tr}(\mathbf{K})}.
\end{aligned}$$

The proof uses Jensen's inequality on the function $\sqrt{\cdot}$ and the fact that σ_i and σ_j are random variables taking values $+1$ or -1 with equal probability. Thus, when $i \neq j$, $\mathbf{E}_\sigma[\sigma_i \sigma_j \mathbf{x}_i^\top \mathbf{x}_i] = 0$ and, otherwise, $\mathbf{E}_\sigma[\sigma_i \sigma_i \mathbf{x}_i^\top \mathbf{x}_i] = \mathbf{E}_\sigma[\mathbf{x}_i^\top \mathbf{x}_i] = \mathbf{x}_i^\top \mathbf{x}_i$. The result follows from the linearity of the expectation operator. ■

5. The dependence of the empirical Rademacher complexity on n and $\mathbf{S}$ is suppressed by writing $\hat{R}(\mathcal{F})$ for brevity.

Roughly speaking, by keeping E small, the classifier's ability to fit arbitrary labels is reduced. This is one way to motivate a maximum margin strategy. Note that $\sqrt{\mathrm{tr}(\mathbf{K})}$ is a coarse measure of the spread of the data. However, most SVM formulations do not directly optimize this term. This motivates to next consider two new function classes.

Theorem 10 $\hat{R}(\mathcal{H}^V_{E,D}) \leq T_2(\mathbf{V},\mathbf{S})$, *where for any training set* $\mathcal{B}$ *and landmark*[6] *set* $\mathcal{A}$, $T_2(\mathcal{A},\mathcal{B}) := \min_{\lambda\geq 0}\ \frac{1}{|\mathcal{B}|}\sum_{\mathbf{x}\in\mathcal{B}}\mathbf{x}^\top\left(\bar{D}\mathbf{I}\sum_{\mathbf{u}\in\mathcal{A}}\lambda_{\mathbf{u}} + D\sum_{\mathbf{u}\in\mathcal{A}}\lambda_{\mathbf{u}}\mathbf{u}\mathbf{u}^\top\right)^{-1}\mathbf{x} + \frac{2E}{|\mathcal{B}|}\sum_{\mathbf{u}\in\mathcal{A}}\lambda_{\mathbf{u}}$.

Proof Start with the definition of the empirical Rademacher complexity:

$$\hat{R}(\mathcal{H}^V_{E,D}) = \mathbf{E}_\sigma\left[\sup_{\mathbf{w}:\frac{1}{2}(\bar{D}\mathbf{w}^\top\mathbf{w}+D(\mathbf{w}^\top\mathbf{v}_i)^2)\leq E}\left|\frac{2}{n}\sum_{i=1}^n\sigma_i(\mathbf{w}^\top\mathbf{x}_i)\right|\right].$$

Consider the supremum inside the expectation. Depending on the sign of the term inside $|\cdot|$, the above corresponds to either a maximization or a minimization. Without loss of generality, consider the case of maximization. When a minimization is involved, the value of the objective still remains the same. The supremum is recovered by solving the following optimization problem:

$$\max_{\mathbf{w}}\ \mathbf{w}^\top\sum_{i=1}^n\sigma_i\mathbf{x}_i\quad\text{s.t.}\qquad\frac{1}{2}(\bar{D}\mathbf{w}^\top\mathbf{w}+D(\mathbf{w}^\top\mathbf{v}_i)^2)\leq E\qquad\forall 1\leq i\leq n_v. \tag{15}$$

Using Lagrange multipliers $\lambda_1\geq 0,\ldots\lambda_{n_v}\geq 0$, the Lagrangian of (15) is: $\mathcal{L}(\mathbf{w},\lambda) = -\mathbf{w}^\top\sum_{i=1}^n\sigma_i\mathbf{x}_i + \sum_{i=1}^{n_v}\lambda_i\left(\frac{1}{2}\left(\bar{D}\mathbf{w}^\top\mathbf{w}+D(\mathbf{w}^\top\mathbf{v}_i)^2\right)-E\right)$. Differentiating this with respect to the primal variable $\mathbf{w}$ and equating it to zero gives: $\mathbf{w} = \boldsymbol{\Sigma}^{-1}_{\lambda,D}\sum_{i=1}^n\sigma_i\mathbf{x}_i$, where $\boldsymbol{\Sigma}_{\lambda,D} := \bar{D}\sum_{i=1}^{n_v}\lambda_i\mathbf{I} + D\sum_{i=1}^{n_v}\lambda_i\mathbf{v}_i\mathbf{v}_i^\top$. Substituting this $\mathbf{w}$ in $\mathcal{L}(\mathbf{w},\lambda)$ gives the dual of (15):

$$\min_{\lambda\geq 0}\ \frac{1}{2}\sum_{i=1}^n\sigma_i\mathbf{x}_i^\top\boldsymbol{\Sigma}^{-1}_{\lambda,D}\sum_{j=1}^n\sigma_j\mathbf{x}_j + E\sum_{i=1}^{n_v}\lambda_i.$$

This permits the following upper bound on the empirical Rademacher complexity since the primal and the dual objectives are equal at the optimum:

$$\begin{aligned}
\hat{R}(\mathcal{H}^V_{E,D}) &= \frac{2}{n}\mathbf{E}_\sigma\left[\min_{\lambda\geq 0}\ \frac{1}{2}\sum_{i=1}^n\sigma_i\mathbf{x}_i^\top\boldsymbol{\Sigma}^{-1}_{\lambda,D}\sum_{j=1}^n\sigma_j\mathbf{x}_j + E\sum_{i=1}^{n_v}\lambda_i\right]\\
&\leq \min_{\lambda\geq 0}\ \frac{2}{n}\mathbf{E}_\sigma\left[\frac{1}{2}\sum_{i=1}^n\sigma_i\mathbf{x}_i^\top\boldsymbol{\Sigma}^{-1}_{\lambda,D}\sum_{j=1}^n\sigma_j\mathbf{x}_j + E\sum_{i=1}^{n_v}\lambda_i\right]\\
&\leq \min_{\lambda\geq 0}\ \frac{1}{n}\sum_{i=1}^n\mathbf{x}_i^\top\boldsymbol{\Sigma}^{-1}_{\lambda,D}\mathbf{x}_i + \frac{2}{n}E\sum_{i=1}^{n_v}\lambda_i = T_2(\mathbf{V},\mathbf{S}).
\end{aligned}$$

On line one, the expectation is over the minimizers over λ; this is less than first taking the expectation and then minimizing over λ in line two. Then, simply recycle the arguments used in Theorem 9 to handle the expectation over σ. ■

6. $T_2(\mathcal{A},\mathcal{B})$ has been defined on generic sets. When an already defined set, such as $\mathbf{V}$ (with a known number n_v of elements) is an argument to T_2, λ will be subscripted with i or j.

Theorem 11 $\hat{R}(\mathcal{G}^U_{B,D}) \leq T_1(\mathbf{U},\mathbf{S})$, *where for any training set* $\mathcal{B}$ *and landmark set* $\mathcal{A}$, $T_1(\mathcal{A},\mathcal{B}) := \frac{2\sqrt{2B}}{|\mathcal{B}|}\left(\sum_{\mathbf{x}\in\mathcal{B}}\mathbf{x}^\top\left(\bar{D}\mathbf{I}+\frac{D}{|\mathcal{A}|}\sum_{\mathbf{u}\in\mathcal{A}}\mathbf{u}\mathbf{u}^\top\right)^{-1}\mathbf{x}\right)^{\frac{1}{2}}$.

Proof The proof is similar to the one for Theorem 10. ■

Thus, the empirical Rademacher complexities of the function classes of interest are bounded using the functions T_0, $T_1(\mathbf{U},\mathbf{S})$ and $T_2(\mathbf{V},\mathbf{S})$. For both $\mathcal{F}_E$ and $\mathcal{G}^U_{E,D}$, the empirical Rademacher complexity is bounded by a closed-form expression. For $\mathcal{H}^V_{E,D}$, optimizing over the Lagrange multipliers (i.e., the λ's) can further reduce the upper bound on empirical Rademacher complexity. This can yield advantages over both $\mathcal{F}_E$ and $\mathcal{G}^U_{E,D}$ in many situations and the overall shape of $\mathbf{\Sigma}_{\lambda,D}$ plays a key role in determining the overall bound; this will be discussed in Section 4.7. Note that the upper bound $T_2(\mathbf{V},\mathbf{S})$ is not a closed-form expression in general but can be evaluated in polynomial time using semi-definite programming by invoking Schur's complement lemma as shown by Boyd and Vandenberghe (2003).

4.4 From Empirical to True Rademacher Complexity

By definition 8, the empirical Rademacher complexity of a function class is dependent on the data sample, $\mathbf{S}$. In many cases, it is not possible to give exact expressions for the Rademacher complexity since the underlying distribution over the data is unknown. However, it is possible to give probabilistic upper bounds on the Rademacher complexity. Since the Rademacher complexity is the expectation of its empirical estimate over the data, by a straightforward application of McDiarmid's inequality (Appendix A), it is possible to show the following:

Lemma 12 *Fix* $\delta \in (0,1)$. *With probability at least* $1-\delta$ *over draws of the samples* $\mathbf{S}$ *the following holds for any function class* $\mathcal{F}$*:*

$$R(\mathcal{F}) \leq \hat{R}(\mathcal{F}) + 2\sqrt{\frac{\ln(2/\delta)}{2n}} \tag{16}$$

and,

$$\hat{R}(\mathcal{F}) \leq R(\mathcal{F}) + 2\sqrt{\frac{\ln(2/\delta)}{2n}}. \tag{17}$$

At this point, the motivation for introducing the landmark sets $\mathbf{U}$ and $\mathbf{V}$ becomes clear. The inequalities (16) and (17) do not hold when the function class $\mathcal{F}$ is dependent on the set $\mathbf{S}$. Specifically, using the sample $\mathbf{S}$ instead of the landmarks breaks the required *iid* assumptions in the derivation of (16) and (17). Thus neither Lemma 12, nor any of the results in Section 4.5 are sound for the function classes $\mathcal{G}^S_{B,D}$ and $\mathcal{H}^S_{E,D}$.

4.5 Generalization Bounds

This section presents generalization bounds for the three different function classes. The derivation largely follows the approach of Shawe-Taylor and Cristianini (2004) and, therefore, several details will be omitted in this article. Recall the theorem from Shawe-Taylor and Cristianini (2004) that leverages the empirical Rademacher complexity to provide a generalization bound.

Theorem 13 *Let $\mathcal{F}$ be a class of functions mapping Z to $[0,1]$; let $\{\mathbf{z}_1,\ldots,\mathbf{z}_n\}$ be drawn from the domain Z independently and identically distributed (*iid*) according to a probability distribution $\mathcal{D}$. Then, for any fixed $\delta \in (0,1)$, the following bound holds for any $f \in \mathcal{F}$ with probability at least $1-\delta$ over random draws of a set of examples of size n:*

$$\mathbf{E}_{\mathcal{D}}[f(\mathbf{z})] \leq \hat{\mathbf{E}}[f(\mathbf{z})] + \hat{R}(\mathcal{F}) + 3\sqrt{\frac{\ln(2/\delta)}{2n}}. \tag{18}$$

Similarly, under the same conditions as above, with probability at least $1-\delta$,

$$\hat{\mathbf{E}}[f(\mathbf{z})] \leq \mathbf{E}_{\mathcal{D}}[f(\mathbf{z})] + \hat{R}(\mathcal{F}) + 3\sqrt{\frac{\ln(2/\delta)}{2n}}. \tag{19}$$

Inequality (18) can be found in Shawe-Taylor and Cristianini (2004) and inequality (19) is obtained by a simple modification of the proof in Shawe-Taylor and Cristianini (2004). The following theorem, found in Shawe-Taylor and Cristianini (2004), gives a probabilistic upper bound on the future error rate based on the empirical error and the function class complexity.

Theorem 14 *Fix $\gamma > 0$. Let $\mathcal{F}$ be the class of functions from $\mathbb{R}^m \times \{\pm 1\} \to \mathbb{R}$ given by $f(\mathbf{x},y) = -yg(\mathbf{x})$. Let $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n)\}$ be drawn* iid *from a probability distribution $\mathcal{D}$. Then, with probability at least $1-\delta$ over the samples of size n, the following bound holds:*

$$\Pr_{\mathcal{D}}[y \neq \operatorname{sign}(g(\mathbf{x}))] \leq \frac{1}{n\gamma}\sum_{i=1}^{n}\xi_i + \frac{2}{\gamma}\hat{R}(\mathcal{F}) + 3\sqrt{\frac{\ln(2/\delta)}{2n}}, \tag{20}$$

where $\xi_i = \max(0, 1 - y_i g(\mathbf{x}_i))$ are the so-called slack variables.

The upper bounds that were derived in Section 4.2, namely: T_0, $T_1(\mathbf{U},\mathbf{S})$ and $T_2(\mathbf{V},\mathbf{S})$ can now be inserted into (20) to give generalization bounds for each class of interest. However, a caveat remains since a separate set of landmark data was necessary to provide such generalization bounds. The next section provides steps to eliminate the landmark data set from the bound.

4.6 Stating Bounds Independently of Landmarks

Note that the original function classes were defined using landmark examples. However, it is possible to eliminate these and state the generalization bounds independent of the landmark examples on function classes defined on the training data. Landmarks are eliminated from the generalization bounds in two steps. First, the empirical Rademacher complexities are shown to be concentrated and, second, the function classes defined using landmarks are shown to be supersets of the original function classes. One mild and standard assumption will be necessary, namely, that all examples from the distribution $\Pr([\mathbf{x}])$ have a norm bounded above by R with probability one.

4.6.1 CONCENTRATION OF EMPIRICAL RADEMACHER COMPLEXITY

Recall the upper bound $T_1(\mathbf{U},\mathbf{S})$ that was derived in Theorem 11. The following bounds show that these quantities are concentrated.

Theorem 15
i) With probability at least $1-\delta$*,*

$$T_1(\mathbf{U},\mathbf{S}) \leq \mathbf{E}_{\mathbf{U}}[T_1(\mathbf{U},\mathbf{S})] + O\left(\frac{1}{\sqrt{n}\sqrt{tr(\mathbf{K})}}\right).$$

ii) With probability at least $1-\delta$*,*

$$T_2(\mathbf{V},\mathbf{S}) \leq \mathbf{E}_{\mathbf{V}}[T_1(\mathbf{V},\mathbf{S})] + O\left(\frac{1}{\sqrt{n_v}\sqrt{tr(\mathbf{K})}}\right).$$

Proof McDiarmid's inequality from Appendix A can be applied to $T_1(\mathbf{U},\mathbf{S})$ since it is possible to compute Lipschitz constants $c_1, c_2, \ldots, c_n$ that correspond to each input of the function. These Lipschitz constants all share the same value c which is derived in Appendix B. With this Lipschitz constant, McDiarmid's inequality (32) is directly applicable and yields: $\Pr[T_1(\mathbf{U},\mathbf{S}) - \mathbf{E}_{\mathbf{U}}[T_1(\mathbf{U},\mathbf{S})] \geq \varepsilon] \leq \exp\left(-2\varepsilon^2/(nc^2)\right)$ Setting the upper bound on probability to δ, the following inequality holds with probability at least $1-\delta$:

$$T_1(\mathbf{U},\mathbf{S}) \leq \mathbf{E}_{\mathbf{U}}[T_1(\mathbf{U},\mathbf{S})] + \frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\left(\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i} - \sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i - \frac{DR^2\mu_{max}}{n\bar{D}+DR^2}}\right). \quad (21)$$

The second term above is:

$$\begin{aligned}
&\frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\left(\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i} - \sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i - \frac{DR^2\mu_{max}}{n\bar{D}+DR^2}}\right)\\
&= \frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\frac{DR^2\mu_{max}/(n\bar{D}+DR^2)}{\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i} + \sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i - \frac{DR^2\mu_{max}}{n\bar{D}+DR^2}}}\\
&\leq \frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\frac{DR^2\mu_{max}/(n\bar{D}+DR^2)}{\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i}}\\
&\leq \frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\frac{DR^4 n}{(n\bar{D}+DR^2)\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i}}\\
&\leq \frac{2\sqrt{\ln(1/\delta)E}}{\bar{D}\sqrt{n}}\frac{DR^4 n}{(n\bar{D})\sqrt{\sum_{i=1}^{n}\mathbf{x}_i^\top\mathbf{x}_i}} = O\left(\frac{1}{\sqrt{n}\sqrt{\mathrm{tr}(\mathbf{K})}}\right).
\end{aligned}$$

Here, $\mu_{max} \leq nR^2$ is the largest eigenvalue of the Gram matrix $\mathbf{K}$. The big oh notation refers to the asymptotic behavior in n. Note that $\mathrm{tr}(\mathbf{K})$ also grows with n. Thus, asymptotically, the above term is better than $O(1/\sqrt{n})$ which is the behavior of (20). So, from (21), with probability at least $1-\delta$: $T_1(\mathbf{U},\mathbf{S}) \leq \mathbf{E}_{\mathbf{U}}[T_1(\mathbf{U},\mathbf{S})] + O\left(1/\sqrt{n\,\mathrm{tr}(\mathbf{K})}\right)$.

The proof for the second claim is similar since $T_2(\mathbf{V},\mathbf{S})$ has the same Lipschitz constants (Appendix B). The only difference is in the number of elements in $\mathbf{V}$ which is reflected in the bound. ■

4.6.2 FUNCTION CLASS INCLUSION

At this point, using Equation 20 and Theorem 15, it is possible to state bounds that hold for functions in $\mathcal{G}^U_{B,D}$ and $\mathcal{H}^U_{B,D}$ but that are independent of $\mathbf{U}$ and $\mathbf{V}$ otherwise. However, the aim is to state uniform convergence bounds for functions in $\mathcal{G}^S_{B,D}$ and $\mathcal{H}^S_{B,D}$. This is achieved by showing the latter two sets are subsets of the former two with high probability. It is not enough to show that each element of one set is inside the other. Since uniform bounds are required for the initial function classes, one has to prove set-inclusion results.[7]

Theorem 16 *For $B = E + \varepsilon$ where $\varepsilon = O\left(\frac{1}{\sqrt{n}}\right)$, with probability at least $1 - 2\delta$ $\mathcal{G}^S_{E,D} \subseteq \mathcal{G}^U_{B,D}$.*

Proof First, note that $\mathcal{G}^S_{E,D} \subseteq \mathcal{F}_{E/\bar{D}}$. Thus, $\mathcal{F}_{E/\bar{D}}$ is a bigger class of functions than $\mathcal{G}^S_{E,D}$. Moreover, $\mathcal{F}_{E/\bar{D}}$ is not dependent on data. Now, consider $\frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2}(\mathbf{w}^\top\mathbf{x})^2$ where $\mathbf{w} \in \mathcal{F}_{E/\bar{D}}$. For $||\mathbf{x}|| \leq R^2$, the Cauchy-Schwarz inequality yields $\sup_{\mathbf{w}\in\mathcal{F}_{E/\bar{D}}} \frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2}(\mathbf{w}^\top\mathbf{x})^2 \leq \kappa$ where $\kappa = E/2 + DER^2/(2\bar{D})$. Now, define the function $h^{\mathbf{w}} : \mathcal{R}^m \to [0,1]$, as : $h^{\mathbf{w}}(\mathbf{x}) = (\frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2}(\mathbf{w}^\top\mathbf{x})^2)/\kappa$. Since the sets $\mathbf{S}$ and $\mathbf{U}$ are drawn *iid* from the distribution $\mathcal{D}_x$, it is now possible to apply (18) and (19) for any $\mathbf{w} \in \mathcal{F}_{E/\bar{D}}$. Applying (19) to $h^{\mathbf{w}}(\cdot)$ on $\mathbf{S}$, $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$, with probability at least $1-\delta$, the following inequality holds:

$$\mathbf{E}_{\mathcal{D}_x}[h^{\mathbf{w}}(\mathbf{x})] \leq \frac{1}{n}\sum_{i=1}^{n} h^{\mathbf{w}}(\mathbf{x}_i) + 2\sqrt{\frac{2E}{n\bar{D}}}\sqrt{\frac{1}{n}\mathrm{tr}(\mathbf{K})} + 3\sqrt{\frac{\ln(2/\delta)}{2n}}, \tag{22}$$

where the value of $\hat{R}(\mathcal{F}_{E/\bar{D}})$ has been obtained from Theorem 9. The expectation is over the draw of $\mathbf{S}$. Similarly, applying (18) to $h^{\mathbf{w}}(\cdot)$ on $\mathbf{U}$, with probability at least $1-\delta$, $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$, the following inequality holds:

$$\frac{1}{n}\sum_{i=1}^{n} h^{\mathbf{w}}(\mathbf{u}_i) \leq \mathbf{E}_{\mathcal{D}_x}[h^{\mathbf{w}}(\mathbf{u})] + 2\sqrt{\frac{2E}{n\bar{D}}}\sqrt{\frac{1}{n}\mathrm{tr}(\mathbf{K}_u)} + 3\sqrt{\frac{\ln(2/\delta)}{2n}} \tag{23}$$

where $\mathbf{K}_u$ is the Gram matrix of the landmark examples in $\mathbf{U}$. Using the fact that expectations in (22) and (23) are the same, $\mathrm{tr}(\mathbf{K}_u) \leq nR^2$, and the union bound, the following inequality holds $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$ with probability at least $1-2\delta$:

$$\frac{1}{n}\sum_{i=1}^{n} h^{\mathbf{w}}(\mathbf{u}_i) \leq \frac{1}{n}\sum_{i=1}^{n} h^{\mathbf{w}}(\mathbf{x}_i) + 4R\sqrt{\frac{2E}{n\bar{D}}} + 6\sqrt{\frac{\ln(2/\delta)}{2n}}.$$

Using the definition of $h^{\mathbf{w}}(\cdot)$, with probability at least $1-2\delta$, $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$,

$$\frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2n}\sum_{i=1}^{n}(\mathbf{w}^\top\mathbf{u}_i)^2 \leq \frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2n}\sum_{i=1}^{n}(\mathbf{w}^\top\mathbf{x}_i)^2 + O\left(\frac{1}{\sqrt{n}}\right).$$

Now, suppose, $\frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2n}\sum_{i=1}^{n}(\mathbf{w}^\top\mathbf{x}_i)^2 \leq E$, which describes the function class $\mathcal{G}^S_{E,D}$. If B is chosen to be $E+\varepsilon$ where $\varepsilon = O(\frac{1}{\sqrt{n}})$, then, $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$, with probability at least $1-2\delta$, $\mathbf{w}^\top\mathbf{w} + \frac{D}{2n}\sum_{i=1}^{n}(\mathbf{w}^\top\mathbf{u}_i)^2 \leq B$. Since $\mathcal{F}_{E/\bar{D}}$ is a superset of $\mathcal{G}^S_{E,D}$, with probability at least $1-2\delta$, $\mathcal{G}^S_{E,D} \subseteq \mathcal{G}^U_{E,D}$. ■

7. The function classes will also be treated as sets of parameters $\mathbf{w}$ without introducing additional notation.

Theorem 17 *For $n_v = O(\sqrt{n})$, with probability at least $1-2\delta$, $\mathcal{H}^S_{E,D} \subseteq \mathcal{H}^V_{E,D}$.*

Proof First define the function, $g^{\mathbf{w}} : \mathbb{R}^m \to \mathbb{R}$, as $g^{\mathbf{w}}(\mathbf{v}) = \frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2}(\mathbf{w}^\top\mathbf{v})^2$. Define the indicator random variable $\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v})>E]}$ which has a value 1 if $g^{\mathbf{w}}(\mathbf{v}) > E$ and a value 0 otherwise. By definition, $\forall \mathbf{w} \in \mathcal{H}^S_{E,D}$, $\forall \mathbf{x}_i \in \mathbf{S}$, $\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x}_i)>E]} = 0$. Similarly, $\forall \mathbf{w} \in \mathcal{H}^V_{E,D}$, $\forall \mathbf{v}_i \in \mathbf{V}$, $\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v}_i)>E]} = 0$. As before, consider a larger class of functions that is independent of $\mathbf{S}$, namely, $\mathcal{F}_{E/\bar{D}}$. For an *iid* sample $\mathbf{S}$ from the distribution $\mathcal{D}_x$, applying (18) to the indicator random variables $\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x})>E]}$ on the set $\mathbf{S}$, with probability at least $1-\delta$,

$$\mathbf{E}_{\mathcal{D}_x}\left[\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x})>E]}\right] \leq \frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x}_i)>E]} + 2\sqrt{\frac{2E}{n\bar{D}}}\sqrt{\frac{1}{n}\mathrm{tr}(\mathbf{K})} + 3\sqrt{\frac{\ln(2/\delta)}{2n}}. \tag{24}$$

Similarly, applying (19) on the set $\mathbf{V}$, with probability at least $1-\delta$,

$$\frac{1}{n_v}\sum_{i=1}^{n_v}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v}_i)>E]} \leq \mathbf{E}_{\mathcal{D}_x}\left[\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x})>E]}\right] + 2\sqrt{\frac{2E}{n\bar{D}}}\sqrt{\frac{1}{n_v}\mathrm{tr}(\mathbf{K}_v)} + 3\sqrt{\frac{\ln(2/\delta)}{2n_v}}. \tag{25}$$

Performing a union bound on (24) and (25), using the fact that $\mathrm{tr}(\mathbf{K}) \leq nR^2$ and $\mathrm{tr}(\mathbf{K}_v) \leq n_v R^2$ with probability at least $1-2\delta$, $\forall \mathbf{w} \in \mathcal{F}_{E/\bar{D}}$,

$$\frac{1}{n_v}\sum_{i=1}^{n_v}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v}_i)>E]} - \frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x}_i)>E]} \leq 4R\sqrt{\frac{2E}{n\bar{D}}} + 3\sqrt{\frac{\ln(2/\delta)}{2}}\left(\frac{1}{\sqrt{n}} + \frac{1}{\sqrt{n_v}}\right). \tag{26}$$

Equating the right hand side of the above inequality to $\frac{1}{n_v}$, the above inequality can be written more succinctly as:

$$\begin{aligned}&\mathbf{P}\left[\exists \mathbf{w} \in \mathcal{F}_{E/\bar{D}} \frac{1}{n_v}\sum_{i=1}^{n_v}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v}_i)>E]} - \frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x}_i)>E]}) \geq \frac{1}{n_v}\right] \\ \leq & 2\exp\left(-\frac{2}{9}\left(\frac{1}{n_v} - 4R\sqrt{\frac{2E}{n\bar{D}}}\right)^2 / \left(\frac{1}{\sqrt{n}} + \frac{1}{\sqrt{n_v}}\right)^2\right)\end{aligned}$$

The left hand side of the equation above is the probability that there exists a $\mathbf{w}$ such that the difference in the fraction of the number of examples that fall outside $\frac{\bar{D}}{2}\mathbf{w}^\top\mathbf{w} + \frac{D}{2}(\mathbf{w}^\top\mathbf{x})^2 \leq E$ over the random draw of the sets $\mathbf{S}$ and $\mathbf{V}$ is at least $\frac{1}{n_v}$. Thus, it gives an upper bound on the probability that $\mathcal{H}^S_{E,D}$ is contained in $\mathcal{H}^V_{E,D}$. This is because, if there is a $\mathbf{w} \in \mathcal{H}^S_{E,D}$ that is not in $\mathcal{H}^V_{E,D}$, for such a $\mathbf{w}$, $\frac{1}{n_v}\sum_{i=1}^{n_v}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{v}_i)>E]} > \frac{1}{n_v}$ and $\frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{[g^{\mathbf{w}}(\mathbf{x}_i)>E]} = 0$. Thus, equating the right hand side of (26) to $\frac{1}{n_v}$ and solving for n_v, the result follows. Both an exact value and the asymptotic behavior of n_v are derived in Appendix C. ■

It is straightforward to write the generalization bounds of Section 4.5 only in terms of $\mathbf{S}$, completely eliminating the landmark set $\mathbf{U}$ from the results in this section. However, the resulting bounds now have additional factors which further loosen them. In spite of this, in principle, using

a landmark set and compensating with McDiarmid's inequality can overcome the difficulties associated with a data-dependent hypothesis class and provide important generalization guarantees. In summary, the following overall bounds can now be provided for the function classes $\mathcal{F}_E$, $\mathcal{H}^S_{E,D}$ and $\mathcal{G}^S_{E,D}$. This result is obtained from a union bound of Theorem 14, Theorem 15, Theorem 16 and Theorem 17.

Theorem 18 *Fix $\gamma > 0$ and let $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be drawn* iid *from a probability distribution $\mathcal{D}$ where $\|x\| \leq R$.*

i) For any g from the function class $\mathcal{F}_E$, the following holds with probability at least $1-\delta$,

$$\Pr_{\mathcal{D}}[y \neq \text{sign}(g(\mathbf{x}))] \leq \frac{1}{\gamma n}\sum_{i=1}^{n}\xi_i + 3\sqrt{\frac{\ln(2/\delta)}{2n}} + \frac{4\sqrt{2E}}{n\gamma}\sqrt{tr(\mathbf{K})}. \tag{27}$$

ii) For any g from the function class $\mathcal{H}^S_{E,D}$, the following inequality (a solution of a semi-definite program) holds for $n_v = O(\sqrt{n})$ with probability at least $1-\delta$,

$$\begin{aligned}\Pr_{\mathcal{D}}[y \neq \text{sign}(g(\mathbf{x}))] &\leq \frac{1}{n\gamma}\sum_{i=1}^{n}\xi_i + 3\sqrt{\frac{\ln(8/\delta)}{2n}} + O\left(\frac{1}{\sqrt{n_v}\sqrt{tr(\mathbf{K})}}\right)\\ &+ \frac{2}{\gamma}\mathbf{E}_{\mathbf{V}}\left(\min_{\lambda\geq 0}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i^\top\left(\bar{D}\sum_{j=1}^{n_v}\lambda_j\mathbf{I} + D\sum_{j=1}^{n_v}\lambda_j\mathbf{v}_j\mathbf{v}_j^\top\right)^{-1}\mathbf{x}_i + \frac{2E}{n}\sum_{i=1}^{n_v}\lambda_i\right).\end{aligned} \tag{28}$$

iii) Similarly, for any g from the function class $\mathcal{G}^S_{E,D}$, the following bound holds for $B = E + O(\frac{1}{\sqrt{n}})$ with probability at least $1-\delta$,

$$\begin{aligned}\Pr_{\mathcal{D}}[y \neq \text{sign}(g(\mathbf{x}))] &\leq \frac{1}{n\gamma}\sum_{i=1}^{n}\xi_i + 3\sqrt{\frac{\ln(8/\delta)}{2n}} + O\left(\frac{1}{\sqrt{n}\sqrt{tr(\mathbf{K})}}\right)\\ &+ \frac{4\sqrt{2B}}{n\gamma}\mathbf{E}_{\mathbf{U}}\left(\sum_{i=1}^{n}\mathbf{x}_i^\top\left(\bar{D}\mathbf{I} + \frac{D}{n}\sum_{j=1}^{n}\mathbf{u}_j\mathbf{u}_j^\top\right)^{-1}\mathbf{x}_i\right)^{\frac{1}{2}},\end{aligned} \tag{29}$$

where $\xi_i = \max(0, \gamma - y_i g(\mathbf{x}_i))$ are the so-called slack variables.

4.7 Discussion of the Bounds

Clearly, all the three bounds, namely (27), (28) and (29) in Theorem (18) have similar asymptotic behavior in n, so how do they differ? Simple, separable scenarios are considered in this section to examine these bounds (which will be referred to as the SVM bound, RMM bound and $\mathbf{\Sigma}$-SVM bound respectively). For the SVM bound, the quantity of interest is $4\frac{\sqrt{2E}}{n\gamma}\sqrt{\text{tr}(\mathbf{K})}$ and, for the $\mathbf{\Sigma}$-SVM bound, the quantity of interest is $\frac{4\sqrt{2E}}{n\hat{\gamma}}\sqrt{\left(\sum_{i=1}^{n}\mathbf{x}_i^\top\left(\bar{D}\mathbf{I} + \frac{D}{n}\sum_{j=1}^{n}\mathbf{u}_j\mathbf{u}_j^\top\right)^{-1}\mathbf{x}_i\right)}$. Similarly, for the RMM bound, the quantity of interest is:

$$\frac{2}{\hat{\gamma}}\left(\min_{\lambda\geq 0}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i^\top\left(\bar{D}\sum_{j=1}^{n_v}\lambda_j\mathbf{I} + D\sum_{j=1}^{n_v}\lambda_j\mathbf{v}_j\mathbf{v}_j^\top\right)^{-1}\mathbf{x}_i + \frac{2E}{n}\sum_{i=1}^{n_v}\lambda_i\right).$$

Here the expectations over $\mathbf{U}$ and $\mathbf{V}$ have been dropped for brevity; in fact, this is how these terms would have appeared without the concentration result (Theorem 15). Moreover, in the latter two cases, γ has been replaced by $\hat{\gamma}$ intentionally.

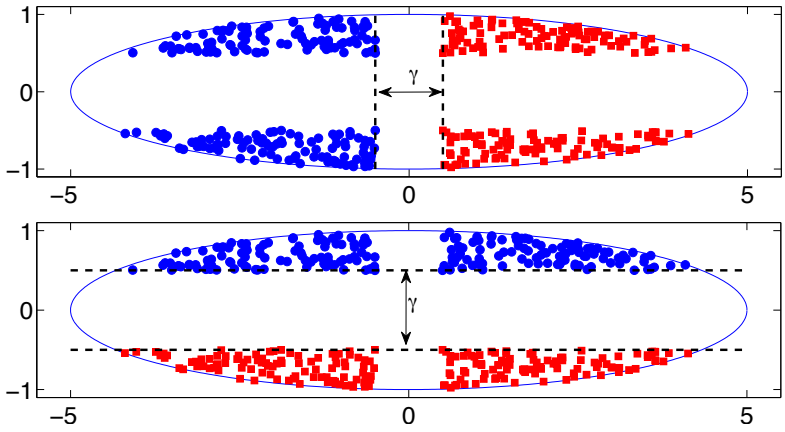


Figure 2: Two labellings of the same examples. Circles and squares denote the two classes (positive and negative). The top case is referred to as **toy example 1** and the bottom case is referred to as **toy example 2** in the sequel. The bound for the function class $\mathcal{F}_E$ does not distinguish between these two cases.

The differences between the three bounds will be illustrated with a toy example. In Figure 2, two different labellings of the same data set are shown. The two different labellings of the data produce completely different classification boundaries. However, in both the cases, the absolute margin of separation γ remains the same. A similar synthetic setting was explored in the context of second order perceptron bounds (Cesa-Bianchi et al., 2005).

The margin γ corresponding to the function class $\mathcal{F}$ is found by solving the following optimization problem:

$$\max_{\gamma,\mathbf{w}} \gamma, \quad \text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i) \geq \gamma, \ \frac{1}{2}\mathbf{w}^\top \mathbf{w} \leq E.$$

This merely recovers the absolute margin γ which is shown in the figure. Similarly, for the function class $\mathcal{G}$, a margin $\hat{\gamma}$ is obtained by solving:

$$\max_{\gamma,\mathbf{w}} \gamma, \text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i) \geq \gamma, \ \frac{1}{2}\mathbf{w}^\top \left(\bar{D}\mathbf{I} + \frac{D}{n} \sum_{j=1}^{n} \mathbf{x}_j \mathbf{x}_j^\top \right) \mathbf{w} \leq E.$$

Through a change of variable, $\mathbf{u} = \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{w}$ where $\mathbf{\Sigma} = \left(\bar{D}\mathbf{I} + \frac{D}{n} \sum_{j=1}^{n} \mathbf{x}_j \mathbf{x}_j^\top \right)$ it is easy to see that the above optimization problem is equivalent to

$$\max_{\gamma,\mathbf{u}} \gamma, \ \text{s.t. } y_i \mathbf{u}^\top \mathbf{\Sigma}^{-\frac{1}{2}} \mathbf{x}_i \geq \gamma, \ \frac{1}{2}\mathbf{u}^\top \mathbf{u} \leq E.$$

	toy example 1	**toy example 2**
SVM bound	0.643	0.643
Σ-SVM bound, D=0	0.643	0.643
Σ-SVM bound, D=0.999	0.859	0.281
RMM bound, D=0	0.643	0.643
RMM bound, D=0.999	1.355	0.315

Table 1: The bound values for the two toy examples. The SVM bound does not distinguish between the two cases. By exploring D values, it is possible to obtain smaller bound values in both cases for Σ-SVM and RMM ($D = 0$ in **toy example 1** and D close to one in **toy example 2**).

Thus, when a linear function is selected from the function class $\mathcal{G}^S_{D,E}$, the margin $\hat{\gamma}$ is estimated from a whitened version of the data. Similarly, for function class $\mathcal{H}^S_{E,D}$, the margin is estimated from a whitened version of the data where the whitening matrix is modified by Lagrange multipliers.

Thus, in the finite sample case, the bounds differ as demonstrated in the above synthetic problem. The bound for the function class $\mathcal{G}^S_{E,D}$ explores a whitening of the data. Suppose $D = 0.999$, the result is a whitening which evens out the spread of the data in all directions. On this whitened data set, the margin $\hat{\gamma}$ appears much larger in **toy example 2** since it is large compared to the spread. This leads to an improvement in the Σ-SVM bound over the usual SVM bound. While such differences could be compensated for by appropriate a priori normalization of features, this is not always an easy preprocessing.

Similarly, the RMM bound also considers a whitening of the data however, it shapes the whitening matrix adaptively by estimating λ. This gives further flexibility and rescales data not only along principal eigen-directions but in any direction where the margin is large relative to the spread of the data. By exploring D values, margin can be measured relative to the spread of the data rather than in the absolute sense. Since Σ-SVM and RMM are strict generalizations of the SVM, through the use of a proper validation set, it is almost always possible to obtain improvements. The various bounds for the toy examples are shown in Table 1.

5. Experiments

In this section, a detailed investigation of the performance of the RMM[8] on several synthetic and real world data sets is provided.

5.1 Synthetic Data Set

First consider a simple two dimensional data set that illustrates the performance differences between the SVM and the RMM. Since this is a synthetic data set, the best classifier can be constructed and Bayes optimal results can be reported. Consider sampling data from two different Gaussian distributions[9] corresponding to two different classes. Samples are drawn from the two following

8. Code available at `http://www1.cs.columbia.edu/~pks2103/RMM`.
9. Due to such Gaussian assumptions, LDA or generative modeling would be appropriate contenders but are omitted to focus the discussion on margin-based approaches.

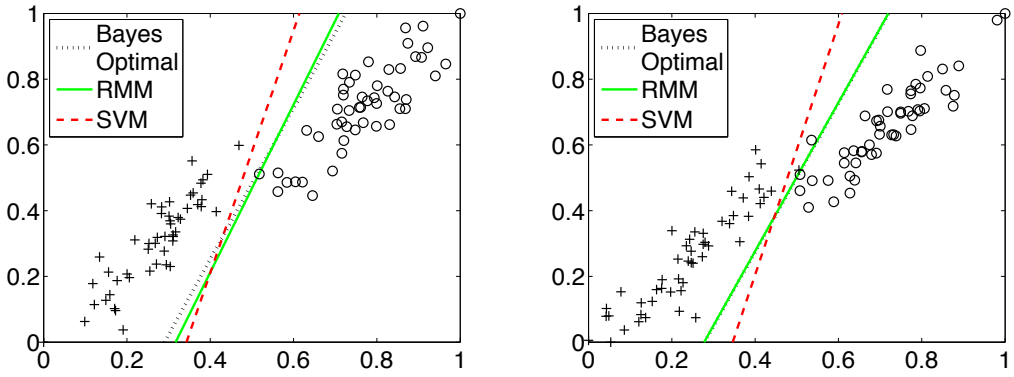


Figure 3: Two typical synthetic data sets (rescaled inside a 0-1 box) with corresponding SVM and RMM solutions are shown along with the Bayes optimal solution. The SVM (the RMM) solution uses the C (C and B) setting that minimized validation error. The RMM produces an estimate that is significantly closer to the Bayes optimal solution.

Gaussian distributions with equal prior probability:

$$\boldsymbol{\mu}_+ = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \ \boldsymbol{\mu}_- = \begin{bmatrix} 19 \\ 13 \end{bmatrix}, \ \boldsymbol{\Sigma} = \begin{bmatrix} 17 & 15 \\ 15 & 17 \end{bmatrix}.$$

The Gaussian distributions have different means, yet identical covariance. A total of 100,000 examples were drawn from each of the Gaussian distributions to create validation and test sets. Large validation and test sets were used to get accurate estimates of validation and test error.

Due to the synthetic nature of the problem, the Bayes optimal classifier is easily recovered (Duda et al., 2000) and is given by the following decision boundary

$$(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \boldsymbol{\Sigma}^{-1}\mathbf{x} - 0.5(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_+ + \boldsymbol{\mu}_-) = 0. \tag{30}$$

The above formula uses the true means and covariances of the Gaussian distributions (not empirical estimates). It is clear that the Bayes optimal solution is a linear decision boundary which is in the hypothesis class explored by both the RMM and the SVM. Note that the synthetic data was subsequently normalized to lie withing the zero-one box. This rescaling was taken into account while constructing the Bayes optimal classifier (30).

Various C values (and B values) were explored during SVM (RMM) training. The settings with minimum error rate on the validation set were used to compute test error rates. Furthermore, the test error rate for the Bayes optimal classifier was computed. Each experiment was repeated fifty times over random draws of train, test and validation sets. Figure 3 shows an example data set from this synthetic experiment along with the (cross-validated) SVM, RMM and Bayes optimal classification boundaries. The SVM decision boundary is biased to separate the data in a direction where it has large spread. The RMM is less biased by the spread and is visibly closer to the Bayes optimal solution.

Figure 4 shows the test error rates achieved for the SVM, the RMM and the Bayes optimal classifier. The SVM performs significantly worse than the RMM, particularly when training data

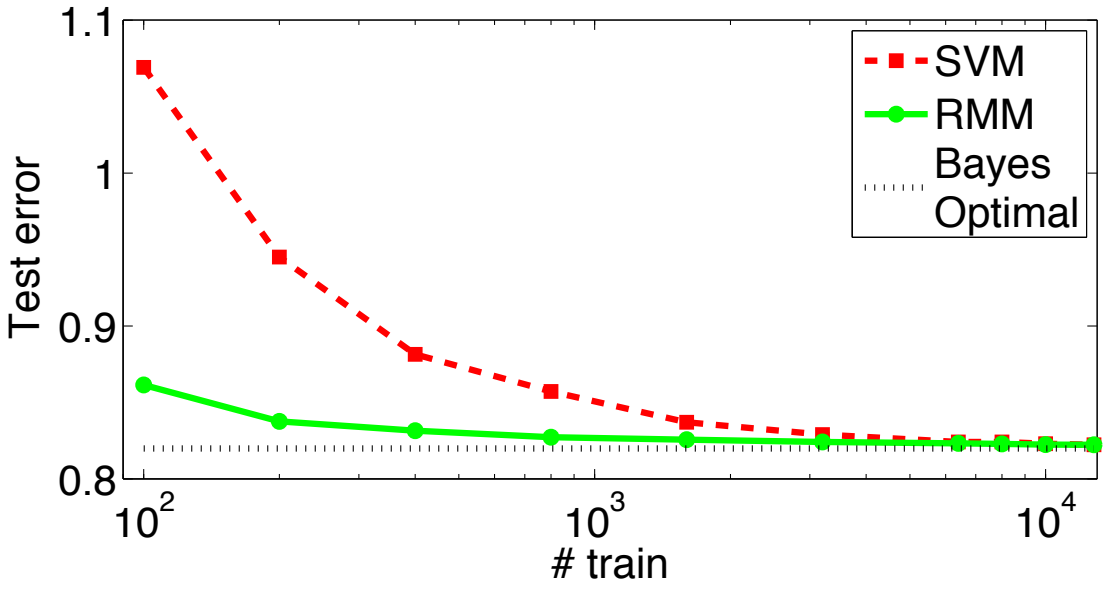


Figure 4: Percent test error rates for the SVM, RMM and Bayes optimal classifier as training data size is increased. The RMM has a statistically significant (at 5% level) advantage over the SVM until 6400 training examples. Subsequently, the advantage remains though with less statistical significance.

is scarce. The RMM maintains a statistically significant advantage over the SVM until the number of training examples grows beyond 6400. With larger training sample size n, regularization plays a smaller role in the future probability of error. This is clear, for instance, from the bound (27). The last term goes to zero at $O(1/\sqrt{n})$, the second term (which is the outcome of regularization) is $O(\sqrt{\mathrm{tr}(\mathbf{K})/n}\sqrt{1/n})$. Both have an $O(1/\sqrt{n})$ rate. However, the first term in the bound is the average slack variables divided by the margin which does not go to zero asymptotically with increasing n and eventually dominates the bound. Thus, the SVM and RMM have asymptotically similar performance but have significant differences in the small sample case.

The effect of scaling, which is a particular affine transformation, was explored next. To explore the effect of scaling in a controlled manner, first, the projection $\mathbf{w}$ recovered by the Bayes optimal classifier was obtained. An orthogonal vector $\mathbf{v}$ (such that $\mathbf{w}^\top \mathbf{v} = 0$) was then obtained. The examples (training, test and validation) were then projected onto the axes defined by $\mathbf{w}$ and $\mathbf{v}$. Each projection along $\mathbf{w}$ was preserved while the projection along $\mathbf{v}$ was scaled by a factor $s > 1$. This merely elongates the data further along directions orthogonal to $\mathbf{w}$ (i.e., along the Bayes optimal classification boundary). More concisely, given an example $\mathbf{x}$, the following scaling transformation was applied:

$$\begin{bmatrix} \mathbf{w} & \mathbf{v} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} \mathbf{w} & \mathbf{v} \end{bmatrix}^{-1} \mathbf{x}. \tag{31}$$

Figure 5 shows the SVM and RMM test error rate across a range of scaling values s. Here, 100 examples were used to construct the training data. As s grows, the SVM further deviates from the

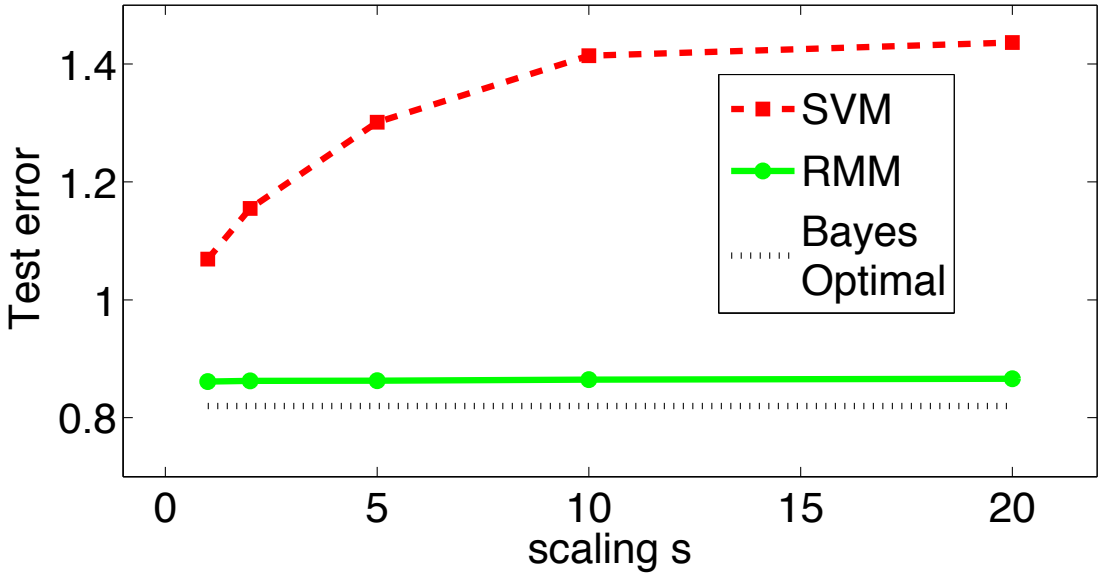


Figure 5: Percent test error rates for the SVM, RMM and Bayes optimal classifier as data is scaled according to (31). The RMM solution remains resilient to scaling while the SVM solution deteriorates significantly. The advantage of the RMM over the SVM is statistically significant (at the 1% level).

Bayes optimal classifier and attempts to separate the data along directions of large spread. Meanwhile, the RMM remains resilient to scaling and maintains a low error rate throughout.

To explore the effect of the B parameter, the average validation and test error rate were computed across many settings of C and B. The setting $C = 100$ was chosen since it obtained the minimum error rate on the validation set. The average test error rate of the RMM is shown in Figure 6 at $C = 100$ for multiple settings of the B parameter. Starting from the SVM solution on the right (i.e., large B) the error rate begins to fall until it attains a minimum and then starts to go increase. A similar behavior is seen in many real world data sets. Surprisingly, some data sets even exhibit monotonic reduction in test error as the value of B is decreased. The following section investigates such real world experiments in more detail.

5.2 Experiments on Digits

Experiments were carried out on three data sets of digits—optical digits from the UCI machine learning repository (Asuncion and Newman, 2007), USPS digits (LeCun et al., 1989) and MNIST digits (LeCun et al., 1998). These data sets vary considerably in terms of their number of features (64 in optical digits, 256 in USPS and 784 in MNIST) and their number of training examples (3823 in optical digits, 7291 in USPS and 60000 in MNIST). In all the multi-class experiments, the one versus one classification strategy was used. The one versus one strategy trains a classifier for every combination of two classes. The final prediction for an example is simply the class that is predicted most often. These results are directly comparable with various methods that have been applied on

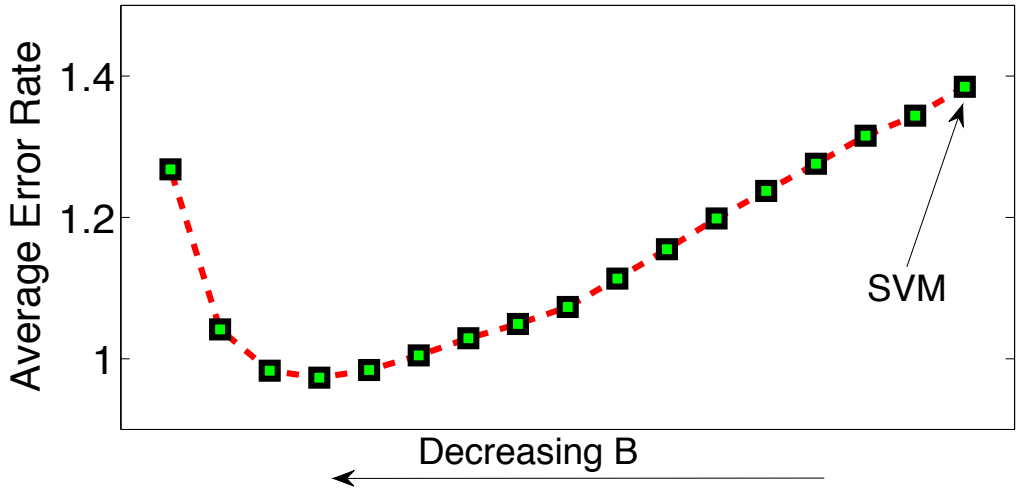


Figure 6: Behavior on the toy data set with $C = 100$. As the B value is decreased, the error rate decreases to a reasonably wide minimum before starting to increase.

		1	2	3	4	5	6	7	RBF
OPT	SVM	71	57	54	47	40	46	46	51
	Σ-SVM	**61**	48	41	36	35	31	**29**	47
	KLDA	71	57	54	47	40	46	46	**45**
	RMM	71	**36**	**32**	**31**	**33**	**30**	**29**	51
USPS	SVM	145	109	109	103	100	95	93	104
	Σ-SVM	**132**	**108**	99	94	**89**	**87**	**90**	**97**
	KLDA	132	119	121	117	114	118	117	101
	RMM	153	109	**94**	**91**	91	90	**90**	98
1000-MNIST	SVM	696	511	422	380	362	338	332	670
	Σ-SVM	**671**	470	373	341	322	309	303	673
	KLDA	1663	848	591	481	430	419	405	1597
	RMM	689	**342**	**319**	**301**	**298**	**290**	**296**	**613**
2/3-MNIST	SVM	552	237	200	183	178	177	164	166
	RMM	**534**	**164**	**148**	**140**	**123**	**129**	**129**	**144**
Full MNIST	SVM	536	198	170	156	157	141	136	146
	RMM	**521**	**146**	**140**	**130**	**119**	**116**	**115**	**129**

Table 2: The number of misclassification in three different digit data sets. Various kernels are explored using the SVM, Σ-SVM, KLDA and RMM methods.

this data set. For a fair comparison, results from contender methods that use special preprocessing or domain knowledge are not explored in this article.[10]

10. Additional results are reported in `http://yann.lecun.com/exdb/mnist/`.

In all experiments, the digits were first normalized to have unit norm. This eliminates numerical problems that may arise in kernel functions such as the polynomial kernel $k(\mathbf{u},\mathbf{v}) = (1+\mathbf{u}^\top\mathbf{v})^d$. Classification results were then examined for various degrees of the polynomial kernel. In addition, kernel values were further normalized so that the trace of the training Gram matrix was equal to the number of training examples.

All parameters were tuned by splitting the training data according to an 80:20 ratio with the larger split being used for training and the smaller split for validation. The process was repeated five times over random splits to select hyper-parameters (C for the SVM, C and D for the $\mathbf{\Sigma}$-SVM and C and B for the RMM). A final classifier was trained for each of the 45 classification problems with the best parameters found by cross validation using all the training examples in its corresponding pair of classes.

For the MNIST digits experiment, the $\mathbf{\Sigma}$-SVM and kernel LDA (KLDA) methods were too computationally demanding due to their use of matrix inversion. To cater to these methods, a smaller experiment was conducted with 1000 examples per training. For the larger experiments, the $\mathbf{\Sigma}$-SVM and KLDA were excluded. The larger experiment on MNIST involved training on two thirds of the digits (i.e., training with an average of 8000 examples for each pair of digits) for each binary classification task. In both these experiments, the remaining training data was used as a validation set. The classifier that performed best on the validation set was used for testing.

After forming all 45 classifiers (corresponding to each pair of digits), testing was done on the standard separate test sets available for each of these three benchmark problems (1797 examples in the case of optical digits, 2007 examples in USPS and 10000 examples in MNIST). The final prediction for each test example was recovered based on the majority of predictions made by the 45 classifiers on the test example with ties broken uniformly at random.

It is important to note that, on the MNIST test set, an error rate improvement of 0.1% has been established as statistically significant (Bengio et al., 2007; Decoste and Schölkopf, 2002). This corresponds to 10 or more test examples being correctly classified by one method over an other.

Table 2 shows results on all three digits data sets for polynomial kernels under varying degrees as well as for RBF kernels. For each data set, the number of misclassified examples using the majority voting scheme above is reported. The $\mathbf{\Sigma}$-SVM typically outperforms the SVM yet the RMM outperforms both. Interestingly, with higher degree kernels, the $\mathbf{\Sigma}$-SVM seems to match the performance of the RMM while in most lower-degree kernels, the RMM outperforms both the SVM and the $\mathbf{\Sigma}$-SVM convincingly. Since the $\mathbf{\Sigma}$-SVM is prohibitive to run on large scale data sets due to the computationally cumbersome matrix inversion, the RMM was clearly the most competitive method in these experiments in terms of both accuracy and computational efficiency.

The best parameters found by validation in the previous experiments were used in a full-scale MNIST experiment which does not have a validation set of its own. All 45 pair-wise classifiers (both SVMs and RMMs) were trained with the previously cross-validated parameters using *all* the training examples for each class in MNIST for various kernels. The test results are reported in Table 2; the RMM advantages persist in this full-scale MNIST experiment.

5.3 Classifying MNIST Digits 3 vs 5

This section presents more detailed results on one particular binary classification problem in the MNIST digits data set: the classification of digit 3 versus 5. Therein, the RMM has a dramatically stronger performance than the SVM. The results reported in this section are with polynomial kernels

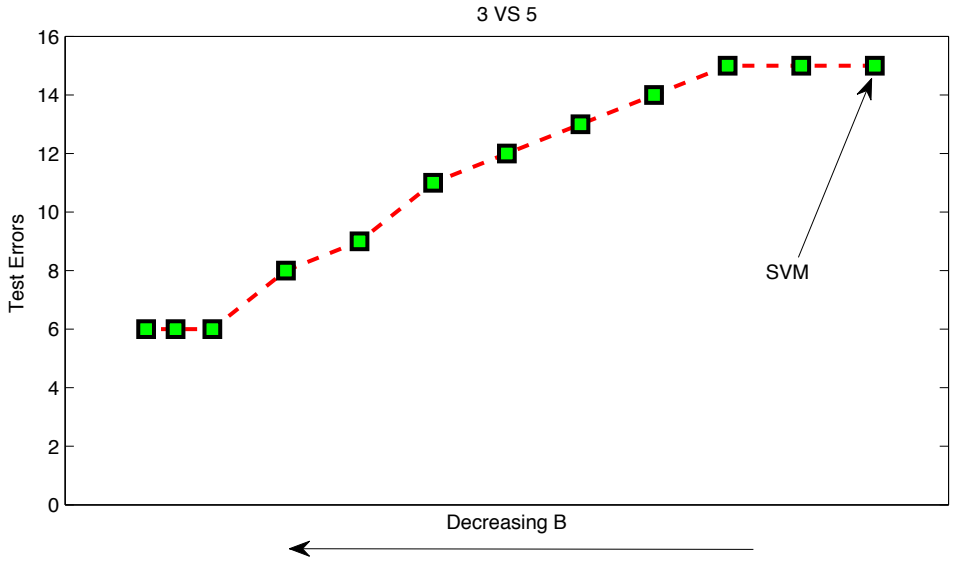


Figure 7: Performance on MNIST test set with digits 3 and 5. The number of errors decreases from 15 to 6 as B decreases from the right.

of degree 5. The parameter C was selected as mentioned above. With the selected C value, an SVM was first trained over the entire MNIST training set containing the digits 3 and 5. After noting the maximum absolute value of the output given on all the training examples, B value was reduced in steps. The number of test errors on the MNIST test set (3 versus 5) was noted. As the B value is reduced, the number of errors starts to diminish as shown in Figure 7. The number of errors produced by the SVM was 15. With the RMM, the number of errors dropped to 6 as the B value approached one. Clearly, as B decreases, the absolute margin is decreased however the test error rate drops drastically. This empirically suggests that maximizing the relative margin can have a beneficial effect on the performance of a classifier. Admittedly, this is only one example and is provided only for illustrative purposes. However, similar behavior was observed in most of the binary digit classification problems though in some cases the error rate did not go down significantly with decreasing B values. The generalization behavior on all 45 individual problems is explored in more detail in Section 5.4.

5.4 All 45 Binary MNIST Problems

This section explores RMM performance on the 45 pairwise digit classification problems in isolation. In these experiments, both C and B values were fixed using validation as in previous sections. A total of 45 binary classifiers were constructed using all MNIST training digits. The resulting error rates are shown in Figure 8. On most problems, the RMM obtains a significantly lower error rate than the SVM and, at times obtains half the error rate.

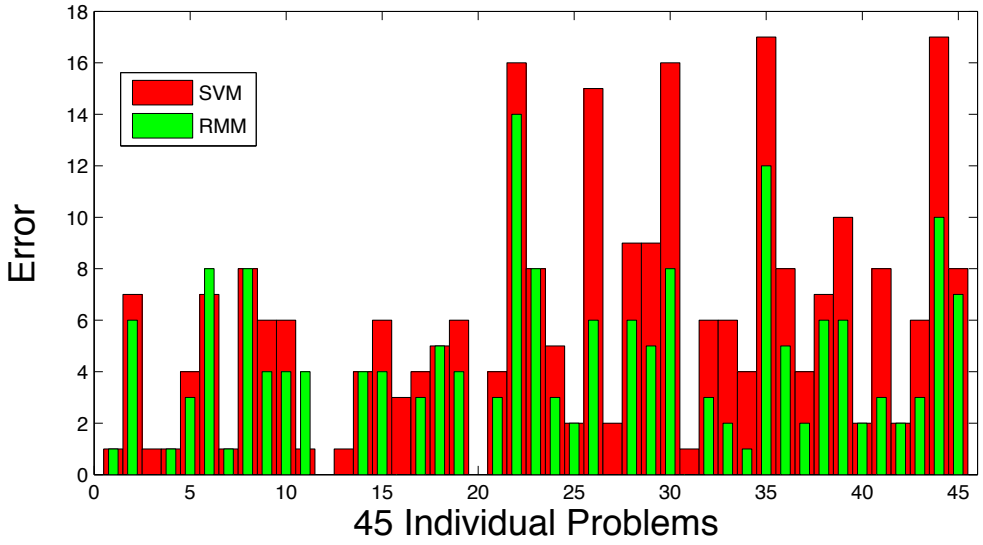


Figure 8: Total test errors on all 45 MNIST classification problems. Various classifiers were trained on the entire MNIST training data set and evaluated on a standardized separate test set.

Method	RMM	$\mathfrak{U}$-SVM		
# Universum	-	1000	3000	all
Error rate	1.081	1.059	1.037	1.020
Error Std Dev	0.138	0.142	0.149	0.159
p-value		0.402	0.148	0.031

Table 3: Percentage error rates for the RMM and the $\mathfrak{U}$-SVM. The rate for the SVM was 1.274 with a standard deviation of 0.179; this is significantly larger than all other results in the table (with a p-value of 0.000). The final row reports the p-value of a paired t-test between the RMM error rate and the $\mathfrak{U}$-SVM error rate (corresponding to the Universum size being considered in that column).

5.4.1 A COMPARISON WITH THE UNIVERSUM METHOD

A new framework known as the Universum (Weston et al., 2006; Sinz et al., 2008) was recently introduced which maximizes the margin while keeping classifier outputs low on an additional collection of non-examples that do not belong to either class of interest. These additional examples are known as Universum examples. Like landmarks, these are examples where a classifier's scalar predictions are forced to remain small. However, these Universum examples are obtained from any other distribution other than the one generating the training data. In the RMM, classification outputs on training examples are bounded; in the Universum, classification outputs on Universum examples are bounded (albeit with a different loss). The following experiments compare the Universum based framework with the RMM.

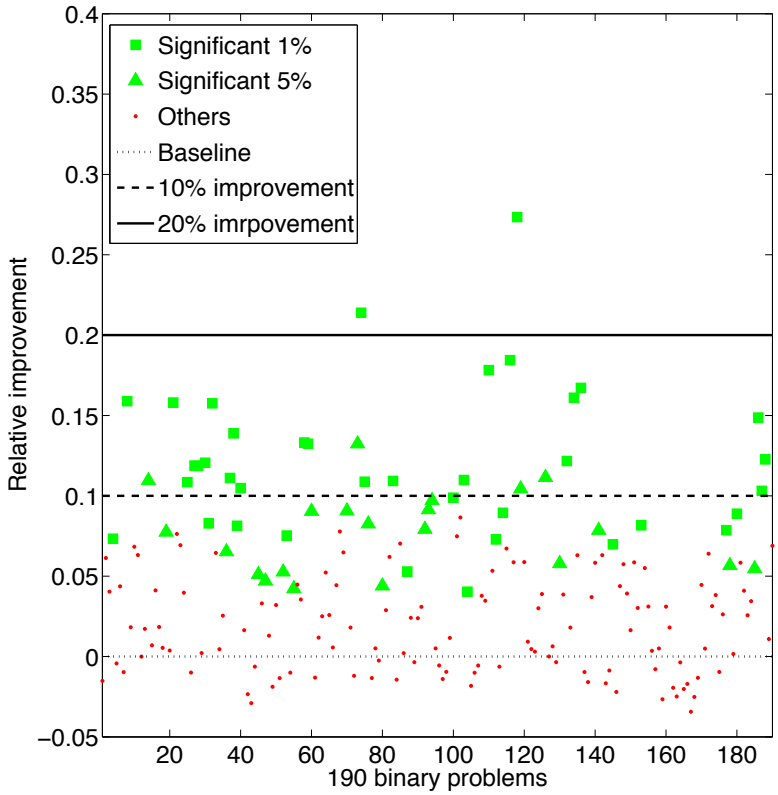


Figure 9: Percentage improvement of the RMM over the SVM on all 190 binary problems. Significance tests were performed using a paired t-test at the indicated levels of significance. On most problems, the RMM shows significant improvement over SVM.

An MNIST experiment was explored for classifying digits 5 vs 8 using 1000 labeled training examples under the RBF kernel. This setup is identical to the experimental conditions described in Weston et al. (2006). Examples of the digit 3 served as Universum examples since these were reported to be the *best* performing Universum examples in previous work (Weston et al., 2006). The experiments used the standard implementation of the Universum provided by the authors Weston et al. (2006) under the default parameter settings (for variables such as ε). The Universum was compared with the RMM which had access to the same 1000 training examples. Furthermore, 3500 examples were used as a test set and another 3500 examples as a validation set to perform model selection. All parameter settings for the RMM and the Universum SVM (or $\mathfrak{U}$-SVM) as well as the variance parameter of the RBF kernel were explored over a wide range of values. The parameter settings that achieved the smallest error on a validation set were then used to evaluate performance on the test set (and vice-versa). This entire experiment was repeated ten-fold over different random draws of the various sets. The average test error rates were compiled for both algorithms.

While the RMM only had access to the 1000 training examples, the $\mathfrak{U}$-SVM was also given a Universum of images of the digit 3. The Universum spanned three different sizes—1000, 3000 and 6131 examples (i.e., all available images of the digit 3 in the MNIST training set). The results are reported in Table 3. First, observe that both the RMM and the $\mathfrak{U}$-SVM improved the baseline SVM performance significantly (as measured by a paired t-test). With 1000 and 3000 Universum examples, even though the error rate of the $\mathfrak{U}$-SVM was slightly lower, a paired t-test revealed that it did not achieve statistically significant improvement over the RMM. Statistically significant advantages for the $\mathfrak{U}$-SVM only emerged when *all* the available images of the digit 3 were used in the Universum.

Note that there is a slight discrepancy between the errors reported here and those in Weston et al. (2006) even though both methods used the digit 3 to generate Universum examples. This may be because the previous authors (Weston et al., 2006) reported the *best test error* on 1865 examples. In this article, a more conservative approach is taken where a good model is first selected using the validation set and then errors are reported on an unseen test set without further tuning. Clearly, picking the minimum error rate on a test set will give more optimistic results but tuning to the test set can be potentially misleading. This makes it difficult to directly compare test error rates with those reported in the previous paper. While the error rate (using all digits 3 as the Universum examples) in our experiments varied from 0.74% to 1.35%, the authors in Weston et al. (2006) reported an error rate of 0.62%.

With 1000 training examples, the RMM (as in Equation (8)) has 1000 classification constraints and 1000 bounding constraints. With 1000 Universum examples, the $\mathfrak{U}$-SVM also has 1000 bounding constraints in addition to the classification constraints. It is interesting to note that the RMM, with no extra data, is not significantly worse than a $\mathfrak{U}$-SVM endowed with an additional 1000 or 3000 *best-possible* Universum examples.

The authors of Weston et al. (2006) observed that Universum examples help most when they are correlated with the training examples. This, coupled with the results in Table 3 and the fact that training examples are correlated most with themselves (or with examples from the same distribution as the training examples), raises the following question: How much of the performance gain with the $\mathfrak{U}$-SVM is due to the extra examples and how much of it is due to its implicit control of the spread (as with an RMM)? This is left as an open question in this article and as motivation for further theoretical work.

5.5 Text Classification

In this section, results are reported on the 20 Newsgroups [11] data set. This data set has posts from 20 different Usenet newsgroups. Each post was represented by a vector which counts the number of words that occurred in the document. In the text classification literature, this is commonly known as the bag of words representation. Each feature vector was divided by the total number of words in the document to normalize it.

All 190 binary pairwise classification problems were considered in this experiment. For each problem, 500 examples were used for training. The remaining examples were divided into a validation and test set of the same size. Both SVMs and RMMs were trained for various values of their parameters. After finding the parameter settings that achieved the lowest error on a validation set,

11. This data set is available online at `http://people.csail.mit.edu/jrennie/20Newsgroups/`.

the test error was evaluated (and vice-versa). This experiment was repeated ten times for random draws of the train, validation and test sets.

Figure 9 summarizes the results. For each binary classification problem, a paired t-test was performed and p-values were obtained. As can be seen from the plot, the RMM outperforms the SVM significantly in almost 30% of the problems. This experiment once again demonstrates that an absolute margin does not always result in a small test error.

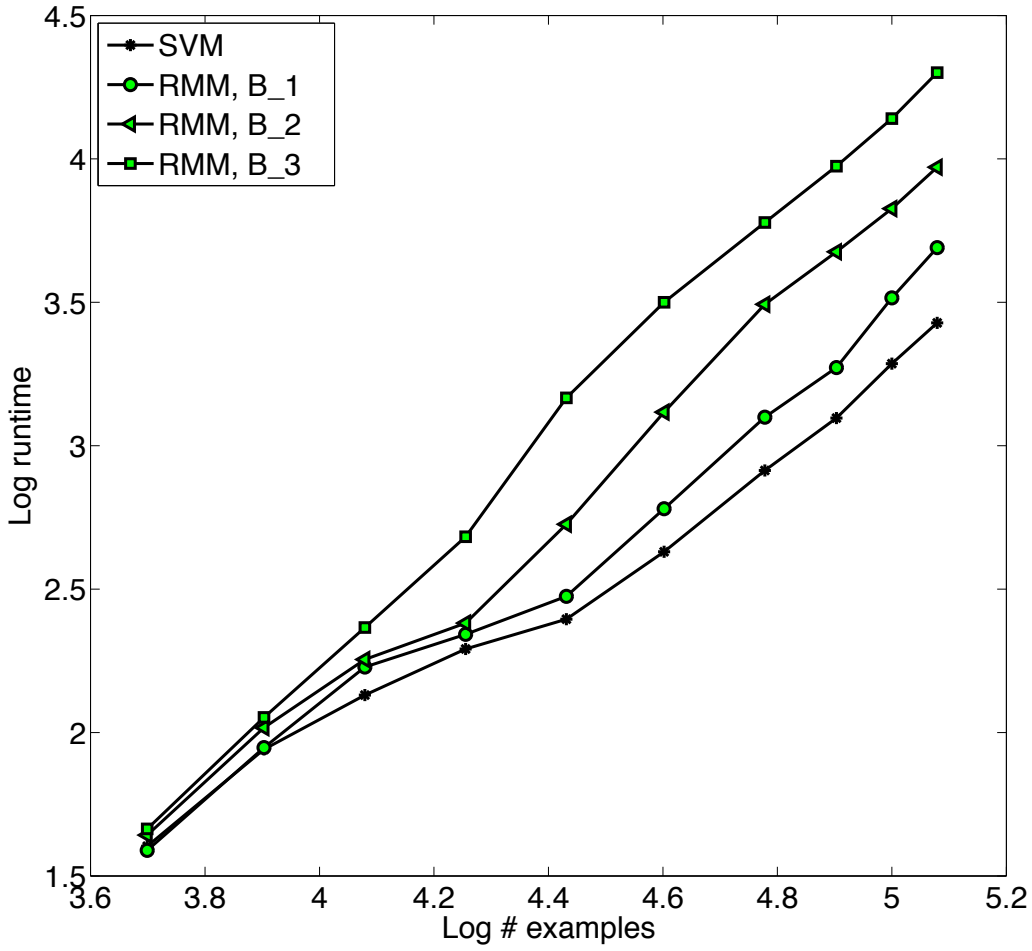


Figure 10: Log run time versus log number of examples. The figure shows that the SVM and the RMM have similar computational requirements overall.

Data Set	SVM	KLDA	Σ-SVM	RMM (C=D)	RMM
banana	10.5 ± 0.4	10.8 ± 0.5	10.5 ± 0.4	$\mathbf{10.4 \pm 0.4}$	$\mathbf{10.4 \pm 0.4}$*
b.cancer	$\mathbf{25.3 \pm 4.6}$*	26.6 ± 4.8	28.8 ± 4.6	25.9 ± 4.5	$\mathbf{25.4 \pm 4.6}$
diabetes	$\mathbf{23.1 \pm 1.7}$	$\mathbf{23.2 \pm 1.8}$	24.2 ± 1.9	$\mathbf{23.1 \pm 1.7}$	$\mathbf{23.0 \pm 1.7}$*
f.solar	$\mathbf{32.3 \pm 1.8}$	33.1 ± 1.6	34.6 ± 2.0	$\mathbf{32.3 \pm 1.8}$*	$\mathbf{32.3 \pm 1.8}$*
German	$\mathbf{23.4 \pm 2.2}$	24.1 ± 2.4	25.9 ± 2.4	$\mathbf{23.4 \pm 2.1}$	$\mathbf{23.2 \pm 2.2}$*
heart	15.5 ± 3.3	15.7 ± 3.2	19.9 ± 3.6	15.4 ± 3.3	$\mathbf{15.2 \pm 3.1}$*
image	$\mathbf{3.0 \pm 0.6}$	3.1 ± 0.6	3.3 ± 0.7	3.0 ± 0.6	$\mathbf{2.9 \pm 0.7}$
ringnorm	1.5 ± 0.1	$\mathbf{1.5 \pm 0.1}$	1.5 ± 0.1	$\mathbf{1.5 \pm 0.1}$	$\mathbf{1.5 \pm 0.1}$*
splice	10.9 ± 0.7	10.6 ± 0.7	10.8 ± 0.6	10.8 ± 0.6	10.8 ± 0.6
thyroid	4.7 ± 2.1	$\mathbf{4.2 \pm 2.1}$	4.5 ± 2.1	$\mathbf{4.2 \pm 1.8}$*	$\mathbf{4.2 \pm 2.2}$
titanic	22.3 ± 1.1	$\mathbf{22.0 \pm 1.3}$*	23.1 ± 2.2	22.3 ± 1.1	22.3 ± 1.0
twonorm	$\mathbf{2.4 \pm 0.1}$*	$\mathbf{2.4 \pm 0.2}$	2.5 ± 0.2	2.4 ± 0.1	$\mathbf{2.4 \pm 0.1}$
waveform	9.9 ± 0.4	9.9 ± 0.4	10.5 ± 0.5	10.0 ± 0.4	$\mathbf{9.7 \pm 0.4}$*

Data Set	RBF	AB	LPAB	QPAB	ABR
banana	10.8 ± 0.4	12.3 ± 0.7	10.7 ± 0.4	10.9 ± 0.5	10.9 ± 0.4
b.cancer	27.6 ± 4.7	30.4 ± 4.7	26.8 ± 6.1	$\mathbf{25.9 \pm 4.6}$	26.5 ± 4.5
diabetes	24.3 ± 1.9	26.5 ± 2.3	24.1 ± 1.9	25.4 ± 2.2	23.8 ± 1.8
f.solar	34.4 ± 1.9	35.7 ± 1.8	34.7 ± 2.0	36.2 ± 1.8	34.2 ± 2.2
German	24.7 ± 2.4	27.5 ± 2.5	24.8 ± 2.2	25.3 ± 2.1	24.3 ± 2.1
heart	17.1 ± 3.3	20.3 ± 3.4	17.5 ± 3.5	17.2 ± 3.4	16.5 ± 3.5
image	3.3 ± 0.7	$\mathbf{2.7 \pm 0.7}$	2.8 ± 0.6	$\mathbf{2.7 \pm 0.6}$*	$\mathbf{2.7 \pm 0.6}$*
ringnorm	1.7 ± 0.2	1.9 ± 0.2	2.2 ± 0.5	1.9 ± 0.2	1.6 ± 0.1
splice	9.9 ± 0.8	10.1 ± 0.5	10.2 ± 1.6	10.1 ± 0.5	$\mathbf{9.5 \pm 0.6}$*
thyroid	$\mathbf{4.5 \pm 2.1}$	$\mathbf{4.4 \pm 2.2}$	$\mathbf{4.6 \pm 2.2}$	$\mathbf{4.3 \pm 2.2}$	4.5 ± 2.2
titanic	23.3 ± 1.3	22.6 ± 1.2	24.0 ± 4.4	22.7 ± 1.0	22.6 ± 1.2
twonorm	2.8 ± 0.3	3.0 ± 0.3	3.2 ± 0.4	3.0 ± 0.3	2.7 ± 0.2
waveform	10.7 ± 1.1	10.8 ± 0.6	10.5 ± 1.0	10.1 ± 0.5	$\mathbf{9.8 \pm 0.8}$

Table 4: UCI results for a number of classification methods. Results are shown for the SVM, regularized kernel Linear Discriminant Analysis, the Σ-SVM, the RMM, an RBF network, Adaboost, LP-regularized Adaboost, QP-regularized Adaboost and Regularized Adaboost. The results have been split into two parts due to lack of space. For each data set, all methods could be placed on the same row in a larger table. For each data set, the algorithm which gave the minimum error rate is starred. All other algorithms that were not significantly different from (at the 5% significance level based on a paired t-test) the minimum error rate are in boldface.

5.6 Benchmark Data Sets

To compare the performance of the RMM with a number of other methods, experiments were performed on several benchmark data sets. In particular, 100 training and test splits of 13 of these data sets have been previously used in Raetsch et al. (2001); Mika et al. (1999); Cawley and Talbot (2003).[12] The RBF kernel was used in these experiments for all kernel-based methods. To handle the noisy nature of these data sets, the kernelized and relaxed version of the RMM (14) was used. All the parameters were tuned using cross-validation using a similar setup as in Raetsch et al. (2001).[13] With the chosen values of these parameters, the error rates were first obtained for all 100 test splits using the corresponding training splits. The results are reported in Table 4. Once again, the RMM exhibits clear performance advantages over other methods.

5.7 Scalability and Run-time

While the asymptotic run time behavior was analyzed in Section 3.3, the run time of the RMM is also studied empirically in this section. In particular, the classification of MNIST digits 0-4 versus 5-9 with a polynomial kernel of degree five was used to benchmark the algorithms. For both the RMM and the SVM, the tolerance parameter (ε mentioned in Section 3.3) was set to 0.001. The size of the sub-problem (12) solved was 800 in all the cases. To evaluate how the algorithms scale, the number of training examples was increased in steps and the training time was noted. Throughout all the experiments, the C value was set to 1. The SVM was first run on the training examples. The value of maximum absolute prediction θ was noted. Three different values of B were then tried for the RMM: $B_1 = 1 + (\theta - 1)/2$, $B_2 = 1 + (\theta - 1)/4$ and $B_3 = 1 + (\theta - 1)/10$. In all experiments, the run time was noted. The experiment was repeated ten times to get an average run time for each B value. A log-log plot comparing the number of examples to the average run time is shown in Figure 10. Both the SVM and the RMM run time exhibit similar asymptotic behavior.

6. Conclusions

The article showed that support vector machines and maximum margin classifiers can be sensitive to affine transformations of the input data and are biased in favor of separating data along directions with large spread. The relative margin machine was proposed to overcome such problems and optimizes the projection direction such that the margin is large only *relative to* the spread of the data. By deriving the dual with quadratic constraints, a geometric interpretation was also formulated for RMMs and led to risk bounds via Rademacher complexity arguments. In practice, the RMM implementation requires only additional linear constraints that complement the SVM quadratic program and maintain its efficient run time. Empirically, the RMM and maximum relative margin approach showed significant improvements in classification accuracy. In addition, an intermediate method known as Σ-SVM was shown that lies between the SVM and the RMM both conceptually and in terms of classification performance.

Generalization bounds with Rademacher averages were derived. The SVM's bound which involves the trace of the kernel matrix was replaced with a more general whitened version of the trace of the kernel matrix. A proof technique using landmark examples led to Rademacher bounds on an

12. These data sets are available at `http://theoval.cmp.uea.ac.uk/~gcc/matlab/default.html\#benchmarks`.
13. The values of the selected parameters and the code for the RMM are available for download at `http://www.cs.columbia.edu/~pks2103/ucirmm/`.

empirical data-dependent hypothesis space. Furthermore, the bounds were stated independently of the particular sample of landmarks.

Directions of future work include exploring the connections between maximum relative margin and generalization bounds based on margin distributions (Schapire et al., 1998; Koltchinskii and Panchenko, 2002). By bounding outputs, the RMM is potentially finding a better margin distribution on the training examples. Previous arguments for such an approach were obtained in the context of voting methods (such as boosting) and may also be relevant here.

Furthermore, the maximization of relative margin is a fairly promising and general concept which may be compatible with other popular problems that have recently been tackled by the maximum margin paradigm. These include regression, ordinal regression, ranking and so forth. These are valuable and natural extensions for the RMM. Finally, since the constraints that bound the projections are unsupervised, RMMs can readily apply in semi-supervised and transductive settings. These are all promising directions for future work.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under Grant Number IIS-0347499. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Appendix A. McDiarmid's Inequality

Assume $X_1, X_2, \ldots, X_n$ are independent random variables from a set $\mathcal{X}$ and $g : \mathcal{X}^n \to \mathbb{R}$. If the function g satisfies $\sup_{X_1,\ldots,X_n,\hat{X}_k} |g(X_1,\ldots,X_n) - g(X_1,\ldots,\hat{X}_k,\ldots,X_n)| \leq c_k$, for all $1 \leq k \leq n$ then, for any $\varepsilon > 0$:

$$\Pr\left[g(X_1,\ldots,X_n) - \mathbf{E}[g(X_1,\ldots,X_n)] \geq \varepsilon\right] \leq \exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^n c_i^2}\right), \tag{32}$$

$$\Pr\left[\mathbf{E}[g(X_1,\ldots,X_n)] - g(X_1,\ldots,X_n) \geq \varepsilon\right] \leq \exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^n c_i^2}\right),$$

where the expectations are over the random draws of $X_1,\ldots,X_n$. Here the constants $c_1, c_2, \cdots, c_n$ are called Lipschitz constants.

Appendix B. Lipschitz Constants for Section 4.6

Lemma 19 *The upper bound on $\hat{R}(\mathcal{G}^U_{B,D})$, namely $T_1(\mathbf{U},\mathbf{S})$, admits the Lipschitz constant:*

$$\frac{2\sqrt{2B}}{\bar{D}n}\left(\sqrt{\sum_{i=1}^n \mathbf{x}_i^\top \mathbf{x}_i} - \sqrt{\sum_{i=1}^n \mathbf{x}_i^\top \mathbf{x}_i - \frac{DR^2\mu_{max}}{n\bar{D}+DR^2}}\right).$$

Proof The quantity of interest is the worst change in

$$\frac{2\sqrt{2B}}{n}\sqrt{\sum_{i=1}^n \mathbf{x}_i(\bar{D}\mathbf{I} + \frac{D}{n}\sum_{j=1}^n \mathbf{u}_j\mathbf{u}_j^\top)^{-1}\mathbf{x}_i^\top}$$

when $\mathbf{u}_k$ is varied for any setting of $\mathbf{u}_1, \ldots, \mathbf{u}_{k-1}, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_n$. Since $\sum_{j=1, j\neq k}^{n} \mathbf{u}_j \mathbf{u}_j^\top$ is positive semi-definite and inside the inverse operator, $\mathbf{u}_k$ will have the most extreme effect on the expression when $\sum_{j=1, j\neq k}^{n} \mathbf{u}_j \mathbf{u}_j^\top = \mathbf{0}$. Thus, consider:

$$\frac{2\sqrt{2B}}{n} \sqrt{\sum_{i=1}^{n} \mathbf{x}_i^\top \left(\bar{D}\mathbf{I} + \frac{D}{n} \mathbf{u}_k \mathbf{u}_k^\top \right)^{-1} \mathbf{x}_i}.$$

Apply the Woodbury matrix inversion identity to the term inside the square root:

$$\begin{aligned} \sum_{i=1}^{n} \mathbf{x}_i^\top \left(\bar{D}\mathbf{I} + \frac{D}{n} \mathbf{u}_k \mathbf{u}_k^\top \right)^{-1} \mathbf{x}_i &= \frac{1}{\bar{D}} \sum_{i=1}^{n} \mathbf{x}_i^\top \left(\mathbf{I} - \frac{\mathbf{u}_k \mathbf{u}_k^\top}{\frac{n\bar{D}}{D} + \mathbf{u}_k^\top \mathbf{u}_k} \right) \mathbf{x}_i \\ &= \frac{1}{\bar{D}} \left(\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \frac{\sum_{i=1}^{n} (\mathbf{x}_i^\top \mathbf{u}_k)^2}{\frac{n\bar{D}}{D} + \mathbf{u}_k^\top \mathbf{u}_k} \right). \end{aligned}$$

The maximum value of this expression occurs when $\mathbf{u}_k = \mathbf{0}$. To find the minimum, write the second term inside the brackets in the above expression as below:

$$\left(\frac{\mathbf{u}_k^\top}{\|\mathbf{u}_k\|} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|} \right) / \left(\frac{n\bar{D}}{D\mathbf{u}_k^\top \mathbf{u}_k} + 1 \right).$$

Clearly, in the numerator, the magnitude of $\mathbf{u}_k$ does not matter. To maximize this expression, $\mathbf{u}_k$ should be set to a vector of maximal length and in the same direction as the maximum eigenvector of $\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top$. Since all examples are assumed to have bounded norm no larger than R, the largest $\mathbf{u}_k$ vector has norm R. Denoting the maximum eigenvalue of $\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top$ by μ_{max}, it is easy to show the claimed value of Lipschitz constant for any k. ■

Lemma 20 *The upper bound on $\hat{R}(\mathcal{H}_{E,D}^{V})$, namely $T_2(\mathbf{V}, \mathbf{S})$, admits the Lipschitz constant:*

$$\frac{2\sqrt{2E}}{\bar{D}n} \left(\sqrt{\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i} - \sqrt{\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \frac{DR^2 \mu_{max}}{n\bar{D} + DR^2}} \right).$$

Proof The quantity of interest is the maximum change in the following optimization problem over $\mathbf{u}_k$ for any setting of $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{k-1}, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_{n_v}$:

$$\min_{\lambda \geq 0} \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^\top \left(\bar{D} \sum_{j=1}^{n_v} \lambda_j \mathbf{I} + D \sum_{j=1}^{n_v} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top \right)^{-1} \mathbf{x}_i + \frac{2}{n} E \sum_{i=1}^{n_v} \lambda_i.$$

As before, this happens when all $\mathbf{u}$'s except $\mathbf{u}_k$ are $\mathbf{0}$. In such a scenario, the expression is minimized for the setting $\lambda_j = 0$ for all $j \neq k$. The minimization only needs to consider variable settings of λ_k. Since this minimization is over a single scalar, it is possible to obtain a closed-form expression for λ_k. The optimal λ_k is merely: $\frac{1}{\sqrt{2E}} \sum_{i=1}^{n} \mathbf{x}_i^\top \left(\bar{D}\mathbf{I} + D\mathbf{u}_k \mathbf{u}_k^\top \right)^{-1} \mathbf{x}_i$. Substituting this into the objective gives an expression which is independent of λ's:

$$\frac{2\sqrt{2E}}{n} \sqrt{\sum_{i=1}^{n} \mathbf{x}_i^\top \left(\bar{D}\mathbf{I} + \frac{D}{n} \mathbf{u}_k \mathbf{u}_k^\top \right)^{-1} \mathbf{x}_i}.$$

This expression is identical to the one obtained in Theorem 19 and the proof follows. ■

Appendix C. Solving for n_v

Let $x = \frac{1}{\sqrt{n_v}}$, $c = 4R\sqrt{\frac{2E}{D}}$ and $b = \frac{3}{2}\sqrt{\frac{\ln(2/\delta)}{2}}$. Consider solving for x in the expression $x^2 - 2bx = (c + 2b)/\sqrt{n}$. Equivalently, solve $(x-b)^2 = b^2 + (c+2b)/\sqrt{n}$. Taking the square root of both sides gives $x = b \pm \sqrt{b^2 + (c+2b)/\sqrt{n}}$. Since $x > 0$, only the positive root is considered. Thus, $\sqrt{n_v} = 1/(b + \sqrt{b^2 + (c+2b)/\sqrt{n}})$ which gives an exact expression for n_v. Dropping terms from the denominator produces the simpler expression: $\sqrt{n_v} \leq 1/\sqrt{(c+2b)/\sqrt{n}}$. Hence, $n_v \leq \frac{\sqrt{n}}{4R\sqrt{\frac{2E}{D}} + 3\sqrt{\frac{\ln(2/\delta)}{2}}}$.

References

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

M. Belkin, P. Niyogi, and V. Sindhwani. On manifold regularization. In *Artificial Intelligence and Statistics*, 2005.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, Cambridge, MA, 2007.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Lecture Notes in Artificial Intelligence 3176*, pages 169–207, Heidelberg, Germany, 2004. Springer.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2003.

G. C. Cawley and N. L. C Talbot. Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers. *Pattern Recognition*, 36:2585–2592, 2003.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. *SIAM J. Comput.*, 34(3):640–668, 2005.

M. Collins and B. Roark. Incremental parsing with the perceptron algorithm. In *Annual Meeting of the Association for Computational Linguistics*, pages 111–118, 2004.

K. Crammer, M. Dredze, and F. Pereira. Exact convex confidence-weighted learning. In *Advances in Neural Information Processing Systems 21*, Cambridge, MA, 2009a. MIT Press.

K. Crammer, M. Mohri, and F. Pereira. Gaussian margin machines. In *Proceedings of the Artificial Intelligence and Statistics*, 2009b.

D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, pages 161–190, 2002.

M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *Internationcal Conference on Machine Leaarning*, 2008.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

P. Haffner. Escaping the convex hull with extrapolated vector machines. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 753–760. MIT Press, Cambridge, MA, 2001.

R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines. *Journal of Machine Learning Research*, 1:245–279, 2001.

T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems 11*, 1999.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, 1999.

T. Joachims. Training linear SVMs in linear time. In *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD)*, pages 217–226, 2006.

T. Joachims. Making large-scale support vector machine learning practical. In A. Smola B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

S.S. Keerthi. Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13:1225–1229, 2002.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30, 2002.

Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

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.

S. Mika, G. Raetsch, J. Weston, B. Scholkopf, and K.-R. Muller. Fisher discriminant analysis with kernels. In *in Neural Networks for Signal Processing IX*, pages 41–48, 1999.

G. Raetsch, T. Onoda, and K.-R. Muller. Soft margins for adaboost. *Machine Learning*, 43:287–320, 2001.

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*, 26:322–330, 1998.

B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, USA, 2002.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *International Conference on Machine Learning*, 2007.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

P. K. Shivaswamy and T. Jebara. Ellipsoidal kernel machines. In *Proceedings of the Artificial Intelligence and Statistics*, 2007.

P. K. Shivaswamy and T. Jebara. Relative margin machines. In *Advances in Neural Information Processing Systems 21*, Cambridge, MA, 2009a. MIT Press.

P. K. Shivaswamy and T. Jebara. Structured prediction with relative margin. In *International Conference on Machine Learning and Applications*, 2009b.

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.

F. Sinz, O. Chapelle, A. Agarwal, and B. Schölkopf. An analysis of inference with the universum. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1369–1376. MIT Press, Cambridge, MA, 2008.

N. Srebro, J. D. M. Rennie, and T. S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, pages 1329–1336. MIT Press, 2005.

B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2004.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, September 2005.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 668–674. MIT Press, Cambridge, MA, 2000.

J. Weston, R. Collobert, F. H. Sinz, L. Bottou, and V. Vapnik. Inference with the universum. In *Proceedings of the International Conference on Machine Learning*, pages 1009–1016, 2006.

B. Zhang, X. Chen, S. Shan, and W. Gao. Nonlinear face recognition based on maximum average margin criterion. In *Computer Vision and Pattern Recognition*, pages 554–559, 2005.

Journal of Machine Learning Research 11 (2010) 789-814 Submitted 11/08; Revised 1/10; Published 2/10

Stability Bounds for Stationary φ-mixing and β-mixing Processes

Mehryar Mohri MOHRI@CIMS.NYU.EDU
Courant Institute of Mathematical Sciences
and Google Research
251 Mercer Street
New York, NY 10012

Afshin Rostamizadeh ROSTAMI@CS.NYU.EDU
Department of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, NY 10012

Editor: Ralf Herbrich

Abstract

Most generalization bounds in learning theory are based on some measure of the complexity of the hypothesis class used, independently of any algorithm. In contrast, the notion of algorithmic stability can be used to derive tight generalization bounds that are tailored to specific learning algorithms by exploiting their particular properties. However, as in much of learning theory, existing stability analyses and bounds apply only in the scenario where the samples are independently and identically distributed. In many machine learning applications, however, this assumption does not hold. The observations received by the learning algorithm often have some inherent temporal dependence.

This paper studies the scenario where the observations are drawn from a stationary φ-mixing or β-mixing sequence, a widely adopted assumption in the study of non-i.i.d. processes that implies a dependence between observations weakening over time. We prove novel and distinct stability-based generalization bounds for stationary φ-mixing and β-mixing sequences. These bounds strictly generalize the bounds given in the i.i.d. case and apply to all stable learning algorithms, thereby extending the use of stability-bounds to non-i.i.d. scenarios.

We also illustrate the application of our φ-mixing generalization bounds to general classes of learning algorithms, including Support Vector Regression, Kernel Ridge Regression, and Support Vector Machines, and many other kernel regularization-based and relative entropy-based regularization algorithms. These novel bounds can thus be viewed as the first theoretical basis for the use of these algorithms in non-i.i.d. scenarios.

Keywords: learning in non-i.i.d. scenarios, weakly dependent observations, mixing distributions, algorithmic stability, generalization bounds, learning theory

1. Introduction

Most generalization bounds in learning theory are based on some measure of the complexity of the hypothesis class used, such as the VC-dimension, covering numbers, or Rademacher complexity. These measures characterize a class of hypotheses, independently of any algorithm. In contrast, the notion of algorithmic stability can be used to derive bounds that are tailored to specific learning algorithms and exploit their particular properties. A learning algorithm is stable if the hypothesis it

outputs varies in a limited way in response to small changes made to the training set. Algorithmic stability has been used effectively in the past to derive tight generalization bounds (Bousquet and Elisseeff, 2001, 2002; Kearns and Ron, 1997).

But, as in much of learning theory, existing stability analyses and bounds apply only in the scenario where the samples are independently and identically distributed (i.i.d.). In many machine learning applications, this assumption, however, does not hold; in fact, the i.i.d. assumption is not tested or derived from any data analysis. The observations received by the learning algorithm often have some inherent temporal dependence. This is clear in system diagnosis or time series prediction problems. Clearly, prices of different stocks on the same day, or of the same stock on different days, may be dependent. But, a less apparent time dependency may affect data sampled in many other tasks as well.

This paper studies the scenario where the observations are drawn from a stationary φ-mixing or β-mixing sequence, a widely adopted assumption in the study of non-i.i.d. processes that implies a dependence between observations weakening over time (Yu, 1994; Meir, 2000; Vidyasagar, 2003; Lozano et al., 2006; Mohri and Rostamizadeh, 2007). We prove novel and distinct stability-based generalization bounds for stationary φ-mixing and β-mixing sequences. These bounds strictly generalize the bounds given in the i.i.d. case and apply to all stable learning algorithms, thereby extending the usefulness of stability-bounds to non-i.i.d. scenarios. Our proofs are based on the independent block technique described by Yu (1994) and attributed to Bernstein (1927), which is commonly used in such contexts. However, our analysis somewhat differs from previous uses of this technique in that the blocks of points we consider are not necessarily of equal size.

For our analysis of stationary φ-mixing sequences, we make use of a generalized version of McDiarmid's inequality given by Kontorovich and Ramanan (2008) that holds for φ-mixing sequences. This leads to stability-based generalization bounds with the standard exponential form. Our generalization bounds for stationary β-mixing sequences cover a more general non-i.i.d. scenario and use the standard McDiarmid's inequality, however, unlike the φ-mixing case, the β-mixing bound presented here is not a purely exponential bound and contains an additive term depending on the mixing coefficient.

We also illustrate the application of our φ-mixing generalization bounds to general classes of learning algorithms, including Support Vector Regression (SVR) (Vapnik, 1998), Kernel Ridge Regression (Saunders et al., 1998), and Support Vector Machines (SVMs) (Cortes and Vapnik, 1995). Algorithms such as SVR have been used in the context of time series prediction in which the i.i.d. assumption does not hold, some with good experimental results (Müller et al., 1997; Mattera and Haykin, 1999). However, to our knowledge, the use of these algorithms in non-i.i.d. scenarios has not been previously supported by any theoretical analysis. The stability bounds we give for SVR, SVMs, and many other kernel regularization-based and relative entropy-based regularization algorithms can thus be viewed as the first theoretical basis for their use in such scenarios.

The following sections are organized as follows. In Section 2, we introduce the definitions relevant to the non-i.i.d. problems that we are considering and discuss the learning scenarios in that context. Section 3 gives our main generalization bounds for stationary φ-mixing sequences based on stability, as well as the illustration of its applications to general kernel regularization-based algorithms, including SVR, KRR, and SVMs, as well as to relative entropy-based regularization algorithms. Finally, Section 4 presents the first known stability bounds for the more general stationary β-mixing scenario.

2. Preliminaries

We first introduce some standard definitions for dependent observations in mixing theory (Doukhan, 1994) and then briefly discuss the learning scenarios in the non-i.i.d. case.

2.1 Non-i.i.d. Definitions

Definition 1 *A sequence of random variables $\mathbf{Z} = \{Z_t\}_{t=-\infty}^{\infty}$ is said to be* stationary *if for any t and non-negative integers m and k, the random vectors $(Z_t, \ldots, Z_{t+m})$ and $(Z_{t+k}, \ldots, Z_{t+m+k})$ have the same distribution.*

Thus, the index t or time, does not affect the distribution of a variable Z_t in a stationary sequence. This does not imply independence however. In particular, for $i < j < k$, $\Pr[Z_j \mid Z_i]$ may not equal $\Pr[Z_k \mid Z_i]$, that is, conditional probabilities may vary at different points in time. The following is a standard definition giving a measure of the dependence of the random variables Z_t within a stationary sequence. There are several equivalent definitions of these quantities, we are adopting here a version convenient for our analysis, as in Yu (1994).

Definition 2 *Let $\mathbf{Z} = \{Z_t\}_{t=-\infty}^{\infty}$ be a stationary sequence of random variables. For any $i, j \in \mathbb{Z} \cup \{-\infty, +\infty\}$, let σ_i^j denote the σ-algebra generated by the random variables Z_k, $i \leq k \leq j$. Then, for any positive integer k, the β-mixing and φ-mixing coefficients of the stochastic process $\mathbf{Z}$ are defined as*

$$\beta(k) = \sup_n \mathop{\mathrm{E}}_{B \in \sigma_{-\infty}^n} \Big[\sup_{A \in \sigma_{n+k}^{\infty}} \Big| \Pr[A \mid B] - \Pr[A] \Big| \Big] \quad \varphi(k) = \sup_{\substack{n \\ A \in \sigma_{n+k}^{\infty} \\ B \in \sigma_{-\infty}^n}} \Big| \Pr[A \mid B] - \Pr[A] \Big|.$$

$\mathbf{Z}$ is said to be β-mixing (φ-mixing) if $\beta(k) \to 0$ (resp. $\varphi(k) \to 0$) as $k \to \infty$. It is said to be algebraically β-mixing (algebraically φ-mixing*) if there exist real numbers $\beta_0 > 0$ (resp. $\varphi_0 > 0$) and $r > 0$ such that $\beta(k) \leq \beta_0 / k^r$ (resp. $\varphi(k) \leq \varphi_0 / k^r$) for all k,* exponentially mixing *if there exist real numbers β_0 (resp. $\varphi_0 > 0$), β_1 (resp. $\varphi_1 > 0$) and $r > 0$ such that $\beta(k) \leq \beta_0 \exp(-\beta_1 k^r)$ (resp. $\varphi(k) \leq \varphi_0 \exp(-\varphi_1 k^r)$) for all k.*

Both $\beta(k)$ and $\varphi(k)$ measure the dependence of an event on those that occurred more than k units of time in the past. β-mixing is a weaker assumption than φ-mixing and thus covers a more general non-i.i.d. scenario.

This paper gives stability-based generalization bounds both in the φ-mixing and β-mixing case. The β-mixing bounds cover a more general case of course, however, the φ-mixing bounds are simpler and admit the standard exponential form. The φ-mixing bounds are based on a concentration inequality that applies to φ-mixing processes only. Except for the use of this concentration bound and two lemmas 5 and 6, all of the intermediate proofs and results to derive a φ-mixing bound in Section 3 are given in the more general case of β-mixing sequences.

It has been argued by Vidyasagar (2003) that β-mixing is "just the right" assumption for the analysis of weakly-dependent sample points in machine learning, in particular because several PAC-learning results then carry over to the non-i.i.d. case. Our β-mixing generalization bounds further contribute to the analysis of this scenario.[1]

1. Some results have also been obtained in the more general context of α-mixing but they seem to require the stronger condition of exponential mixing (Modha and Masry, 1998).

We describe in several instances the application of our bounds in the case of algebraic mixing. Algebraic mixing is a standard assumption for mixing coefficients that has been adopted in previous studies of learning in the presence of dependent observations (Yu, 1994; Meir, 2000; Vidyasagar, 2003; Lozano et al., 2006). Let us also point out that mixing assumptions can be checked in some cases such as with Gaussian or Markov processes (Meir, 2000) and that mixing parameters can also be estimated in such cases.

Most previous studies use a technique originally introduced by Bernstein (1927) based on *independent blocks* of equal size (Yu, 1994; Meir, 2000; Lozano et al., 2006). This technique is particularly relevant when dealing with stationary β-mixing. We will need a related but somewhat different technique since the blocks we consider may not have the same size. The following lemma is a special case of Corollary 2.7 from Yu (1994).

Lemma 3 (Yu, 1994, Corollary 2.7) *Let $\mu \geq 1$ and suppose that h is measurable function, with absolute value bounded by M, on a product probability space $\left(\prod_{j=1}^{\mu} \Omega_j, \prod_{i=1}^{\mu} \sigma_{r_i}^{s_i}\right)$ where $r_i \leq s_i \leq r_{i+1}$ for all i. Let Q be a probability measure on the product space with marginal measures Q_i on $(\Omega_i, \sigma_{r_i}^{s_i})$, and let Q^{i+1} be the marginal measure of Q on $\left(\prod_{j=1}^{i+1} \Omega_j, \prod_{j=1}^{i+1} \sigma_{r_j}^{s_j}\right)$, $i = 1, \ldots, \mu-1$. Let $\beta(Q) = \sup_{1 \leq i \leq \mu-1} \beta(k_i)$, where $k_i = r_{i+1} - s_i$, and $P = \prod_{i=1}^{\mu} Q_i$. Then,*

$$|\mathop{\mathrm{E}}_{Q}[h] - \mathop{\mathrm{E}}_{P}[h]| \leq (\mu - 1) M \beta(Q).$$

The lemma gives a measure of the difference between the distribution of μ blocks where the blocks are independent in one case and dependent in the other case. The distribution within each block is assumed to be the same in both cases. For a monotonically decreasing function β, we have $\beta(Q) = \beta(k^*)$, where $k^* = \min_i(k_i)$ is the smallest gap between blocks.

2.2 Learning Scenarios

We consider the familiar supervised learning setting where the learning algorithm receives a sample of m labeled points $S = (z_1, \ldots, z_m) = ((x_1, y_1), \ldots, (x_m, y_m)) \in (X \times Y)^m$, where X is the input space and Y the set of labels ($Y \subseteq \mathbb{R}$ in the regression case), both assumed to be measurable.

For a fixed learning algorithm, we denote by h_S the hypothesis it returns when trained on the sample S. The error of a hypothesis on a pair $z \in X \times Y$ is measured in terms of a cost function $c : Y \times Y \to \mathbb{R}_+$. Thus, $c(h(x), y)$ measures the error of a hypothesis h on a pair (x, y), $c(h(x), y) = (h(x) - y)^2$ in the standard regression cases. We will often use the shorthand $c(h, z) := c(h(x), y)$ for a hypothesis h and $z = (x, y) \in X \times Y$ and will assume that c is upper bounded by a constant $M > 0$. We denote by $\widehat{R}(h)$ the empirical error of a hypothesis h for a training sample $S = (z_1, \ldots, z_m)$:

$$\widehat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} c(h, z_i).$$

In the standard machine learning scenario, the sample pairs $z_1, \ldots, z_m$ are assumed to be i.i.d., a restrictive assumption that does not always hold in practice. We will consider here the more general case of dependent samples drawn from a stationary mixing sequence $\mathbf{Z}$ over $X \times Y$. As in the i.i.d. case, the objective of the learning algorithm is to select a hypothesis with small error over future samples. But, here, we must distinguish two versions of this problem.

In the most general version, future samples depend on the training sample S and thus the generalization error or true error of the hypothesis h_S trained on S must be measured by its expected error conditioned on the sample S:

$$R(h_S) = \mathop{\mathrm{E}}_{z}[c(h_S, z) \mid S]. \tag{1}$$

This is the most realistic setting in this context, which matches time series prediction problems. A somewhat less realistic version is one where the samples are dependent, but the test points are assumed to be independent of the training sample S. The generalization error of the hypothesis h_S trained on S is then:

$$R(h_S) = \mathop{\mathrm{E}}_{z}[c(h_S, z) \mid S] = \mathop{\mathrm{E}}_{z}[c(h_S, z)].$$

This setting seems less natural since, if samples are dependent, future test points must also depend on the training points, even if that dependence is relatively weak due to the time interval after which test points are drawn. Nevertheless, it is this somewhat less realistic setting that has been studied by all previous machine learning studies that we are aware of Yu (1994), Meir (2000), Vidyasagar (2003) and Lozano et al. (2006), even when examining specifically a time series prediction problem (Meir, 2000). Thus, the bounds derived in these studies cannot be directly applied to the more general setting. We will consider instead the most general setting with the definition of the generalization error based on Equation 1. Clearly, our analysis also applies to the less general setting just discussed as well.

Let us also briefly discuss the more general scenario of *non-stationary* mixing sequences, that is one where the distribution may change over time. Within that general case, the generalization error of a hypothesis h_S, defined straightforwardly by

$$R(h_S, t) = \mathop{\mathrm{E}}_{z_t \sim \sigma'_t}[c(h_S, z_t) \mid S],$$

would depend on the time t and it may be the case that $R(h_S,t) \neq R(h_S,t')$ for $t \neq t'$, making the definition of the generalization error a more subtle issue. To remove the dependence on time, one could define a weaker notion of the generalization error based on an expected loss over all time:

$$R(h_S) = \mathop{\mathrm{E}}_{t}[R(h_S, t)].$$

It is not clear however whether this term could be easily computed and be useful. A stronger condition would be to minimize the generalization error for any particular target time. Studies of this type have been conducted for smoothly changing distributions, such as in Zhou et al. (2008), however, to the best of our knowledge, the scenario of a both non-identical and non-independent sequences has not yet been studied.

3. φ-Mixing Generalization Bounds and Applications

This section gives generalization bounds for $\widehat{\beta}$-stable algorithms over a mixing stationary distribution.[2] The first two sections present our supporting lemmas which hold for either β-mixing or φ-mixing stationary distributions. In the third section, we will briefly discuss concentration inequalities that apply to φ-mixing processes only. Then, in the final section, we will present our main results.

2. The standard variable used for the stability coefficient is β. To avoid the confusion with the β-mixing coefficient, we will use $\widehat{\beta}$ instead.

The condition of $\widehat{\beta}$-stability is an algorithm-dependent property first introduced by Devroye and Wagner (1979) and Kearns and Ron (1997). It has been later used successfully by Bousquet and Elisseeff (2001, 2002) to show algorithm-specific stability bounds for i.i.d. samples. Roughly speaking, a learning algorithm is said to be *stable* if small changes to the training set do not cause large deviations in its output. The following gives the precise technical definition.

Definition 4 *A learning algorithm is said to be (uniformly)* $\widehat{\beta}$-stable *if the hypotheses it returns for any two training samples S and S' that differ by removing a single point satisfy*

$$\forall z \in X \times Y, \quad |c(h_S, z) - c(h_{S'}, z)| \leq \widehat{\beta}.$$

We note that a $\widehat{\beta}$-stable algorithm is also stable with respect to *replacing* a single point. Let S and S_i be two sequences differing in the ith coordinate, and $S_{/i}$ be equivalent to S and S_i but with the ith point removed. Then for a $\widehat{\beta}$-stable algorithm we have,

$$\begin{aligned} |c(h_S, z) - c(h_{S_i}, z)| &= |c(h_S, z) - c(h_{S_{/i}}) + c(h_{S_{/i}}) - c(h_{S_i}, z)| \\ &\leq |c(h_S, z) - c(h_{S_{/i}})| + |c(h_{S_{/i}}) - c(h_{S_i}, z)| \\ &\leq 2\widehat{\beta}. \end{aligned}$$

The use of stability in conjunction with McDiarmid's inequality will allow us to derive generalization bounds. McDiarmid's inequality is an exponential concentration bound of the form

$$\Pr[|\Phi - \mathrm{E}[\Phi]| \geq \varepsilon] \leq \exp\left(-\frac{m\varepsilon^2}{\tau^2}\right),$$

where the probability is over a sample of size m and where $\frac{\tau}{m}$ is the Lipschitz parameter of Φ, with τ a function of m. Unfortunately, this inequality cannot be applied when the sample points are not distributed in an i.i.d. fashion. We will use instead a result of Kontorovich and Ramanan (2008) that extends McDiarmid's inequality to φ-mixing distributions (Theorem 8). To obtain a stability-based generalization bound, we will apply this theorem to

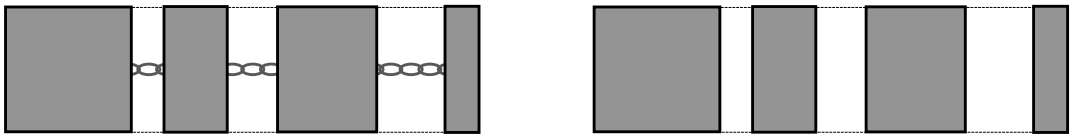
$$\Phi(S) = R(h_S) - \widehat{R}(h_S).$$

To do so, we need to show, as with the standard McDiarmid's inequality, that Φ is a Lipschitz function and, to make it useful, bound $\mathrm{E}[\Phi]$. The next two sections describe how we achieve both of these in this non-i.i.d. scenario.

Let us first take a brief look at the problem faced when attempting to give stability bounds for dependent sequences and give some idea of our solution for that problem. The stability proofs given by Bousquet and Elisseeff (2001) assume the i.i.d. property, thus replacing an element in a sequence with another does not affect the expected value of a random variable defined over that sequence. In other words, the following equality holds,

$$\mathop{\mathrm{E}}_{S}[V(Z_1, \ldots, Z_i, \ldots, Z_m)] = \mathop{\mathrm{E}}_{S, Z'}[V(Z_1, \ldots, Z', \ldots, Z_m)], \tag{2}$$

for a random variable V that is a function of the sequence of random variables $S = (Z_1, \ldots, Z_m)$. However, clearly, if the points in that sequence S are dependent, this equality may not hold anymore.



(a) (b)

Figure 1: Illustration of dependent (a) and independent (b) blocks. Although there is no dependence between blocks of points in (b), the distribution within each block remains the same as in (a) and thus points within a block remain dependent.

The main technique to cope with this problem is based on the so-called "independent block sequence" originally introduced by Bernstein (1927). This consists of eliminating from the original dependent sequence several blocks of contiguous points, leaving us with some remaining blocks of points. Instead of these dependent blocks, we then consider independent blocks of points, each with the same size and the same distribution (within each block) as the dependent ones. By Lemma 3, for a β-mixing distribution, the expected value of a random variable defined over the dependent blocks is close to the one based on these independent blocks. Working with these independent blocks brings us back to a situation similar to the i.i.d. case, with i.i.d. blocks replacing i.i.d. points. Figure 1 illustrates the two types of blocks just discussed.

Our use of this method somewhat differs from previous ones (see Yu, 1994; Meir, 2000) where many blocks of equal size are considered. We will be dealing with four blocks and with typically unequal sizes. More specifically, note that for Equation 2 to hold, we only need that the variable Z_i be independent of the other points in the sequence. To achieve this, roughly speaking, we will be "discarding" some of the points in the sequence surrounding Z_i. This results in a sequence of three blocks of contiguous points. If our algorithm is stable and we do not discard too many points, the hypothesis returned should not be greatly affected by this operation. In the next step, we apply the independent block lemma, which then allows us to assume each of these blocks as independent modulo the addition of a mixing term. In particular, Z_i becomes independent of all other points. Clearly, the number of points discarded is subject to a trade-off: removing too many points could excessively modify the hypothesis returned; removing too few would maintain the dependency between Z_i and the remaining points, thereby inducing a larger penalty when applying Lemma 3. This trade-off is made explicit in the following section where an optimal solution is sought.

3.1 Lipschitz Bound

As discussed in Section 2.2, in the most general scenario, test points depend on the training sample. We first present a lemma that relates the expected value of the generalization error in that scenario and the same expectation in the scenario where the test point is independent of the training sample. We denote by $R(h_S) = \mathrm{E}_z[c(h_S, z)|S]$ the expectation in the dependent case and by $\widetilde{R}(h_{S_b}) = \mathrm{E}_{\widetilde{z}}[c(h_{S_b}, \widetilde{z})]$ the expectation where the test points are assumed independent of the training, with S_b denoting a sequence similar to S but with the last b points removed. Figure 2(a) illustrates that sequence. The block S_b is assumed to have exactly the same distribution as the corresponding block of the same size in S.

Lemma 5 *Assume that the learning algorithm is $\widehat{\beta}$-stable and that the cost function c is bounded by M. Then, for any sample S of size m drawn from a φ-mixing stationary distribution and for any $b \in \{0,\ldots,m\}$, the following holds:*

$$|R(h_S) - \widetilde{R}(h_{S_b})| \leq b\widehat{\beta} + M\varphi(b)$$

Proof The $\widehat{\beta}$-stability of the learning algorithm implies that

$$|R(h_S) - R(h_{S_b})| = |\mathop{\mathrm{E}}_{z}[c(h_S,z)|S] - \mathop{\mathrm{E}}_{z}[c(h_{S_b},z)|S_b]| \leq b\widehat{\beta}. \tag{3}$$

Now, in order to remove the dependence on S_b we bound the following difference

$$\begin{aligned}
&|\mathop{\mathrm{E}}_{z}[c(h_{S_b},z)|S_b] - \mathop{\mathrm{E}}_{\widetilde{z}}[c(h_{S_b},\widetilde{z})]| \\
&\qquad = \Big|\sum_{z\in Z} c(h_{S_b},z)(\Pr[z|S_b] - \Pr[z])\Big| \\
&\qquad = \Big|\sum_{z\in Z^+} c(h_{S_b},z)(\Pr[z|S_b] - \Pr[z]) + \sum_{z\in Z^-} c(h_{S_b},z)(\Pr[z|S_b] - \Pr[z])\Big| \\
&\qquad = \Big|\sum_{z\in Z^+} c(h_{S_b},z)\Big|\Pr[z|S_b] - \Pr[z]\Big| - \sum_{z\in Z^-} c(h_{S_b},z)\Big|\Pr[z|S_b] - \Pr[z]\Big|\Big| \\
&\qquad \leq \max_{Z\in\{Z^-,Z^+\}} \sum_{z\in Z} c(h_{S_b},z)\Big|\Pr[z|S_b] - \Pr[z]\Big| \qquad (4) \\
&\qquad \leq \max_{Z\in\{Z^-,Z^+\}} M\sum_{z\in Z}\Big|\Pr[z|S_b] - \Pr[z]\Big| \\
&\qquad = \max_{Z\in\{Z^-,Z^+\}} M\Big|\sum_{z\in Z}\Pr[z|S_b] - \Pr[z]\Big| \\
&\qquad = \max_{Z\in\{Z^-,Z^+\}} M\Big|\Pr[Z|S_b] - \Pr[Z]\Big| \leq M\varphi(b),
\end{aligned}$$

where the sum has been separated over the set of zs Z^+ for which the difference $\Pr[z|S_b] - \Pr[z]$ is non-negative, and its complement Z^-. Using (3) and (4) and the triangle inequality yields the statement of the lemma. ■

Note that we assume that z immediately follows the sample S, which is the strongest dependent scenario. The following bounds can be improved in a straightforward manner if the test point z is assumed to be observed say k units of time after the sample S. The bound would then contain the mixing term $\varphi(k+b)$ instead of $\varphi(b)$.

We can now prove a Lipschitz bound for the function Φ.

Lemma 6 *Let $S = (z_1,\ldots,z_i,\ldots,z_m)$ and $S^i = (z_1,\ldots,z_i',\ldots,z_m)$ be two sequences drawn from a φ-mixing stationary process that differ only in point z_i for some $i \in \{1,\ldots,m\}$, and let h_S and h_{S^i} be the hypotheses returned by a $\widehat{\beta}$-stable algorithm when trained on each of these samples. Then, for any $i \in \{1,\ldots,m\}$, the following inequality holds:*

$$|\Phi(S) - \Phi(S^i)| \leq (b+2)2\widehat{\beta} + 2\varphi(b)M + \frac{M}{m}.$$

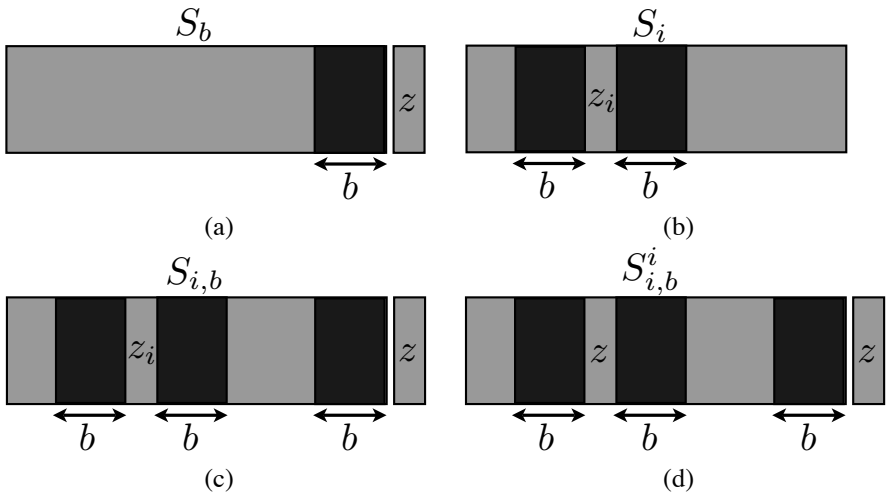


Figure 2: Illustration of the sequences derived from S that are considered in the proofs.

Proof To prove this inequality, we first bound the difference of the empirical errors as in Bousquet and Elisseeff (2002), then the difference of the generalization errors. Bounding the difference of costs on agreeing points with $\widehat{\beta}$ and the one that disagrees with M gives

$$\begin{aligned}|\widehat{R}(h_S) - \widehat{R}(h_{S^i})| &\leq \frac{1}{m}\sum_{j\neq i}|c(h_S, z_j) - c(h_{S^i}, z_j)| + \frac{1}{m}|c(h_S, z_i) - c(h_{S^i}, z_i')| \\ &\leq 2\widehat{\beta} + \frac{M}{m}. \end{aligned} \tag{5}$$

Since both $R(h_S)$ and $R(h_{S^i})$ are defined with respect to a (different) dependent point, we can apply Lemma 5 to both generalization error terms and use $\widehat{\beta}$-stability. Using this and the triangle inequality, we can write

$$\begin{aligned}|R(h_S) - R(h_{S^i})| &\leq |R(h_S) - \widetilde{R}(h_{S_b}) + \widetilde{R}(h_{S_b}) - \widetilde{R}(h_{S_b^i}) + \widetilde{R}(h_{S_b^i}) - R(h_{S^i})| \\ &\leq |\widetilde{R}(h_{S_b}) - \widetilde{R}(h_{S_b^i})| + 2b\widehat{\beta} + 2\varphi(b)M \\ &= \mathop{\mathrm{E}}_{\widetilde{z}}[c(h_{S_b}, \widetilde{z}) - c(h_{S_b^i}, \widetilde{z})] + 2b\widehat{\beta} + 2\varphi(b)M \\ &\leq 2\widehat{\beta} + 2b\widehat{\beta} + 2\varphi(b)M. \end{aligned} \tag{6}$$

The statement of the lemma is obtained by combining inequalities 5 and 6. ■

3.2 Bound on Expectation

As mentioned earlier, to obtain an explicit bound after application of a generalized McDiarmid's inequality, we also need to bound $\mathrm{E}_S[\Phi(S)]$. This is done by analyzing independent blocks using Lemma 3.

Lemma 7 *Let h_S be the hypothesis returned by a $\widehat{\beta}$-stable algorithm trained on a sample S drawn from a β-mixing stationary distribution. Then, for all $b \in [1,m]$, the following inequality holds:*

$$\mathop{\mathrm{E}}_{S}[|\Phi(S)|] \leq (6b+2)\widehat{\beta} + 3\beta(b)M.$$

Proof Let S_b be defined as in the proof of Lemma 5. To deal with independent block sequences defined with respect to the same hypothesis, we will consider the sequence $S_{i,b} = S_i \cap S_b$, which is illustrated by Figure 2(a-c). This can result in as many as four blocks. As before, we will consider a sequence $\widetilde{S}_{i,b}$ with a similar set of blocks each with the same distribution as the corresponding blocks in $S_{i,b}$, but such that the blocks are independent as seen in Figure 2(d).

Since three blocks of at most b points are removed from each hypothesis, by the $\widehat{\beta}$-stability of the learning algorithm, the following holds:

$$\begin{aligned}
\mathop{\mathrm{E}}_{S}[\Phi(S)] &= \mathop{\mathrm{E}}_{S}[\widehat{R}(h_S) - R(h_S)] \\
&= \mathop{\mathrm{E}}_{S,z}\left[\frac{1}{m}\sum_{i=1}^{m} c(h_S, z_i) - c(h_S, z)\right] \\
&\leq \mathop{\mathrm{E}}_{S_{i,b},z}\left[\frac{1}{m}\sum_{i=1}^{m} c(h_{S_{i,b}}, z_i) - c(h_{S_{i,b}}, z)\right] + 6b\widehat{\beta}.
\end{aligned}$$

The application of Lemma 3 to the difference of two cost functions also bounded by M as in the right-hand side leads to

$$\mathop{\mathrm{E}}_{S}[\Phi(S)] \leq \mathop{\mathrm{E}}_{\widetilde{S}_{i,b},\widetilde{z}}\left[\frac{1}{m}\sum_{i=1}^{m} c(h_{\widetilde{S}_{i,b}}, \widetilde{z}_i) - c(h_{\widetilde{S}_{i,b}}, \widetilde{z})\right] + 6b\widehat{\beta} + 3\beta(b)M.$$

Now, since the points $\widetilde{z}$ and $\widetilde{z}_i$ are independent and since the distribution is stationary, they have the same distribution and we can replace $\widetilde{z}_i$ with $\widetilde{z}$ in the empirical cost. Thus, we can write

$$\mathop{\mathrm{E}}_{S}[\Phi(S)] \leq \mathop{\mathrm{E}}_{\widetilde{S}_{i,b},\widetilde{z}}\left[\frac{1}{m}\sum_{i=1}^{m} c(h_{\widetilde{S}^i_{i,b}}, \widetilde{z}) - c(h_{\widetilde{S}_{i,b}}, \widetilde{z})\right] + 6b\widehat{\beta} + 3\beta(b)M \leq 2\widehat{\beta} + 6b\widehat{\beta} + 3\beta(b)M,$$

where $\widetilde{S}^i_{i,b}$ is the sequence derived from $\widetilde{S}_{i,b}$ by replacing $\widetilde{z}_i$ with $\widetilde{z}$. The last inequality holds by $\widehat{\beta}$-stability of the learning algorithm. The other side of the inequality in the statement of the lemma can be shown following the same steps. ■

3.3 φ-Mixing Concentration Bound

We are now prepared to make use of a concentration inequality to provide a generalization bound in the φ-mixing scenario. Several concentration inequalities have been shown in the φ-mixing case, for example, Marton (1998), Samson (2000), Chazottes et al. (2007) and Kontorovich and Ramanan (2008). We will use that of Kontorovich and Ramanan (2008), which is very similar to that of Chazottes et al. (2007), modulo the fact that the latter requires a finite sample space.

The following is a concentration inequality derived from that of Kontorovich and Ramanan (2008).[3]

Theorem 8 *Let $\Phi\colon Z^m \to \mathbb{R}$ be a measurable function that is c-Lipschitz with respect to the Hamming metric for some $c>0$ and let $Z_1,\ldots,Z_m$ be random variables distributed according to a φ-mixing distribution. Then, for any $\varepsilon>0$, the following inequality holds:*

$$\Pr\left[\left|\Phi(Z_1,\ldots,Z_m)-\mathrm{E}[\Phi(Z_1,\ldots,Z_m)]\right|\geq\varepsilon\right]\leq 2\exp\left(\frac{-2\varepsilon^2}{mc^2||\Delta_m||_\infty^2}\right),$$

where $||\Delta_m||_\infty \leq 1+2\sum_{k=1}^{m}\varphi(k)$.

It should be pointed out that the statement of the theorem in this paper is improved by a factor of 4 in the exponent with respect to that of Kontorovich and Ramanan (2008, Theorem 1.1). This can be achieved straightforwardly by following the same steps as in the proof of Kontorovich and Ramanan (2008), but by making use of the following general form of McDiarmid's inequality (Theorem 9) instead of Azuma's inequality. In particular, Theorem 5.1 of Kontorovich and Ramanan (2008) shows that for a φ-mixing distribution and a 1-Lipschitz function, the constants c_i can be bounded as follows in Theorem 9:

$$c_i \leq 1+2\sum_{k=1}^{m-i}\varphi(k).$$

Theorem 9 (McDiarmid, 1989, 6.10) *Let $Z_1,\ldots,Z_m$ be arbitrary random variables taking values in Z and let $\Phi\colon Z^m\to\mathbb{R}$ be a measurable function satisfying for all $z_i,z_i'\in Z$, $i=1,\ldots,m$, the following inequalities:*

$$\left|\mathrm{E}\left[\Phi(Z_1,\ldots,Z_m)\middle|Z_1=z_1,\ldots,Z_i=z_i\right]-\mathrm{E}\left[\Phi(Z_1,\ldots,Z_m)\middle|Z_1=z_1,\ldots,Z_i=z_i'\right]\right|\leq c_i,$$

where $c_i>0$, $i=1,\ldots,m$, are constants. Then, for any $\varepsilon>0$, the following inequality holds :

$$\Pr\left[\left|\Phi(Z_1,\ldots,Z_m)-\mathrm{E}[\Phi(Z_1,\ldots,Z_m)]\right|\geq\varepsilon\right]\leq 2\exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^m c_i^2}\right).$$

In the i.i.d. case, McDiarmid's theorem can be restated in the following simpler form that we shall use in Section 4.

Theorem 10 (McDiarmid, i.i.d. scenario) *Let $Z_1,\ldots,Z_m$ be independent random variables taking values in Z and let $\Phi\colon Z^m\to\mathbb{R}$ be a measurable function satisfying for all $z_i,z_i'\in Z$, $i=1,\ldots,m$, the following inequalities:*

$$\left|\Phi(z_1,\ldots,z_i,\ldots z_m)-\Phi(z_1,\ldots,z_i',\ldots z_m)\right|\leq c_i,$$

where $c_i>0$, $i=1,\ldots,m$, are constants. Then, for any $\varepsilon>0$, the following inequality holds:

$$\Pr\left[\left|\Phi(Z_1,\ldots,Z_m)-\mathrm{E}[\Phi(Z_1,\ldots,Z_m)]\right|\geq\varepsilon\right]\leq 2\exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^m c_i^2}\right).$$

3. We should note that original bound is expressed in terms of η-mixing coefficients. To simplify presentation, we are adapting it to the case of stationary φ-mixing sequences by using the following straightforward inequality for a stationary process: $2\varphi(j-i)\geq\eta_{ij}$. Furthermore, the bound presented in Kontorovich and Ramanan (2008) holds when the sample space is countable, it is extended to the continuous case in Kontorovich (2007).

3.4 φ-Mixing Generalization Bounds

This section presents several theorems that constitute the main results of this paper in the φ-mixing case. The following theorem is constructed from the bounds shown in the previous three sections.

Theorem 11 (General Non-i.i.d. Stability Bound) *Let h_S denote the hypothesis returned by a $\widehat{\beta}$-stable algorithm trained on a sample S drawn from a φ-mixing stationary distribution and let c be a measurable non-negative cost function upper bounded by $M>0$, then for any $b \in \{0,\ldots,m\}$ and any $\varepsilon>0$, the following generalization bound holds:*

$$\Pr_S\left[\left|R(h_S)-\widehat{R}(h_S)\right| > \varepsilon + (6b+2)\widehat{\beta} + 6M\varphi(b)\right] \leq 2\exp\left(\frac{-2\varepsilon^2(1+2\sum_{i=1}^{m}\varphi(i))^{-2}}{m((b+2)2\widehat{\beta}+2M\varphi(b)+M/m)^2}\right).$$

Proof The theorem follows directly the application of Lemma 6 and Lemma 7 to Theorem 8. ■

The theorem gives a general stability bound for φ-mixing stationary sequences. If we further assume that the sequence is algebraically φ-mixing, that is for all k, $\varphi(k)=\varphi_0 k^{-r}$ for some $r>1$, then we can solve for the value of b to optimize the bound.

Theorem 12 (Non-i.i.d. Stability Bound for Algebraically Mixing Sequences) *Let h_S denote the hypothesis returned by a $\widehat{\beta}$-stable algorithm trained on a sample S drawn from an algebraically φ-mixing stationary distribution, $\varphi(k)=\varphi_0 k^{-r}$ with $r>1$, and let c be a measurable non-negative cost function upper bounded by $M>0$, then, for any $\varepsilon>0$, the following generalization bound holds:*

$$\Pr_S\left[\left|R(h_S)-\widehat{R}(h_S)\right| > \varepsilon + 8\widehat{\beta} + (r+1)6M\varphi(b)\right] \leq 2\exp\left(\frac{-2\varepsilon^2(1+2\varphi_0 r/(r-1))^{-2}}{m(6\widehat{\beta}+(r+1)2M\varphi(b)+M/m)^2}\right),$$

where $b=\left(\frac{\widehat{\beta}}{r\varphi_0 M}\right)^{-1/(r+1)}$.

Proof For an algebraically mixing sequence, the value of b minimizing the bound of Theorem 11 satisfies the equation $\widehat{\beta}b^* = rM\varphi(b^*)$. Since b must be an integer, we use the approximation $b=\left\lceil\left(\frac{\widehat{\beta}}{r\varphi_0 M}\right)^{-1/(r+1)}\right\rceil$ when applying Theorem 11. However, observing the inequalities $\varphi(b^*)\geq\varphi(b)$ and $(b^*+1)\geq b$, allows us to write the statement of Theorem 12 in terms of the fractional choice b^*.

The term in the numerator can be bounded as

$$\begin{aligned}1+2\sum_{i=1}^{m}\varphi(i) &= 1+2\sum_{i=1}^{m}\varphi_0 i^{-r}\\ &\leq 1+2\varphi_0\left(1+\int_1^m x^{-r}dx\right)\\ &= 1+2\varphi_0\left(1+\frac{m^{1-r}-1}{1-r}\right).\end{aligned}$$

Using the assumption $r > 1$, we can upper bound m^{1-r} with 1 and obtain

$$1 + 2\varphi_0 \left(1 + \frac{m^{1-r} - 1}{1-r}\right) \leq 1 + 2\varphi_0 \left(1 + \frac{1}{r-1}\right) = 1 + \frac{2\varphi_0 r}{r-1}.$$

Plugging in this value and the minimizing value of b in the bound of Theorem 11 yields the statement of the theorem. ■

In the case of a zero mixing coefficient ($\varphi = 0$ and $b = 0$), the bounds of Theorem 11 coincide with the i.i.d. stability bound of Bousquet and Elisseeff (2002).

In the general case, in order for the right-hand side of these bounds to converge, we must have $\widehat{\beta} = o(1/\sqrt{m})$ and $\varphi(b) = o(1/\sqrt{m})$. The first condition holds for several families of algorithms with $\widehat{\beta} \leq O(1/m)$ (Bousquet and Elisseeff, 2002).

In the case of algebraically mixing sequences with $r > 1$, as assumed in Theorem 12, $\widehat{\beta} \leq O(1/m)$ implies $\varphi(b) \approx \varphi_0(\widehat{\beta}/(r\varphi_0 M))^{(r/(r+1))} < O(1/\sqrt{m})$. More specifically, for the scenario of algebraic mixing with $1/m$-stability, the following bound holds with probability at least $1 - \delta$:

$$\left|R(h_S) - \widehat{R}(h_S)\right| \leq O\left(\sqrt{\frac{\log(1/\delta)}{m^{\frac{r-1}{r+1}}}}\right).$$

This is obtained by setting the right-hand side of Theorem 12 equal to δ and solving for ε. Furthermore, if we choose $\varepsilon = \sqrt{\frac{C\log(m)}{m^{(r-1)/(r+1)}}}$ for a large enough constant $C > 0$, the right-hand side of Theorem 12 is summable over m and thus, by the Borel-Cantelli lemma, the following inequality holds almost surely:

$$\left|R(h_S) - \widehat{R}(h_S)\right| \leq O\left(\sqrt{\frac{\log(m)}{m^{\frac{r-1}{r+1}}}}\right).$$

Similar bounds can be given for the exponential mixing setting ($\varphi(k) = \varphi_0 \exp(-\varphi_1 k^r)$). If we choose $b = O(\sqrt{\log(m)^3/m})$ and assume $\widehat{\beta} = O(1/m)$, then, with probability at least $1 - \delta$,

$$\left|R(h_S) - \widehat{R}(h_S)\right| \leq O\left(\sqrt{\frac{\log(1/\delta)\log^2(m)}{m}}\right).$$

If we instead set $\varepsilon = C\sqrt{\frac{\log^3(m)}{m}}$ for a large enough constant C, the right-hand side of Theorem 12 is summable and again by the Borel-Cantelli lemma we have

$$\left|R(h_S) - \widehat{R}(h_S)\right| \leq O\left(\sqrt{\frac{\log^3(m)}{m}}\right),$$

almost surely.

3.5 Applications

We now present the application of our stability bounds for algebraically φ-mixing sequences to several algorithms, including the family of kernel-based regularization algorithms and that of relative entropy-based regularization algorithms. The application of our learning bounds will benefit from the previous analysis of the stability of these algorithms by Bousquet and Elisseeff (2002).

3.5.1 KERNEL-BASED REGULARIZATION ALGORITHMS

We first apply our bounds to a family of algorithms minimizing a regularized objective function based on the norm $\|\cdot\|_K$ in a reproducing kernel Hilbert space, where K is a positive definite symmetric kernel:

$$\operatorname*{argmin}_{h\in H} \frac{1}{m}\sum_{i=1}^{m} c(h,z_i) + \lambda\|h\|_K^2. \tag{7}$$

The application of our bound is possible, under some general conditions, since kernel regularized algorithms are stable with $\widehat{\beta} \leq O(1/m)$ (Bousquet and Elisseeff, 2002). For the sake of completeness, we briefly present the proof of this $\widehat{\beta}$-stability.

We will assume that the cost function c is σ-*admissible*, that is there exists $\sigma \in \mathbb{R}_+$ such that for any two hypotheses $h, h' \in H$ and for all $z = (x,y) \in X \times Y$,

$$|c(h,z) - c(h',z)| \leq \sigma|h(x) - h'(x)|.$$

This assumption holds for the quadratic cost and most other cost functions when the hypothesis set and the set of output labels are bounded by some $M \in \mathbb{R}_+$: $\forall h \in H, \forall x \in X, |h(x)| \leq M$ and $\forall y \in Y, |y| \leq M$. We will also assume that c is differentiable. This assumption is in fact not necessary and all of our results hold without it, but it makes the presentation simpler.

We denote by B_F the Bregman divergence associated to a convex function F: $B_F(f\|g) = F(f) - F(g) - \langle f - g, \nabla F(g)\rangle$. In what follows, it will be helpful to define F as the objective function of a general regularization based algorithm,

$$F_S(h) = \widehat{R}_S(h) + \lambda N(h),$$

where $\widehat{R}_S$ is the empirical error as measured on the sample S, $N : H \to \mathbb{R}^+$ is a regularization function and $\lambda > 0$ is the familiar trade-off parameter. Finally, we shall use the shorthand $\Delta h = h' - h$.

Lemma 13 (Bousquet and Elisseeff, 2002) *A kernel-based regularization algorithm of the form (7), with bounded kernel $K(x,x) \leq \kappa < \infty$ and σ-admissible cost function, is $\widehat{\beta}$-stable with coefficient*

$$\widehat{\beta} \leq \frac{\sigma^2\kappa^2}{m\lambda}.$$

Proof Let h and h' be the minimizers of F_S and F'_S respectively where S and S' differ in the first coordinate (choice of coordinate is without loss of generality), then,

$$B_N(h'\|h) + B_N(h\|h') \leq \frac{2\sigma}{m\lambda}\sup_{x\in S}|\Delta h(x)|. \tag{8}$$

To see this, we notice that since $B_F = B_{\widehat{R}} + \lambda B_N$, and since a Bregman divergence is non-negative,

$$\lambda\big(B_N(h'\|h) + B_N(h\|h')\big) \leq B_{F_S}(h'\|h) + B_{F_{S'}}(h\|h').$$

By the definition of h and h' as the minimizers of F_S and $F_{S'}$,

$$B_{F_S}(h'\|h) + B_{F_{S'}}(h\|h') = \widehat{R}_{F_S}(h') - \widehat{R}_{F_S}(h) + \widehat{R}_{F_{S'}}(h) - \widehat{R}_{F_{S'}}(h').$$

Finally, by the σ-admissibility of the cost function c and the definition of S and S',

$$\begin{aligned}\lambda\big(B_N(h'\|h)+B_N(h\|h')\big) &\leq \widehat{R}_{F_S}(h')-\widehat{R}_{F_S}(h)+\widehat{R}_{F_{S'}}(h)-\widehat{R}_{F_{S'}}(h')\\ &= \frac{1}{m}\Big[c(h',z_1)-c(h,z_1)+c(h,z_1')-c(h',z_1')\Big]\\ &\leq \frac{1}{m}\Big[\sigma|\Delta h(x_1)|+\sigma|\Delta h(x_1')|\Big]\\ &\leq \frac{2\sigma}{m}\sup_{x\in S}|\Delta h(x)|,\end{aligned}$$

which establishes (8).

Now, if we consider $N(\cdot)=\|\cdot\|_K^2$, we have $B_N(h'\|h)=\|h'-h\|_K^2$, thus $B_N(h'\|h)+B_N(h\|h')=2\|\Delta h\|_K^2$ and by (8) and the reproducing kernel property,

$$\begin{aligned}2\|\Delta h\|_K^2 &\leq \frac{2\sigma}{m\lambda}\sup_{x\in S}|\Delta h(x)|\\ &\leq \frac{2\sigma}{m\lambda}\kappa||\Delta h||_K.\end{aligned}$$

Thus $\|\Delta h\|_K \leq \frac{\sigma\kappa}{m\lambda}$. And using the σ-admissibility of c and the kernel reproducing property we obtain

$$\forall z\in X\times Y, |c(h',z)-c(h,z)|\leq \sigma|\Delta h(x)|\leq \kappa\sigma\|\Delta h\|_K.$$

Therefore,

$$\forall z\in X\times Y, |c(h',z)-c(h,z)|\leq \frac{\sigma^2\kappa^2}{m\lambda},$$

which completes the proof. ■

Three specific instances of kernel regularization algorithms are SVR, for which the cost function is based on the ε-insensitive cost:

$$c(h,z)=\begin{cases}0 & \text{if } |h(x)-y|\leq\varepsilon,\\ |h(x)-y|-\varepsilon & \text{otherwise.}\end{cases}$$

Kernel Ridge Regression (Saunders et al., 1998), for which

$$c(h,z)=(h(x)-y)^2\,,$$

and finally Support Vector Machines with the hinge-loss,

$$c(h,z)=\begin{cases}0 & \text{if } 1-yh(x)\leq 0,\\ 1-yh(x) & \text{if } yh(x)<1,\end{cases}$$

For kernel regularization algorithms, as pointed out in Bousquet and Elisseeff (2002, Lemma 23), a bound on the labels immediately implies a bound on the output of the hypothesis returned by the algorithm. We formally state this lemma below.

Lemma 14 *Let h^* be the solution of the optimization problem (7), let c be a cost function and let $B(\cdot)$ be a real-valued function such that for all $h \in H$, $x \in X$, and $y' \in Y$,*

$$c(h(x), y') \leq B(h(x)).$$

Then, the output of h^* is bounded as follows,

$$\forall x \in X, |h^*(x)| \leq \kappa \sqrt{\frac{B(0)}{\lambda}},$$

where λ is the regularization parameter, and $\kappa^2 \geq K(x,x)$ for all $x \in X$.

Proof Let $F(h) = \frac{1}{m} \sum_{i=1}^{m} c(h, z_i) + \lambda \|h\|_K^2$ and let $\mathbf{0}$ be the zero hypothesis, then by definition of F and h^*,

$$\lambda \|h^*\|_K^2 \leq F(h^*) \leq F(\mathbf{0}) \leq B(0).$$

Then, using the reproducing kernel property and the Cauchy-Schwarz inequality we note,

$$\forall x \in X, |h^*(x)| = \langle h^*, K(x, \cdot) \rangle \leq \|h^*\|_K \sqrt{K(x,x)} \leq \kappa \|h^*\|_K.$$

Combining the two inequalities proves the lemma. ■

We note that in Bousquet and Elisseeff (2002), the following bound is also stated: $c(h^*(x), y') \leq B(\kappa \sqrt{B(0)/\lambda})$. However, when later applied, it seems that the authors use an incorrect upper bound function $B(\cdot)$, which we remedy in the following.

Corollary 15 *Assume a bounded output $Y = [0, B]$, for some $B > 0$, and assume that $K(x,x) \leq \kappa^2$ for all x for some $\kappa > 0$. Let h_S denote the hypothesis returned by the algorithm when trained on a sample S drawn from an algebraically φ-mixing stationary distribution. Let $u = r/(r+1) \in [\frac{1}{2}, 1]$, $M' = 2(r+1)\varphi_0 M/(r\varphi_0 M)^u$, and $\varphi_0' = (1 + 2\varphi_0 r/(r-1))$. Then, with probability at least $1-\delta$, the following generalization bounds hold for*

a. Support Vector Machines (SVM, with hinge-loss)

$$R(h_S) \leq \widehat{R}(h_S) + \frac{8\kappa^2}{\lambda m} + \left(\frac{2\kappa^2}{\lambda}\right)^u \frac{3M'}{m^u} + \varphi_0' \left(M + \frac{3\kappa^2}{\lambda} + \left(\frac{2\kappa^2}{\lambda}\right)^u \frac{M'}{m^{u-1}}\right) \sqrt{\frac{2\log(2/\delta)}{m}},$$

where $M = \kappa \sqrt{\frac{1}{\lambda}} + B$.

b. Support Vector Regression (SVR):

$$R(h_S) \leq \widehat{R}(h_S) + \frac{8\kappa^2}{\lambda m} + \left(\frac{2\kappa^2}{\lambda}\right)^u \frac{3M'}{m^u} + \varphi_0' \left(M + \frac{3\kappa^2}{\lambda} + \left(\frac{2\kappa^2}{\lambda}\right)^u \frac{M'}{m^{u-1}}\right) \sqrt{\frac{2\log(2/\delta)}{m}},$$

where $M = \kappa \sqrt{\frac{2B}{\lambda}} + B$.

c. Kernel Ridge Regression (KRR):

$$R(h_S) \leq \widehat{R}(h_S) + \frac{32\kappa^2 B^2}{\lambda m} + \left(\frac{8\kappa^2 B^2}{\lambda}\right)^u \frac{3M'}{m^u} + \varphi_0'\left(M + \frac{12\kappa^2 B^2}{\lambda} + \left(\frac{8\kappa^2 B^2}{\lambda}\right)^u \frac{M'}{m^{u-1}}\right)\sqrt{\frac{2\log(2/\delta)}{m}},$$

where $M = 2\kappa^2 B^2/\lambda + B^2$.

Proof For SVM, the hinge-loss is 1-admissible giving $\widehat{\beta} \leq \kappa^2/(\lambda m)$. Using Lemma 14, with $B(0) = 1$, the loss can be bounded $\forall x \in X, y \in Y, 1 + |h^*(x)| \leq \kappa\sqrt{\frac{1}{\lambda}} + B$.

Similarly, SVR has a loss function that is 1-admissible, thus, applying Lemma 13 gives us $\widehat{\beta} \leq \kappa^2/(\lambda m)$. Using Lemma 14, with $B(0) = B$, we can bound the loss as follows, $\forall x \in X, y \in Y, |h^*(x) - y| \leq \kappa\sqrt{\frac{B}{\lambda}} + B$.

Finally for KRR, we have a loss function that is $2B$-admissible and again using Lemma 13 $\widehat{\beta} \leq 4\kappa^2 B^2/(\lambda m)$. Again, applying Lemma 14 with $B(0) = B^2$ and $\forall x \in X, y \in Y, (h^*(x) - y)^2 \leq \kappa^2 B^2/\lambda + B^2$.

Plugging these values into the bound of Theorem 12 and setting the right-hand side to δ yields the statement of the corollary. ■

3.5.2 RELATIVE ENTROPY-BASED REGULARIZATION ALGORITHMS

In this section, we apply the results of Theorem 12 to a family of learning algorithms based on relative entropy-regularization. These algorithms learn hypotheses h that are mixtures of base hypotheses in $\{h_\theta : \theta \in \Theta\}$, where Θ is measurable set. The output of these algorithms is a mixture $g : \Theta \to \mathbb{R}$, that is a distribution over Θ. Let G denote the set of all such distributions and let $g_0 \in G$ be a fixed distribution. Relative entropy based-regularization algorithms output the solution of a minimization problem of the following form:

$$\operatorname*{argmin}_{g \in G} \frac{1}{m}\sum_{i=1}^{m} c(g, z_i) + \lambda D(g\|g_0), \tag{9}$$

where the cost function $c : G \times Z \to \mathbb{R}$ is defined in terms of a second internal cost function $c' : H \times Z \to \mathbb{R}$:

$$c(g, z) = \int_\Theta c'(h_\theta, z) g(\theta) d\theta,$$

and where $D(g\|g_0)$ is the relative entropy between g and g_0:

$$D(g\|g_0) = \int_\Theta g(\theta) \log \frac{g(\theta)}{g_0(\theta)} d\theta.$$

As shown by Bousquet and Elisseeff (2002, Theorem 24), a relative entropy-based regularization algorithm defined by (9) with bounded loss $c'(\cdot) \leq M$, is $\widehat{\beta}$-stable with the following bound on the stability coefficient:

$$\widehat{\beta} \leq \frac{M^2}{\lambda m}.$$

Theorem 12 combined with this inequality immediately yields the following generalization bound.

Corollary 16 *Let h_S be the hypothesis solution of the optimization (9) trained on a sample S drawn from an algebraically φ-mixing stationary distribution with the internal cost function c' bounded by M. Then, with probability at least $1-\delta$, the following holds:*

$$R(h_S) \leq \widehat{R}(h_S) + \frac{8M^2}{\lambda m} + \frac{3M'}{\lambda^u m^u} + \varphi_0' \left(M + \frac{3M^2}{\lambda} + \frac{2^u M'}{\lambda^u m^{u-1}} \right) \sqrt{\frac{2\log(2/\delta)}{m}},$$

where $u = r/(r+1) \in [\frac{1}{2}, 1]$, $M' = 2(r+1)\varphi_0 M^{u+1}/(r\varphi_0)^u$, and $\varphi_0' = (1 + 2\varphi_0 r/(r-1))$.

3.6 Discussion

The results presented here are, to the best of our knowledge, the first stability-based generalization bounds for the class of algorithms just studied in a non-i.i.d. scenario. These bounds are non-trivial when the condition on the regularization parameter $\lambda \gg 1/m^{1/2-1/r}$ parameter holds for all large values of m. This condition coincides with the one obtained in the i.i.d. setting by Bousquet and Elisseeff (2002), in the limit, as r tends to infinity. The next section gives stability-based generalization bounds that hold even in the scenario of β-mixing sequences.

4. β-Mixing Generalization Bounds

In this section, we prove a stability-based generalization bound that only requires the training sequence to be drawn from a β-mixing stationary distribution. The bound is thus more general and covers the φ-mixing case analyzed in the previous section. However, unlike the φ-mixing case, the β-mixing bound presented here is not a purely exponential bound. It contains an additive term, which depends on the mixing coefficient.

As in the previous section, $\Phi(S)$ is defined by $\Phi(S) = R(h_S) - \widehat{R}(h_S)$. To simplify the presentation, here, we define the generalization error of h_S by $R(h_S) = \mathrm{E}_z[c(h_S, z)]$. Thus, test samples are assumed independent of S.[4] Note that for any block of points $Z = z_1 \ldots z_k$ drawn independently of S, the following equality holds:

$$\mathop{\mathrm{E}}_{Z}\left[\frac{1}{|Z|}\sum_{z\in Z} c(h_S, z)\right] = \frac{1}{k}\sum_{i=1}^{k} \mathop{\mathrm{E}}_{Z}[c(h_S, z_i)] = \frac{1}{k}\sum_{i=1}^{k} \mathop{\mathrm{E}}_{z_i}[c(h_S, z_i)] = \mathop{\mathrm{E}}_{z}[c(h_S, z)]$$

since, by stationarity, $\mathrm{E}_{z_i}[c(h_S, z_i)] = \mathrm{E}_{z_j}[c(h_S, z_j)]$ for all $1 \leq i, j \leq k$. Thus, for any such block Z, we can write $R(h_S) = \mathrm{E}_Z\left[\frac{1}{|Z|}\sum_{z\in Z} c(h_S, z)\right]$. For convenience, we extend the cost function c to blocks as follows:

$$c(h, Z) = \frac{1}{|Z|}\sum_{z\in Z} c(h, z).$$

With this notation, $R(h_S) = \mathrm{E}_Z[c(h_S, Z)]$ for any block drawn independently of S, regardless of the size of Z.

To derive a generalization bound for the β-mixing scenario, we apply McDiarmid's inequality (Theorem 10) to Φ defined over a sequence of independent blocks. The independent blocks we consider are non-symmetric and thus more general than those considered by previous authors (Yu, 1994; Meir, 2000).

4. In the β-mixing scenario, a result similar to that of Lemma 5 can be shown to hold in expectation with respect the sample S. Using Markov's inequality, the inequality can be shown to hold with high probability. Thus, the results that follow can all be be extended to the case where the test points depend on the training sample, at the expense of an an additional confidence term.

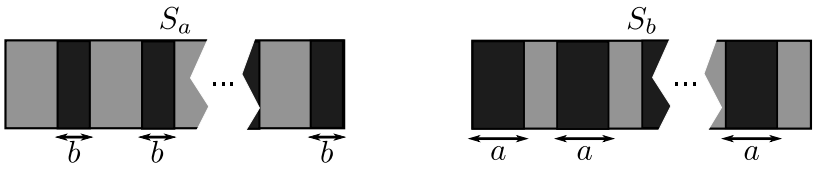


Figure 3: Illustration of the sequences S_a and S_b derived from S that are considered in the proofs. The darkened regions are considered as being removed from the sequence.

From a sample S made of a sequence of m points, we construct two sequences of blocks S_a and S_b, each containing μ blocks. Each block in S_a contains a points and each block in S_b contains b points (see Figure 3). S_a and S_b form a partitioning of S; for any $a,b \in \{0,\ldots,m\}$ such that $(a+b)\mu = m$, they are defined precisely as follows:

$$S_a = (Z_1^{(a)},\ldots,Z_\mu^{(a)}), \text{ with } Z_i^{(a)} = z_{(i-1)(a+b)+1},\ldots,z_{(i-1)(a+b)+a}$$
$$S_b = (Z_1^{(b)},\ldots,Z_\mu^{(b)}), \text{ with } Z_i^{(b)} = z_{(i-1)(a+b)+a+1},\ldots,z_{(i-1)(a+b)+a+b},$$

for all $i \in \{1,\ldots,\mu\}$. We shall consider similarly sequences of i.i.d. blocks $\widetilde{Z}_i^a$ and $\widetilde{Z}_i^b$, $i \in \{1,\ldots,\mu\}$, such that the points within each block are drawn according to the same original β-mixing distribution and shall denote by $\widetilde{S}_a$ the block sequence $(\widetilde{Z}_1^{(a)},\ldots,\widetilde{Z}_\mu^{(a)})$.

In preparation for the application of McDiarmid's inequality, we give a bound on the expectation of $\Phi(\widetilde{S}_a)$. Since the expectation is taken over a sequence of i.i.d. blocks, this brings us to a situation similar to the i.i.d. scenario analyzed by Bousquet and Elisseeff (2002), with the exception that we are dealing with i.i.d. blocks instead of i.i.d. points.

Lemma 17 *Let $\widetilde{S}_a$ be an independent block sequence as defined above, then the following bound holds for the expectation of $|\Phi(\widetilde{S}_a)|$:*

$$\mathop{\mathrm{E}}_{\widetilde{S}_a}[|\Phi(\widetilde{S}_a)|] \leq 2a\widehat{\beta}.$$

Proof Since the blocks $\widetilde{Z}^{(a)}$ are independent, we can replace any one of them with any other block Z drawn from the same distribution. However, changing the training set also changes the hypothesis, in a limited way. This is shown precisely below:

$$\begin{aligned}
\mathop{\mathrm{E}}_{\widetilde{S}_a}[|\Phi(\widetilde{S}_a)|] &= \mathop{\mathrm{E}}_{\widetilde{S}_a}\left[\left|\frac{1}{\mu}\sum_{i=1}^{\mu} c(h_{\widetilde{S}_a},\widetilde{Z}_i^{(a)}) - \mathop{\mathrm{E}}_{Z}[c(h_{\widetilde{S}_a},Z)]\right|\right] \\
&\leq \mathop{\mathrm{E}}_{\widetilde{S}_a,Z}\left[\left|\frac{1}{\mu}\sum_{i=1}^{\mu} c(h_{\widetilde{S}_a},\widetilde{Z}_i^{(a)}) - c(h_{\widetilde{S}_a},Z)\right|\right] \\
&= \mathop{\mathrm{E}}_{\widetilde{S}_a,Z}\left[\left|\frac{1}{\mu}\sum_{i=1}^{\mu} c(h_{\widetilde{S}_a^i},Z) - c(h_{\widetilde{S}_a},Z)\right|\right],
\end{aligned}$$

where $\widetilde{S}_a^i$ corresponds to the block sequence $\widetilde{S}_a$ obtained by replacing the ith block with Z. The $\widehat{\beta}$-stability of the learning algorithm gives

$$\mathop{\mathrm{E}}_{\widetilde{S}_a,Z}\left[\frac{1}{\mu}\Big|\sum_{i=1}^{\mu} c(h_{\widetilde{S}_a^i},Z) - c(h_{\widetilde{S}_a},Z)\Big|\right] \leq \mathop{\mathrm{E}}_{\widetilde{S}_a,Z}\left[\frac{1}{\mu}\sum_{i=1}^{\mu} 2a\widehat{\beta}\right] \leq 2a\widehat{\beta}.$$

■

We now relate the non-i.i.d. event $\Pr[\Phi(S) \geq \varepsilon]$ to an independent block sequence event to which we can apply McDiarmid's inequality.

Lemma 18 *Assume a $\widehat{\beta}$-stable algorithm. Then, for a sample S drawn from a β-mixing stationary distribution, the following bound holds:*

$$\Pr_S[|\Phi(S)| \geq \varepsilon] \leq \Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| - \mathrm{E}[|\Phi(\widetilde{S}_a)|] \geq \varepsilon_0'\right] + (\mu-1)\beta(b),$$

where $\varepsilon_0' = \varepsilon - \frac{\mu b M}{m} - 2\mu b\widehat{\beta} - \mathrm{E}_{\widetilde{S}_a'}[|\Phi(\widetilde{S}_a')|]$.

Proof The proof consists of first rewriting the event in terms of S_a and S_b and bounding the error on the points in S_b in a trivial manner. This can be afforded since b will be eventually chosen to be small. Since $|\mathrm{E}_{Z'}[c(h_S,Z')] - c(h_S,z')| \leq M$ for any $z' \in S_b$, we can write

$$\begin{aligned}
\Pr_S[|\Phi(S)| \geq \varepsilon] &= \Pr_S[|R(h_S) - \widehat{R}(h_S)| \geq \varepsilon] \\
&= \Pr_S\left[\frac{1}{m}\Big|\sum_{z\in S}\mathop{\mathrm{E}}_{Z}[c(h_S,Z)] - c(h_S,z)\Big| \geq \varepsilon\right] \\
&\leq \Pr_S\left[\frac{1}{m}\Big|\sum_{z\in S_a}\mathop{\mathrm{E}}_{Z}[c(h_S,Z)] - c(h_S,z)\Big| + \frac{1}{m}\Big|\sum_{z'\in S_b}\mathop{\mathrm{E}}_{Z'}[c(h_S,Z')] - c(h_S,z')\Big| \geq \varepsilon\right] \\
&\leq \Pr_S\left[\frac{1}{m}\Big|\sum_{z\in S_a}\mathop{\mathrm{E}}_{Z}[c(h_S,Z)] - c(h_S,z)\Big| + \frac{\mu b M}{m} \geq \varepsilon\right].
\end{aligned}$$

By $\widehat{\beta}$-stability and $\mu a/m \leq 1$, this last term can be bounded as follows

$$\begin{aligned}
&\Pr_S\left[\frac{1}{m}\Big|\sum_{z\in S_a}\mathop{\mathrm{E}}_{Z}[c(h_S,Z)] - c(h_S,z)\Big| + \frac{\mu b M}{m} \geq \varepsilon\right] \leq \\
&\qquad\qquad \Pr_{S_a}\left[\frac{1}{\mu a}\Big|\sum_{z\in S_a}\mathop{\mathrm{E}}_{Z}[c(h_{S_a},Z)] - c(h_{S_a},z)\Big| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right].
\end{aligned}$$

The right-hand side can be rewritten in terms of Φ and bounded in terms of a β-mixing coefficient:

$$\begin{aligned}
&\Pr_{S_a}\left[\frac{1}{\mu a}\Big|\sum_{z\in S_a}\mathop{\mathrm{E}}_{Z}[c(h_{S_a},Z)] - c(h_{S_a},z)\Big| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right] \\
&\qquad\qquad = \Pr_{S_a}\left[|\Phi(S_a)| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right] \\
&\qquad\qquad \leq \Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right] + (\mu-1)\beta(b),
\end{aligned}$$

by applying Lemma 3 to the indicator function of the event $\left\{|\Phi(S_a)| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right\}$. Since $\mathrm{E}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|]$ is a constant, the probability in this last term can be rewritten as

$$\begin{aligned}
&\Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon\right] \\
&\qquad = \Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| - \mathrm{E}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|] + \frac{\mu b M}{m} + 2\mu b\widehat{\beta} \geq \varepsilon - \mathrm{E}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|]\right] \\
&\qquad = \Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| - \mathrm{E}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|] \geq \varepsilon'_0\right],
\end{aligned}$$

which ends the proof of the lemma. ■

The last two lemmas will help us prove the main result of this section formulated in the following theorem.

Theorem 19 *Assume a $\widehat{\beta}$-stable algorithm and let ε' denote $\varepsilon - \frac{\mu b M}{m} - 2\mu b\widehat{\beta} - 2a\widehat{\beta}$ as in Lemma 18. Then, for any sample S of size m drawn according to a β-mixing stationary distribution, any choice of the parameters $a, b, \mu > 0$ such that $(a+b)\mu = m$, and $\varepsilon \geq 0$ such that $\varepsilon' \geq 0$, the following generalization bound holds:*

$$\Pr_{S}\left[|R(h_S) - \widehat{R}(h_S)| \geq \varepsilon\right] \leq \exp\left(\frac{-2\varepsilon'^2 m}{\left(4a\widehat{\beta}m + (a+b)M\right)^2}\right) + (\mu - 1)\beta(b).$$

Proof To prove the statement of theorem, it suffices to bound the probability term appearing in the right-hand side of Equation 18, $\Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| - \mathrm{E}[|\Phi(\widetilde{S}_a)|] \geq \varepsilon'_0\right]$, which is expressed only in terms of independent blocks. We can therefore apply McDiarmid's inequality by viewing the blocks as i.i.d. "points".

To do so, we must bound the quantity $\left||\Phi(\widetilde{S}_a)| - |\Phi(\widetilde{S}^i_a)|\right|$ where the sequence S_a and S^i_a differ in the ith block. We will bound separately the difference between the generalization errors and empirical errors.[5] The difference in empirical errors can be bounded as follows using the bound on the cost function c:

$$\begin{aligned}
|\widehat{R}(h_{S_a}) - \widehat{R}(h_{S^i_a})| &= \left|\frac{1}{\mu}\left[\sum_{j\neq i} c(h_{S_a}, Z_j) - c(h_{S^i_a}, Z_j)\right] + \frac{1}{\mu}\left[c(h_{S_a}, Z_i) - c(h_{S^i_a}, Z'_i)\right]\right| \\
&\leq 2a\widehat{\beta} + \frac{M}{\mu} = 2a\widehat{\beta} + \frac{(a+b)M}{m}.
\end{aligned}$$

The difference in generalization error can be straightforwardly bounded using $\widehat{\beta}$-stability:

$$|R(h_{S_a}) - R(h_{S^i_a})| = |\mathrm{E}_Z[c(h_{S_a}, Z)] - \mathrm{E}_Z[c(h_{S^i_a}, Z)]| = |\mathrm{E}_Z[c(h_{S_a}, Z) - c(h_{S^i_a}, Z)]| \leq 2a\widehat{\beta}.$$

5. We drop the superscripts on $Z^{(a)}$ since we will not be considering the sequence S_b in what follows.

Using these bounds in conjunction with McDiarmid's inequality yields

$$\Pr_{\widetilde{S}_a}[|\Phi(\widetilde{S}_a)| - \mathop{\mathrm{E}}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|] \geq \varepsilon'_0] \leq \exp\left(\frac{-2\varepsilon'^2_0 m}{\left(4a\widehat{\beta}m + (a+b)M\right)^2}\right)$$
$$\leq \exp\left(\frac{-2\varepsilon'^2 m}{\left(4a\widehat{\beta}m + (a+b)M\right)^2}\right).$$

Note that to show the second inequality we make use of Lemma 17 to establish the fact that

$$\varepsilon'_0 = \varepsilon - \frac{\mu b M}{m} - 2\mu b\widehat{\beta} - \mathop{\mathrm{E}}_{\widetilde{S}'_a}[|\Phi(\widetilde{S}'_a)|] \geq \varepsilon - \frac{\mu b M}{m} - 2\mu b\widehat{\beta} - 2a\widehat{\beta} = \varepsilon'.$$

Finally, we make use of Lemma 18 to establish the proof,

$$\Pr_S[|\Phi(S)| \geq \varepsilon] \leq \Pr_{\widetilde{S}_a}\left[|\Phi(\widetilde{S}_a)| - \mathrm{E}[|\Phi(\widetilde{S}_a)|] \geq \varepsilon'_0\right] + (\mu-1)\beta(b)$$
$$\leq \exp\left(\frac{-2\varepsilon'^2 m}{\left(4a\widehat{\beta}m + (a+b)M\right)^2}\right) + (\mu-1)\beta(b).$$

This concludes the proof of the theorem. ■

In order to make use of this bound, we must determine the values of parameters b and μ (a is then equal to $\mu/m - u$). There is a trade-off between selecting a large enough value for b to ensure that the mixing term decreases and choosing a large enough value of μ to minimize the remaining terms of the bound. The exact choice of parameters will depend on the type of mixing that is assumed (e.g., algebraic or exponential). In order to choose optimal parameters, it will be useful to view the bound as it holds with high probability, in the following corollary.

Corollary 20 *Assume a $\widehat{\beta}$-stable algorithm and let δ' denote $\delta - (\mu-1)\beta(b)$. Then, for any sample S of size m drawn according to a β-mixing stationary distribution, any choice of the parameters $a, b, \mu > 0$ such that $(a+b)\mu = m$, and $\delta \geq 0$ such that $\delta' \geq 0$, the following generalization bound holds with probability at least $(1-\delta)$:*

$$|R(h_S) - \widehat{R}(h_S)| < \mu b\left(\frac{M}{m} + 2\widehat{\beta}\right) + 2a\widehat{\beta}\left(4a\widehat{\beta}m + M\frac{m}{\mu}\right)\sqrt{\frac{\log(1/\delta')}{2m}}.$$

In the case of a fast mixing distribution, it is possible to select the values of the parameters to retrieve a bound as in the i.i.d. case, that is, $|R(h_S) - \widehat{R}(h_S)| = O\left(m^{-\frac{1}{2}}\sqrt{\log 1/\delta}\right)$. In particular, for $\beta(b) \equiv 0$, we can choose $a = 0$, $b = 1$, and $\mu = m$ to retrieve the i.i.d. bound of Bousquet and Elisseeff (2001).

In the following, we examine slower mixing algebraic β-mixing distributions, which are thus not close to the i.i.d. scenario. For algebraic mixing, the mixing parameter is defined as $\beta(b) = b^{-r}$. In that case, we wish to minimize the following function in terms of μ and b:

$$s(\mu, b) = \frac{\mu}{b^r} + \frac{m^{3/2}\widehat{\beta}}{\mu} + \frac{m^{1/2}}{\mu} + \mu b\left(\frac{1}{m} + \widehat{\beta}\right). \tag{10}$$

The first term of the function captures the condition $\delta > (\mu+1)\beta(b) \approx \mu/b^r$ and the remaining terms capture the shape of the bound in Corollary 20.

Setting the derivative with respect to each variable μ and b to zero and solving for each parameter results in the following expressions:

$$b = C_r \gamma^{-\frac{1}{r+1}} \qquad \mu = \frac{m^{3/4}\gamma^{\frac{1}{2(r+1)}}}{\sqrt{C_r(1+1/r)}},$$

where $\gamma = (m^{-1} + \widehat{\beta})$ and $C_r = r^{\frac{1}{r+1}}$ is a constant defined by the parameter r.

Now, assuming $\widehat{\beta} = O(m^{-\alpha})$ for some $0 < \alpha \leq 1$, we analyze the convergence behavior of Corollary 20. First, we observe that the terms b and μ have the following asymptotic behavior,

$$b = O\left(m^{\frac{\alpha}{r+1}}\right) \qquad \mu = O\left(m^{\frac{3}{4}-\frac{\alpha}{2(r+1)}}\right).$$

Next, we consider the condition $\delta' > 0$ which is equivalent to,

$$\delta > (\mu-1)\beta(b) = O\left(m^{\frac{3}{4}-\alpha\left(1-\frac{1}{2(r+1)}\right)}\right). \tag{11}$$

In order for the right-hand side of the inequality to converge, it must be the case that $\alpha > \frac{3r+3}{4r+2}$. In particular, if $\alpha = 1$, as is the case for several algorithms in Section 3.5, then it suffices that $r > 1$.

Finally, in order to see how the bound itself converges, we study the asymptotic behavior of the terms of Equation 10 (without the first term, which corresponds to the quantity already analyzed in Equation 11):

$$\underbrace{\frac{m^{3/2}\widehat{\beta}}{\mu} + \mu b\widehat{\beta}}_{(a)} + \underbrace{\frac{m^{1/2}}{\mu} + \frac{\mu b}{m}}_{(b)} = O\left(\underbrace{m^{\frac{3}{4}-\alpha\left(1-\frac{1}{2(r+1)}\right)}}_{(a)} \underbrace{+ m^{\frac{\alpha}{2(r+1)}-\frac{1}{4}}}_{(b)}\right).$$

This expression can be further simplified by noticing that $(b) \leq (a)$ for all $0 < \alpha \leq 1$ (with equality at $\alpha = 1$). Thus, both the bound and the condition on δ decrease asymptotically as the term in (a), resulting in the following corollary.

Corollary 21 *Assume a $\widehat{\beta}$-stable algorithm with $\widehat{\beta} = O(m^{-1})$ and let $\delta' = \delta - m^{\frac{1}{2(r+1)}-\frac{1}{4}}$. Then, for any sample S of size m drawn according to a algebraic β-mixing stationary distribution, and $\delta \geq 0$ such that $\delta' \geq 0$, the following generalization bound holds with probability at least $(1-\delta)$:*

$$|R(h_S) - \widehat{R}(h_S)| < O\left(m^{\frac{1}{2(r+1)}-\frac{1}{4}}\sqrt{\log(1/\delta')}\right).$$

As in previous bounds $r > 1$ is required for convergence. Furthermore, as expected, a larger mixing parameter r leads to a more favorable bound.

5. Conclusion

We presented stability bounds for both φ-mixing and β-mixing stationary sequences. Our bounds apply to large classes of algorithms, including common algorithms such as SVR, KRR, and SVMs, and extend to non-i.i.d. scenarios existing i.i.d. stability bounds. Since they are algorithm-specific, these bounds can often be tighter than other generalization bounds based on general complexity measures for families of hypotheses. As in the i.i.d. case, weaker notions of stability might help further improve and refine these bounds. These stability bounds complement general data-dependent learning bounds we have shown elsewhere for stationary β-mixing sequences using the notion of Rademacher complexity (Mohri and Rostamizadeh, 2009).

The stability bounds we presented can be used to analyze the properties of stable algorithms when used in the non-i.i.d settings studied. But, more importantly, they can serve as a tool for the design of novel and accurate learning algorithms. Of course, some mixing properties of the distributions need to be known to take advantage of the information supplied by our generalization bounds. In some problems, it is possible to estimate the shape of the mixing coefficients. This should help devising such algorithms.

Acknowledgments

We thank the editor and the reviewers for several comments that helped improve the original version of this paper.

References

Sergei Natanovich Bernstein. Sur l'extension du théorème limite du calcul des probabilités aux sommes de quantités dépendantes. *Mathematische Annalen*, 97:1–59, 1927.

Olivier Bousquet and André Elisseeff. Algorithmic stability and generalization performance. In *Advances in Neural Information Processing Systems*, 2001.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Jean-René Chazottes, Pierre Collet, Christof Külske, and Frank Redig. Concentration inequalities for random fields via coupling. *Probability Theory and Related Fields*, 137(1):201–225, 2007.

Corinna Cortes and Vladimir N. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Luc Devroye and T. J. Wagner. Distribution-free performance bounds for potential function rules. In *Information Theory, IEEE Transactions on*, volume 25, pages 601–604, 1979.

Paul Doukhan. *Mixing: Properties and Examples*. Springer-Verlag, 1994.

Michael Kearns and Dana Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. In *Computational Learning Theory*, pages 152–162, 1997.

Leo Kontorovich. *Measure Concentration of Strongly Mixing Processes with Applications*. PhD thesis, Carnegie Mellon University, 2007.

Leo Kontorovich and Kavita Ramanan. Concentration inequalities for dependent random variables via the martingale method. *Annals of Probability*, 36(6):2126–2158, 2008.

Aurélie Lozano, Sanjeev Kulkarni, and Robert Schapire. Convergence and consistency of regularized boosting algorithms with stationary β-mixing observations. In *Advances in Neural Information Processing Systems*, 2006.

Katalin Marton. Measure concentration for a class of random processes. *Probability Theory and Related Fields*, 110(3):427–439, 1998.

Davide Mattera and Simon Haykin. Support vector machines for dynamic reconstruction of a chaotic system. In *Advances in Kernel Methods: Support Vector Learning*, pages 211–241. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-19416-3.

Colin McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

Ron Meir. Nonparametric time series prediction through adaptive model selection. *Machine Learning*, 39(1):5–34, April 2000.

Dharmendra Modha and Elias Masry. On the consistency in nonparametric estimation under mixing assumptions. *IEEE Transactions of Information Theory*, 44:117–133, 1998.

Mehryar Mohri and Afshin Rostamizadeh. Stability bounds for non-iid processes. In *Advances in Neural Information Processing Systems*, 2007.

Mehryar Mohri and Afshin Rostamizadeh. Rademacher complexity bounds for non-i.i.d. processes. In *Advances in Neural Information Processing Systems (NIPS 2008)*, pages 1097–1104, Vancouver, Canada, 2009. MIT Press.

Klaus-Robert Müller, Alex Smola, Gunnar Rätsch, Bernhard Schölkopf, Jens Kohlmorgen, and Vladimir Vapnik. Predicting time series with support vector machines. In *Proceedings of the International Conference on Artificial Neural Networks*, Lecture Notes in Computer Science, pages 999–1004. Springer, 1997.

Paul-Marie Samson. Concentration of measure inequalities for Markov chains and-mixing processes. *Annals Probability*, 28(1):416–461, 2000.

Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 515–521. Morgan Kaufmann Publishers Inc., 1998.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.

Mathukumalli Vidyasagar. *Learning and Generalization: with Applications to Neural Networks*. Springer, 2003.

Bin Yu. Rates of convergence for empirical processes of stationary mixing sequences. *The Annals of Probability*, 22(1):94–116, Jan. 1994.

Shuheng Zhou, John Lafferty, and Larry Wasserman. Time varying undirected graphs. In *Proceedings of the 21st Annual Conference on Learning Theory*, 2008.

Journal of Machine Learning Research 11 (2010) 815-848 Submitted 5/09; Published 2/10

Iterative Scaling and Coordinate Descent Methods for Maximum Entropy Models

Fang-Lan Huang D93011@CSIE.NTU.EDU.TW
Cho-Jui Hsieh B92085@CSIE.NTU.EDU.TW
Kai-Wei Chang B92084@CSIE.NTU.EDU.TW
Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: Michael Collins

Abstract

Maximum entropy (Maxent) is useful in natural language processing and many other areas. Iterative scaling (IS) methods are one of the most popular approaches to solve Maxent. With many variants of IS methods, it is difficult to understand them and see the differences. In this paper, we create a general and unified framework for iterative scaling methods. This framework also connects iterative scaling and coordinate descent methods. We prove general convergence results for IS methods and analyze their computational complexity. Based on the proposed framework, we extend a coordinate descent method for linear SVM to Maxent. Results show that it is faster than existing iterative scaling methods.

Keywords: maximum entropy, iterative scaling, coordinate descent, natural language processing, optimization

1. Introduction

Maximum entropy (Maxent) is widely used in many areas such as natural language processing (NLP) and document classification. It is suitable for problems needing probability interpretations. For many NLP tasks, given a word sequence, we can use Maxent models to predict the label sequence with the maximal probability (Berger et al., 1996). Such tasks are different from traditional classification problems, which assign label(s) to a single instance.

Maxent models the conditional probability as:

$$\begin{aligned} &P_{\mathbf{w}}(y|x) \equiv \frac{S_{\mathbf{w}}(x,y)}{T_{\mathbf{w}}(x)}, \\ &S_{\mathbf{w}}(x,y) \equiv e^{\sum_t w_t f_t(x,y)},\ T_{\mathbf{w}}(x) \equiv \sum_y S_{\mathbf{w}}(x,y), \end{aligned} \tag{1}$$

where x indicates a context, y is the label of the context, and $\mathbf{w} \in R^n$ is the weight vector. A real-valued function $f_t(x,y)$ denotes the t-th feature extracted from the context x and the label y. We assume a finite number of features. In some cases, $f_t(x,y)$ is 0/1 to indicate a particular property. $T_{\mathbf{w}}(x)$ is a normalization term applied to make $\sum_y P_{\mathbf{w}}(y|x) = 1$.

Given an empirical probability distribution $\tilde{P}(x,y)$ obtained from training samples, Maxent minimizes the following negative log-likelihood:

$$\min_{\mathbf{w}} \; -\sum_{x,y} \tilde{P}(x,y) \log P_{\mathbf{w}}(y|x),$$

or equivalently,

$$\min_{\mathbf{w}} \; \sum_{x} \tilde{P}(x) \log T_{\mathbf{w}}(x) - \sum_{t} w_t \tilde{P}(f_t), \tag{2}$$

where $\tilde{P}(x,y) = N_{x,y}/N$, $N_{x,y}$ is the number of times that (x,y) occurs in training data, and N is the total number of training samples. $\tilde{P}(x) = \sum_y \tilde{P}(x,y)$ is the marginal probability of x, and $\tilde{P}(f_t) = \sum_{x,y} \tilde{P}(x,y) f_t(x,y)$ is the expected value of $f_t(x,y)$. To avoid overfitting the training samples, some add a regularization term to (2) and solve:

$$\min_{\mathbf{w}} \; L(\mathbf{w}) \equiv \min_{\mathbf{w}} \; \sum_{x} \tilde{P}(x) \log T_{\mathbf{w}}(x) - \sum_{t} w_t \tilde{P}(f_t) + \frac{1}{2\sigma^2} \sum_{t} w_t^2, \tag{3}$$

where σ is a regularization parameter. More discussion about regularization terms for Maxent can be seen in, for example, Chen and Rosenfeld (2000). We focus on (3) in this paper because it is strictly convex. Note that (2) is convex, but may not be strictly convex. We can further prove that a unique global minimum of (3) exists. The proof, omitted here, is similar to Theorem 1 in Lin et al. (2008).

Iterative scaling (IS) methods are popular in training Maxent models. They all share the same property of *solving a one-variable sub-problem at a time*. Existing IS methods include generalized iterative scaling (GIS) by Darroch and Ratcliff (1972), improved iterative scaling (IIS) by Della Pietra et al. (1997), and sequential conditional generalized iterative scaling (SCGIS) by Goodman (2002). The approach by Jin et al. (2003) is also an IS method, but it assumes that every class uses the same set of features. As this assumption is not general, in this paper we do not include this approach for discussion. In optimization, coordinate descent (CD) is a popular method which also solves a one-variable sub-problem at a time. With these many IS and CD methods, it is difficult to see their differences. In Section 2, we propose a unified framework to describe IS and CD methods from an optimization viewpoint. We further analyze the theoretical convergence as well as computational complexity of IS and CD methods. In particular, general linear convergence is proved. In Section 3, based on a comparison between IS and CD methods, we propose a new and more efficient CD method. These two results (a unified framework and a faster CD method) are the main contributions of this paper.

Besides IS methods, numerous optimization methods have been applied to train Maxent. For example, Liu and Nocedal (1989), Bottou (2004), Daumé (2004), Keerthi et al. (2005), McDonald and Pereira (2006), Vishwanathan et al. (2006), Koh et al. (2007), Genkin et al. (2007), Andrew and Gao (2007), Schraudolph et al. (2007), Gao et al. (2007), Collins et al. (2008), Lin et al. (2008) and Friedman et al. (2008). They do not necessarily solve the optimization problem (3). Some handle more complicated log linear models such as Conditional Random Fields (CRF), but their approaches can be modified for Maxent. Some focus on logistic regression, which is a special form of Maxent if the number of labels is two. Moreover, some consider the L1 regularization term $\sum_t |w_t|$ in (3). Several papers have compared optimization methods for Maxent, though it is difficult to have a complete study. Malouf (2002) compares methods for NLP data, while Minka (2003) focuses on logistic regression for synthesis data. In this paper, we are interested in a detailed investigation of

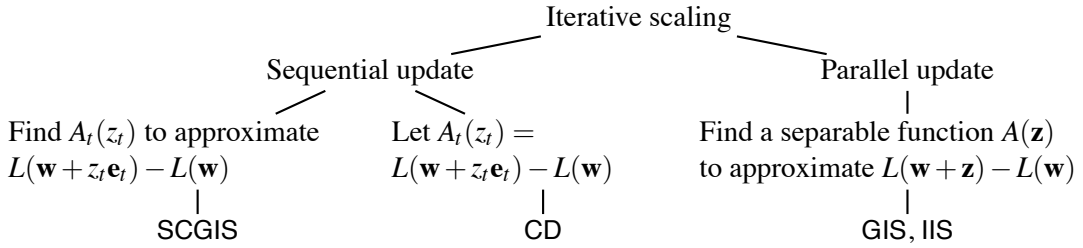


Figure 1: An illustration of various iterative scaling methods.

IS methods because they remain one of the most used approaches to train Maxent. This fact can be easily seen from popular NLP software. The Stanford Log-linear POS Tagger[1] supports two optimization methods, where one is IIS. The OpenNLP Maxent package (Baldridge et al., 2001) provides only one optimization method, which is GIS.

This paper is organized as follows. In Section 2, we present a unified framework for IS/CD methods and give theoretical results. Section 3 proposes a new CD method. Its advantages over existing IS/CD methods are discussed. In Section 4, we investigate some implementation issues for IS/CD methods. Section 5 presents experimental results. With a careful implementation, our CD outperforms IS and quasi-Newton techniques. Finally, Section 6 gives discussion and conclusions.

Part of this work appears in a short conference paper (Huang et al., 2009).

Notation X, Y, and n are the numbers of contexts, class labels, and features, respectively. The total number of nonzeros in training data and the average number of nonzeros per feature are respectively

$$\#\text{nz} \equiv \sum_{x,y} \sum_{t:f_t(x,y)\neq 0} 1 \quad \text{and} \quad \bar{l} \equiv \frac{\#\text{nz}}{n}. \tag{4}$$

In this paper, we assume non-negative feature values:

$$f_t(x,y) \geq 0,\ \forall t,x,y. \tag{5}$$

Most NLP applications have non-negative feature values. All existing IS methods use this property.

2. A Framework for Iterative Scaling and Coordinate Descent Methods

An important characteristic of IS and CD methods is that they solve a one-variable optimization problem and then modify the corresponding element in $\mathbf{w}$. Conceptually, the one-variable subproblem is related to the function reduction

$$L(\mathbf{w}+z_t\mathbf{e}_t)-L(\mathbf{w}),$$

where $\mathbf{e}_t \equiv [\underbrace{0,\ldots,0}_{t-1},1,0,\ldots,0]^T$. Then IS methods differ in how they approximate the function reduction. They can also be categorized according to whether $\mathbf{w}$'s components are updated in a sequential or parallel way. In this section, we create a framework for these methods. A hierarchical illustration of the framework is in Figure 1.

1. Stanford Log-linear POS Tagger can be found at `http://nlp.stanford.edu/software/tagger.shtml`.

2.1 The Framework

To introduce the framework, we separately discuss coordinate descent methods according to whether $\mathbf{w}$ is sequentially or parallely updated.

2.1.1 SEQUENTIAL UPDATE

For a sequential-update algorithm, once a one-variable sub-problem is solved, the corresponding element in $\mathbf{w}$ is updated. The new $\mathbf{w}$ is then used to construct the next sub-problem. The procedure is sketched in Algorithm 1. If the t-th component is selected for update, a sequential IS method solves the following one-variable sub-problem:

$$\min_{z_t} \; A_t(z_t),$$

where $A_t(z_t)$ is twice differentiable and bounds the function difference:

$$A_t(z_t) \geq L(\mathbf{w}+z_t\mathbf{e}_t) - L(\mathbf{w}), \; \forall z_t. \tag{6}$$

We hope that by minimizing $A_t(z_t)$, the resulting $L(\mathbf{w}+z_t\mathbf{e}_t)$ can be smaller than $L(\mathbf{w})$. However, (6) is not enough to ensure this property, so we impose an additional condition

$$A_t(0) = 0 \tag{7}$$

on the approximate function $A_t(z_t)$. The explanation below shows that we can strictly decrease the function value. If $A_t'(0) \neq 0$ and assume $\bar{z}_t \equiv \arg\min_{z_t} A_t(z_t)$ exists, with the condition $A_t(0) = 0$, we have $A_t(\bar{z}_t) < 0$. This property and (6) then imply $L(\mathbf{w}+\bar{z}_t\mathbf{e}_t) < L(\mathbf{w})$. If $A_t'(0) = 0$, we can prove that $\nabla_t L(\mathbf{w}) = 0$,[2] where $\nabla_t L(\mathbf{w}) = \partial L(\mathbf{w})/\partial w_t$. In this situation, the convexity of $L(\mathbf{w})$ and $\nabla_t L(\mathbf{w}) = 0$ imply that we cannot decrease the function value by modifying w_t, so we should move on to modify other components of $\mathbf{w}$.

A CD method can be viewed as a sequential-update IS method. Its approximate function $A_t(z_t)$ is simply the function difference:

$$A_t^{\mathsf{CD}}(z_t) = L(\mathbf{w}+z_t\mathbf{e}_t) - L(\mathbf{w}). \tag{8}$$

Other IS methods consider approximations so that $A_t(z_t)$ is simpler for minimization. More details are in Section 2.2. Note that the name "sequential" comes from the fact that each sub-problem $A_t(z_t)$ depends on $\mathbf{w}$ obtained from the previous update. Therefore, sub-problems must be sequentially solved.

2.1.2 PARALLEL UPDATE

A parallel-update IS method simultaneously constructs n independent one-variable sub-problems. After (approximately) solving all of them, the whole vector $\mathbf{w}$ is updated. Algorithm 2 gives the procedure. The function $A(\mathbf{z})$, $\mathbf{z} \in R^n$, is an approximation of $L(\mathbf{w}+\mathbf{z}) - L(\mathbf{w})$ satisfying

$$A(\mathbf{z}) \geq L(\mathbf{w}+\mathbf{z}) - L(\mathbf{w}), \; \forall \mathbf{z}, \quad A(\mathbf{0}) = 0, \quad \text{and} \quad A(\mathbf{z}) = \sum_{t=1}^{n} A_t(z_t). \tag{9}$$

2. Define a function $D(z_t) \equiv A(z_t) - (L(\mathbf{w}+z_t\mathbf{e}_t) - L(\mathbf{w}))$. We have $D'(0) = A'(0) - \nabla_t L(\mathbf{w})$. If $\nabla_t L(\mathbf{w}) \neq 0$ and $A_t'(0) = 0$, then $D'(0) \neq 0$. Since $D(0) = 0$, we can find a z_t such that $A(z_t) - (L(\mathbf{w}+z_t\mathbf{e}_t) - L(\mathbf{w})) < 0$, a contradiction to (6).

Algorithm 1 A sequential-update IS method

While $\mathbf{w}$ is not optimal

For $t = 1, \ldots, n$

1. Find an approximate function $A_t(z_t)$ satisfying (6)-(7).
2. Approximately solve $\min_{z_t} A_t(z_t)$ to get $\bar{z}_t$.
3. $w_t \leftarrow w_t + \bar{z}_t$.

Algorithm 2 A parallel-update IS method

While $\mathbf{w}$ is not optimal

1. Find approximate functions $A_t(z_t)\ \forall z_t$ satisfying (9).
2. For $t = 1, \ldots, n$

   Approximately solve $\min_{z_t} A_t(z_t)$ to get $\bar{z}_t$.
3. For $t = 1, \ldots, n$

   $w_t \leftarrow w_t + \bar{z}_t$.

The first two conditions are similar to (6) and (7). By a similar argument, we can ensure that the function value is strictly decreasing. The last condition indicates that $A(\mathbf{z})$ is separable, so

$$\min_{\mathbf{z}} A(\mathbf{z}) = \sum_{t=1}^{n} \min_{z_t} A_t(z_t).$$

That is, we can minimize $A_t(z_t),\ \forall z_t$ simultaneously, and then update $w_t\ \forall t$ together. We show in Section 4 that a parallel-update method possesses some nicer implementation properties than a sequential method. However, as sequential approaches update $\mathbf{w}$ as soon as a sub-problem is solved, they often converge faster than parallel methods.

If $A(\mathbf{z})$ satisfies (9), taking $\mathbf{z} = z_t \mathbf{e}_t$ implies that (6) and (7) hold for $A_t(z_t),\ \forall t = 1, \ldots, n$. A parallel-update method could thus be transformed to a sequential-update method using the same approximate function. Contrarily, a sequential-update algorithm cannot be directly transformed to a parallel-update method because the summation of the inequality in (6) does not imply (9).

2.2 Existing Iterative Scaling Methods

We introduce GIS, IIS and SCGIS via the proposed framework. GIS and IIS use a parallel update, but SCGIS is sequential. Their approximate functions aim to bound the change of the function values

$$L(\mathbf{w}+\mathbf{z}) - L(\mathbf{w}) = \sum_x \tilde{P}(x) \log \frac{T_{\mathbf{w}+\mathbf{z}}(x)}{T_{\mathbf{w}}(x)} + \sum_t Q_t(z_t), \tag{10}$$

where $T_{\mathbf{w}}(x)$ is defined in (1) and

$$Q_t(z_t) \equiv \frac{2w_t z_t + z_t^2}{2\sigma^2} - z_t \tilde{P}(f_t). \tag{11}$$

Then GIS, IIS and SCGIS use similar inequalities to get approximate functions. With

$$\begin{aligned} \frac{T_{\mathbf{w}+\mathbf{z}}(x)}{T_{\mathbf{w}}(x)} &= \frac{\sum_y S_{\mathbf{w}+\mathbf{z}}(x,y)}{T_{\mathbf{w}}(x)} = \frac{\sum_y S_{\mathbf{w}}(x,y)\left(e^{\sum_t z_t f_t(x,y)}\right)}{T_{\mathbf{w}}(x)} \\ &= \sum_y P_{\mathbf{w}}(y|x) e^{\sum_t z_t f_t(x,y)}, \end{aligned}$$

they apply $\log\alpha \le \alpha - 1\ \forall\alpha > 0$ and $\sum_y P_{\mathbf{w}}(y|x) = 1$ to get

$$\begin{aligned}(10) &\le \sum_t Q_t(z_t) + \sum_x \tilde{P}(x)\left(\sum_y P_{\mathbf{w}}(y|x)e^{\sum_t z_t f_t(x,y)} - 1\right)\\ &= \sum_t Q_t(z_t) + \sum_{x,y} \tilde{P}(x)P_{\mathbf{w}}(y|x)\left(e^{\sum_t z_t f_t(x,y)} - 1\right).\end{aligned} \tag{12}$$

GIS defines

$$f^{\#} \equiv \max_{x,y} f^{\#}(x,y), \quad f^{\#}(x,y) \equiv \sum_t f_t(x,y),$$

and adds a feature $f_{n+1}(x,y) \equiv f^{\#} - f^{\#}(x,y)$ with $z_{n+1} = 0$. Using Jensen's inequality and the assumption of non-negative feature values (5),

$$\begin{aligned}&e^{\sum_{t=1}^n z_t f_t(x,y)} = e^{\sum_{t=1}^{n+1} \frac{f_t(x,y)}{f^{\#}} z_t f^{\#}}\\ &\le \sum_{t=1}^{n+1} \frac{f_t(x,y)}{f^{\#}} e^{z_t f^{\#}} = \sum_{t=1}^{n} \frac{f_t(x,y)}{f^{\#}} e^{z_t f^{\#}} + \frac{f^{\#} - f^{\#}(x,y)}{f^{\#}} = \sum_{t=1}^{n}\left(\frac{e^{z_t f^{\#}} - 1}{f^{\#}} f_t(x,y)\right) + 1.\end{aligned} \tag{13}$$

Substituting (13) into (12), the approximate function of GIS is

$$A^{\mathsf{GIS}}(\mathbf{z}) = \sum_t Q_t(z_t) + \sum_{x,y} \tilde{P}(x)P_{\mathbf{w}}(y|x)\sum_t \left(\frac{e^{z_t f^{\#}} - 1}{f^{\#}} f_t(x,y)\right).$$

Then we obtain n independent one-variable functions:

$$A_t^{\mathsf{GIS}}(z_t) = Q_t(z_t) + \frac{e^{z_t f^{\#}} - 1}{f^{\#}} \sum_{x,y} \tilde{P}(x)P_{\mathbf{w}}(y|x) f_t(x,y).$$

IIS assumes $f_t(x,y) \ge 0$ and applies Jensen's inequality

$$e^{\sum_t z_t f_t(x,y)} = e^{\sum_t \frac{f_t(x,y)}{f^{\#}(x,y)} z_t f^{\#}(x,y)} \le \sum_t \frac{f_t(x,y)}{f^{\#}(x,y)} e^{z_t f^{\#}(x,y)}$$

on (12) to get the approximate function

$$A_t^{\mathsf{IIS}}(z_t) = Q_t(z_t) + \sum_{x,y} \tilde{P}(x)P_{\mathbf{w}}(y|x) f_t(x,y) \frac{e^{z_t f^{\#}(x,y)} - 1}{f^{\#}(x,y)}.$$

SCGIS is a sequential-update algorithm. It replaces $f^{\#}$ in GIS with

$$f_t^{\#} \equiv \max_{x,y} f_t(x,y). \tag{14}$$

Using $z_t\mathbf{e}_t$ as $\mathbf{z}$ in (10), a derivation similar to (13) gives

$$e^{z_t f_t(x,y)} \le \frac{f_t(x,y)}{f_t^{\#}} e^{z_t f_t^{\#}} + \frac{f_t^{\#} - f_t(x,y)}{f_t^{\#}}. \tag{15}$$

The approximate function of SCGIS is

$$A_t^{\mathsf{SCGIS}}(z_t) = Q_t(z_t) + \frac{e^{z_t f_t^\#} - 1}{f_t^\#} \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y).$$

As a comparison, we expand $A_t^{\mathsf{CD}}(z_t)$ in (8) to the following form:

$$A_t^{\mathsf{CD}}(z_t) = Q_t(z_t) + \sum_x \tilde{P}(x) \log \frac{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{T_{\mathbf{w}}(x)} \tag{16}$$

$$= Q_t(z_t) + \sum_x \tilde{P}(x) \log \left(1 + \sum_y P_{\mathbf{w}}(y|x)(e^{z_t f_t(x,y)} - 1)\right), \tag{17}$$

where (17) is from (1) and

$$S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y) = S_{\mathbf{w}}(x,y) e^{z_t f_t(x,y)}, \tag{18}$$

$$T_{\mathbf{w}+z_t\mathbf{e}_t}(x) = T_{\mathbf{w}}(x) + \sum_y S_{\mathbf{w}}(x,y)(e^{z_t f_t(x,y)} - 1). \tag{19}$$

A summary of approximate functions of IS and CD methods is in Table 1.

2.3 Convergence of Iterative Scaling and Coordinate Descent Methods

The convergence of CD methods has been well studied (e.g., Bertsekas, 1999; Luo and Tseng, 1992). However, for methods like IS which use only an approximate function to bound the function difference, the convergence is less studied. In this section, we generalize the linear convergence proof in Chang et al. (2008) to show the convergence of IS and CD methods. To begin, we consider any convex and differentiable function L: $R^n \to R$ satisfying the following conditions in the set

$$U = \{\mathbf{w} \mid L(\mathbf{w}) \le L(\mathbf{w}^0)\}, \tag{20}$$

where $\mathbf{w}^0$ is the initial point of an IS/CD algorithm:

1. ∇L is bi-Lipschitz: there are two positive constants $\tau_{\max}$ and $\tau_{\min}$ such that for any $\mathbf{u}, \mathbf{v} \in U$,

$$\tau_{\min}\|\mathbf{u} - \mathbf{v}\| \le \|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\| \le \tau_{\max}\|\mathbf{u} - \mathbf{v}\|. \tag{21}$$

2. Quadratic bound property: there is a constant $K > 0$ such that for any $\mathbf{u}, \mathbf{v} \in U$

$$|L(\mathbf{u}) - L(\mathbf{v}) - \nabla L(\mathbf{v})^T(\mathbf{u} - \mathbf{v})| \le K\|\mathbf{u} - \mathbf{v}\|^2. \tag{22}$$

The following theorem proves that (3) satisfies these two conditions.

Theorem 1 *$L(\mathbf{w})$ defined in* (3) *satisfies* (21) *and* (22).

The proof is in Section 7.1.

We denote $\mathbf{w}^k$ as the point after each iteration of the while loop in Algorithm 1 or 2. Hence from $\mathbf{w}^k$ to $\mathbf{w}^{k+1}$, n sub-problems are solved. The following theorem establishes our main linear convergence result for IS methods.

$$A_t^{\mathsf{GIS}}(z_t) = Q_t(z_t) + \frac{e^{z_t f^\#} - 1}{f^\#} \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)$$

$$A_t^{\mathsf{IIS}}(z_t) = Q_t(z_t) + \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) \frac{e^{z_t f^\#(x,y)} - 1}{f^\#(x,y)}$$

$$A_t^{\mathsf{SCGIS}}(z_t) = Q_t(z_t) + \frac{e^{z_t f_t^\#} - 1}{f_t^\#} \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)$$

$$A_t^{\mathsf{CD}}(z_t) = Q_t(z_t) + \sum_x \tilde{P}(x) \log\left(1 + \sum_y P_{\mathbf{w}}(y|x)(e^{z_t f_t(x,y)} - 1)\right)$$

Table 1: Approximate functions of IS and CD methods.

Theorem 2 *Consider Algorithm 1 or 2 to minimize a convex and twice differentiable function $L(\mathbf{w})$. Assume $L(\mathbf{w})$ attains a unique global minimum $\mathbf{w}^*$ and $L(\mathbf{w})$ satisfies* (21)-(22). *If the algorithm satisfies*

$$\|\mathbf{w}^{k+1} - \mathbf{w}^k\| \geq \eta \|\nabla L(\mathbf{w}^k)\|, \tag{23}$$

$$L(\mathbf{w}^{k+1}) - L(\mathbf{w}^k) \leq -\nu \|\mathbf{w}^{k+1} - \mathbf{w}^k\|^2, \tag{24}$$

for some positive constants η and ν, then the sequence $\{\mathbf{w}^k\}$ generated by the algorithm linearly converges. That is, there is a constant $\mu \in (0,1)$ such that

$$L(\mathbf{w}^{k+1}) - L(\mathbf{w}^*) \leq (1-\mu)(L(\mathbf{w}^k) - L(\mathbf{w}^*)), \forall k.$$

The proof is in Section 7.2. Note that this theorem is not restricted to $L(\mathbf{w})$ in (3). Next, we show that IS/CD methods discussed in this paper satisfy (23)-(24), so they all possess the linear convergence property.

Theorem 3 *Consider $L(\mathbf{w})$ defined in* (3) *and assume $A_t(z_t)$ is exactly minimized in* GIS, IIS, SCGIS, *or* CD. *Then $\{\mathbf{w}^k\}$ satisfies* (23)-(24).

The proof is in Section 7.3.

2.4 Solving One-variable Sub-problems

After generating approximate functions, GIS, IIS, SCGIS and CD need to minimize one-variable sub-problems. In general, the approximate function possesses a unique global minimum. We do not discuss some rare situations where this property does not hold (for example, $\min_{z_t} A_t^{\mathsf{GIS}}(z_t)$ has an optimal solution $z_t = -\infty$ if $\tilde{P}(f_t) = 0$ and the regularization term is not considered).

Without the regularization term, by $A_t'(z_t) = 0$, GIS and SCGIS both have a simple closed-form solution of the sub-problem:

$$z_t = \frac{1}{f^s} \log\left(\frac{\tilde{P}(f_t)}{\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)}\right), \quad \text{where } f^s \equiv \begin{cases} f^\# \text{ if } s \text{ is GIS}, \\ f_t^\# \text{ if } s \text{ is SCGIS}. \end{cases} \tag{25}$$

For IIS, the term $e^{z_t f^\#(x,y)}$ in $A_t^{\mathsf{IIS}}(z_t)$ depends on x and y, so it does not have a closed-form solution. CD does not have a closed-form solution either.

With the regularization term, the sub-problems no longer have a closed-form solution. While many optimization methods can be applied, in this section we analyze the complexity of using the Newton method to solve one-variable sub-problems. The Newton method minimizes $A_t^s(z_t)$ by iteratively updating z_t:

$$z_t \leftarrow z_t - A_t^{s\prime}(z_t)/A_t^{s\prime\prime}(z_t), \tag{26}$$

where s indicates an IS or a CD method. This iterative procedure may diverge, so we often need a line search procedure to ensure the function value is decreasing (Fletcher, 1987, p. 47). Due to the many variants of line searches, here we discuss only the cost for finding the Newton direction. The Newton directions of GIS and SCGIS are similar:

$$-\frac{A_t^{s\prime}(z_t)}{A_t^{s\prime\prime}(z_t)} = -\frac{Q_t'(z_t) + e^{z_t f^s}\sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)}{Q_t''(z_t) + f^s e^{z_t f^s}\sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)}, \tag{27}$$

where f^s is defined in (25). For IIS, the Newton direction is:

$$-\frac{A_t^{\mathsf{IIS}\prime}(z_t)}{A_t^{\mathsf{IIS}\prime\prime}(z_t)} = -\frac{Q_t'(z_t) + \sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)e^{z_t f^\#(x,y)}}{Q_t''(z_t) + \sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)f^\#(x,y)e^{z_t f^\#(x,y)}}. \tag{28}$$

The Newton directions of CD is:

$$-\frac{A_t^{\mathsf{CD}\prime}(z_t)}{A_t^{\mathsf{CD}\prime\prime}(z_t)}, \tag{29}$$

where

$$A_t^{\mathsf{CD}\prime}(z_t) = Q_t'(z_t) + \sum_{x,y}\tilde{P}(x)P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y), \tag{30}$$

$$A_t^{\mathsf{CD}\prime\prime}(z_t) = Q_t''(z_t) + \sum_{x,y}\tilde{P}(x)P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)^2 - \sum_x \tilde{P}(x)\left(\sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)\right)^2. \tag{31}$$

Eqs. (27)-(28) can be easily obtained using formulas in Table 1. We show details of deriving (30)-(31) in Section 7.4.

We separate the complexity analysis to two parts. One is on calculating of $P_{\mathbf{w}}(y|x)\ \forall x,y$, and the other is on the remaining operations.

For $P_{\mathbf{w}}(y|x) = S_{\mathbf{w}}(x,y)/T_w(x)$, parallel-update approaches calculate it once every n sub-problems. To get $S_{\mathbf{w}}(x,y)\ \forall x,y$, the operation

$$\sum_t w_t f_t(x,y) \quad \forall x,y$$

needs $O(\#\text{nz})$ time. If $XY \le \#\text{nz}$, the cost for obtaining $P_{\mathbf{w}}(y|x),\ \forall x,y$ is $O(\#\text{nz})$, where X and Y are respectively the numbers of contexts and labels.[3] Therefore, on average each sub-problem shares $O(\#\text{nz}/n) = O(\bar{l})$ cost. For sequential-update methods, they expensively update $P_{\mathbf{w}}(y|x)$ after every

3. If $XY > \#\text{nz}$, one can calculate $e^{w_t f_t(x,y)},\ \forall f_t(x,y) \neq 0$ and then the product $\prod_{t:f_t(x,y)\neq 0} e^{w_t f_t(x,y)}$. The complexity is still $O(\#\text{nz})$.

	CD	GIS	SCGIS	IIS
1st Newton direction	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$
Each subsequent Newton direction	$O(\bar{l})$	$O(1)$	$O(1)$	$O(\bar{l})$

Table 2: Cost for finding Newton directions if the Newton method is used to minimize $A_t(z_t)$.

sub-problem. A trick to trade memory for time is to store all $S_{\mathbf{w}}(x,y)$ and $T_{\mathbf{w}}(x)$, and use (18) and (19). Since $S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y) = S_{\mathbf{w}}(x,y)$, if $f_t(x,y) = 0$, this procedure reduces the number of operations from the $O(\#\text{nz})$ operations to $O(\bar{l})$. However, it needs $O(XY)$ extra spaces to store all $S_{\mathbf{w}}(x,y)$ and $T_{\mathbf{w}}(x)$. This trick has been used in the SCGIS method (Goodman, 2002).

From (27) and (28), all remaining operations of GIS, IIS, and SCGIS involve the calculation of

$$\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) (\text{a function of } z_t), \tag{32}$$

which needs $O(\bar{l})$ under a fixed t. For GIS and SCGIS, since the function of z_t in (32) is independent of x, y, we can calculate and store $\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)$ in the first Newton iteration. Therefore, the overall cost (including calculating $P_{\mathbf{w}}(y|x)$) is $O(\bar{l})$ for the first Newton iteration and $O(1)$ for each subsequent iteration. For IIS, because $e^{z_t f^{\#}(x,y)}$ in (28) depends on x and y, we need $O(\bar{l})$ for every Newton direction. For CD, it calculates $P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)$ for every z_t, so the cost per Newton direction is $O(\bar{l})$. We summarize the cost for solving sub-problems of GIS, SCGIS, IIS and CD in Table 2.

2.5 Related Work

Our framework for IS methods includes two important components:

1. Approximate $L(\mathbf{w} + z_t\mathbf{e}_t) - L(\mathbf{w})$ or $L(\mathbf{w} + \mathbf{z}) - L(\mathbf{w})$ to obtain functions $A_t(z_t)$.

2. Sequentially or parallely minimize approximate functions.

Each component has been well discussed in many places. However, ours may be the first to investigate IS methods in detail. Below we discuss some related work.

The closest work to our framework might be Lange et al. (2000) from the statistics community. They discuss "optimization transfer" algorithms which construct $A_t(z_t)$ or $A(\mathbf{z})$ satisfying conditions similar to (6)-(7) or (9). However, they do not require one-variable sub-problems, so $A(\mathbf{z})$ of a parallel-update method may be non-separable. They discuss that "optimization transfer" algorithms can be traced back to EM (Expectation Maximization). In their paper, the function $A_t(z_t)$ or $A(\mathbf{z})$ is called a "surrogate" function or a "majorizing" function. Some also call it an "auxiliary" function. Lange et al. (2000) further discuss several ways to construct $A(\mathbf{z})$, where Jensen's inequality used in (13) is one of them. An extension along this line of research is by Zhang et al. (2007).

The concept of sequential- and parallel-update algorithms is well known in many subjects. For example, these algorithms are used in iterative methods for solving linear systems (Jacobi and Gauss-Seidel methods). Some recent machine learning works which mention them include, for example, Collins et al. (2002) and Dudík et al. (2004). Dudík et al. (2004) propose a variant of IS methods for L1-regularized maximum entropy. They consider both sequential- and parallel-update

algorithms using certain approximate functions. Their sequential methods greedily choose coordinates minimizing $A_t(z_t)$, while ours in Section 2.1.1 chooses coordinates cyclicly.

Regarding the convergence, if the sub-problem has a closed-form solution like (25), it is easy to apply the result in Lange et al. (2000). However, the case with regularization is more complicated. For example, Dudík et al. (2004) point out that Goodman (2002) does not give a "complete proof of convergence." Note that the strict decrease of function values following conditions (6)-(7) or (9) does not imply the convergence to the optimal function value. In Section 2.3, we prove not only the global convergence but also the linear convergence for a general class of IS/CD methods.

3. Comparison and a New Coordinate Descent Method

Using the framework in Section 2, we compare CD and IS methods in this section. Based on the comparison, we propose a new and fast CD method.

3.1 Comparison of Iterative Scaling and Coordinate Descent Methods

An IS or CD method falls into a place between two extreme designs:

$$\begin{array}{l} A_t(z_t) \text{ a loose bound} \\ \text{Easy to minimize } A_t(z_t) \end{array} \iff \begin{array}{l} A_t(z_t) \text{ a tight bound} \\ \text{Hard to minimize } A_t(z_t) \end{array}$$

That is, there is a tradeoff between the tightness to bound the function difference and the hardness to solve the sub-problem. To check how IS and CD methods fit into this explanation, we obtain the following relationship of their approximate functions:

$$\begin{aligned} &A_t^{\mathsf{CD}}(z_t) \le A_t^{\mathsf{SCGIS}}(z_t) \le A_t^{\mathsf{GIS}}(z_t), \\ &A_t^{\mathsf{CD}}(z_t) \le A_t^{\mathsf{IIS}}(z_t) \le A_t^{\mathsf{GIS}}(z_t) \quad \forall\, z_t. \end{aligned} \tag{33}$$

The derivation is in Section 7.5. From (33), CD considers more accurate sub-problems than SCGIS and GIS. However, when solving the sub-problem, from Table 2, CD's each Newton step takes more time. The same situation occurs in comparing IIS and GIS.

The above discussion indicates that while a tight $A_t(z_t)$ can give faster convergence by handling fewer sub-problems, the total time may not be less due to the higher cost of each sub-problem.

3.2 A Fast Coordinate Descent Method

Based on the discussion in Section 3.1, we develop a CD method which is cheaper in solving each sub-problem but still enjoys fast final convergence. This method is modified from Chang et al. (2008), a CD approach for linear SVM. They approximately minimize $A_t^{\mathsf{CD}}(z_t)$ by applying only one Newton iteration. This approach is a truncated Newton method: In the early stage of the coordinate descent method, we roughly minimize $A_t^{\mathsf{CD}}(z_t)$ but in the final stage, one Newton update can quite accurately solve the sub-problem. The Newton direction at $z_t = 0$ is

$$d = -\frac{A_t^{\mathsf{CD}\prime}(0)}{A_t^{\mathsf{CD}\prime\prime}(0)}. \tag{34}$$

We discuss in Section 2.4 that the update rule (26) may not decrease the function value. Hence we need a line search procedure to find $\lambda \ge 0$ such that $z_t = \lambda d$ satisfies the following sufficient

Algorithm 3 A fast coordinate descent method for Maxent

- Choose $\beta \in (0,1)$ and $\gamma \in (0,1/2)$. Give initial $\mathbf{w}$ and calculate $S_{\mathbf{w}}(x,y)$, $T_{\mathbf{w}}(x)$, $\forall x,y$.
- While $\mathbf{w}$ is not optimal
  - For $t = 1,\ldots,n$
    1. Calculate the Newton direction

$$\begin{aligned} d &= -A_t^{\mathsf{CD}'}(0)/A_t^{\mathsf{CD}''}(0) \\ &= \frac{-\left(\sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)+\frac{w_t}{\sigma^2}\right)}{\sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y)^2-\sum_x\tilde{P}(x)\left(\sum_y P_{\mathbf{w}}(y|x)f_t(x,y)\right)^2+\frac{1}{\sigma^2}}, \end{aligned}$$

       where

$$P_{\mathbf{w}}(y|x) = \frac{S_{\mathbf{w}}(x,y)}{T_{\mathbf{w}}(x)}.$$

    2. While $\lambda = 1,\beta,\beta^2,\ldots$
       (a) Let $z_t = \lambda d$
       (b) Calculate

$$A_t^{\mathsf{CD}}(z_t) = Q_t(z_t) + \sum_x \tilde{P}(x)\log\left(1+\sum_y \frac{S_{\mathbf{w}}(x,y)}{T_{\mathbf{w}}(x)}(e^{z_t f_t(x,y)}-1)\right)$$

       (c) If $A_t^{\mathsf{CD}}(z_t) \le \gamma z_t A_t^{\mathsf{CD}'}(0)$, then break.
    3. $w_t \leftarrow w_t + z_t$
    4. Update $S_{\mathbf{w}}(x,y)$ and $T_{\mathbf{w}}(x)$ $\forall x,y$ by (18)-(19)

decrease condition:

$$A_t^{\mathsf{CD}}(z_t) - A_t^{\mathsf{CD}}(0) = A_t^{\mathsf{CD}}(z_t) \le \gamma z_t A_t^{\mathsf{CD}'}(0) \le 0, \tag{35}$$

where γ is a constant in $(0,1/2)$. Note that $z_t A_t^{\mathsf{CD}'}(0)$ is negative under the definition of d in (34). Instead of (35), Grippo and Sciandrone (1999) and Chang et al. (2008) use

$$A_t^{\mathsf{CD}}(z_t) \le -\gamma z_t^2 \tag{36}$$

as the sufficient decrease condition. We prefer (35) as it is scale-invariant. That is, if A_t^{CD} is linearly scaled, then (35) holds under the same γ. In contrast, γ in (36) may need to be changed. To find λ for (35), a simple way is by sequentially checking $\lambda = 1,\beta,\beta^2,\ldots$, where $\beta \in (0,1)$. We choose β as 0.5 for experiments. The following theorem proves that the condition (35) can always be satisfied.

Theorem 4 *Given the Newton direction d as in* (34). *There is $\bar{\lambda} > 0$ such that $z_t = \lambda d$ satisfies* (35) *for all $0 \le \lambda < \bar{\lambda}$.*

The proof is in Section 7.6. The new CD procedure is in Algorithm 3. In the rest of this paper, we refer to CD as this new algorithm.

In Section 2.3 we prove the linear convergence of IS/CD methods. In Section 7.7, we use the same framework to prove that Algorithm 3 linearly converges:

Theorem 5 *Algorithm 3 satisfies* (23)-(24) *and linearly converges to the global optimum of* (3).

As evaluating $A_t^{\mathsf{CD}}(z_t)$ via (17)-(19) needs $O(\bar{l})$ time, the line search procedure takes

$$O(\bar{l}) \times (\#\text{ line search steps}).$$

This causes the cost of solving a sub-problem higher than that of GIS/SCGIS (see Table 2). Fortunately, we show that near the optimum, the line search procedure needs only one step:

Theorem 6 *In a neighborhood of the optimal solution, the Newton direction d defined in* (34) *satisfies the sufficient decrease condition* (35) *with* $\lambda = 1$.

The proof is in Section 7.8. If the line search procedure succeeds at $\lambda = 1$, then the cost for each sub-problem is similar to that of GIS and SCGIS.

Next we show that near the optimum, one Newton direction of CD's tight $A_t^{\mathsf{CD}}(z_t)$ already reduces the objective function $L(\mathbf{w})$ more rapidly than directions by exactly minimizing a loose $A_t(z_t)$ of GIS, IIS or SCGIS. Thus Algorithm 3 has faster final convergence than GIS, IIS, or SCGIS.

Theorem 7 *Assume* $\mathbf{w}^*$ *is the global optimum of* (3). *There is an* $\varepsilon > 0$ *such that the following result holds. For any* $\mathbf{w}$ *satisfying* $\|\mathbf{w} - \mathbf{w}^*\| \le \varepsilon$, *if we select an index t and generate directions*

$$d = -A_t^{\mathsf{CD}'}(0)/A_t^{\mathsf{CD}''}(0) \quad \textit{and} \quad d^s = \arg\min_{z_t} A_t^s(z_t), \quad s = \mathsf{GIS}, \mathsf{IIS} \textit{ or } \mathsf{SCGIS}, \tag{37}$$

then

$$\delta_t(d) < \min\left(\delta_t(d^{\mathsf{GIS}}), \delta_t(d^{\mathsf{IIS}}), \delta_t(d^{\mathsf{SCGIS}})\right),$$

where

$$\delta_t(z_t) \equiv L(\mathbf{w} + z_t \mathbf{e}_t) - L(\mathbf{w}).$$

The proof is in Section 7.9. Theorems 6 and 7 show that Algorithm 3 improves upon the traditional CD by approximately solving sub-problems, while still maintaining fast convergence. That is, it attempts to take both advantages of the two designs mentioned in Section 3.1.

3.2.1 EFFICIENT LINE SEARCH

We propose a technique to speed up the line search procedure. We derive a function $\bar{A}_t^{\mathsf{CD}}(z_t)$ so that it is cheaper to calculate than $A_t^{\mathsf{CD}}(z_t)$ and satisfies $\bar{A}_t^{\mathsf{CD}}(z_t) \ge A_t^{\mathsf{CD}}(z_t)\ \forall z_t$. Then,

$$\bar{A}_t^{\mathsf{CD}}(z_t) \le \gamma z_t A_t^{\mathsf{CD}'}(0) \tag{38}$$

implies (35), so we can save time by replacing step 2 of Algorithm 3 with

2'. While $\lambda = 1, \beta, \beta^2, \ldots$
   (a) Let $z_t = \lambda d$
   (b) Calculate $\bar{A}_t^{\mathsf{CD}}(z_t)$
   (c) If $\bar{A}_t^{\mathsf{CD}}(z_t) \le \gamma z_t A_t^{\mathsf{CD}'}(0)$, then break.
   (d) Calculate $A_t^{\mathsf{CD}}(z_t)$
   (e) If $A_t^{\mathsf{CD}}(z_t) \le \gamma z_t A_t^{\mathsf{CD}'}(0)$, then break.

We assume non-negative feature values and obtain

$$\bar{A}_t^{\mathsf{CD}}(z_t) \equiv Q_t(z_t) + \tilde{P}_t \log\left(1 + \frac{e^{z_t f_t^{\#}} - 1}{f_t^{\#} \tilde{P}_t} \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)\right), \tag{39}$$

where $f_t^{\#}$ is defined in (14),

$$\tilde{P}_t \equiv \sum_{\Omega_t} \tilde{P}(x), \quad \text{and} \quad \Omega_t \equiv \{x : \exists y \text{ such that } f_t(x,y) \neq 0\}. \tag{40}$$

The derivation is in Section 7.10. Because

$$\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y),\ t = 1,\ldots,n \tag{41}$$

are available from finding $A_t^{\mathsf{CD}'}(0)$, getting $\bar{A}_t^{\mathsf{CD}}(z_t)$ and checking (38) take only $O(1)$, smaller than $O(\bar{l})$ for (35). Using $\log\alpha \leq \alpha - 1\ \forall \alpha > 0$, it is easy to see that

$$\bar{A}_t^{\mathsf{CD}}(z_t) \leq A_t^{\mathsf{SCGIS}}(z_t),\ \forall z_t.$$

Therefore, we can simply replace $A_t^{\mathsf{SCGIS}}(z_t)$ of the SCGIS method with $\bar{A}_t^{\mathsf{CD}}(z_t)$ to have a new IS method.

4. Implementation Issues

In this section we analyze some implementation issues of IS and CD methods.

4.1 Row Versus Column Format

In many Maxent applications, data are sparse with few nonzero $f_t(x,y)$. We store such data by a sparse matrix. Among many ways to implement sparse matrices, two common ones are "row format" and "column format." For the row format, each (x,y) corresponds to a list of nonzero $f_t(x,y)$, while for the column format, each feature t is associated with a list of (x,y). The loop to access data in the row format is $(x,y) \rightarrow t$, while for the column format it is $t \rightarrow (x,y)$. By $(x,y) \rightarrow t$ we mean that the outer loop goes through (x,y) values and for each (x,y), there is an inner loop for a list of feature values. For sequential-update algorithms such as SCGIS and CD, as we need to maintain $S_{\mathbf{w}}(x,y)\ \forall x,y$ via (18) after solving the t-th sub-problem, an easy access of t's corresponding (x,y) elements is essential. Therefore, the column format is more suitable. In contrast, parallel-update methods can use either row or column formats. For GIS, we can store all n elements of (41) before solving n sub-problems by (25) or (27). The calculation of (41) can be done by using the row format and a loop of $(x,y) \rightarrow t$. For IIS, an implementation by the row format is more complicated due to the $e^{z_t f^{\#}(x,y)}$ term in $A_t^{\mathsf{IIS}}(z_t)$. Take the Newton method to solve the sub-problem as an example. We can calculate and store (28) for all $t = 1,\ldots,n$ by a loop of $(x,y) \rightarrow t$. That is, n Newton directions are obtained together before conducting n updates.

4.2 Memory Requirement

For sequential-update methods, to save the computational time of calculating $P_{\mathbf{w}}(y|x)$, we use (18)-(19), so $S_{\mathbf{w}}(x,y)\ \forall x,y$ must be stored. Therefore, $O(XY)$ storage is needed. For parallel-update

Data set	X	Y	n	#nz
CoNLL2000-P	197,979	44	168,674	48,030,163
CoNLL2000-C	197,252	22	273,680	53,396,844
BROWN	935,137	185	626,726	601,216,661

Table 3: Statistics of NLP data (0/1 features). X: number of contexts, Y: number of class labels, n: number of features, and #nz: number of total non-zero feature values; see (4).

methods, they also need $O(XY)$ spaces if using the column format: To calculate $e^{\sum_t w_t f_t(x,y)}\ \forall x,y$ via a loop of $t \to (x,y)$, we need $O(XY)$ positions to store $\sum_t w_t f_t(x,y)\ \forall x,y$. In contrast, if using the row format, the loop is $x \to y \to t$, so for each fixed x, we need only $O(Y)$ spaces to store $S(x,y)\ \forall y$ and then obtain $T_{\mathbf{w}}(x)$. This advantage makes the parallel update a viable approach if Y (the number of labels) is very large.

4.3 Number of exp and log Operations

Many exp/log operations are needed in training a Maxent model. On most computers, exp/log operations are much more expensive than multiplications/divisions. It is important to analyze the number of exp/log operations in IS and CD methods.

We first discuss the number of exp operations. A simple check of (27)-(31) shows that the numbers are the same as those in Table 2. IIS and CD need $O(\bar{l})$ exp operations for every Newton direction because they calculate $e^{z_t f^{\#}(x,y)}$ in (28) and $e^{z_t f_t(x,y)}$ in (17), respectively. CD via Algorithm 3 takes only one Newton iteration, but each line search step also needs $O(\bar{l})$ exp operations. If feature values are binary, $e^{z_t f_t(x,y)}$ in (17) becomes e^{z_t}, a value independent of x,y. Thus the number of exp operations is significantly reduced from $O(\bar{l})$ to $O(1)$. This property implies that Algorithm 3 is more efficient if data are binary valued. In Section 5, we will confirm this result through experiments.

Regarding log operations, GIS, IIS and SCGIS need none as they remove the log function in $A_t(z_t)$. CD via Algorithm 3 keeps log in $A_t^{\mathsf{CD}}(z_t)$ and requires $O(|\Omega_t|)$ log operations at each line search step, where Ω_t is defined in (40).

4.4 Permutation of Indices in Solving Sub-problems

For sequential-update methods, one does not have to follow a cyclic way to update $w_1,\ldots,w_n$. Chang et al. (2008) report that in their CD method, a permutation of $\{1,\ldots,n\}$ as the order for solving n sub-problems leads to faster convergence. For sequential-update IS methods adopting this strategy, the linear convergence in Theorem 2 still holds.

5. Experiments

In this section, we compare IS/CD methods to reconfirm properties discussed in earlier sections. We consider two types of data for NLP (Natural Language Processing) applications. One is Maxent for 0/1-featured data and the other is Maxent (logistic regression) for document data with non-negative real-valued features. Programs used for experiments in this paper are online available at
`http://www.csie.ntu.edu.tw/~cjlin/liblinear/exp.html`.

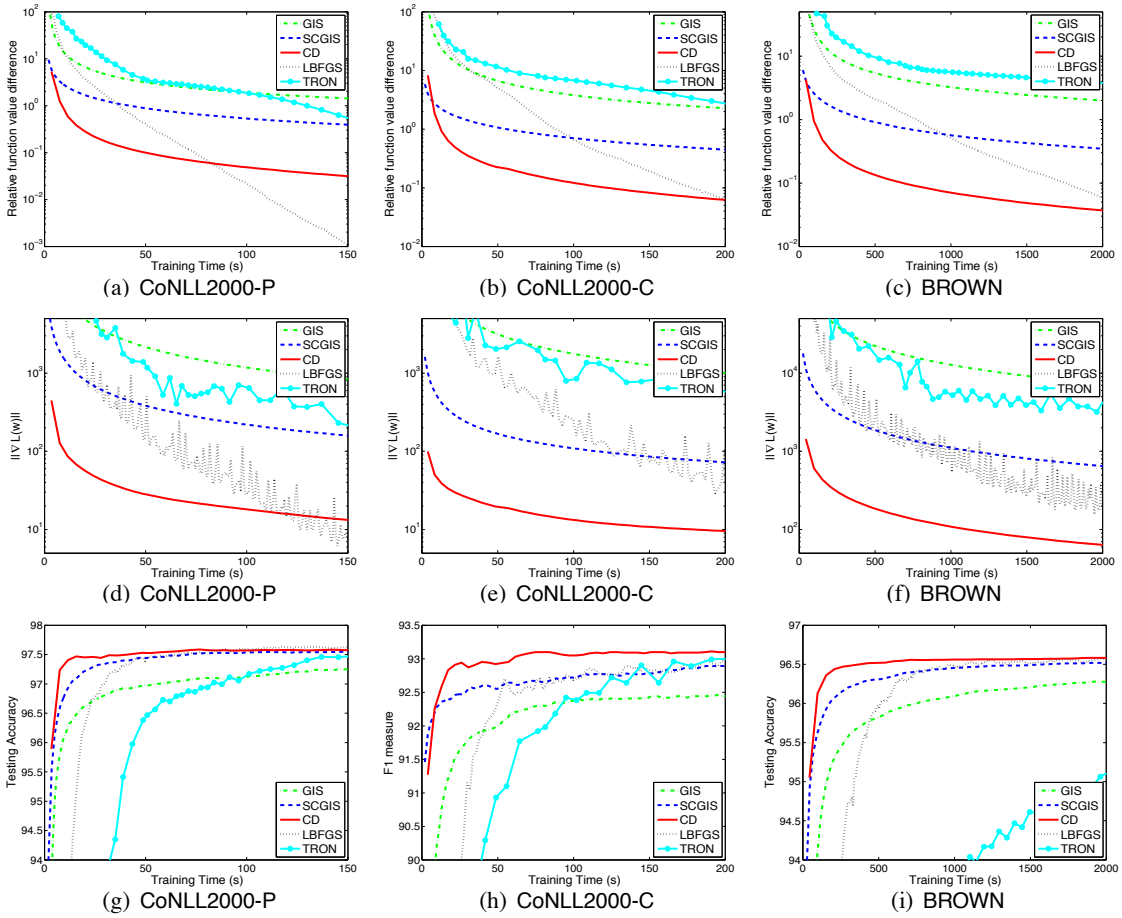


Figure 2: Results on 0/1-featured data. The first row shows time versus the relative function difference (42). The second and third rows show $\|\nabla L(\mathbf{w})\|$ and testing performances along time, respectively. Time is in seconds.

5.1 Maxent for 0/1-featured Data in NLP

We apply Maxent models to part of speech (POS) tagging and chunking tasks. In POS tagging, we mark a POS tag to the word in a text based on both its definition and context. In a chunking task, we divide a text into syntactically correlated parts of words. That is, given words in a sentence annotated with POS tags, we label each word with a chunk tag. Other learning models such as CRF (Conditional Random Fields) may outperform Maxent for these NLP applications. However, we do not consider other learning models as the focus of this paper is to study IS methods for Maxent.

We use CoNLL2000 shared task data[4] for chunking and POS tagging, and BROWN corpus[5] for POS tagging. CoNLL2000-P indicates CoNLL2000 for POS tagging, and CoNLL2000-C means CoNLL2000 for chunking. CoNLL2000 data consist of Sections 15-18 of the Wall Street Journal corpus as training data and Section 20 as testing data. For the BROWN corpus, we randomly se-

4. Data can be found at `http://www.cnts.ua.ac.be/conll2000/chunking`.
5. Corpus can be found at `http://www.nltk.org`.

lect four-fifth articles for training and use the rest for testing. We omit the stylistic tag modifiers "fw,""tl,""nc,"and "hl," so the number of labels is reduced from 472 to 185. Our implementation is built upon the OpenNLP package (Baldridge et al., 2001). We use the default setting of OpenNLP to extract binary features (0/1 values) suggested by Ratnaparkhi (1998). The OpenNLP implementation assumes that each feature index t corresponds to a unique label y. In prediction, we approximately maximize the probability of tag sequences to the word sequences by a beam search (Ratnaparkhi, 1998). Table 3 lists the statistics of data sets.

We implement the following methods for comparisons.

1. GIS and SCGIS: To minimize $A_t(z_t)$, we run Newton updates (without line search) until $|A'_t(z_t)| \leq 10^{-5}$. We can afford many Newton iterations because, according to Table 2, each Newton direction costs only $O(1)$ time.

2. CD: the coordinate descent method proposed in Section 3.2.

3. LBFGS: a limited memory quasi Newton method for general unconstrained optimization problems (Liu and Nocedal, 1989).

4. TRON: a trust region Newton method for logistic regression (Lin et al., 2008). We extend the method for Maxent.

We consider LBFGS as Malouf (2002) reports that it is better than other approaches including GIS and IIS. Lin et al. (2008) show that TRON is faster than LBFGS for document classification, so we include TRON for comparison. We exclude IIS because of its higher cost per Newton direction than GIS/SCGIS (see Table 2). Indeed Malouf (2002) reports that GIS outperforms IIS. Our implementation of all methods takes the property of 0/1 features. We use the regularization parameter $\sigma^2 = 10$ as under this value Maxent models achieve good testing performances. We set $\beta = 0.5$ and $\gamma = 0.001$ for the line search procedure (35) in CD. The initial $\mathbf{w}$ of all methods is $\mathbf{0}$.

We begin at checking time versus the relative difference of the function value to the optimum:

$$\frac{L(\mathbf{w}) - L(\mathbf{w}^*)}{L(\mathbf{w}^*)}, \tag{42}$$

where $\mathbf{w}^*$ is the optimal solution of (3). As $\mathbf{w}^*$ is not available, we obtain a reference point satisfying $\|\nabla L(\mathbf{w})\| \leq 0.01$. Results are in the first row of Figure 2. Next, we check these methods' gradient values. As $\|\nabla L(\mathbf{w})\| = 0$ implies that $\mathbf{w}$ is the global minimum, usually $\|\nabla L(\mathbf{w})\|$ is used in a stopping condition. The second row of Figure 2 shows time versus $\|\nabla L(\mathbf{w})\|$. We are also interested in the time needed to achieve a reasonable testing result. We measure the performance of POS tagging by accuracy and chunking by F1 measure. The third row of Figure 2 presents the testing accuracy/F1 versus training time. Note that (42) and $\|\nabla L(\mathbf{w})\|$ in Figure 2 are both log scaled.

We give some observations from Figure 2. Among the three IS/CD methods compared, the new CD approach discussed in Section 3.2 is the fastest. SCGIS comes the second, while GIS is the last. This result is consistent with the tightness of their approximate functions; see (33). Regarding IS/CD methods versus LBFGS/TRON, the three IS/CD methods more quickly decrease the function value in the beginning, but LBFGS has faster final convergence. In fact, if we draw figures with longer training time, TRON's final convergence is the fastest. This result is reasonable as LBFGS and TRON respectively have superlinear and quadratic convergence, higher than the linear rate proved in Theorem 2 for IS methods. The choice of methods thus relies on whether one prefers getting a

Problem	l	n	#nz	σ^2
astro-physic	62,369	99,757	4,834,550	$8l$
yahoo-japan	176,203	832,026	23,506,415	$4l$
rcv1	677,399	47,236	49,556,258	$8l$

Table 4: Statistics of document data (real-valued features). l: number of instances, n: number of features, #nz: number of total non-zero feature values, and σ^2: best regularization parameter from five-fold cross validation.

reasonable model quickly (IS/CD methods) or accurately minimizing the function (LBFGS/TRON). Practically CD/IS may be more useful as they reach the final testing accuracy rapidly. Finally, we compare LBFGS and TRON. Surprisingly, LBFGS outperforms TRON, a result opposite to that in Lin et al. (2008). We do not have a clear explanation yet. A difference is that Lin et al. (2008) deal with document data of real-valued features, but here we have 0/1-featured NLP applications. Therefore, one should always be careful that for the same approaches, observations made on one type of data may not extend to another.

In Section 4, we discussed a strategy of permuting n sub-problems to speed up the convergence of sequential-update IS methods. However, in training Maxent models for 0/1-featured NLP data, with/without permutation gives similar performances. We find that this strategy tends to work better if features are related. Hence we suspect that features used in POS tagging or chunking tasks are less correlated than those in documents and the order of sub-problems is not very important.

5.2 Maxent (Logistic Regression) for Document Classification

In this section, we experiment with logistic regression on document data with non-negative real-valued features. Chang et al. (2008) report that their CD method is very efficient for linear SVM, but is slightly less effective for logistic regression. They attribute the reason to that logistic regression requires expensive exp/log operations. In Section 4, we show that for 0/1 features, the number of IS methods' exp operations is smaller. Experiments here help to check if IS/CD methods are more suitable for 0/1 features than real values.

Logistic regression is a special case of maximum entropy with two labels $+1$ and -1. Consider training data $\{\bar{\mathbf{x}}_i, \bar{y}_i\}_{i=1}^{l}, \bar{\mathbf{x}}_i \in R^n, \bar{y}_i = \{1,-1\}$. Assume $\bar{x}_{it} \geq 0,\ \forall i,t$. We set the feature $f_t(x_i,y)$ as

$$f_t(x_i,y) = \begin{cases} \bar{x}_{it} & \text{if } y = 1, \\ 0 & \text{if } y = -1, \end{cases}$$

where x_i denotes the index of the i-th training instance $\bar{\mathbf{x}}_i$. Then

$$S_{\mathbf{w}}(x_i,y) = e^{\sum_t w_t f_t(x_i,y)} = \begin{cases} e^{\mathbf{w}^T \bar{\mathbf{x}}_i} & \text{if } y = 1, \\ 1 & \text{if } y = -1, \end{cases}$$

and

$$P_{\mathbf{w}}(y|x_i) = \frac{S_{\mathbf{w}}(x_i,y)}{T_{\mathbf{w}}(x_i)} = \frac{1}{1+e^{-y\mathbf{w}^T \bar{\mathbf{x}}_i}}. \tag{43}$$

From (2) and (43),

$$L(\mathbf{w}) \quad = \frac{1}{2\sigma^2}\sum_t w_t^2 + \frac{1}{l}\sum_i \log\left(1+e^{-\bar{y}_i \mathbf{w}^T \bar{\mathbf{x}}_i}\right)$$

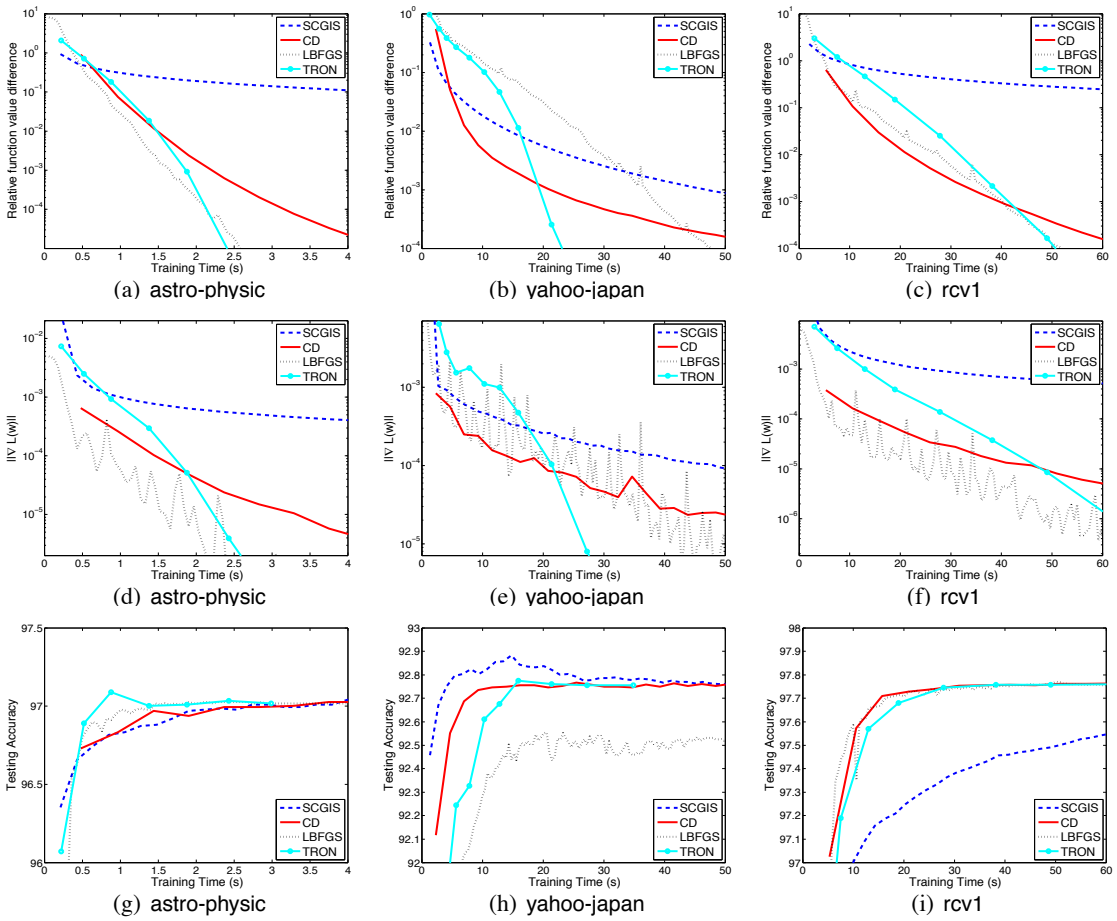


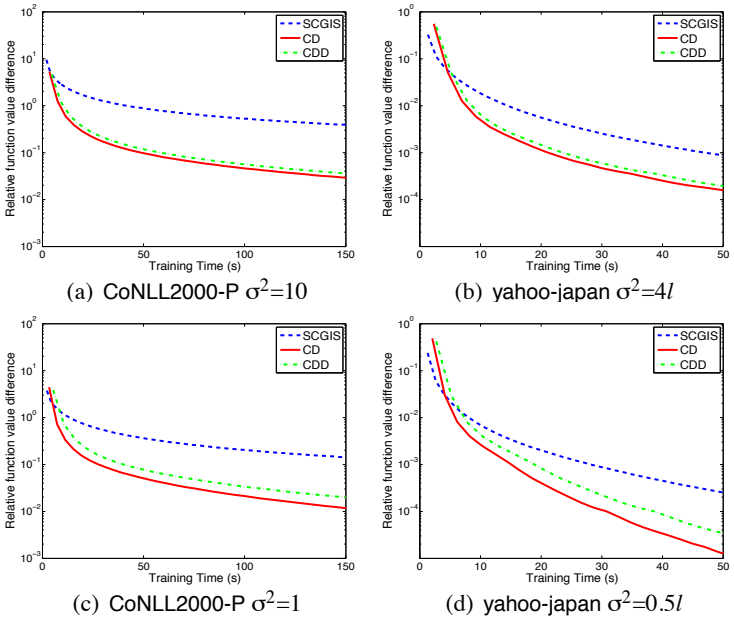
Figure 3: Results on real-valued document data. The first row shows time versus the relative function difference (42). The second and third rows show $\|\nabla L(\mathbf{w})\|$ and testing performances along time, respectively. Time is in seconds.

is the common form of regularized logistic regression. We give approximate functions of IS/CD methods in Section 7.11.

We compare the same methods: SCGIS, CD, LBFGS, and TRON. GIS is not included because of its slow convergence shown in Section 5.1. Our implementations are based on sources used in Chang et al. (2008).[6] We select three data sets considered in Chang et al. (2008). Each instance has been normalized to $\|\bar{\mathbf{x}}_i\| = 1$. Data statistics and σ^2 for each problem are in Table 4. We set $\beta = 0.5$ and $\gamma = 0.01$ for the line search procedure (35) in CD. Figure 3 shows the results of the relative function difference to the optimum, the gradient $\|\nabla L(\mathbf{w})\|$, and the testing accuracy.

From Figure 3, the relation between the two IS/CD methods is similar to that in Figure 2, where CD is faster than SCGIS. However, in contrast to Figure 2, here TRON/LBFGS may surpass IS/CD in an earlier stage. Some preliminary analysis on the cost per iteration seems to indicate that IS/CD

6. Source can be found at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/exp.html`.



(a) CoNLL2000-P σ^2=10 (b) yahoo-japan σ^2=4l

(c) CoNLL2000-P σ^2=1 (d) yahoo-japan σ^2=0.5l

Figure 4: This figure shows the effect of using (38) to do line search. The first and second rows show time versus the relative function difference with different σ^2. CDD indicates the CD method without using (38). Time is in seconds.

methods are more efficient on 0/1-featured data due to a smaller number of exp operations, but more experiments/data are needed to draw definitive conclusions.

In Figure 3, TRON is only similarly to or moderately better than LBFGS, but Lin et al. (2008) show that TRON is much better. The only difference between their setting and ours is that Lin et al. (2008) add one feature to each data instance. That is, they modify $\bar{\mathbf{x}}_i$ to $\begin{bmatrix} \bar{\mathbf{x}}_i \\ 1 \end{bmatrix}$, so weights of Maxent become $\begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$, where b is called the bias term. It is surprising that this difference affects LBFGS' performance that much.

6. Discussion and Conclusions

In (38), we propose a way to speed up the line search procedure of Algorithm 3. Figure 4 shows how effective this trick is by varying the value of σ^2. Clearly, the trick is more useful if σ^2 is small. In this situation, the function $L(\mathbf{w})$ is well conditioned (as it is closer to a quadratic function $\sum_t w_t^2$). Hence (38) more easily holds at $\lambda = 1$. Then the line search procedure costs only $O(1)$ time. However, a too small σ^2 may downgrade the testing accuracy. For example, the final accuracy for yahoo-japan is 92.75% with $\sigma^2 = 4l$, but is 92.31% with $\sigma^2 = 0.5l$.

Some work has concluded that approaches like LBFGS or nonlinear conjugate gradient are better than IS methods for training Maxent (e.g., Malouf, 2002; Daumé, 2004). However, experiments in this paper show that comparison results may vary under different circumstances. For example, comparison results can be affected by:

1. Data of the target application. IS/CD methods seem to perform better if features are 0/1 and if implementations have taken this property.

2. The IS method being compared. Our experiments indicate that GIS is inferior to many methods, but other IS/CD methods like SCGIS or CD (Algorithm 3) are more competitive.

In summary, we create a general framework for iterative scaling and coordinate descent methods for maximum entropy. Based on this framework, we discuss the convergence, computational complexity, and other properties of IS/CD methods. We further develop a new coordinate decent method for Maxent. It is more efficient than existing iterative scaling methods.

7. Proofs and Derivations

We define 1-norm and 2-norm of a vector $\mathbf{w} \in R^n$:

$$\|\mathbf{w}\|_1 \equiv \sum_{t=1}^{n} |w_t|, \quad \|\mathbf{w}\|_2 \equiv \sqrt{\sum_{t=1}^{n} w_t^2}.$$

The following inequality is useful in our proofs.

$$\|\mathbf{w}\|_2 \le \|\mathbf{w}\|_1 \le \sqrt{n}\|\mathbf{w}\|_2, \quad \forall \mathbf{w} \in R^n. \tag{44}$$

Subsequently we simplify $\|\mathbf{w}\|_2$ to $\|\mathbf{w}\|$.

7.1 Proof of Theorem 1

Due to the regularization term $\frac{1}{2\sigma^2}\mathbf{w}^T\mathbf{w}$, one can prove that the set U defined in (20) is bounded; see, for example, Theorem 1 of Lin et al. (2008). As $\nabla^2 L(\mathbf{w})$ is continuous in the bounded set U, the following $\tau_{\max}$ and $\tau_{\min}$ exist:

$$\tau_{\max} \equiv \max_{\mathbf{w}\in U} \lambda_{\max}(\nabla^2 L(\mathbf{w})) \quad \text{and} \quad \tau_{\min} \equiv \min_{\mathbf{w}\in U} \lambda_{\min}(\nabla^2 L(\mathbf{w})), \tag{45}$$

where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ mean the largest and the smallest eigenvalues of a matrix, respectively. To show that $\tau_{\max}$ and $\tau_{\min}$ are positive, it is sufficient to prove $\tau_{\min} > 0$. As $\nabla^2 L(\mathbf{w})$ is I/σ^2 plus a positive semi-definite matrix, it is easy to see $\tau_{\min} \ge 1/(\sigma^2) > 0$.

To prove (21), we apply the multi-dimensional Mean-Value Theorem (Apostol, 1974, Theorem 12.9) to $\nabla L(\mathbf{w})$. If $\mathbf{u}, \mathbf{v} \in R^n$, then for any $\mathbf{a} \in R^n$, there is a $\mathbf{c} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v}$ with $0 \le \alpha \le 1$ such that

$$\mathbf{a}^T(\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})) = \mathbf{a}^T \nabla^2 L(\mathbf{c})(\mathbf{u}-\mathbf{v}). \tag{46}$$

Set

$$\mathbf{a} = \mathbf{u} - \mathbf{v}.$$

Then for any $\mathbf{u}, \mathbf{v} \in U$, there is a point $\mathbf{c}$ such that

$$(\mathbf{u}-\mathbf{v})^T(\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})) = (\mathbf{u}-\mathbf{v})^T \nabla^2 L(\mathbf{c})(\mathbf{u}-\mathbf{v}). \tag{47}$$

Since U is a convex set from the convexity of $L(\mathbf{w})$, $\mathbf{c} \in U$. With (45) and (47),

$$\|\mathbf{u}-\mathbf{v}\|\|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\| \ge (\mathbf{u}-\mathbf{v})^T(\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})) \ge \tau_{\min}\|\mathbf{u}-\mathbf{v}\|^2.$$

Hence

$$\|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\| \geq \tau_{\min}\|\mathbf{u}-\mathbf{v}\|. \tag{48}$$

By applying (46) again with $\mathbf{a} = \nabla L(\mathbf{u}) - \nabla L(\mathbf{v})$,

$$\begin{aligned}\|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\|^2 \leq & \|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\|\|\nabla^2 L(\mathbf{c})(\mathbf{u}-\mathbf{v})\| \\ \leq & \|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\|\|\mathbf{u}-\mathbf{v}\|\tau_{\max}.\end{aligned}$$

Therefore,

$$\|\nabla L(\mathbf{u}) - \nabla L(\mathbf{v})\| \leq \tau_{\max}\|\mathbf{u}-\mathbf{v}\|. \tag{49}$$

Then (21) follows from (48) and (49)

To prove the second property (22), we write the Taylor expansion of $L(\mathbf{u})$:

$$L(\mathbf{u}) = L(\mathbf{v}) + \nabla L(\mathbf{v})^T(\mathbf{u}-\mathbf{v}) + \frac{1}{2}(\mathbf{u}-\mathbf{v})^T\nabla^2 L(\mathbf{c})(\mathbf{u}-\mathbf{v}),$$

where $\mathbf{c} \in U$. With (45), we have

$$\frac{\tau_{\min}}{2}\|\mathbf{u}-\mathbf{v}\|^2 \leq L(\mathbf{u}) - L(\mathbf{v}) - \nabla L(\mathbf{v})^T(\mathbf{u}-\mathbf{v}) \leq \frac{\tau_{\max}}{2}\|\mathbf{u}-\mathbf{v}\|^2.$$

Since $\tau_{\max} \geq \tau_{\min} > 0$, L satisfies (22) by choosing $K = \tau_{\max}/2$.

7.2 Proof of Theorem 2

The following proof is modified from Chang et al. (2008). Since $L(\mathbf{w})$ is convex and $\mathbf{w}^*$ is the unique solution, the optimality condition shows that

$$\nabla L(\mathbf{w}^*) = \mathbf{0}. \tag{50}$$

From (21) and (50),

$$\|\nabla L(\mathbf{w}^k)\| \geq \tau_{\min}\|\mathbf{w}^k - \mathbf{w}^*\|. \tag{51}$$

With (23) and (51),

$$\|\mathbf{w}^{k+1} - \mathbf{w}^k\| \geq \eta\tau_{\min}\|\mathbf{w}^k - \mathbf{w}^*\|. \tag{52}$$

From (24) and (52),

$$L(\mathbf{w}^k) - L(\mathbf{w}^{k+1}) \geq \nu\eta^2\tau_{\min}^2\|\mathbf{w}^k - \mathbf{w}^*\|^2. \tag{53}$$

Combining (22) and (50),

$$L(\mathbf{w}^k) - L(\mathbf{w}^*) \leq K\|\mathbf{w}^k - \mathbf{w}^*\|^2. \tag{54}$$

Using (53) and (54),

$$L(\mathbf{w}^k) - L(\mathbf{w}^{k+1}) \geq \frac{\nu\eta^2\tau_{\min}^2}{K}\left(L(\mathbf{w}^k) - L(\mathbf{w}^*)\right).$$

This is equivalent to

$$\left(L(\mathbf{w}^k) - L(\mathbf{w}^*)\right) + \left(L(\mathbf{w}^*) - L(\mathbf{w}^{k+1})\right) \geq \frac{\nu\eta^2\tau_{\min}^2}{K}\left(L(\mathbf{w}^k) - L(\mathbf{w}^*)\right).$$

Finally, we have

$$L(\mathbf{w}^{k+1}) - L(\mathbf{w}^*) \leq \left(1 - \frac{\nu\eta^2\tau_{\min}^2}{K}\right)\left(L(\mathbf{w}^k) - L(\mathbf{w}^*)\right). \tag{55}$$

Let $\mu \equiv \nu\eta^2\tau_{\min}^2/K$. As all constants are positive, $\mu > 0$. If $\mu > 1$, $L(\mathbf{w}^k) > L(\mathbf{w}^*)$ implies that $L(\mathbf{w}^{k+1}) < L(\mathbf{w}^*)$, a contradiction to the definition of $L(\mathbf{w}^*)$. Thus we have either $\mu \in (0,1)$ or $\mu = 1$, which suggests we get the optimum in finite steps.

7.3 Proof of Theorem 3

We prove the result for GIS and IIS first. Let $\bar{\mathbf{z}} = \arg\min_{\mathbf{z}} A^s(\mathbf{z})$, where s indicates GIS or IIS method.[7] From the definition of $A^s(\mathbf{z})$ and its convexity,[8]

$$\nabla A^s(\mathbf{0}) = \nabla L(\mathbf{w}^k) \text{ and } \nabla A^s(\bar{\mathbf{z}}) = \mathbf{0}.^{9} \tag{56}$$

Note that $\nabla A^s(\mathbf{z})$ is the gradient with respect to $\mathbf{z}$, but $\nabla L(\mathbf{w})$ is the gradient with respect to $\mathbf{w}$. Since U is bounded, the set $\{(\mathbf{w},\mathbf{z}) \mid \mathbf{w} \in U \text{ and } \mathbf{w}+\mathbf{z} \in U\}$ is also bounded. Thus we have that

$$\max_{\mathbf{w}\in U} \max_{\mathbf{z}:\mathbf{w}+\mathbf{z}\in U} \lambda_{\max}\left(\nabla^2 A^s(\mathbf{z})\right)$$

is bounded by a constant K. Here $\lambda_{\max}(\cdot)$ means the largest eigenvalue of a matrix. To prove (23), we use

$$\begin{aligned} &\|\mathbf{w}^{k+1} - \mathbf{w}^k\| = \|\bar{\mathbf{z}} - \mathbf{0}\| \\ \geq &\frac{1}{K}\|\nabla A^s(\bar{\mathbf{z}}) - \nabla A^s(\mathbf{0})\| = \frac{1}{K}\|\nabla A^s(\mathbf{0})\| = \frac{1}{K}\|\nabla L(\mathbf{w}^k)\|, \end{aligned} \tag{57}$$

where the inequality is from the same derivation for (49) in Theorem 1. The last two equalities follow from (56).

Next, we prove (24). By (56) and the fact that the minimal eigenvalue of $\nabla^2 A^s(\mathbf{z})$ is greater than or equal to $1/(\sigma^2)$, we have

$$A^s(\mathbf{0}) \geq A^s(\bar{\mathbf{z}}) - \nabla A^s(\bar{\mathbf{z}})^T\bar{\mathbf{z}} + \frac{1}{2\sigma^2}\bar{\mathbf{z}}^T\bar{\mathbf{z}} = A^s(\bar{\mathbf{z}}) + \frac{1}{2\sigma^2}\bar{\mathbf{z}}^T\bar{\mathbf{z}}. \tag{58}$$

From (9) and (58),

$$L(\mathbf{w}^k) - L(\mathbf{w}^{k+1}) = L(\mathbf{w}^k) - L(\mathbf{w}^k + \bar{\mathbf{z}}) \geq A^s(\mathbf{0}) - A^s(\bar{\mathbf{z}}) \geq \frac{1}{2\sigma^2}\bar{\mathbf{z}}^T\bar{\mathbf{z}} = \frac{1}{2\sigma^2}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|^2.$$

Let $\nu = 1/(2\sigma^2)$ and we obtain (24).

We then prove results for SCGIS and CD. For the convenience, we define some notation. A sequential algorithm starts from an initial point $\mathbf{w}^0$, and produces a sequence $\{\mathbf{w}^k\}_{k=0}^{\infty}$. At each iteration, $\mathbf{w}^{k+1}$ is constructed by sequentially updating each component of $\mathbf{w}^k$. This process generates vectors $\mathbf{w}^{k,t} \in R^n, t = 1,\ldots,n$, such that $\mathbf{w}^{k,1} = \mathbf{w}^k$, $\mathbf{w}^{k,n+1} = \mathbf{w}^{k+1}$, and

$$\mathbf{w}^{k,t} = [w_1^{k+1},\ldots,w_{t-1}^{k+1}, w_t^k,\ldots,w_n^k]^T \text{ for } t = 2,\ldots,n.$$

7. The existence of $\bar{z}$ follows from that U is bounded. See the explanation in the beginning of Section 7.1.
8. It is easy to see that all $A_t(z_t)$ in Table 1 are strictly convex.
9. Because $\nabla_t L(\mathbf{w}^k) = A_t^{\mathsf{CD}'}(0)$, we can easily obtain $\nabla A^s(\mathbf{0}) = \nabla L(\mathbf{w}^k)$ by checking $A_t^{s\prime}(0) = A_t^{\mathsf{CD}'}(0)$, where s is GIS or IIS. See formulas in Table 1.

By an argument similar to (57) and (58), we can prove that the one-variable function $A_t^s(z_t)$, where s is SCGIS or CD, satisfies

$$|w_t^{k,t+1} - w_t^{k,t}| \geq \bar{\eta}|A_t^{s\prime}(0)| = \bar{\eta}|\nabla L(\mathbf{w}^{k,t})_t| \text{ and} \tag{59}$$

$$L(\mathbf{w}^{k,t}) - L(\mathbf{w}^{k,t+1}) \geq \frac{1}{2\sigma^2}|w_t^{k,t+1} - w_t^{k,t}|^2. \tag{60}$$

Note that $\bar{\eta} > 0$ is a positive constant. To prove (23), taking the summation of (59) from $t = 1$ to n,

$$\begin{aligned}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|_1 &\geq \bar{\eta}\sum_{t=1}^{n}|\nabla L(\mathbf{w}^{k,t})_t| \geq \bar{\eta}\sum_{t=1}^{n}\left(|\nabla L(\mathbf{w}^{k,1})_t| - |\nabla L(\mathbf{w}^{k,t})_t - \nabla L(\mathbf{w}^{k,1})_t|\right)\\ &= \bar{\eta}\left(\|\nabla L(\mathbf{w}^{k,1})\|_1 - \sum_{t=1}^{n}|\nabla L(\mathbf{w}^{k,t})_t - \nabla L(\mathbf{w}^{k,1})_t|\right).\end{aligned} \tag{61}$$

Since $L(\mathbf{w})$ satisfies (21), using (44),

$$\begin{aligned}&\sum_{t=1}^{n}|\nabla L(\mathbf{w}^{k,t})_t - \nabla L(\mathbf{w}^{k,1})_t| \leq \sum_{t=1}^{n}\|\nabla L(\mathbf{w}^{k,t}) - \nabla L(\mathbf{w}^{k,1})\|_1\\ \leq &\sum_{t=1}^{n}\sqrt{n}\tau_{\max}\|\mathbf{w}^{k,t} - \mathbf{w}^{k,1}\|_1 \leq n\sqrt{n}\tau_{\max}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|_1.\end{aligned} \tag{62}$$

From (61) and (62), we have

$$\|\mathbf{w}^{k+1} - \mathbf{w}^k\|_1 \geq \frac{\bar{\eta}}{1 + \bar{\eta}n\sqrt{n}\tau_{\max}}\|\nabla L(\mathbf{w}^{k,1})\|_1.$$

This inequality and (44) imply

$$\|\mathbf{w}^{k+1} - \mathbf{w}^k\| \geq \tfrac{1}{\sqrt{n}}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|_1 \geq \tfrac{\bar{\eta}}{\sqrt{n} + \bar{\eta}n^2\tau_{\max}}\|\nabla L(\mathbf{w}^k)\|.$$

Let $\eta = \bar{\eta}/(\sqrt{n} + \bar{\eta}n^2\tau_{\max})$. We then have (23).

Taking the summation of (60) from $t = 1$ to n, we get (24):

$$L(\mathbf{w}^k) - L(\mathbf{w}^{k+1}) \geq \frac{1}{2\sigma^2}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|^2.$$

7.4 Derivation of (30)-(31)

Using (18)-(19), we have

$$\frac{dS_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)}{dz_t} = S_{\mathbf{w}}(x,y)e^{z_t f_t(x,y)}f_t(x,y) = S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)f_t(x,y)$$

and

$$\frac{dT_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{dz_t} = \sum_y S_{\mathbf{w}}(x,y)e^{z_t f_t(x,y)}f_t(x,y) = \sum_y S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)f_t(x,y).$$

Then (30) can be obtained from (16), the definition of $T_{\mathbf{w}+z_t\mathbf{e}_t}(x)$ in (1), and the following calculation:

$$\frac{d\log T_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{dz_t} = \frac{\sum_y S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)f_t(x,y)}{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)} = \sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y).$$

For (31), we use

$$\begin{aligned}
&\frac{d\sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)}{dz_t} \\
&= \sum_y f_t(x,y)\frac{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)\frac{dS_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)}{dz_t} - \frac{dT_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{dz_t}S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)}{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)^2} \\
&= \sum_y f_t(x,y)\left(f_t(x,y)P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x) - \frac{S_{\mathbf{w}+z_t\mathbf{e}_t}(x,y)}{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)}\sum_{y'} P_{\mathbf{w}+z_t\mathbf{e}_t}(y'|x)f_t(x,y')\right) \\
&= \sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)^2 - \sum_y\left(P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)\sum_{y'} P_{\mathbf{w}+z_t\mathbf{e}_t}(y'|x)f_t(x,y')\right) \\
&= \sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)^2 - \left(\sum_y P_{\mathbf{w}+z_t\mathbf{e}_t}(y|x)f_t(x,y)\right)^2.
\end{aligned}$$

7.5 Derivation of (33)

From (6) and (9), we immediately have $A_t^{\mathsf{SCGIS}}(z_t) \geq A_t^{\mathsf{CD}}(z_t)$ and $A_t^{\mathsf{IIS}}(z_t) \geq A_t^{\mathsf{CD}}(z_t)$. Next we prove that $A_t^{\mathsf{GIS}}(z_t) \geq A_t^{\mathsf{SCGIS}}(z_t)$. Assume $D(z_t) \equiv A_t^{\mathsf{GIS}}(z_t) - A_t^{\mathsf{SCGIS}}(z_t)$. Then

$$D'(z_t) = \left(e^{z_t f^\#} - e^{z_t f_t^\#}\right)\sum_{x,y}\tilde{P}(x)P_{\mathbf{w}}(y|x)f_t(x,y).$$

Since $f^\# \geq f_t^\# \geq 0$,

$$\begin{aligned}
&D'(z_t) \geq 0 \text{ if } z_t > 0, \\
&D'(z_t) \leq 0 \text{ if } z_t < 0.
\end{aligned} \tag{63}$$

From Taylor expansion, there exists h between 0 and z_t such that

$$D(z_t) = D(0) + z_t D'(h).$$

From $A_t^{\mathsf{SCGIS}}(0) = A_t^{\mathsf{GIS}}(0) = 0$, we have $D(0) = 0$. By (63), $z_t D'(h) \geq 0$, so

$$A_t^{\mathsf{GIS}}(z_t) - A_t^{\mathsf{SCGIS}}(z_t) = D(z_t) \geq D(0) = 0.$$

We can use a similar method to prove $A_t^{\mathsf{GIS}}(z_t) \geq A_t^{\mathsf{IIS}}(z_t)$.

7.6 Proof of Theorem 4

From (31), we can define

$$H = \max_t\left(\frac{1}{\sigma^2} + \sum_{x,y}\tilde{P}(x)f_t(x,y)^2\right) \geq A_t^{\mathsf{CD}\prime\prime}(z_t),\ \forall z_t. \tag{64}$$

From the Taylor expansion of $A_t^{\mathsf{CD}}(z_t)$ at $z_t = 0$, there exists h between 0 and d such that $z_t = \lambda d$ satisfies

$$\begin{aligned}
& A_t^{\mathsf{CD}}(\lambda d) - \gamma\lambda d A_t^{\mathsf{CD}'}(0) \\
= \;& A_t^{\mathsf{CD}}(0) + A_t^{\mathsf{CD}'}(0)\lambda d + \frac{1}{2}A_t^{\mathsf{CD}''}(h)\lambda^2 d^2 - \gamma\lambda d A_t^{\mathsf{CD}'}(0) \\
\leq \;& A_t^{\mathsf{CD}'}(0)\lambda d + \frac{1}{2}H\lambda^2 d^2 - \gamma\lambda d A_t^{\mathsf{CD}'}(0) \\
= \;& -\lambda\frac{A_t^{\mathsf{CD}'}(0)^2}{A_t^{\mathsf{CD}''}(0)} + \frac{1}{2}H\lambda^2\frac{A_t^{\mathsf{CD}'}(0)^2}{A_t^{\mathsf{CD}''}(0)^2} + \gamma\lambda\frac{A_t^{\mathsf{CD}'}(0)^2}{A_t^{\mathsf{CD}''}(0)} \\
= \;& \lambda\frac{A_t^{\mathsf{CD}'}(0)^2}{A_t^{\mathsf{CD}''}(0)}\left(\lambda\left(\frac{H}{2A_t^{\mathsf{CD}''}(0)}\right) - 1 + \gamma\right). \qquad (65)
\end{aligned}$$

If we choose

$$\bar{\lambda} = \frac{2A_t^{\mathsf{CD}''}(0)(1-\gamma)}{H}, \qquad (66)$$

then for $\lambda \leq \bar{\lambda}$, (65) is non-positive. Therefore, (35) is satisfied for all $0 \leq \lambda \leq \bar{\lambda}$.

7.7 Proof of Theorem 5

Following the proof in Section 7.3, it is sufficient to prove inequalities in the same form as (59) and (60). By Theorem 4, any $\lambda \in [\beta\bar{\lambda}, \bar{\lambda}]$, where $\beta \in (0,1)$ and $\bar{\lambda}$ is defined in (66), satisfies the sufficient decrease condition (35). Since Algorithm 3 selects λ by trying $\{1, \beta, \beta^2, \dots\}$, with (64), the selected value λ satisfies

$$\lambda \geq \beta\bar{\lambda} = \beta\frac{2A_t^{\mathsf{CD}''}(0)(1-\gamma)}{H}.$$

This and (34) suggest that the step size $z_t = \lambda d$ in Algorithm 3 satisfies

$$|z_t| = \lambda\left|\frac{-A_t^{\mathsf{CD}'}(0)}{A_t^{\mathsf{CD}''}(0)}\right| \geq \frac{2\beta(1-\gamma)}{H}\left|A_t^{\mathsf{CD}'}(0)\right|. \qquad (67)$$

From (34), (35), $z_t = \lambda d$, $A_t^{\mathsf{CD}''}(0) \geq 1/\sigma^2$ and $\lambda \leq 1$, we have

$$A_t^{\mathsf{CD}}(z_t) - A_t^{\mathsf{CD}}(0) \leq \gamma z_t A_t^{\mathsf{CD}'}(0) = -\gamma z_t d A_t^{\mathsf{CD}''}(0) \leq -\frac{\gamma}{\lambda\sigma^2}z_t^2 \leq -\frac{\gamma}{\sigma^2}z_t^2. \qquad (68)$$

Note that z_t is the step taken for updating $w_t^{k,t}$ to $w_t^{k,t+1}$. With $A_t^{\mathsf{CD}}(z_t) = L(\mathbf{w}^{k,t+1}) - L(\mathbf{w}^{k,t})$, (67)-(68) are in the same form as (59)-(60). In Section 7.3, (59)-(60) are sufficient to prove the desired conditions (23)-(24) for the linear convergence (Theorem 2). Therefore, Algorithm 3 linearly converges.

7.8 Proof of Theorem 6

A direct calculation of $A_t^{\mathsf{CD}'''}(z_t)$ shows that it is bounded for all z_t and w_t^k. We assume that a bound is M. Using $A_t^{\mathsf{CD}}(0) = 0$, (34) and Taylor expansion, there exists h between 0 and d such that

$$\begin{aligned} A_t^{\mathsf{CD}}(d) &= A_t^{\mathsf{CD}'}(0)d + \frac{1}{2}A_t^{\mathsf{CD}''}(0)d^2 + \frac{1}{6}A_t^{\mathsf{CD}'''}(h)d^3 \\ &= -\frac{1}{2}\frac{A_t^{\mathsf{CD}'}(0)^2}{A_t^{\mathsf{CD}''}(0)} + \frac{1}{6}A_t^{\mathsf{CD}'''}(h)d^3 \\ &\le -\frac{1}{2}A_t^{\mathsf{CD}''}(0)d^2 + \frac{1}{6}M\left|d^3\right| \\ &= -\gamma d^2 A_t^{\mathsf{CD}''}(0) + \left(\gamma A_t^{\mathsf{CD}''}(0) - \frac{1}{2}A_t^{\mathsf{CD}''}(0) + \frac{1}{6}M|d|\right)d^2 \\ &= \gamma d A_t^{\mathsf{CD}'}(0) + \left(\gamma A_t^{\mathsf{CD}''}(0) - \frac{1}{2}A_t^{\mathsf{CD}''}(0) + \frac{1}{6}M|d|\right)d^2. \end{aligned} \tag{69}$$

Note that $\gamma < 1/2$. As $A_t^{\mathsf{CD}''}(0) \ge 1/\sigma^2$ and $|A_t^{\mathsf{CD}'}(0)| \to 0$ when $\mathbf{w}$ converges to the optimal solution $\mathbf{w}^*$, near the optimum, d is small enough so that

$$0 \le |d| \le \frac{6}{M}\left(\frac{1}{2} - \gamma\right)A_t^{\mathsf{CD}''}(0).$$

Then we obtain $A_t^{\mathsf{CD}}(d) \le \gamma d A_t^{\mathsf{CD}'}(0)$ and (35) is satisfied.

7.9 Proof of Theorem 7

The following lemma, needed for proving Theorem 7, shows that the direction taken by CD is bigger than that of GIS, IIS, or SCGIS.

Lemma 8 *There exists a positive constant* λ *such that in a neighborhood of* $\mathbf{w}^*$,

$$|d^s|(1+\lambda) \le |d| = \left|\frac{\delta_t'(0)}{\delta_t''(0)}\right|, \tag{70}$$

where d *and* d^s *are defined in* (37).

Proof. Since $d^s = \arg\min_{z_t} A_t^s(z_t)$ and $A_t^s(z_t)$ is strictly convex,

$$A_t^{s\prime}(d^s) = 0. \tag{71}$$

We separate the proof to two cases: $A_t^{s\prime}(0) > 0$ and $A_t^{s\prime}(0) < 0$. If $A_t^{s\prime}(0) = 0$, then $d^s = 0$, so (70) immediately holds.

If $A_t^{s\prime}(0) > 0$, from the strict convexity of $A_t^s(z_t)$ and (71), $d^s < 0$. It is sufficient to prove that there is λ such that $A_t^{s\prime}(d/(1+\lambda)) \le 0$. This result implies $d/(1+\lambda) \le d^s$, so we obtain (70) .

Using Taylor expansion, if $z_t f^s(x,y) < 0$, then

$$e^{z_t f^s(x,y)} \le 1 + z_t f^s(x,y) + \frac{1}{2}z_t^2 (f^s(x,y))^2, \tag{72}$$

where

$$f^s(x,y) \equiv \begin{cases} f^\# \text{ if } s \text{ is GIS}, \\ f_t^\# \text{ if } s \text{ is SCGIS}, \\ f^\#(x,y) \text{ if } s \text{ is IIS}. \end{cases}$$

From Table 1 and (72),

$$\begin{aligned} A_t^{s\prime}(z_t) &= \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) e^{z_t f^s(x,y)} + Q_t'(z_t) \\ &\le \left(\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) + \frac{w_t}{\sigma^2} - \tilde{P}(f_t) \right) + \left(R_1(\mathbf{w}) + \frac{1}{\sigma^2} \right) z_t + \frac{1}{2} z_t^2 R_2(\mathbf{w}) \\ &= A_t^{s\prime}(0) + \left(R_1(\mathbf{w}) + \frac{1}{\sigma^2} - \frac{1}{2}|z_t| R_2(\mathbf{w}) \right) z_t, \end{aligned} \tag{73}$$

where

$$\begin{aligned} R_1(\mathbf{w}) &\equiv \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) f^s(x,y) \quad \text{and} \\ R_2(\mathbf{w}) &\equiv \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) f^s(x,y)^2. \end{aligned}$$

Now the Newton direction is

$$d = -\frac{A_t^{\mathsf{CD}\prime}(0)}{A_t^{\mathsf{CD}\prime\prime}(0)} = -\frac{\delta_t'(0)}{\delta_t''(0)} = -\frac{A_t^{s\prime}(0)}{\delta_t''(0)} < 0. \tag{74}$$

From (31),

$$\begin{aligned} \delta_t''(0) &= \sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y)^2 - \sum_x \tilde{P}(x) \left(\sum_y P_{\mathbf{w}}(y|x) f_t(x,y) \right)^2 + \frac{1}{\sigma^2} \\ &\le R_1(\mathbf{w}) - R_3(\mathbf{w}) + \frac{1}{\sigma^2}, \end{aligned} \tag{75}$$

where

$$R_3(\mathbf{w}) \equiv \sum_x \tilde{P}(x) \left(\sum_y P_{\mathbf{w}}(y|x) f_t(x,y) \right)^2.$$

When $\mathbf{w} \to \mathbf{w}^*, R_1(\mathbf{w}), R_2(\mathbf{w}), R_3(\mathbf{w})$ respectively converge to $R_1(\mathbf{w}^*), R_2(\mathbf{w}^*), R_3(\mathbf{w}^*)$. Moreover, as $\mathbf{w} + d^s \mathbf{e}_t \in \{\bar{\mathbf{w}} \mid L(\bar{\mathbf{w}}) \le L(\mathbf{w})\}$, $d^s \to 0$ when $\mathbf{w} \to \mathbf{w}^*$. Therefore,

$$\lim_{\mathbf{w} \to \mathbf{w}^*} \frac{\delta_t''(0)}{R_1(\mathbf{w}) + \frac{1}{\sigma^2} - \frac{1}{2}|d^s| R_2(\mathbf{w})} = 1 - \frac{R_3(\mathbf{w}^*)}{R_1(\mathbf{w}^*) + \frac{1}{\sigma^2}}. \tag{76}$$

Here we can assume $R_3(\mathbf{w}^*) > 0$. If not, $f_t(x,y) = 0$ for all x, y. Then $w_t^* = 0$ is obtained in just one iteration. From (76), we can choose a positive λ such that

$$\frac{1}{1+\lambda} > 1 - \frac{R_3(\mathbf{w}^*)}{R_1(\mathbf{w}^*) + \frac{1}{\sigma^2}}. \tag{77}$$

From (76) and (77), for any $\mathbf{w}$ in a neighborhood of $\mathbf{w}^*$,

$$\delta_t''(0) \le \frac{R_1(\mathbf{w}) + \frac{1}{\sigma^2} - \frac{1}{2}|d^s| R_2(\mathbf{w})}{1+\lambda}.$$

From (74),

$$\frac{d}{1+\lambda} \le -\frac{A_t^{s\prime}(0)}{R_1(\mathbf{w}) + \frac{1}{\sigma^2} - \frac{1}{2}|d^s| R_2(\mathbf{w})}. \tag{78}$$

From (73) with $z_t = d/(1+\lambda)$ and (78),

$$A_t^{s\prime}\left(\frac{d}{1+\lambda}\right) \le A_t^{s\prime}(0) - A_t^{s\prime}(0) = 0.$$

Therefore, $d/(1+\lambda) \le d^s < 0$.

If $A_t^{s\prime}(0) < 0$, then $d^s > 0$. Using Taylor expansion, if $z_t f^s(x,y) > 0$, we have

$$e^{z_t f^s(x,y)} \ge 1 + z_t f^s(x,y).$$

Then (73) becomes

$$\begin{aligned} A_t^{s\prime}(z_t) &\ge \left(\sum_{x,y} \tilde{P}(x) P_{\mathbf{w}}(y|x) f_t(x,y) + \frac{w_t}{\sigma^2} - \tilde{P}(f_t)\right) + \left(R_1(\mathbf{w}) + \frac{1}{\sigma^2}\right) z_t \\ &= A_t^{s\prime}(0) + \left(R_1(\mathbf{w}) + \frac{1}{\sigma^2}\right) z_t. \end{aligned} \tag{79}$$

From (75) and a derivation similar to (76), there is a $\lambda > 0$ such that

$$\delta_t''(0)(1+\lambda) \le R_1(\mathbf{w}) + \frac{1}{\sigma^2}.$$

Let $z_t = d/(1+\lambda)$ in (79). With (74),

$$A_t^{s\prime}\left(\frac{d}{1+\lambda}\right) \ge A_t^{s\prime}(0) - A_t^{s\prime}(0) = 0.$$

Therefore, $0 < d^s \le d/(1+\lambda)$.

Proof of Theorem 7 We prove this theorem by calculating a lower bound of $\delta_t(d^s) - \delta_t(d)$. From (69),

$$\delta_t(d) \le -\frac{1}{2}\frac{\delta_t'(0)^2}{\delta_t''(0)} + \frac{1}{6} M d^3, \tag{80}$$

where M is an upper bound of $\delta_t'''(z_t)$. If $\mathbf{w}$ is sufficiently close to $\mathbf{w}^*$,

$$\begin{aligned} \delta_t(d^s) &= \delta_t'(0) d^s + \frac{1}{2}\delta_t''(0)(d^s)^2 + \frac{1}{6}\delta_t'''(h)(d^s)^3 \\ &= \frac{1}{2}\delta_t''(0)\left(\frac{\delta_t'(0)}{\delta_t''(0)} - d^s\right)^2 - \frac{1}{2}\frac{\delta_t'(0)^2}{\delta_t''(0)} + \frac{1}{6}\delta_t'''(h)(d^s)^3 \\ &\ge \frac{1}{2}\delta_t''(0)\left(\frac{\lambda}{1+\lambda}\right)\frac{\delta_t'(0)^2}{\delta_t''(0)^2} - \frac{1}{2}\frac{\delta_t'(0)^2}{\delta_t''(0)} - \frac{1}{6}M|d^3| \\ &= \frac{1}{2}\left(\frac{-1}{1+\lambda}\right)\frac{\delta_t'(0)^2}{\delta_t''(0)} - \frac{1}{6}M|d^3|, \end{aligned} \tag{81}$$

where h is between 0 and d^s and the inequality is from Lemma 8. Combining (80) and (81),

$$\begin{aligned}\delta_t(d^s)-\delta_t(d) &\geq \frac{1}{2}\left(1-\frac{1}{1+\lambda}\right)\frac{\delta_t'(0)^2}{\delta_t''(0)}-\frac{1}{3}M|d^3| \\ &= \left(\frac{1}{2}\left(\frac{\lambda}{1+\lambda}\right)-\frac{1}{3}M\frac{|\delta_t'(0)|}{\delta_t''(0)^2}\right)\frac{\delta_t'(0)^2}{\delta_t''(0)}.\end{aligned}$$

Since $\delta_t''(0)\geq 1/(\sigma^2)$ and $\delta_t'(0)\to\nabla_t L(\mathbf{w}^*)=0$, there is a neighborhood of $\mathbf{w}^*$ so that $\delta_t'(0)$ is small enough and

$$\frac{1}{2}\left(\frac{\lambda}{1+\lambda}\right)>\frac{1}{3}M\frac{|\delta_t'(0)|}{\delta_t''(0)^2}.$$

Therefore, $\delta_t(d^s)>\delta_t(d)$ in a neighborhood of $\mathbf{w}^*$.

7.10 Derivation of (39)

From Jensen's inequality and the fact that $\log(x)$ is a concave function,

$$\frac{\sum_{\Omega_t}\tilde{P}(x)\log\frac{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{T_{\mathbf{w}}(x)}}{\sum_{\Omega_t}\tilde{P}(x)}\leq\log\left(\frac{\sum_{\Omega_t}\tilde{P}(x)\frac{T_{\mathbf{w}+z_t\mathbf{e}_t}(x)}{T_{\mathbf{w}}(x)}}{\sum_{\Omega_t}\tilde{P}(x)}\right).$$

With (16), (19) and (40), we have

$$A_t^{\mathsf{CD}}(z_t)\leq Q_t(z_t)+\tilde{P}_t\log\left(1+\frac{\sum_{\Omega_t}\tilde{P}(x)\sum_y P_{\mathbf{w}}(y|x)(e^{z_t f_t(x,y)}-1)}{\tilde{P}_t}\right).$$

By the inequality (15),

$$\begin{aligned}A_t^{\mathsf{CD}}(z_t) &\leq Q_t(z_t)+\tilde{P}_t\log\left(1+\frac{\sum_{\Omega_t}\tilde{P}(x)\sum_y P_{\mathbf{w}}(y|x)\left(\frac{f_t(x,y)e^{z_t f_t^\#}}{f_t^\#}+\frac{f_t^\#-f_t(x,y)}{f_t^\#}-1\right)}{\tilde{P}_t}\right) \\ &= Q_t(z_t)+\tilde{P}_t\log\left(1+\frac{\left(e^{z_t f_t^\#}-1\right)\sum_{\Omega_t}\tilde{P}(x)\sum_y P_{\mathbf{w}}(y|x)f_t(x,y)}{f_t^\#\tilde{P}_t}\right) \\ &= \bar{A}_t^{\mathsf{CD}}(z_t).\end{aligned}$$

Note that replacing $\tilde{P}_t$ with $\sum_x\tilde{P}(x)$ leads to another upper bound of $A_t^{\mathsf{CD}}(z_t)$. It is, however, looser than $\bar{A}_t^{\mathsf{CD}}(z_t)$.

7.11 Logistic Regression

We list approximate functions of IS/CD methods for logistic regression. Note that

$$\tilde{P}(x_i,y)=\begin{cases}\frac{1}{l} & \text{if } y=\bar{y}_i,\\ 0 & \text{otherwise,}\end{cases}\quad\text{and }\tilde{P}(f_t)=\sum_{i:\bar{y}_i=1}\frac{1}{l}\bar{x}_{it}. \tag{82}$$

For GIS, using the formula in Table 1 and (43),

$$A_t^{\mathsf{GIS}}(z_t) = Q_t(z_t) + \frac{1}{l}\left(\frac{e^{z_t f^\#} - 1}{f^\#} \sum_i \frac{\bar{x}_{it}}{1 + e^{-\mathbf{w}^T \bar{\mathbf{x}}_i}}\right),$$

where from (11) and (82),

$$Q_t(z_t) = \frac{2w_t z_t + z_t^2}{2\sigma^2} - \frac{z_t}{l} \sum_{i:\bar{y}_i = 1} \bar{x}_{it} \quad \text{and} \quad f^\# \equiv \max_j f^\#(i).$$

Similarly, IIS and SCGIS respectively solve

$$A_t^{\mathsf{IIS}}(z_t) = Q_t(z_t) + \frac{1}{l}\left(\sum_i \frac{\bar{x}_{it}}{1 + e^{-\mathbf{w}^T \mathbf{x}_i}} \frac{e^{z_t f^\#(i)} - 1}{f^\#(i)}\right),$$

$$A_t^{\mathsf{SCGIS}}(z_t) = Q_t(z_t) + \frac{1}{l}\left(\frac{e^{z_t f_t^\#} - 1}{f_t^\#} \sum_i \frac{\bar{x}_{it}}{1 + e^{-\mathbf{w}^T \mathbf{x}_i}}\right),$$

where $f_t^\# = \max_i \bar{x}_{it}$ and $f^\#(i) = \sum_t \bar{x}_{it}$. Finally, from (17), (30), and (31),

$$A_t^{\mathsf{CD}}(z_t) = Q_t(z_t) + \frac{1}{l} \sum_i \log\left(1 + \frac{e^{z_t \bar{x}_{it}} - 1}{1 + e^{-\mathbf{w}^T \bar{\mathbf{x}}_i}}\right),$$

$$A_t^{\mathsf{CD}\prime}(0) = \frac{w_t}{\sigma^2} + \frac{1}{l}\left(\sum_i \frac{\bar{x}_{it}}{1 + e^{-\mathbf{w}^T \bar{\mathbf{x}}_i}} - \sum_{i:\bar{y}_i = 1} \bar{x}_{it}\right),$$

$$A_t^{\mathsf{CD}\prime\prime}(0) = \frac{1}{\sigma^2} + \frac{1}{l}\left(\sum_i \frac{e^{-\mathbf{w}^T \bar{\mathbf{x}}_i} \bar{x}_{it}^2}{(1 + e^{-\mathbf{w}^T \bar{\mathbf{x}}_i})^2}\right).$$

Acknowledgments

The authors thank anonymous reviewers and associate editor for helpful comments.

References

Galen Andrew and Jianfeng Gao. Scalable training of L1-regularized log-linear models. In *Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML)*, 2007.

Tom M. Apostol. *Mathematical Analysis*. Addison-Wesley, second edition, 1974.

Jason Baldridge, Tom Morton, and Gann Bierner. OpenNLP package, 2001. URL `http://opennlp.sourceforge.net/`.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA 02178-9998, second edition, 1999.

Léon Bottou. Stochastic learning. In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, LNAI 3176, pages 146–168. Springer Verlag, 2004.

Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. Coordinate descent method for large-scale L2-loss linear SVM. *Journal of Machine Learning Research*, 9:1369–1398, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/cdl2.pdf`.

Stanley F. Chen and Ronald Rosenfeld. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, January 2000.

Michael Collins, Robert E. Schapire, and Yoram Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48(1–3):253–285, 2002.

Michael Collins, Amir Globerson, Terry Koo, Xavier Carreras, and Peter Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *Journal of Machine Learning Research*, 9:1775–1822, 2008.

John N. Darroch and Douglas Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

Hal Daumé, III. Notes on CG and LM-BFGS optimization of logistic regression. 2004. URL `http://www.cs.utah.edu/~hal/megam/`.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.

Miroslav Dudík, Steven J. Phillips, and Robert E. Schapire. Performance guarantees for regularized maximum entropy density estimation. In *Proceedings of the 17th Annual Conference on Computational Learning Theory*, pages 655–662, New York, 2004. ACM press.

Roger Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, 1987.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. 2008.

Jianfeng Gao, Galen Andrew, Mark Johnson, and Kristina Toutanova. A comparative study of parameter estimation methods statistical natural language processing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 824–831, 2007.

Alexandar Genkin, David D. Lewis, and David Madigan. Large-scale Bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

Joshua Goodman. Sequential conditional generalized iterative scaling. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 9–16, 2002.

Luigi Grippo and Marco Sciandrone. Globally convergent block-coordinate techniques for unconstrained optimization. *Optimization Methods and Software*, 10:587–637, 1999.

Fang-Lan Huang, Cho-Jui Hsieh, Kai-Wei Chang, and Chih-Jen Lin. Iterative scaling and coordinate descent methods for maximum entropy. In *Proceedings of the 47th Annual Meeting of the Association of Computational Linguistics (ACL)*, 2009. Short paper.

Rong Jin, Rong Yan, Jian Zhang, and Alex G. Hauptmann. A faster iterative scaling algorithm for conditional exponential model. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, 2003.

S. Sathiya Keerthi, Kaibo Duan, Shirish Shevade, and Aun Neow Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61:151–165, 2005.

Kwangmoo Koh, Seung-Jean Kim, and Stephen Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, 2007. URL `http://www.stanford.edu/~boyd/l1_logistic_reg.html`.

Kenneth Lange, David R. Hunter, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9(1):1–20, March 2000.

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`.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

Zhi-Quan Luo and Paul Tseng. On the convergence of coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.

Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the 6th conference on Natural language learning*, pages 1–7. Association for Computational Linguistics, 2002.

Ryan McDonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 81–88, 2006.

Thomas P. Minka. A comparison of numerical optimizers for logistic regression, 2003. URL `http://research.microsoft.com/~minka/papers/logreg/`.

Adwait Ratnaparkhi. *Maximum Entropy Models For Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania, 1998.

Nicol N. Schraudolph, Jin Yu, and Simon Gunter. A stochastic quasi-Newton method for online convex optimization. In *Proceedings of the 11th International Conference Artificial Intelligence and Statistics (AISTATS)*, pages 433–440, 2007.

S.V.N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 969–976, 2006.

Zhihua Zhang, James T. Kwok, and Dit-Yan Yeung. Surrogate maximization/minimization algorithms and extensions. *Machine Learning*, 69(1):1–33, October 2007.

Journal of Machine Learning Research 11 (2010) 849-872 Submitted 2/09; Revised 12/09; Published 2/10

A Streaming Parallel Decision Tree Algorithm

Yael Ben-Haim YAELBH@IL.IBM.COM
Elad Tom-Tov YOMTOV@IL.IBM.COM
IBM Haifa Research Lab
Haifa University Campus
Mount Carmel, Haifa 31905, ISRAEL

Editor: Soeren Sonnenburg

Abstract

We propose a new algorithm for building decision tree classifiers. The algorithm is executed in a distributed environment and is especially designed for classifying large data sets and streaming data. It is empirically shown to be as accurate as a standard decision tree classifier, while being scalable for processing of streaming data on multiple processors. These findings are supported by a rigorous analysis of the algorithm's accuracy.

The essence of the algorithm is to quickly construct histograms at the processors, which compress the data to a fixed amount of memory. A master processor uses this information to find near-optimal split points to terminal tree nodes. Our analysis shows that guarantees on the local accuracy of split points imply guarantees on the overall tree accuracy.

Keywords: decision tree classifiers, distributed computing, streaming data, scalability

1. Introduction

We propose a new algorithm for building decision tree classifiers for classifying both large data sets and streaming data. As recently noted (Bottou and Bousquet, 2008), the challenge which distinguishes large-scale learning from small-scale learning is that training time is limited compared to the amount of available data. Thus, in our algorithm both training and testing are executed in a distributed environment, using only one pass on the data. We refer to the new algorithm as the Streaming Parallel Decision Tree (SPDT).

Decision trees are simple yet effective classification algorithms. One of their main advantages is that they provide human-readable rules of classification. Decision trees have several drawbacks, one of which is the need to sort all numerical attributes in order to decide where to split a node. This becomes costly in terms of running time and memory size, especially when decision trees are trained on large data. The various techniques for handling large data can be roughly grouped into two approaches: performing pre-sorting of the data, as in SLIQ (Mehta et al., 1996) and its successors SPRINT (Shafer et al., 1996) and ScalParC (Joshi et al., 1998), or replacing sorting with approximate representations of the data such as sampling and/or histogram building, for example, BOAT (Gehrke et al., 1999), CLOUDS (AlSabti et al., 1998), and SPIES (Jin and Agrawal, 2003). While pre-sorting techniques are more accurate, they cannot accommodate very large data sets or streaming data.

Faced with the challenge of handling large data, a large body of work has been dedicated to parallel decision tree algorithms (Shafer et al., 1996; Joshi et al., 1998; Narlikar, 1998; Jin and Agrawal,

2003; Srivastava et al., 1999; Sreenivas et al., 1999; Goil and Choudhary, 1999). There are several ways to parallelize decision trees, described in detail in Amado et al. (2001), Srivastava et al. (1999) and Narlikar (1998). Horizontal parallelism partitions the data so that different processors see different examples.[1] Vertical parallelism enables different processors to see different attributes. Task parallelism distributes the tree nodes among the processors. Finally, hybrid parallelism combines horizontal or vertical parallelism in the first stages of tree construction with task parallelism towards the end.

Like their serial counterparts, parallel decision trees overcome the sorting obstacle by applying pre-sorting, distributed sorting, and approximations. Following our interest in streaming data, we focus on approximate algorithms. Our proposed algorithm builds the decision tree in a breadth-first mode, using horizontal parallelism. The core of our algorithm is an on-line method for building histograms from streaming data at the processors. The histograms are essentially compressed representations of the data, so that each processor can transmit an approximate description of the data that it sees to a master processor, with low communication complexity. The master processor integrates the information received from all the processors and determines which terminal nodes to split and how.

This paper is organized as follows. In Section 2 we introduce the SPDT algorithm and the underlying histogram building algorithm. We dwell upon the advantages of SPDT over existing algorithms. In Section 3 we analyze the tree accuracy. In Section 4 we present experiments that compare the SPDT algorithm with the standard decision tree. The experiments show that the SPDT algorithm compares favorably with the traditional, single-processor algorithm. Moreover, it is scalable to streaming data and multiple processors. We conclude in Section 5.

2. Algorithm Description

Consider the following problem: given a (possibly infinite) series of training examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, \ldots, c\}$, our goal is to construct a decision tree that will accurately classify test examples. The classifier is built using multiple processing nodes (i.e., CPUs), where each of the processing nodes observes approximately $1/W$ of the training examples (where W is the number of processing nodes). This partitioning happens for one of several reasons: for example, the data may not be stored in a single location, and may not arrive at a single location, or it may be too abundant to be handled by a single node in a timely manner.

Because of the large number of training examples, it is not feasible to store the examples (even in each separate processor). Therefore, a processor can either save a short buffer of examples and use them to improve (or construct) the classifier, or build a representative summary statistic from the examples, improving it over time, but never saving the examples themselves. In this paper we take the latter approach.

Although the setting described here is generally applicable to streams of data, it is also applicable to the classification of large data sets in batch mode, where memory and processing power constraints require the distribution of data across multiple processors and with limited memory for each processor.

We first present our histogram data structure and the methods related to it. We then describe the tree building process.

1. We refer to processing nodes as processors, to avoid confusion with tree nodes.

Algorithm 1 Update Procedure

input A histogram $h = \{(p_1, m_1), \ldots, (p_B, m_B)\}$, a point p.

output A histogram with B bins that represents the set $S \cup \{p\}$, where S is the set represented by h.

1: **if** $p = p_i$ for some i **then**

2: $m_i = m_i + 1$

3: **else**

4: Add the bin $(p, 1)$ to the histogram, resulting in a histogram of $B+1$ bins $h \cup \{(p,1)\}$. Denote $p_{B+1} = p$ and $m_{B+1} = 1$.

5: Sort the sequence $p_1, \ldots, p_{B+1}$. Denote by $q_1, \ldots, q_{B+1}$ the sorted sequence, and let π be a permutation on $1, \ldots, B+1$ such that $q_i = p_{\pi(i)}$ for all $i = 1, \ldots, B+1$. Denote $k_i = m_{\pi(i)}$, namely, the histogram $h \cup (p,1)$ is equivalent to $(q_1, k_1), \ldots, (q_{B+1}, k_{B+1})$, $q_1 < \ldots < q_{B+1}$.

6: Find a point q_i that minimizes $q_{i+1} - q_i$.

7: Replace the bins (q_i, k_i), (q_{i+1}, k_{i+1}) by the bin

$$\left(\frac{q_i k_i + q_{i+1} k_{i+1}}{k_i + k_{i+1}}, k_i + k_{i+1} \right).$$

8: **end if**

2.1 On-line Histogram Building

A histogram is a set of B pairs (called bins) of real numbers $\{(p_1, m_1), \ldots, (p_B, m_B)\}$, where B is a preset constant integer. The histogram is a compressed and approximate representation of a set S of real numbers. At any time we have $|S| = \sum_{i=1}^{B} m_i$, where $|S|$ is the number of points in S. The histogram data structure supports four procedures, named `update`, `merge`, `sum`, and `uniform`. The `update` procedure is based on an on-line clustering algorithm developed by Guedalia et al. (1999). A demonstration of the algorithms on actual input is given in the appendix.

Algorithm 1 presents the `update` procedure, which adds a new point to a set that is already represented by a given histogram. The `merge` procedure (Algorithm 2) creates a histogram that represents the union $S_1 \cup S_2$ of the sets S_1, S_2, whose representing histograms are given. The algorithm is similar to the update algorithm; in the first step, the two histograms form a single histogram with many bins. In the second step, bins which are closest are merged together (as in lines 5 and 6 in Algorithm 1) to form a single bin. The process repeats until the histogram has B bins.

The `sum` procedure estimates the number of points in a given interval $[a, b]$, that belong to a set whose histogram is given. Algorithm 3 describes how to calculate the sum for $[-\infty, b]$, and can be used to calculate the sum for $[a, b]$, since it is equal to the sum for $[-\infty, b]$ minus the sum for $[-\infty, a]$.

The algorithm assumes that for a bin (p, m), there are m points surrounding p, of which $m/2$ points are to the left of the bin and $m/2$ points are to the right. Consequently, the number of points in the interval $[p_i, p_{i+1}]$ is equal to $(m_i + m_{i+1})/2$, which is the area of the trapezoid $(p_i, 0), (p_i, m_i), (p_{i+1}, m_{i+1}), (p_{i+1}, 0)$, divided by $(p_{i+1} - p_i)$. To estimate the number of points in the interval $[p_i, b]$, for $p_i < b < p_{i+1}$, we draw a straight line from (p_i, m_i) to (p_{i+1}, m_{i+1}). We set $m_b = m_i + \frac{m_{i+1} - m_i}{p_{i+1} - p_i}(b - p_i)$, so that (b, m_b) is on this line. The estimated number of points in the interval $[p_i, b]$ is then the area of the trapezoid $(p_i, 0), (p_i, m_i), (b, m_b), (b, 0)$, divided again by $(p_{i+1} - p_i)$. The case where $b < p_1$ or $b > p_B$ requires special treatment. One possibility is to add two dummy bins $(p_0, 0)$ and $(p_{B+1}, 0)$, where p_0 and p_{B+1} are chosen using prior knowledge, according to which all

Algorithm 2 Merge Procedure

input Histograms $h_1 = \{(p_1^{(1)}, m_1^{(1)}), \ldots, (p_{B_1}^{(1)}, m_{B_1}^{(1)})\}$, $h_2 = \{(p_1^{(2)}, m_1^{(2)}), \ldots, (p_{B_2}^{(2)}, m_{B_2}^{(2)})\}$, an integer B .

output A histogram with B bins that represents the set $S_1 \cup S_2$, where S_1 and S_2 are the sets represented by h_1 and h_2, respectively.

1: For $i = 1, \ldots, B_1$, denote $p_i = p_i^{(1)}$ and $m_i = m_i^{(1)}$. For $i = 1, \ldots, B_2$, denote $p_{B_1+i} = p_i^{(2)}$ and $m_{B_1+i} = m_i^{(2)}$.

2: Sort the sequence $p_1, \ldots, p_{B_1+B_2}$. Denote by $q_1, \ldots, q_{B_1+B_2}$ the sorted sequence, and let π be a permutation on $1, \ldots, B_1+B_2$ such that $q_i = p_{\pi(i)}$ for all $i = 1, \ldots, B_1+B_2$. Denote $k_i = m_{\pi(i)}$, namely, the histogram $h_1 \cup h_2$ is equivalent to $(q_1, k_1), \ldots, (q_{B_1+B_2}, k_{B_1+B_2})$, $q_1 < \ldots < q_{B_1+B_2}$.

3: **repeat**

4: Find a point q_i that minimizes $q_{i+1} - q_i$.

5: Replace the bins (q_i, k_i), (q_{i+1}, k_{i+1}) by the bin

$$\left(\frac{q_i k_i + q_{i+1} k_{i+1}}{k_i + k_{i+1}}, k_i + k_{i+1} \right).$$

6: **until** The histogram has B bins

Algorithm 3 Sum Procedure

input A histogram $\{(p_1, m_1), \ldots, (p_B, m_B)\}$, a point b such that $p_1 < b < p_B$.

output Estimated number of points in the interval $[-\infty, b]$.

1: Find i such that $p_i \leq b < p_{i+1}$.

2: Set

$$s = \frac{m_i + m_b}{2} \cdot \frac{b - p_i}{p_{i+1} - p_i}$$

where

$$m_b = m_i + \frac{m_{i+1} - m_i}{p_{i+1} - p_i}(b - p_i).$$

3: **for all** $j < i$ **do**

4: $s = s + m_j$

5: **end for**

6: $s = s + m_i/2$

or almost all the points in S are in the interval $[p_0, p_{B+1}]$ (p_0 and p_{B+1} can be determined on the fly during the histogram's construction).

The `uniform` (Algorithm 4) procedure receives as input a histogram $\{(p_1, m_1), \ldots, (p_B, m_B)\}$ and an integer $\tilde{B}$ and outputs a set of real numbers $u_1 < \ldots < u_{\tilde{B}-1}$, with the property that the number of points between two consecutive numbers u_j, u_{j+1}, and the number of data points to the left of u_1 and to the right of $u_{\tilde{B}-1}$, is $\frac{|S|}{\tilde{B}}$. The algorithm works like the `sum` procedure in the inverse direction: After the point u_j was determined, we analytically find a point u_{j+1} such that the number of points in $[u_j, u_{j+1}]$ is estimated to be equal to $\frac{|S|}{\tilde{B}}$. This is very similar to the calculations performed in

Algorithm 4 Uniform Procedure

input A histogram $\{(p_1,m_1),\ldots,(p_B,m_B)\}$, an integer $\tilde{B}$.

output A set of real numbers $u_1 < \ldots < u_{\tilde{B}}$, with the property that the number of points between two consecutive numbers u_j, u_{j+1}, as well as the number of data points to the left of u_1 and to the right of $u_{\tilde{B}}$, is $\frac{1}{\tilde{B}}\sum_{i=1}^{B} m_i$.

1: **for all** $j = 1,\ldots,\tilde{B}-1$ **do**

2: Set $s = \frac{j}{\tilde{B}}\sum_{i=1}^{B} m_i$

3: Find i such that $\mathtt{sum}([-\infty,p_i]) < s < \mathtt{sum}([-\infty,p_{i+1}])$.

4: Set d to be the difference between s and $\mathtt{sum}([-\infty,p_i])$.

5: Search for u_j such that

$$d = \frac{m_i + m_{u_j}}{2} \cdot \frac{u_j - p_i}{p_{i+1} - p_i},$$

where

$$m_{u_j} = m_i + \frac{m_{i+1} - m_i}{p_{i+1} - p_i}(u_j - p_i).$$

Substituting

$$z = \frac{u_j - p_i}{p_{i+1} - p_i},$$

we obtain a quadratic equation $az^2 + bz + c = 0$ with $a = m_{i+1} - m_i$, $b = 2m_i$, and $c = -2d$. Hence set $u_j = p_i + (p_{i+1} - p_i)z$, where

$$z = \frac{-b + \sqrt{b^2 - 4ac}}{2a}.$$

6: **end for**

sum, where this time we are given the area of a trapezoid and have to compute the coordinates of its vertices (see line 5 in Algorithm 4).

2.2 Tree Growing Algorithm

We construct a decision tree based on a set of training examples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n)\}$, where $\mathbf{x}_1,\ldots,\mathbf{x}_n \in \mathtt{R}^d$ are the feature vectors and $y_1,\ldots,y_n \in \{1,\ldots,c\}$ are the labels. Every internal node in the tree possesses two ordered child nodes and a decision rule of the form $\mathbf{x}^{(i)} < a$, where $\mathbf{x}^{(i)}$ is the ith attribute and a is a real number. Feature vectors that satisfy the decision rule are directed to the node's left child node, and the other vectors are directed to the right child node. Thus, every example $\mathbf{x}$ has a path from the root to one of the leaves, denoted $l(\mathbf{x})$. Every leaf has a label t, so that an example $\mathbf{x}$ is assigned the label $t(l(\mathbf{x}))$.

Algorithm 5 provides an overview of the tree construction algorithm. We note that this description fits standard decision trees as well. Each time that line 3 is executed, we say that a new iteration has begun. If there are too many samples (possibly infinite in number), we read a predefined number of samples; otherwise, we use the complete data set. A new level of nodes is appended to the tree in each iteration. In line 5 we decide whether a leaf v is to be split or labeled, according to a stopping criterion. Possible stopping criteria can be some threshold on the number of samples reaching the node, or on the node's impurity. A node's impurity is a function G that measures the homogeneity

Algorithm 5 Decision Tree

input Training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$

1: Initialize T to be a single unlabeled node.
2: **while** there are unlabeled leaves in T **do**
3: Navigate data samples to their corresponding leaves.
4: **for all** unlabeled leaves v in T **do**
5: **if** v satisfies the stopping criterion **or** there are no samples reaching v **then**
6: Label v with the most frequent label among the samples reaching v
7: **else**
8: Choose candidate splits for v and estimate Δ for each of them.
9: Split v with the highest estimated Δ among all possible candidate splits.
10: **end if**
11: **end for**
12: **end while**

of labels in samples reaching the node. Its parameters are $q_1, \ldots, q_c$, where q_j is the probability that a sample reaching v has label j and c is the number of labels. The most popular impurity functions are the Gini criterion,

$$1 - \sum_j q_j^2$$

and the entropy function

$$-\sum_j q_j \ln q_j \text{ where } 0 \ln 0 \equiv 0 \,.$$

In our analysis in Section 3, we require G to be continuous and satisfy $G(\{q_j\}) \geq 1 - \max_j\{q_j\}$. These properties hold for the Gini and entropy functions.

The notation Δ, appearing in lines 8 and 9, represents the gap in the impurity function before and after splitting. Suppose that an attribute i and a threshold a are chosen, so that a node v is split according to the rule $\mathbf{x}^{(i)} < a$. Denote by τ the probability that a sample reaching v is directed to v's left child node. Denote further by $q_{L,j}$ and $q_{R,j}$ the probabilities of label j in the left and right child nodes, respectively. We define the function $\Delta(\tau, \{q_j\}, \{q_{L,j}\}, \{q_{R,j}\}) = \Delta(v, i, a)$ as

$$\Delta = G(\{q_j\}) - \tau G(\{q_{L,j}\}) - (1-\tau) G(\{q_{R,j}\}). \tag{1}$$

To complete the algorithm's description, we need to specify what are the candidate splits, mentioned in lines 8 and 9, and how the function Δ for each split is estimated in a distributed environment. We begin by providing an interpretation for these notions in the classical setting, that is, for the standard, serial algorithm. Most algorithms sort every attribute in the training set, and test splits of the form $\mathbf{x}^{(i)} < \frac{a+b}{2}$, where a and b are two consecutive numbers in the sorted sequence of the ith attribute. For every candidate split, Δ can be calculated precisely, as in (1).

In the parallel setting, we apply a distributed architecture that consists of W processors (also called workers). Each processor can observe $1/W$ of the data, but has a view of the complete classification tree built so far. We do not wish each processor to sort its share of the data set, because this operation is not scalable to extremely large data sets. Moreover, the communication complexity between the processors must be a constant that does not depend on the size of the data set. Our algorithm addresses these issues by trading time and communication complexity with

Algorithm 6 Compress Data Sets

input $1/W$ of the training set, where W is the number of processors
output histograms to be transmitted to the master processor

```
Initialize an empty histogram h(v,i,j) for every unlabeled leaf v, attribute i, and class j.
for all observed training samples (x_k, y_k), where x_k = (x_k^(1), ... x_k^(d)) do
    if the sample is directed to an unlabeled leaf v then
        for all attributes i do
            Update the histogram h(v,i,y_k) with the point x_k^(i), using the update procedure.
        end for
    end if
end for
```

classification accuracy. The processors build histograms describing the data they observed and send them to a master processor. Algorithm 6 specifies which histograms are built and how. The number of bins in the histograms is specified through a trade-off between accuracy and computational load: A large number of bins allows a more accurate description, whereas small histograms are beneficial for avoiding time, memory, and communications overloads.

For every unlabeled leaf v, attribute i, and class j, the master processor merges the W histograms $h(v,i,j)$ received from the processors. The master node now has an exact knowledge of the frequency of each label in each tree node, and hence the ability to calculate the impurity of all unlabeled leaves. Leaves that satisfy the stopping criterion are labeled. For the other leaves, the questions remain of how to choose candidate splits and how to estimate their Δ. They are answered as follows. Let v be an unlabeled leaf (that remains unlabeled after the application of the stopping criterion) and let i be an attribute. We first merge the histograms $h(v,i,1),\ldots,h(v,i,c)$ (c denotes the number of labels). The new histogram, denoted $h(v,i)$, represents the ith dimension of feature vectors that reach v, with no distinction between vectors of different labels. We now apply the `uniform` procedure on $h(v,i)$ with some chosen $\tilde{B}$. The resulting set $u_1 < \ldots < u_{\tilde{B}-1}$ constitutes the locations of the candidate splits for the ith attribute. Finally, Δ for each candidate split is estimated using the `sum` procedure and the histograms $h(v,i,j)$. We clarify the rationale behind this choice of split locations. Suppose that the best split is $\mathbf{x}^{(i)} < a$, where $u_k < a < u_{k+1}$. The number of points in the interval $[u_k, a]$ is bounded, implying a bound on the degree of change in Δ if one splits at u_k instead of a. This issue is discussed in more detail in Section 3.

Decision trees are frequently pruned during or after training to obtain smaller trees and better generalization. In the experiments presented in Section 4, we adapted the MDL-based pruning algorithm of Mehta et al. (1996), which is similar to the one used in CART (Breiman et al., 1984). This algorithm involves simple calculations during node splitting that reflect the node's purity. In a bottom-up pass on the complete tree, some subtrees are chosen to be pruned, based on estimates of the expected error rate before and after pruning. The distributed environment neither changes this pruning algorithm nor does it affect its output.

2.3 Complexity Analysis

Every iteration consists of an updating phase performed simultaneously by all the processors and a merging phase performed by the master processor. In the update phase, every processor makes one pass on the data batch assigned to it. The only memory allocation is for the histograms being

constructed. The number of bins in the histograms is constant; hence, operations on histograms take a constant amount of time. Every processor performs at most N/W histogram updates, where N is the size of the data batch and W is the number of processors. There are $W \times L \times c \times d$ histograms, where L is the number of leaves in the current iteration, c is the number of labels, and d is the number of attributes. Assuming that W, L, c, and d are all independent of N, it follows that the space complexity is $O(1)$. The histograms are communicated to the master processor, which merges them and applies the `sum` and `uniform` procedures. If the `uniform` procedure is applied with a constant parameter $\tilde{B}$, then the time complexity of the merging phase is $O(1)$.

To summarize, each iteration requires the following:

- At most N/W operations by each processor in the updating phase.
- Constant space and communication complexities.
- Constant time in the merging phase.

2.4 Related Work

In this section we discuss previous work on histogram and quantile approximations, as well as procedures for building decision trees on parallel platforms.

2.4.1 HISTOGRAMS AND QUANTILES APPROXIMATIONS

Data structures that summarize large sets are substantial components of a variety of algorithms in database management and data mining. Our histogram algorithms tackle two related problems: data compression and quantile approximations.[2] There is broad coverage of these topics in the literature, with an inclination towards one pass algorithms, see Gilbert et al. (2002), Guha et al. (2006), Ioannidis (2003) and Lin (2007) and references therein. Proposed solutions can be divided into two categories: The first category consists of algorithms with proven approximation guarantees (Cormode and Muthukrishnan, 2005; Gilbert et al., 2002; Greenwald and Khanna, 2001; Guha et al., 2006). The demand for a guaranteed accuracy level forces these algorithms to use large amounts of memory, that is, their space requirements are increasing functions of the data size. An exception is the probabilistic algorithm of Manku et al. (1998), which receives an input parameter δ and returns approximate quantiles whose guarantees hold with probability δ. The space complexity of this algorithm increases with δ but not with the data size. The second category, to which our algorithm belongs, consists of heuristics that work well empirically and demand low amounts of space, but lack any rigorous accuracy analysis (Agrawal and Swami, 1995; Jain and Chlamtac, 1985). To our knowledge, distributed environments are not addressed in either of the two categories, except for a brief mention by Manku et al. (1998).

Guaranteed accuracy at the cost of non-constant memory and increasing processing time are problematic because of the inherent nature of streaming data. For example, the algorithm proposed by Guha et al. (2006) requires roughly $O(B^2 \log n)$ memory, where n is the number of data points and B the number of bins. Thus, for example, a stream of 10^{10} data points (not a large number in today's data environments) requires more than 20 times the memory of a comparable fixed-memory algorithm.

2. For a sequence S of real numbers, the ϕ-quantile, $0 \leq \phi \leq 1$, is defined to be an element $x \in S$ such that $\lceil \phi |S| \rceil$ elements of S are smaller or equal to x.

The use of a fixed-memory algorithm, like the one proposed in this paper, naturally comes at a cost in accuracy. As we show, when the data distribution is highly skewed, the accuracy of the on-line histogram decays. Therefore, in cases where the data can be assumed to have originated in categorical distributions with a limited number of values or in distributions which are not highly skewed, the proposed algorithm is sufficiently accurate. In other cases, where distributions are known to be highly skewed, or memory sizes are not a major factor when executing the algorithm, practitioners may prefer to resort to guaranteed accuracy algorithms. This replaces the first part of the proposed algorithm, but keeps its higher levels intact.

2.4.2 PARALLEL DECISION TREES

The SPIES (Jin and Agrawal, 2003) and pCLOUDS (Sreenivas et al., 1999) algorithms build decision trees for streaming data and work in a distributed environment. They are similar to the SPDT algorithm in that they use histograms to process the data in constant time and memory. There are, however, three major differences between these algorithms and the SPDT algorithm and its analysis. The first difference is in the histogram building algorithm. Unlike SPDT, both SPIES and pCLOUDS sample the data. The second difference is in the need of a second pass. CLOUDS (AlSabti et al., 1998) has two versions, named SS and SSE.[3] SSE and SPIES may require several passes over the data, and therefore hold each data batch in memory. The purpose of the second pass is to locate exactly the best split location for every node, and hence eventually to construct the same tree as the standard algorithm. SS is more similar to SPDT, since both algorithms build histograms with an equal number of points in each bin and take the boundaries of the histograms to be the candidate splits. Since only a constant number of split locations is checked, it is possible that a suboptimal split is chosen, which may cause the entire tree to be different from the one constructed by the standard algorithm. The third difference between our work and previous works is our ability to analytically show that the error rate of the parallel tree approaches the error rate of the serial tree, even though the trees are not identical.

3. Bounding the Error of SPDT

In this section, we investigate the training error rate of SPDT. We adopt a simpler version of the framework set by Kearns and Mansour (1999), which views tree nodes as weak learners. This approach allows us to obtain an overall estimate of the tree by studying the local improvements in classification accuracy induced by the internal nodes.

3.1 Background

Let n be the number of training samples used to train a decision tree T. For a tree node v, denote by n_v the number of training samples that reach v, and by $q_{v,j}$ the probability that a sample reaching v has label j, for $j = 1, \ldots, c$. The training error rate of T is

$$e_T = \frac{1}{n} \sum_{v \text{ leaf in } T} n_v (1 - \max_j \{q_{v,j}\}).$$

3. pCLOUDS is a parallelization of the SSE version of CLOUDS. We mention the SS version as well because it can be similarly parallelized.

Henceforth, we require that the impurity function G is continuous and satisfies $G(\{q_j\}) \geq 1 - \max_j\{q_j\}$. The last inequality implies that we have $e_T \leq G_T$, where

$$G_T = \frac{1}{n} \sum_{v \text{ leaf in } T} n_v G(\{q_{v,j}\}). \tag{2}$$

For our analysis, we rewrite Algorithm 5 such that only one new leaf is added to the tree in each iteration (see Algorithm 7). The resulting full-grown tree is identical to the tree constructed by Algorithm 5. Let T_t be the tree produced by Algorithm 7 after the tth iteration. Suppose that the node v is split in the tth iteration and assigned the rule $\mathbf{x}^{(i)} < a$, and let v_L, v_R denote its left and right child nodes respectively. Then

$$\begin{aligned} &G_{T_{t-1}} - G_{T_t} \\ &= \frac{1}{n}(n_v G(\{q_{v,j}\}) - n_{v_L} G(\{q_{v_L,j}\}) - n_{v_R} G(\{q_{v_R,j}\})) \\ &= \frac{n_v}{n}\Delta(v,i,a). \end{aligned}$$

It follows that a lower bound on $\Delta(v,i,a)$ yields an upper bound on G_{T_t} and hence also on e_{T_t}.

Definition 1 *An internal node v, split by a rule $\mathbf{x}^{(i)} < a$, is said to perform locally well with respect to a function $f(\{q_j\})$ if it satisfies $\Delta(v,i,a) \geq f(\{q_{v,j}\})$. A tree T is said to perform locally well if every internal node v in it performs locally well. Finally, a decision tree building algorithm performs locally well if for every training set, the output tree performs locally well.*

Suppose that T_{t-1} has a leaf for which $\frac{n_v}{n} f(\{q_{v,j}\})$ can be lower-bounded by a quantity $h(t, G_{T_{t-1}})$ that depends only on t and $G_{T_{t-1}}$. Then a lower bound on the training error rate of an algorithm that performs locally well can be derived by solving the recurrence $G_{T_t} \leq G_{T_{t-1}} - h(t, G_{T_{t-1}})$. As a simple example, consider $f(\{q_j\}) = \alpha G(\{q_j\})$ for some positive constant α. By (2), and since the number of leaves in T_{t-1} is t, there exists a leaf v in T_{t-1} for which $\frac{n_v}{n} G(\{q_{v,j}\}) \geq G_{T_{t-1}}/t$, hence $\frac{n_v}{n} f(\{q_{v,j}\}) \geq \frac{\alpha}{t} G_{T_{t-1}}$. Let $\tilde{v}$ be the node which is split in the tth iteration. By definition (see line 10 in Algorithm 7), $\frac{n_{\tilde{v}}}{n}\Delta_{\tilde{v}} \geq \frac{n_v}{n}\Delta_v$, where Δ_v and $\Delta_{\tilde{v}}$ are the best splits for v and $\tilde{v}$. We have

$$G_{T_{t-1}} - G_{T_t} = \frac{n_{\tilde{v}}}{n}\Delta_{\tilde{v}} \geq \frac{n_v}{n}\Delta_v \geq \frac{n_v}{n} f(\{q_{v,j}\}) \geq \frac{\alpha}{t} G_{T_{t-1}}.$$

Let G_0 be an upper bound on G_{T_0}. Solving the recurrence $G_{T_t} \leq (1 - \alpha/t) G_{T_{t-1}}$ with initial value G_0, we obtain $G_{T_t} \leq G_0(t-1)^{-\alpha/2}$, therefore $e_{T_t} \leq G_0(t-1)^{-\alpha/2}$.

Kearns and Mansour (1999) made a stronger assumption, named the Weak Hypothesis Assumption, on the local performance of tree nodes. For binary classification and a finite feature space, it is shown that if $G(q_1, q_2)$ is the Gini index, the entropy function, or $G(q_1, q_2) = \sqrt{q_1 q_2}$, then the Weak Hypothesis Assumption implies good local performance (each splitting criterion with respect to its own $f(q_1, q_2)$). Lower bounds on the training error of trees with these splitting criteria are then derived, as described above. These bounds are subject to the validity of the Weak Hypothesis Assumption.

3.2 Main Result

To build an SPDT, we have to set the parameters B and $\tilde{B}$. Recall that B is the number of bins in the histograms constructed by the processors, and $\tilde{B}$ is the size of the output of `uniform`. Encouraged by empirical results concerning the histograms' accuracy, (see Section 4), we set $B = \tilde{B}$ and assume that all applications of the `uniform` and `sum` procedures during SPDT runtime provide us with exact information on the data set. For example, it is assumed that Δ is calculated exactly and not only "estimated" (see line 9 in Algorithm 5). We note that all our results remain intact also if we allow the calculations to be somewhat biased (the empirical evidence points to a bias of about 5%).

It follows that the only source for sub-optimality with respect to standard decision trees is in the choice of the candidate splits. We recall that for the standard decision tree, the number of candidate splits for a node v is equal to the number of training samples that reach v minus one. This luxury is out of the reach of the SPDT because of scalability requirements. The SPDT thus must test only a constant number of candidate splits before it announces the winning split. The following theorem asserts that Δ for the split chosen by the SPDT algorithm can be arbitrarily close to the optimal Δ (of the split chosen by the standard algorithm). The number of bins depends on how close to the real Δ we wish to be, and also on the shape of the training set, but not on its size.

Theorem 2 *Assume that the functions operating on histograms return exact answers. Let v be a leaf in a decision tree which is under construction, and let $\mathbf{x}^{(i)} < a$ be the best split for v according to the standard algorithm. Denote $\tau, q_j, q_{L,j}, q_{R,j}$ as in Section 2.2. Then for every $\delta > 0$ there exists B that depends on $\tau, \{q_j\}, \{q_{L,j}\}, \{q_{R,j}\}$, and δ, such that the split $\mathbf{x}^{(\tilde{i})} < \tilde{a}$ chosen by the SPDT algorithm with B bins satisfies $\Delta(v, \tilde{i}, \tilde{a}) \geq \Delta(v, i, a) - \delta$.*

Proof. Fix B and consider the split $\mathbf{x}^{(i)} < u_k$, where $u_k < a < u_{k+1}$ (take $u_k = u_1$ if $a < u_1$ or $u_k = u_r$ if $a > u_r$; in the sequel we assume without loss of generality that $a > u_1$). Denote by $\tilde{\tau}, \tilde{q}_L, \tilde{q}_R$ the quantities relevant to this split. Let ρ_j denote the probability that a training sample $\mathbf{x}$ that reaches v satisfies $u_k < \mathbf{x}^{(i)} < a$ and has label j. Then

$$\begin{aligned} \tilde{\tau} &= \tau - \rho_0 - \rho_1 \\ \tilde{q}_{L,j} &= \frac{\tau \cdot q_{L,j} - \rho_j}{\tilde{\tau}} \\ \tilde{q}_{R,j} &= \frac{(1-\tau) q_{R,j} + \rho_j}{1 - \tilde{\tau}}. \end{aligned}$$

By the continuity of $\Delta(\tau, \{q_j\}, \{q_{L,j}\}, \{q_{R,j}\})$, for every $\delta > 0$ there exists ε such that

$$\Delta(\tau, \{q_j\}, \{q_{L,j}\}, \{q_{R,j}\}) - \Delta(\tilde{\tau}, \{q_j\}, \{\tilde{q}_{L,j}\}, \{\tilde{q}_{R,j}\}) < \delta.$$

for all $\rho_j < \varepsilon$. Since $\rho_j \leq \frac{1}{B+1}$, we can guarantee that $\rho_j < \varepsilon$ for all j by setting $B = 1/\varepsilon$. We thus have $\Delta(v, \tilde{i}, \tilde{a}) \geq \Delta(v, i, u_k) \geq \Delta(v, i, a) - \delta$, as required.

Theorem 2 implies the following corollary.

Corollary 3 *Assume that the standard decision tree algorithm performs locally well with respect to a function $f(\{q_j\})$, and that the functions operating on histograms return exact answers. Then for every positive function $\delta(\{q_j\})$, the SPDT algorithm performs locally well with respect to $f(\{q_j\}) - \delta(\{q_j\})$, in the sense that for every training set there exists B such that the tree constructed by the SPDT algorithm with B bins performs locally well. Moreover, B does not depend on the size of the*

Algorithm 7 Decision Tree One Node per Iteration

input training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$

Initialize T to be a single node.
while there are unlabeled leaves in T **do**
  for all unlabeled leaves v in T **do**
    if v satisfies the stopping criterion **or** there are no samples reaching v **then**
      Label v with the most frequent label among the samples reaching v
    else
      Choose candidate splits for v and estimate Δ for each of them.
    end if
  end for
  Split an unlabeled leaf v such that $n_v\Delta$ is maximal among all unlabeled leaves and all possible candidate splits, where n_v is the number of samples reaching v.
end while

training set, implying constant memory and communication complexity and constant running time at the master processor.

We conclude this section with an example in which we explicitly derive an upper bound on the error rate of SPDT. Set $f(\{q_j\}) = \alpha G(\{q_j\})$ for a positive constant α, for which we have seen in Section 3.1 that $e_{T_t} \leq G_0(t-1)^{-\alpha/2}$. We note that Kearns and Mansour (1999) show that for $G(q_1, q_2) = \sqrt{q_1 q_2}$, the Weak Hypothesis Assumption implies good local performance with $f(q_1, q_2) = \alpha G(q_1, q_2)$. Applying Corollary 3 with $\delta(\{q_j\}) = \frac{\alpha}{2} G(\{q_j\}) = f(\{q_j\})/2$, we deduce that when using histograms with enough bins, the SPDT's error rate is guaranteed to be no more than $G_0(t-1)^{-\alpha/4}$.

4. Empirical Results

In the following section we empirically test the proposed algorithms. We first show the accuracy of the histogram building and merging procedures, and later compare the accuracy of SPDT compared to a standard decision tree algorithm.

4.1 Histogram Algorithms

We evaluated the accuracy of the histogram building and information extraction algorithms. We ran experiments on seven synthetic sets, generated via different kinds of probability distributions, summarized in Table 1. Each set S, consisting of 10^5 points, was partitioned into four equal parts, denoted $S_1 - S_4$. For each part S_k we built a histogram h_k with $B = 100$ bins, using the `update` procedure. We then ran the `uniform` procedure on h_k with $\tilde{B} = 100$, resulting in a sequence of points $u_1, \ldots, u_{99}$. For each pair of subsequent numbers u_i, u_{i+1}, we checked how many points of S_k are in the interval $[u_i, u_{i+1}]$. We expect to see $\frac{|S_k|}{\tilde{B}} = 25000/100 = 250$ points in each such interval. Our findings are summarized in Table 2. We observe that the mean absolute difference between 250 and the actual number of points in an interval is equal to 11.17 (4.47% of the expected quantity).

We repeat the same experiment on the histograms $h_{1,2}, h_{3,4}$, obtained after merging h_1 with h_2 and h_3 with h_4. The mean absolute difference between $50000/100 = 500$ and the number of points

Distribution	Probability density function
Normal	$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2}$
Uniform	$f(x) = 1,\ 0 \le x \le 1$
Exponential	$f(x) = \frac{1}{\mu} e^{-(x/\mu)},\ \mu = 0.5,\ x \le 0$
Beta	$f(x) = \frac{1}{\int_0^1 t^{a-1}(1-t)^{b-1}dt} x^{a-1}(1-x)^{b-1},\ a = 0.5,\ b = 0.5,\ 0 < x < 1$
Gamma	$f(x) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-x/b},\ a = 3,\ b = 1,\ x \ge 0$
Lognormal	$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-(\ln(x)-\mu)^2/2\sigma^2},\ \mu = 1,\ \sigma = 0.5,\ x > 0$
Chi-square	$f(x) = \frac{1}{2^{v/2}\Gamma(v/2)} x^{(v-2)/2} e^{-x/2},\ v = 10,\ x \ge 0$

Table 1: Probability density functions of synthetic sets used in the experiments described in Section 4.1.

Distribution	Mean			Standard Deviation		
	Average of $h_1 - h_4$	Average of $h_{1,2}$ and $h_{3,4}$	$h_{1,2,3,4}$	Average of $h_1 - h_4$	Average of $h_{1,2}$ and $h_{3,4}$	$h_{1,2,3,4}$
Normal	11.53	26.22	68.89	15.8	36.83	107.45
Uniform	5.99	18.57	34.13	7.55	24.09	46.84
Exponential	13.78	30.5	18.36	39.28	31.52	83.93
Beta	6.95	18.51	30.91	9.56	24.7	45.26
Gamma	11.87	20.4	61.7	15.68	32.08	84.41
Lognormal	15.93	34.75	72.62	21.59	45.03	93.84
Chi-square	12.12	28.17	56	16.42	38	73.75
Average over all data sets	11.17	25.87	55.36	14.99	34.29	76.5
Percent error, averaged over all data sets	4.47	5.17	5.54			

Table 2: Mean absolute difference between the number of points in $[u_i, u_{i+1}]$ and the desired number and standard deviation of the number of points in $[u_i, u_{i+1}]$. Details are in Section 4.1.

in $(S_k \cup S_{k+1}) \cap [u_i, u_{i+1}],\ k = 1,3$, is 25.87 (5.17% of the expected quantity). Finally, we merged $h_{1,2}$ with $h_{3,4}$. Applying the `uniform` procedure, the obtained mean absolute difference between 1000 and $S \cap [u_i, u_{i+1}]$ is 55.36 (5.54% of the expected quantity).

The `sum` and `uniform` procedures assume that there are $(m_i + m_{i+1})/2$ points in every interval $[p_i, p_{i+1}]$. We tested this assumption on the histograms $h_1 - h_4, h_{1,2}, h_{3,4}$ and $h_{1,2,3,4}$. For $h_{1,2,3,4}$, the mean absolute differences between $(m_i + m_{i+1})/2$ and the actual number of points in $[p_i, p_{i+1}]$ is 28.79. Recall that on average there are 1000 points in each interval, implying an error of 2.88%. Details are in Table 3.

Figure 1 shows how accuracy is affected by the distribution's skewness.[4] The figure was obtained by calculating the histograms $h_{1,2,3,4}$ and points $u_1, \ldots, u_{99}$ for different values of the param-

4. The skewness of a distribution is defined to be κ_3/σ^3, where κ_3 is the third moment and σ is the standard deviation.

Distribution	Average of $h_1 - h_4$	Average of $h_{1,2}$ and $h_{3,4}$	$h_{1,2,3,4}$
Normal	4.22	11.07	23.01
Uniform	5.06	14.18	30.28
Exponential	3.74	12.17	24.21
Beta	6.6	15.98	33.2
Gamma	4.02	12.56	18.94
Lognormal	3.68	13.52	29.29
Chi-square	4.14	12.42	28.58
Average over all data sets	4.5	13.13	28.79
Percent error, averaged over all data sets	1.8	2.63	2.88

Table 3: Mean absolute difference between the number of points in $[p_i, p_{i+1}]$ and $(m_i + m_{i+1})/2$. Details are in Section 4.1.

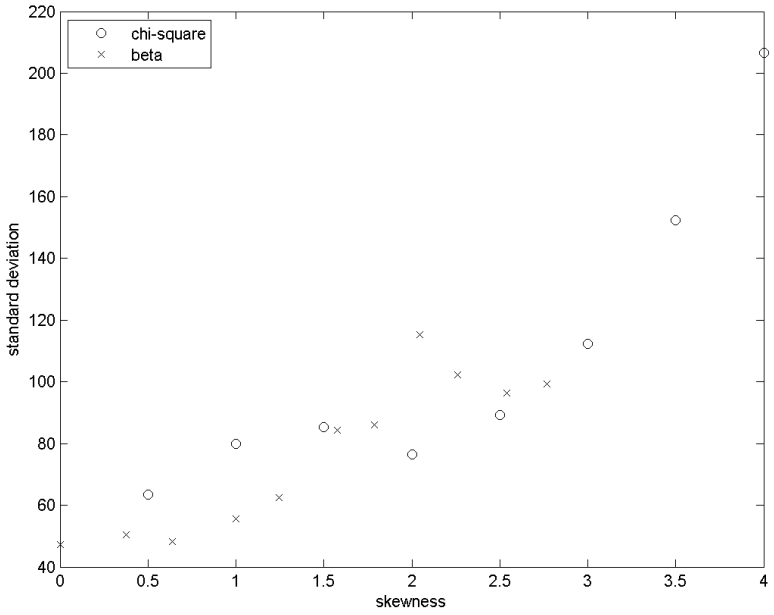


Figure 1: Standard deviation of the number if points in $[u_i, u_{i+1}]$ as a function of the distribution's skewness. The different degrees of skewness are obtained by varying the parameter v of the chi-square distribution and the parameter b of the beta distribution with $a = 0.5$ (see Table 1). More details are given in Section 4.1.

eters of the beta and chi-square distributions. We observe that highly skewed distributions exhibit less accurate results.

Data Set	Number of examples	Number of features	Number of classes
Adult	32561 (16281)	105	2
Isolet	6238 (1559)	617	2
Letter	20000	16	2
Nursery	12960	25	2
Page Blocks	5473	10	2
Pen Digits	7494 (3498)	16	2
Spam Base	4601	57	2
Magic	19020	10	2
Abalone	4177	10	28
Multiple Features	2000	649	11
Face Detection	100000 (10000)	900	2
OCR	100000 (10000)	1156	2

Table 4: Properties of the data sets used in the experiments. The number of examples in parentheses is the number of test examples (if a train/test partition exists).

4.2 Evaluation of the SPDT Algorithms

We ran our experiments on ten medium-sized data sets taken from the UCI repository (Blake et al., 1998) and two large data sets taken from the Pascal Large Scale Learning Challenge (Pascal, 2008). The characteristics of the data sets are summarized in Table 4. For the UCI data sets, we applied ten-fold cross validation when a train/test partition was not given. For the Pascal data sets, we extracted 10^5 examples to constitute a training set, and additional 10^4 examples to constitute a test set. We set the number of bins to 50, and limited the depth of the trees to no more than 100 for the UCI data sets and 10 for the Pascal data sets. We implemented our algorithm in the IBM Parallel Machine Learning toolbox (PML), which runs using MPICH2, and executed it on an 8-CPU Power5 machine with 16GB memory using a Linux operating system. We note that none of the experiments reported in previous works involved both a large number of examples and a large number of attributes.

We began by testing the assumption that splits chosen by the SPDT algorithm are close to optimal. To this end, we extracted four continuous attributes from the training sets (we chose the training set of the first fold if there was no train/test partition). For every attribute, we calculated the following three quantities: Δ of the optimal splitting point, Δ of the splitting point chosen by SPDT with 8 processors, and average Δ over all splitting points (chosen by random splitting). We then normalized by $G(\{q_j\})$, that is,

$$\tilde{\Delta} = \frac{\Delta}{G(\{q_j\})} = 1 - \frac{\tau G(\{q_{L,j}\}) + (1-\tau) G(\{q_{R,j}\})}{G(\{q_j\})}.$$

The normalized value $\tilde{\Delta}$ can be interpreted as the split's efficiency. Since $G(\{q_j\})$ is the maximum possible value of Δ, $\tilde{\Delta}$ represents the ratio between what is actually achieved and the maximum that can be achieved. Table 7 displays the gain of the various splitting algorithms.

Data Set	Constant classification	Standard tree	SPDT 1 worker	SPDT 2 workers	SPDT 4 workers	SPDT 8 workers
Adult	24	15.73	15.79	15.88	**15.69**	15.83
Isolet	50	**14.95**	22.58	26.62	23.09	26.17
Letter	50	**8.52**	8.59	8.59	8.59	8.59
Nursery	34	**2.07**	2.17	2.17	2.17	2.17
Page Blocks	10	**2.89**	3.29	3.09	3.03	3.42
Pen Digits	48	5.37	3.77	3.63	3.63	**3.63**
Spam Base	39	8.17	**6.91**	7.02	7.15	7.22
Magic	35	**17.91**	18.38	18.41	17.95	17.92
Abalone	83.5	**79.33**	79.93	80.6	79.93	80
Multiple Features	90	8.85	8.5	8.15	**8.5**	8.7
Face Detection	8.5	-	**3.31**	4.18	4.13	4.03
OCR	48	-	44.1	42.85	**39.35**	40.73

Table 5: Percent error for UCI and Pascal data sets. The lowest error rate for each data set is marked in bold. The "constant classification" column is the percent error of a classifier that always outputs the most frequent class, that is, it is 100% minus the frequency of the most frequent class.

Data Set	Standard tree	SPDT 1 worker	SPDT 2 workers	SPDT 4 workers	SPDT 8 workers
Adult	81.18	80.75	80.84	80.69	**81.38**
Isolet	**89.7**	77.72	69.45	73.93	70.71
Letter	**95.56**	94.89	94.89	94.89	94.91
Nursery	**99.72**	99.69	99.69	99.69	99.69
Page Blocks	95.48	94.69	95.84	**96.28**	95.05
Pen Digits	97.2	**97.48**	97.37	97.37	97.37
Spam Base	**95.25**	94.95	93.68	94.32	94.22
Magic	80.17	79.81	79.69	80.1	**80.27**
Face Detection	-	97.76	97.32	97.25	95.44
OCR	-	61.72	61.48	**63.85**	62.57

Table 6: Area under ROC curve (%) for UCI and Pascal data sets with binary classification problems. The highest AUC for each data set is marked in bold.

Data Set	Attribute	$\tilde{\Delta}_{OPTIMAL}$	$\tilde{\Delta}_{SPDT}$	$\tilde{\Delta}_{RANDOM}$
Isolet	1	0.0239	0.0231	0.0108
Page Blocks	9	0.1125	0.0985	0.0199
Spam Base	55	0.2044	0.1393	0.1295
Magic	1	0.128	0.1228	0.0304

Table 7: $\tilde{\Delta}$ of splits chosen by the standard tree, SPDT, and random splitting. Details are given in Section 4.2.

Data Set	Err. (%) before pruning	Err. (%) after pruning	AUC (%) before pruning	AUC (%) after pruning	Tree size before pruning	Tree size after pruning
Adult	15.83	13.83	81.38	88.08	5731	359
Isolet	26.17	25.79	70.71	69.9	403	281
Letter	8.59	9.9	94.91	95.29	1069	433
Nursery	2.17	2.28	99.69	99.66	210	194
Page Blocks	3.42	3.46	95.05	95.19	62	29
Pen Digits	3.63	4	97.37	96.75	87	77
Spam Base	7.22	9.48	94.22	94.39	384	95
Magic	17.92	14.75	80.27	88.81	3690	258
Abalone	80	73.5	-	-	4539	93
Multiple Features	8.7	8.25	-	-	173	52
Face Detection	4.03	3.91	95.44	97.75	253	169
OCR	40.73	40.63	62.57	62.63	625	447

Table 8: Percent error, areas under ROC curves, and tree sizes (number of tree nodes) before and after pruning, with eight processors.

We proceed to inspect the tree's accuracy. Tables 5 and 6 display the error rates and areas under the ROC curves of the standard decision tree and the SPDT algorithm with $1, 2, 4$, and 8 processors.[5] We note that it is infeasible to apply the standard algorithm on the Pascal data sets, due to their size. For the UCI data sets, we observe that the approximations undertaken by the SPDT algorithm do not necessarily have a detrimental effect on its error rate. The FF statistics combined with Holm's procedure (Demšar, 2006) with a confidence level of 95% shows that the SPDT algorithm exhibited accuracy that could not be detected as statistically significantly different from that of the standard algorithm.

It is also interesting to study the effect of pruning on the error rate and tree size. Using the procedure described in Section 2.2, we pruned the trees obtained by SPDT. Table 8 shows that pruning usually improves the error rate (though not to a statistically significant threshold, using sign test with $p < 0.05$) while reducing the tree size by 54% on average.

Figure 2 shows the speedup for different sized subsets of the face detection and OCR data sets. Referring to data set size as the number of examples multiplied by the number of dimensions, we found that data set size and speedup are highly correlated (Spearman correlation of 0.90). We further checked the running time as a function of the data set size. In a logarithmic scale, we obtain approximate regression curves (average $R^2 = 0.99$, see Figure 3). The slopes of the curves decrease as the number of processors increases, and drops below 1 for eight processors. In other words, if we multiply the data size by a factor of 10, the running time is multiplied by less than 10.

The results presented here fit the theoretical analysis of Section 2.3. For large data sets, the communication between the processors in the merging phase is negligible relative to the gain in the update phase. Therefore, increasing the number of processors is especially beneficial for large data sets.

5. The results for the OCR data set can be somewhat improved if we increase the tree depth to 25 instead of 10. For four processors, we obtain an error of 32.56% and AUC of 67.5%.

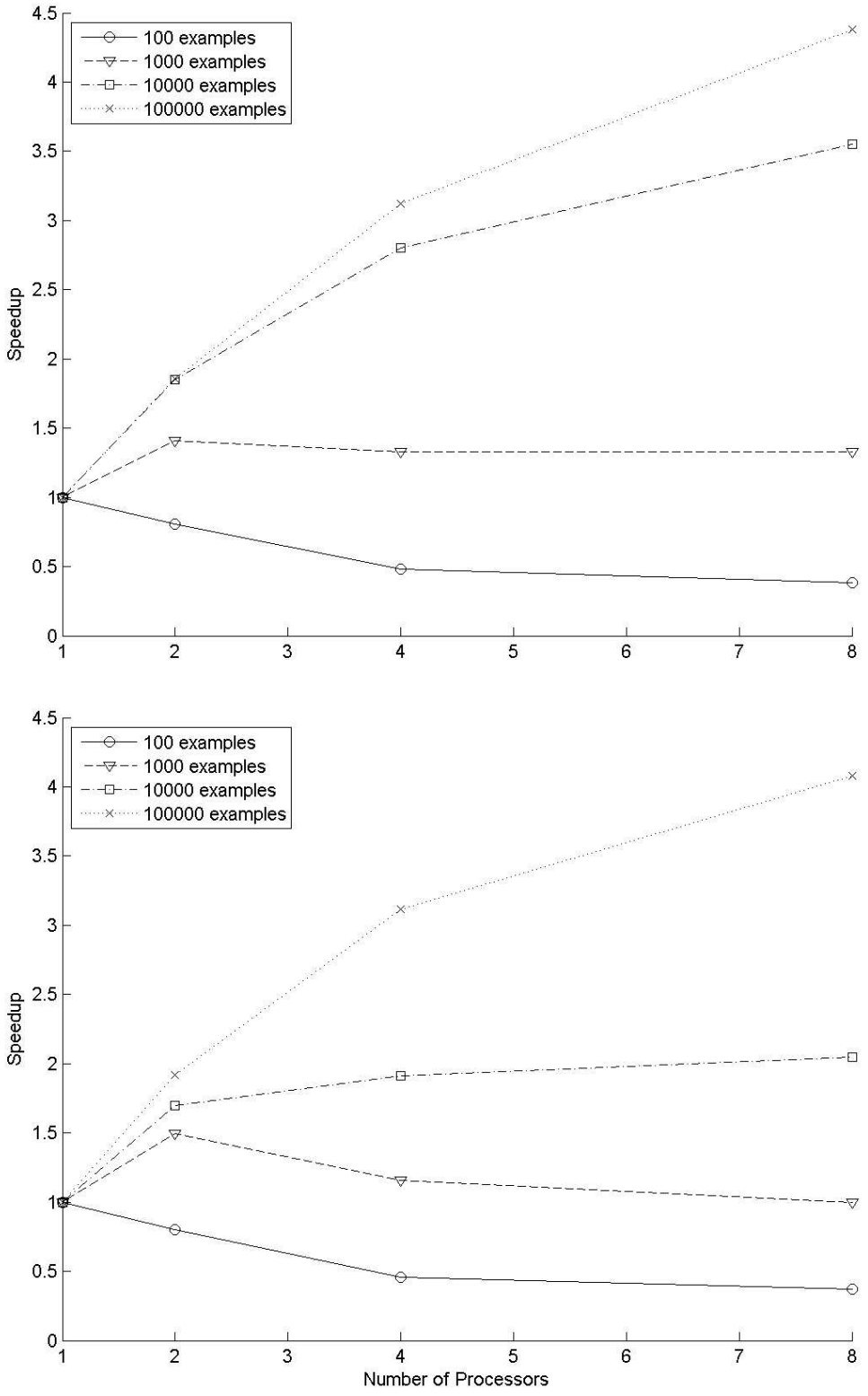


Figure 2: Speedup of the SPDT algorithm for the `face detection` (top) and `OCR` (bottom) data sets.

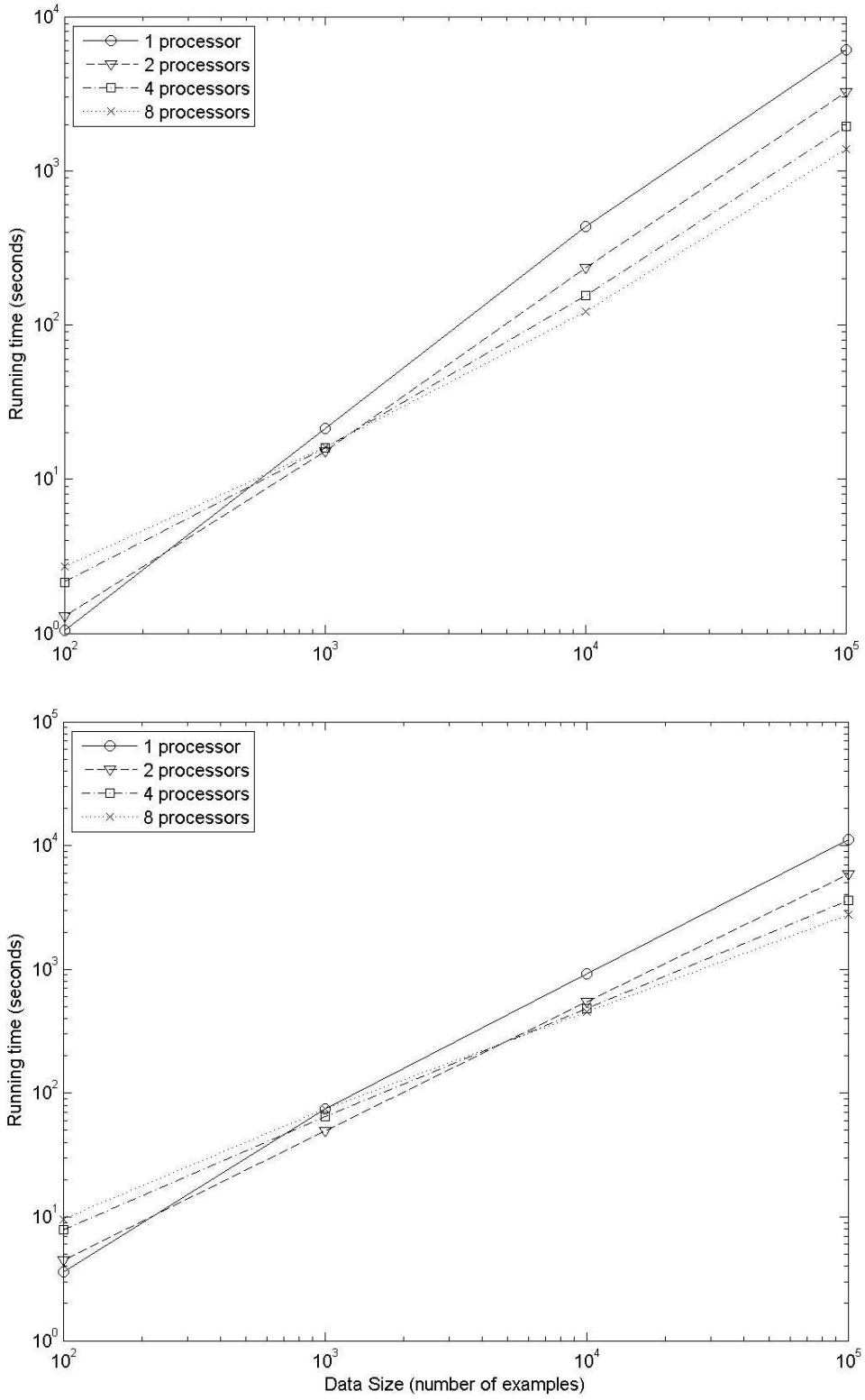


Figure 3: Running time vs. data size for the `face detection` (top) and `OCR` (bottom) data sets.

5. Conclusions

We propose a new algorithm for building decision trees, which we refer to as the Streaming Parallel Decision Tree (SPDT). The algorithm is specially designed for large data sets and streaming data, and is executed in a distributed environment. Our experiments reveal that the error rate of SPDT is approximately the same as for the serial algorithm. We also provide a way to analytically compare the error rate of trees constructed by serial and parallel algorithms without comparing similarities between the trees themselves.

Acknowledgments

We thank the referees for their valuable comments.

Appendix A.

We demonstrate how the histogram algorithms run on the following input sequence:

$$23, 19, 10, 16, 36, 2, 9, 32, 30, 45. \tag{3}$$

Suppose that we wish to build a histogram with five bins for the first seven elements. To this end, we perform seven executions of the `update` procedure. After reading the first five elements, we obtain the histogram

$$(23,1), (19,1), (10,1), (16,1), (36,1).$$

as depicted in Figure 4(a). We then add the bin $(2,1)$ and merge the two closest bins, $(16,1)$ and $(19,1)$, to a single bin $(17.5,2)$. This results in the following histogram, depicted in Figure 4(b):

$$(2,1), (10,1), (17.5,2), (23,1), (36,1).$$

We repeat this process for the seventh element: the bin $(9,1)$ is added, and the two closest bins, $(9,1)$ and $(10,1)$, form a new bin $(9.5,2)$. The resulting histogram is given in Figure 4(c):

$$(2,1), (9.5,2), (17.5,2), (23,1), (36,1).$$

Let us now merge the last histogram with the following one:

$$(32,1), (30,1), (45,1).$$

Figure 5 follows the changes in the histogram during the three iterations of the `merge` procedure. We omit details due to the similarity to the `update` examples given above. The final histogram is given in Figure 5(d):

$$(2,1), (9.5,2), (19.33,3), (32.67,3), (45,1).$$

This histogram represents the set in (3).

We now wish to estimate the number of points smaller than 15. The leftmost bin $(2,1)$ gives 1 point. The second bin, (9.5,2), has $2/2 = 1$ points to its left. The challenge is to estimate how many points to its right are smaller than 15. We first estimate that there are $(2+3)/2 = 2.5$ points inside the trapezoid whose vertices are $(9.5,0), (9.5,2), (19.33,3)$, and $(19.33,0)$ (see Figure 6). Assuming that the number of points inside a trapezoid is proportional to its area, the number of points

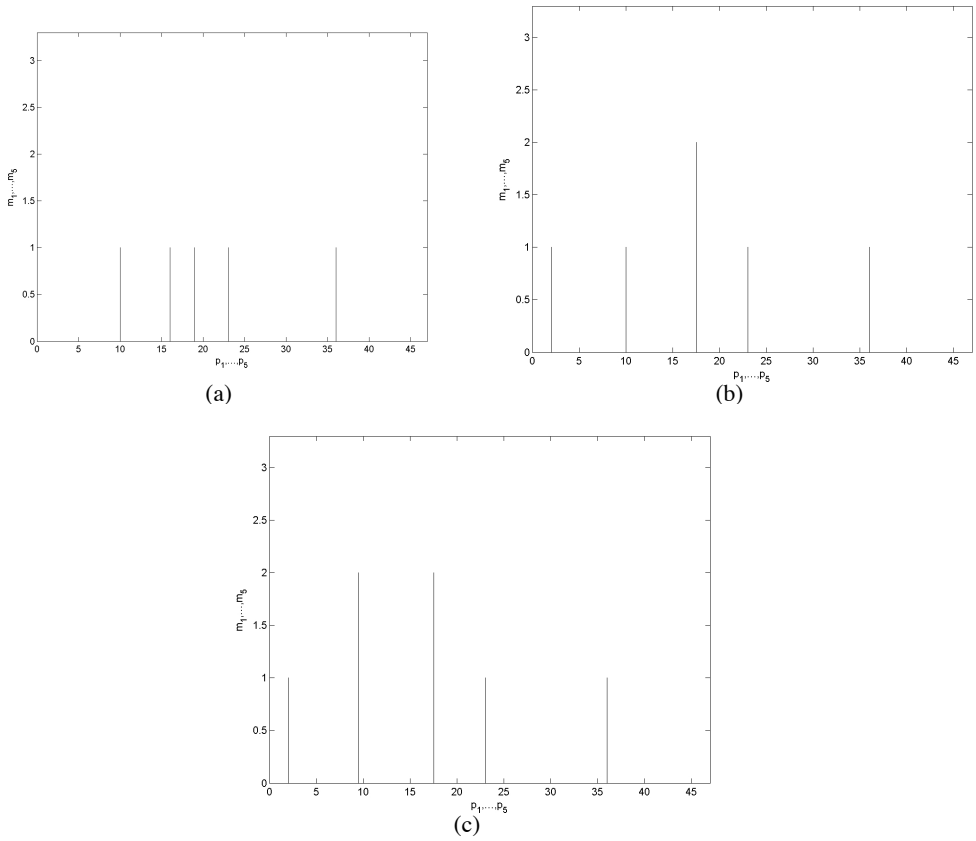


Figure 4: Examples of executions of the `update` procedure.

inside the trapezoid defined by the vertices $(9.5,0),(9.5,2),(15,2.56)$, and $(15,0)$ is estimated to be

$$\frac{2+2.56}{2} \times \frac{15-9.5}{19.33-9.5} = 1.28.$$

We thus estimate that there are in total $1+1+1.28 = 3.28$ points smaller than 15. The true answer, obtained by looking at the set represented by the histogram (see Equation (3)), is three points: $2, 9$, and 10.

The reader can readily verify that the `uniform` procedure with $\tilde{B} = 3$ returns the points 15.21 and 28.98. Each one of the intervals $[-\infty, 15.21], [15.21, 28.98]$, and $[28.98, \infty]$ is expected to contain 3.33 points. The true values are 3, 2, and 4, respectively.

References

Rakesh Agrawal and Arun Swami. A one-pass space-efficient algorithm for finding quantiles. In *Proceedings of COMAD, Pune, India*, 1995.

Khaled AlSabti, Sanjay Ranka, and Vineet Singh. CLOUDS: Classification for large or out-of-core datasets. In *Conference on Knowledge Discovery and Data Mining*, August 1998.

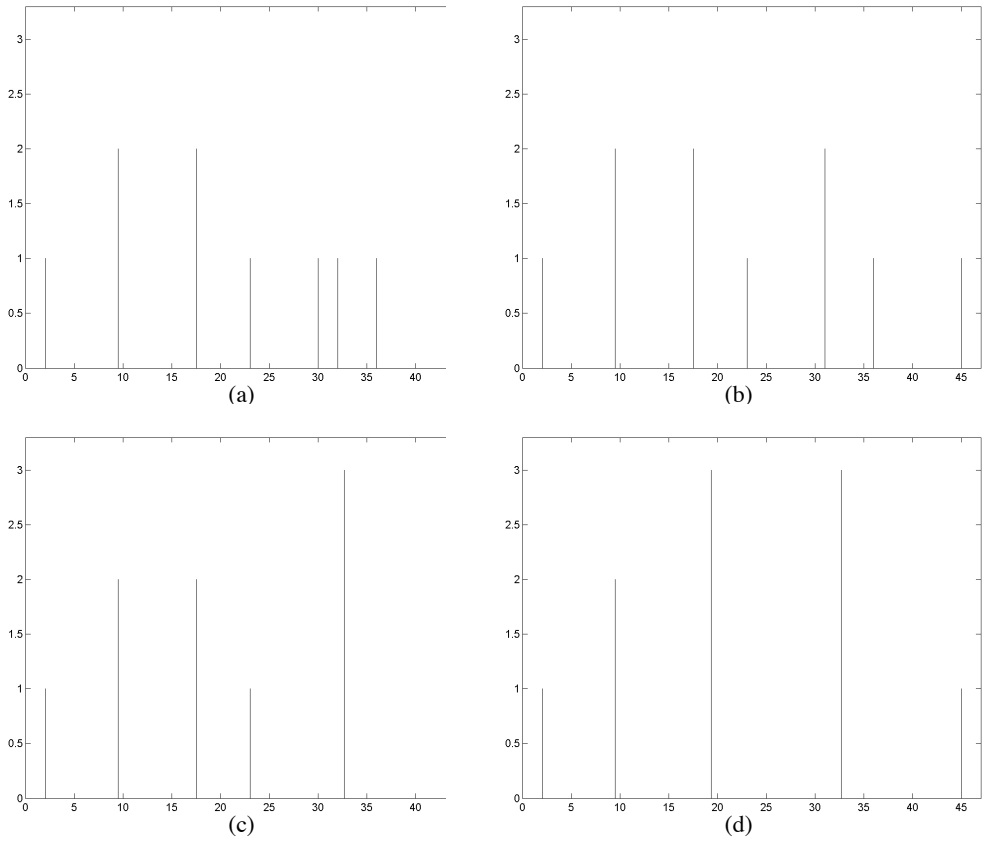


Figure 5: An example of an execution of the `merge` procedure.

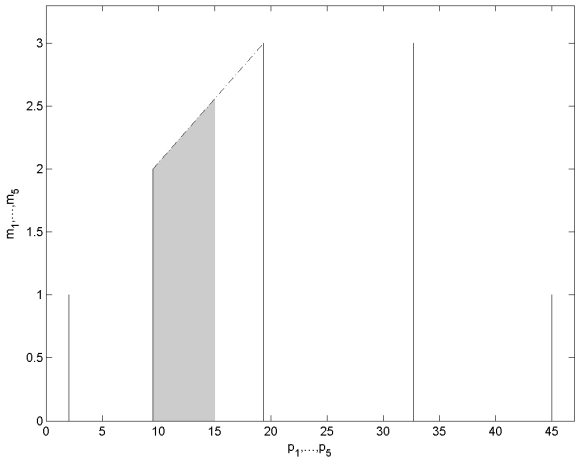


Figure 6: The `sum` procedure.

Nuno Amado, Joao Gama, and Fernando Silva. Parallel implementation of decision tree learning algorithms. In *The 10th Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Knowledge Extraction, Multi-agent Systems, Logic Programming and Constraint Solving*, pages 6–13, December 2001.

Catherine L. Blake, Eamonn J. Keogh, and Christopher J. Merz. UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Leon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, volume 20. MIT Press, Cambridge, MA, 2008. URL `http://leon.bottou.org/papers/bottou-bousquet-2008`. to appear.

Leo Breiman, Jerome H. Friedman, Richard Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth, Monterrey, CA, 1984.

Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Johannes Gehrke, Venkatesh Ganti, Raghu Ramakrishnan, and Wei-Yin Loh. BOAT — optimistic decision tree construction. In *ACM SIGMOD International Conference on Management of Data*, pages 169–180, June 1999.

Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, and Martin J. Strauss. How to summarize the universe: dynamic maintenance of quantiles. In *Proceedings of the 28th VLDB Conference*, pages 454–465, 2002.

Sanjay Goil and Alok Choudhary. Efficient parallel classification using dimensional aggregates. In *Workshop on Large-Scale Parallel KDD Systems, SIGKDD*, pages 197–210, August 1999.

Michael Greenwald and Sanjeev Khanna. Space-efficient online computation of quantile summaries. In *Proceedings of ACM SIGMOD, Santa Barbara, California*, pages 58–66, 'may' 2001.

Isaac D. Guedalia, Mickey London, and Michael Werman. An on-line agglomerative clustering method for nonstationary data. *Neural Comp.*, 11(2):521–540, 1999.

Sudipto Guha, Nick Koudas, and Kyuseok Shim. Approximation and streaming algorithms for histogram construction problems. *ACM Trans. on Database Systems*, 31(1):396–438, 'mar' 2006.

Yannis E. Ioannidis. The history of histograms (abridged). In *Proceedings of the VLDB Conference*, pages 19–30, 2003.

Raj Jain and Imrich Chlamtac. The P^2 algorithm for dynamic calculation of quantiles and histograms without storing observations. *Communications of the ACM*, 28(10):1076–1085, 'oct' 1985.

Ruoming Jin and Gagan Agrawal. Communication and memory efficient parallel decision tree construction. In *The 3rd SIAM International Conference on Data Mining*, May 2003.

Mahesh V. Joshi, George Karypis, and Vipin Kumar. ScalParC: A new scalable and efficient parallel classification algorithm for mining large datasets. In *The 12th International Parallel Processing Symposium*, pages 573–579, March 1998.

Michael Kearns and Yishay Mansour. On the boosting ability of top-down decision tree learning algorithms. *Journal of Computer and System Sciences*, 58(1):109–128, 'feb' 1999.

Xuemin Lin. Continuously maintaining order statistics over data streams. In *Proceedings of the 18th Australian Database Conference, Ballarat, Victoria, Australia*, 2007.

Gurmeet Singh Manku, Sridhar Rajagopalan, and Bruce G. Lindsay. Approximate medians and other quantiles in one pass and with limited memory. In *Proceedings of ACM SIGMOD, Seattle, WA, USA*, pages 426–435, 1998.

Manish Mehta, Rakesh Agrawal, and Jorma Rissanen. SLIQ: A fast scalable classifier for data mining. In *The 5th International Conference on Extending Database Technology*, pages 18–32, 1996.

Girija J. Narlikar. A parallel, multithreaded decision tree builder. Technical Report CMU-CS-98-184, Carnegie Mellon University, 1998.

Pascal, 2008. Pascal large scale learning challenge, 2008. `http://largescale.first.fraunhofer.de`, datasets can be downloaded from `http://ftp.first.fraunhofer.de/pub/projects/largescale`.

PML. IBM Parallel Machine Learning Toolbox, 2009. `http://www.alphaworks.ibm.com/tech/pml`.

John Shafer, Rakesh Agrawal, and Manish Mehta. SPRINT: A scalable parallel classifier for data mining. In *The 22nd International Conference on Very Large Databases*, pages 544–555, September 1996.

Mahesh K. Sreenivas, Khaled Alsabti, and Sanjay Ranka. Parallel out-of-core divide-and-conquer techniques with applications to classification trees. In *The 13th International Symposium on Parallel Processing and the 10th Symposium on Parallel and Distributed Processing*, pages 555–562, 1999. Available as preprint in `http://ipdps.cc.gatech.edu/1999/papers/207.pdf`.

Anurag Srivastava, Eui-Hong Han, Vipin Kumar, , and Vineet Singh. Parallel formulations of decision-tree classification algorithms. *Data Mining and Knowledge Discovery*, 3(3):237–261, September 1999.

Journal of Machine Learning Research 11 (2010) 873-903 Submitted 9/08; Revised 10/09; Published 2/10

Image Denoising with Kernels Based on Natural Image Relations

Valero Laparra VALERO.LAPARRA@UV.ES
Image Processing Lab, Universitat de València
46100 Burjassot, València, Spain.

Juan Gutiérrez JUAN.GUTIERREZ@UV.ES
Dept. Informàtica, Universitat de València
46100 Burjassot, València, Spain.

Gustavo Camps-Valls GUSTAVO.CAMPS@UV.ES
Jesús Malo JESUS.MALO@UV.ES
Image Processing Lab, Universitat de València
46100 Burjassot, València, Spain.

Editor: Donald Geman

Abstract

A successful class of image denoising methods is based on Bayesian approaches working in wavelet representations. The performance of these methods improves when relations among the local frequency coefficients are explicitly included. However, in these techniques, analytical estimates can be obtained *only* for particular combinations of analytical models of signal and noise, thus precluding its straightforward extension to deal with other arbitrary noise sources.

In this paper, we propose an alternative non-explicit way to take into account the relations among natural image wavelet coefficients for denoising: we use support vector regression (SVR) in the wavelet domain to enforce these relations in the estimated signal. Since relations among the coefficients are specific to the signal, the regularization property of SVR is exploited to remove the noise, which does not share this feature. The specific signal relations are encoded in an anisotropic kernel obtained from mutual information measures computed on a representative image database. In the proposed scheme, training considers minimizing the Kullback-Leibler divergence (KLD) between the estimated and actual probability functions (or histograms) of signal and noise in order to enforce similarity up to the higher (computationally estimable) order. Due to its non-parametric nature, the method can eventually cope with different noise sources without the need of an explicit re-formulation, as it is strictly necessary under parametric Bayesian formalisms.

Results under several noise levels and noise sources show that: (1) the proposed method outperforms conventional wavelet methods that assume coefficient independence, (2) it is similar to state-of-the-art methods that do explicitly include these relations when the noise source is Gaussian, and (3) it gives better numerical and visual performance when more complex, realistic noise sources are considered. Therefore, the proposed machine learning approach can be seen as a more flexible (model-free) alternative to the explicit description of wavelet coefficient relations for image denoising.

Keywords: natural images, statistical relations, image denoising, wavelets, non-parametric methods, kernel, mutual information, regularization

1. Introduction

Denoising requires representing the distorted signal in a domain where signal and noise display different enough behavior. In such a representation, noise is removed by imposing the known properties of the signal to the distorted samples. In image denoising, classical regularization techniques are used to impose smoothness in the spatial domain since noise is typically white (Banham and Katsaggelos, 1997). Smoothness in the spatial domain means predictability of the signal from the neighborhood, and thus classical approaches exploit the low-pass behavior of the power spectrum to rely on band-limitation or autoregressive models of the signal (Andrews and Hunt, 1977; Banham and Katsaggelos, 1997; Bertero et al., 1988). Several image denoising methods working in the spatial domain have been presented in the literature, either based on splines (Takeda et al., 2007), patch-based approximations (Kervrann and Boulanger, 2007), local auto-regressive models (Gutiérrez et al., 2006), or support vector regression (Kai Tick Chow and Lee, 2001; van Ginneken and Mendrik, 2006) to perform smooth (regularized) approximations of the noisy signal. Recently, successful methods use adaptive local basis representations (Dabov et al., 2007). Approaches to the problem using local basis is qualitatively related to wavelet descriptions. In fact, wavelet representations have been recognized as quite appropriate domains for image denoising.[1]

Wavelet representations are convenient in image denoising because natural image samples have a very specific statistical behavior in this domain. On the one hand, smoothness is represented by a strong energy compaction in coarse scales. On the other hand, the combination of smooth regions with local, high contrast features, such as edges, gives rise to sparse activation of wavelet sensors. This leads to very particular, heavy-tailed, marginal probability density functions (PDFs) of the wavelet coefficients (Burt and Adelson, 1983; Field, 1987; Simoncelli, 1997; Hyvärinen, 1999). These basic features were incorporated in the classical wavelet-based image denoising techniques (Donoho and Johnstone, 1995; Simoncelli, 1999; Figueiredo and Nowak, 2001). Classical techniques such as hard and soft thresholding (Donoho and Johnstone, 1995) have been derived by using Bayesian approaches in non-redundant wavelets, looking for either *Maximum a Posteriori* (MAP) or *Bayesian Least Squares* (BLS) estimators, in combination with simple marginal models and assuming statistical independence among coefficients (Simoncelli, 1999; Figueiredo and Nowak, 2001).

It is well-known, however, that marginal models in the wavelet domain are not enough for a proper signal characterization: relevant relations among coefficients still remain after wavelet transforms (Simoncelli, 1999). For instance, edges lead to strong coupling between the energy of neighboring wavelet coefficients of natural images. These relations among wavelet coefficients have proven to be a key issue in applications such as image coding (Malo et al., 2006; Camps-Valls et al., 2008), texture analysis and synthesis (Portilla and Simoncelli, 2000) or image quality metrics (Laparra et al., 2010). The use of these relations is in the roots of the most recent and successful image denoising approaches as well (Portilla et al., 2003; Siwei and Simoncelli, 2007; Goossens et al., 2009). In this case, more complex image models explicitly including the relations among coefficients have to be plugged and fitted into the Bayesian framework to obtain the image estimates.

Unfortunately, all these model-based Bayesian techniques have three common problems:

1. They critically depend on the accuracy of the image model, whose definition is not trivial;

1. In the 2007 IEEE International Symposium on Information Theory (ISIT2007), the tutorial "Recent Trends in Denoising" (`http://www.stanford.edu/~slansel/tutorial/summary.htm`) pointed out that state-of-the-art methodologies are usually defined in the wavelet domain.

2. MAP or LS estimations can only be derived analytically for particular, typically Gaussian, noise sources. For different noise sources, the whole technique has to be reformulated which may not be analytically tractable;

3. The estimation of the parameters of the image model from the noisy observation is difficult in general.

Conversely, non-parametric approaches can include the above qualitative properties in an indirect way without the restriction of being analytically attached to particular image or noise models. These approaches are based on *learning* the underlying model directly from the image samples.

In this work we apply support vector regression (SVR) in a redundant (overcomplete) wavelet domain to the noisy image. The proposed method has the following advantages in front of the Bayesian framework:

1. It does not use a particular parametric image model to be fitted, for example, no analytical PDF is required.

2. Its solution may be found for arbitrary noise sources even without knowing the functional form of the noise PDF since it can work with just noise histograms. Therefore, the procedure does not need to be reformulated for different noise sources.

3. It is capable to take into account the relations among wavelet coefficients of natural images through the use of a suitable kernel. In this way, the method preserves the relevant relations among the coefficients of the true signal and better removes the degradation.

The proposed method does not assume independence among the signal coefficients in the wavelet domain, as opposed to Simoncelli (1999) and Figueiredo and Nowak (2001), nor an explicit model of signal relations, as done in Portilla et al. (2003). Therefore, the proposed machine learning approach can be seen as a more flexible (model-free) alternative to the explicit description of wavelet coefficient relations for image denoising. Even though the selection of a particular SVR may be seen as a signal parametrization, the model is still non-parametric in the sense that no functional form of the signal (or noise) characteristics (e.g., the PDF) is assumed.

Non-explicit use of dependencies in local frequency domains for denoising was also introduced in Gutiérrez et al. (2006). In that case, relations were embedded into a perceptual model used for non-parametric spectrum estimation, and offered better results than local parametric autoregressive models not including these relations. Here we pursue the same goal (a model-free technique including local frequency relations), but with a completely different framework (SVR instead of perceptual information). The idea of using SVR regularization in the wavelet domain for image denoising has been recently introduced in Kai Tick Chow and Lee (2001), Cheng et al. (2004) and van Ginneken and Mendrik (2006). However, in these works, (1) the qualitative effect of the different parameters of the SVR was not analyzed, (2) these parameters were set without plausible justification of their values, and more importantly, (3) the relevance of the relations among the wavelet coefficients of the signal was not an issue, so the ability of SVR to take these relations into account in the kernel was neither assessed nor compared to other methods that do consider them. In fact, a trivial isotropic Gaussian kernel was used in all cases. On the contrary, in this paper we address the key following issues:

- **Natural images features in redundant wavelet domains.** Interesting insight about the problem can be obtained by analyzing the mutual information between the coefficients of wavelet representations (Buccigrossi and Simoncelli, 1999; Liu and Moulin, 2001). However, in redundant domains, it is strictly necessary to discern what are the relations specific to the signal and those due to the transform.
- **General constraints of the SVR parameters in image denoising.** Generic recommendations about the SVR parameters have been adapted to propose specific subband-dependent profiles for the insensitivity and the penalization parameters, and to propose a mutual information based kernel.
- **Effect of the SVR parameters.** We show the qualitative effect of varying the values of the parameters under the constrained parameter space.
- **Procedure to optimize the SVR parameters.** We propose an automatic procedure to select the SVR parameters based on the Kullback-Leibler divergence, under certain assumptions on signal and noise.

Even though this methodological framework is proposed in the context of achromatic image denoising, it can be readily extended to other denoising problems in which wavelet coefficients exhibit particular relations, such as in color or multispectral images, speech signals, etc.

The remainder of the paper is outlined as follows. In Section 2, we point out relevant signal features in redundant wavelet domains through mutual information measurements. These key properties will be used by the proposed algorithm presented in Section 3. In Section 4, the effect of SVR parameters and the validity of the proposed criterion for its selection is addressed experimentally. Section 5 shows the performance of the proposed method compared to standard denoising methods in the wavelet domain. Several experiments dealing with different amount and nature of noise illustrate the capabilities of our proposal. Finally, Section 6 draws some conclusions and outlines the further work.

2. Features of Natural Images in the Steerable Wavelet Domain

The starting hypothesis for image denoising is that signal and noise display different characteristics and thus it is possible to separate them in a certain domain. Natural images show non-trivial relationships among wavelet transform coefficients. In the following, we review the reported statistical properties of natural images in orthogonal wavelet domains, and then analyze them in the redundant steerable wavelet domain selected in our implementation. Specifically we will use mutual information (MI) to assess the statistical relations among wavelet coefficients of natural images as in Buccigrossi and Simoncelli (1999) and Liu and Moulin (2001).

2.1 Intraband Versus Interband Signal Relations in Orthogonal Wavelets

Dependencies among *orthogonal wavelet* coefficients were measured using mutual information in Liu and Moulin (2001). The dependencies were studied at interband and intraband levels, and the results suggested that the mutual information between intraband neighbors is typically *larger* than the interband relations for several models and types of interaction. In Buccigrossi and Simoncelli (1999), the authors analyzed the linear predictability of a coefficient's magnitude from a conditioning coefficient set, either its parent, neighbors (left and upper), cousins (coefficients at the same

location but in different orientation subbands), or aunts (cousins of the parent). After an exhaustive mutual information analysis, the parent provided less information content than the neighbors. These evidences suggest that the dependencies among spatial neighboring coefficients (intraband) in orthogonal wavelet descriptions are stronger than the interband dependencies.

2.2 Natural Images Relations in Steerable Wavelets

Redundant (non-orthogonal) wavelet representations may be more suited to image denoising applications since redundant representation of the image features may make the signal inherent relations clearer. Specifically, some redundant representations are designed to be translation or rotation invariant (Freeman and Adelson, 1991; Coifman and Donoho, 1995; Kingsbury, 2006). This behavior is convenient to ensure that a particular feature in different spatial regions (or with different orientations) gives rise to the same neighboring relations. Some translation invariant wavelets (Simoncelli and Freeman, 1995) have also a smoother rotation behavior than non-redundant transforms. This justifies applying the same processing all over a particular subband and dealing with the different orientations in similar ways. Besides, this prevents aliasing artifacts appearing in critically-sampled wavelets. In this work we choose a redundant steerable pyramid representation (Simoncelli and Freeman, 1995) to take advantage of these properties.

Despite the reported results on the relations of signal coefficients in orthogonal transforms, a number of questions have to be answered in the case of redundant representations, and in particular, in the steerable wavelet domain:

1. How relevant are the relations among coefficients of natural images in this domain?
2. How relatively important are interorientation, interscale and intraband signal relations?
3. How is the spatial arrangement of these signal relations?

The first question is particularly important since, even though the steerable transform may intensify the relations among signal coefficients, its redundant nature may also introduce relations which could be due to the transform but not to the signal. The second question allows us to focus on the most significant relations. Answering the third question is crucial to design suitable kernels for image denoising.

In the following, we get some insight on these concerns by performing two experiments on a representative database of 920 achromatic images of size 256×256 extracted from the McGill Calibrated Colour Image Database.[2]

2.2.1 SIGNAL RELATIONS ARE SPECIFIC TO THE SIGNAL

In our first test, following Liu and Moulin (2001), we computed the mutual information among steerable wavelet coefficients of the data set for different spatial, orientation, and scale distances. We used a steerable pyramid with 8 orientations and 4 scales. The mutual information was estimated from the uniformly binned empirical data (256 bins) by computing the histogram of all available sample pairs (721280 samples) for the three considered neighborhoods. In addition, as stated above, in redundant domains it is necessary to know whether these relations come from the images or they are due to the transform. Note that, considering i.i.d. signals, any relation among the coefficients

2. See `http://tabby.vision.mcgill.ca/`.

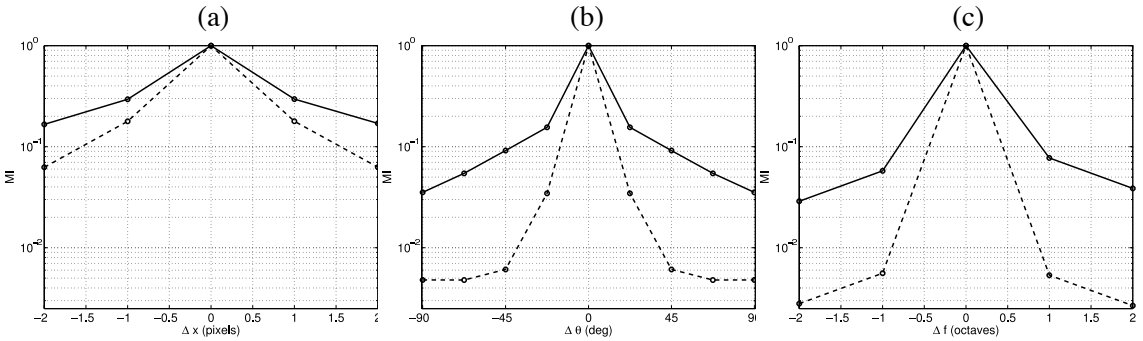


Figure 1: Comparison between redundancy of natural image coefficients in the steerable wavelet representation (solid), and the redundancy due to this representation (dashed). Redundancy is measured in terms of relative mutual information in logarithmic scale among (a) spatial (b) orientation and (c) scale neighbors.

after a linear transform will be due to the transform no matter their PDF in the original domain. Therefore, in order to assess the amount of relations due to the transform, we compared the MI among natural images coefficients, and the MI among the coefficients of an i.i.d. signal (Fig. 1). The relations displayed by i.i.d. signals in the transformed domain may be seen as a lower bound for the mutual information of signal coefficients. From Fig. 1, it can be noticed that, in every case, relations found in natural images are bigger than those introduced by the transform.

2.2.2 INTRABAND SIGNAL RELATIONS DOMINATE OVER INTERSCALE OR ORIENTATION

Besides, the results show that intraband relations in the signal are also more important than interorientation or interscale relations. Note that mutual information measures are defined to depend on logarithms of probability so that comparisons have to be done by subtraction, not by division. Beyond consistency with previously reported results for orthogonal wavelet transforms (Buccigrossi and Simoncelli, 1999; Liu and Moulin, 2001), it has been observed that the relations are specific to the signal and not just induced by the transform.

2.2.3 INTRABAND RELATIONS ARE STRONGLY ORIENTED

In our second test, we studied the spatial arrangement of the relations among intraband coefficients since they display the most relevant relations. To this end, we computed the mutual information in a 2D 5×5 neighborhood for the different orientations and scales. Figure 2[top] shows the above mentioned results for the set of natural images (finest scale). We also provide the relations introduced by the transform (i.i.d. signal, Fig. 2 [bottom]). Similar results were obtained for the other (coarser) scales. Again, the relations among the signal coefficients are higher than those introduced by the transform. Another key issue observed in Fig. 2 [top] is the specific spatial arrangement of these relations: the presence of oriented structures in natural images gives rise to strong anisotropic intraband relations in the different subbands. Coefficients following these relations are expected to be representative of natural features. These mutual information results match recently reported results on autocorrelation of intraband wavelet coefficients (Goossens et al., 2009). The results obtained in these experiments will be further used in Section 4 to design specific kernels that take into account the *observed* natural image relations.

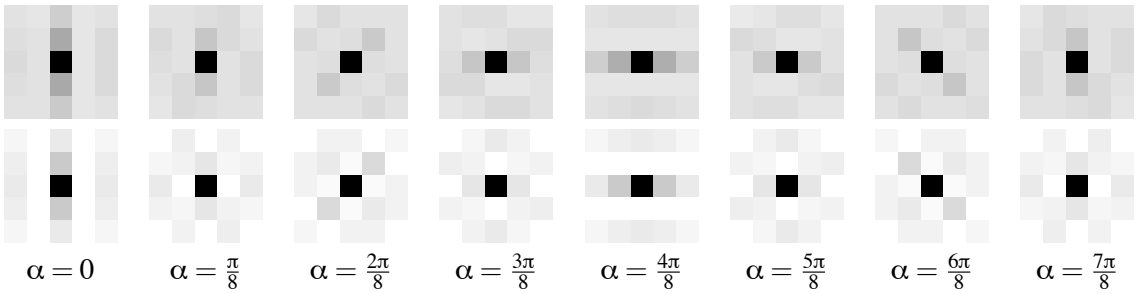


Figure 2: Mutual information among the central coefficient and its spatial neighbors in the same subband (intraband) in the steerable wavelet domain. Darker gray values indicate higher mutual information. Top row shows the results for the different orientations of the finest scale of the natural image database, and bottom row shows the equivalent results for Gaussian noise.

Summarizing, natural images have singular features in the steerable wavelet domain (Figs. 1 and 2): given a distorted image, enforcing these singular oriented relations among coefficients in every subband (with the appropriate kernels) will eventually preserve the natural signal relations and remove the noise. Of course, the bigger the difference between the shape of the intraband relations in signal and noise the better the results are expected to be.

3. Restoring Wavelet Relations with SVR

The effect of noise in the wavelet domain is introducing artificial deviations from the original signal and hiding the natural relations among the coefficients (see an illustrative example in Fig. 3). In the more general case, the degraded observation, $\mathbf{i_d}$, can be written as the result of the addition of a certain realization of noise, $\mathbf{n}$, to the original signal, $\mathbf{i}$:

$$\mathbf{i_d} = \mathbf{i} + \mathbf{n}. \tag{1}$$

Note that this (convenient) way to state the problem does not necessarily mean that the physical degradation has to be additive. In fact, the nature of the degradation should ideally be expressed through a probabilistic noise model that may depend on the original signal, $p(\mathbf{n}|\mathbf{i})$. The other desirable piece of information is a probabilistic model of the signal, $p(\mathbf{i})$. However, in most practical situations, the complete probabilistic description of the problem, that is, having $p(\mathbf{i})$ and $p(\mathbf{n}|\mathbf{i})$, is not available in analytical form.

In order to avoid this lack of information, we propose to use the regularization ability of SVRs. In this section, first we review the capabilities of the SVR for signal approximation. Afterwards, general constraints to the SVR parameter space are given for the particular problem of natural image denoising. Finally, we present an automatic procedure to choose the appropriate SVR parameters (from the above restricted space) to be used for any combination of image and noise.

3.1 Capabilities of SVR for Signal Estimation

Throughout this work, a wavelet transform, matrix T, is applied to the observed image, leading to a set of (noisy) coefficients, $\mathbf{y} = T \cdot \mathbf{i_d}$. The original set of wavelet coefficients, $\mathbf{x} = T \cdot \mathbf{i}$, has to be

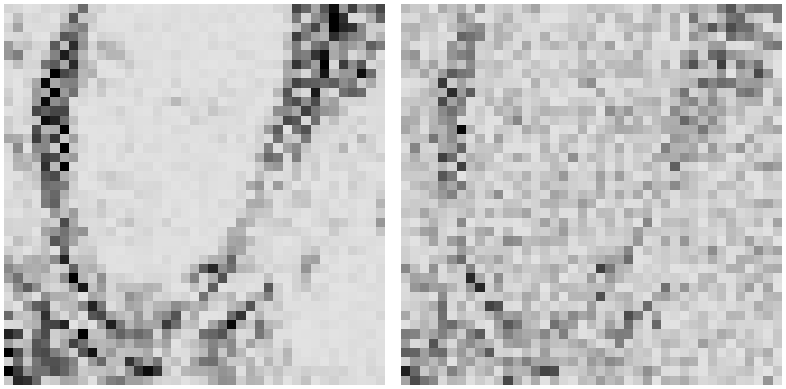


Figure 3: Effect of noise on the wavelet coefficients. Patch of a subband of a wavelet representation of the original image Barbara (left) and its noisy version (right). Darker values indicate higher amplitudes.

estimated from the distorted observation, $\mathbf{y}$. Due to the observed strong intraband relations, we will use the SVR in the wavelet domain in patches inside each subband. Subbands are decomposed into non-overlapping 16×16 patches, leading to sets of $N = 256$ samples. Now, given input-output pairs $\{p_i, y_i\}_{i=1}^{N}$, where p_i are the wavelet indices and y_i are the corresponding noisy wavelet coefficients in a patch, we train the *adaptive* SVR (Camps-Valls et al., 2001; Navia et al., 2001; Gómez et al., 2005) to approximate the signal.

Let ϕ be a non-linear mapping to a higher dimensional feature space, then the adaptive SVR computes the weights $\mathbf{w}$ to obtain the estimation, $\hat{x}_i = \phi^\top(p_i)\mathbf{w}$, by minimizing the following regularized functional:

$$\|\mathbf{w}\|^2 + \sum_i C_i \, \xi_i,$$

subject to $|y_i - \phi^\top(p_i)\mathbf{w}| \leq \varepsilon_i + \xi_i, \forall i = 1, \ldots, N$, where ξ_i are the magnitude of the deviations of the estimated signal from the observed noisy data outside the (sample-dependent) insensitivity zones ε_i. Sample-dependent penalization parameters, C_i, tune the trade-off between fitting the model to the observed noisy data (minimizing the deviations) and keeping model weights $\|\mathbf{w}\|$ small (enforcing flatness in the feature space).

This adaptive SVR differs from the standard formulation (Smola and Schölkopf, 2004), in two aspects: (1) the loss function given by (ε_i, C_i) is sample-dependent, which is convenient in wavelet domains where signal and noise variances strongly depend on the subband, and (2) the usual bias term in SVM formulations has been intentionally dropped to account for the fact that the expected value of wavelet coefficients is zero. The appropriate design of C_i and ε_i profiles is analyzed in Section 3.2.

Explicitly working with the non-linearity ϕ is no longer necessary since the whole formulation can be expressed in the form of dot products of the mapping functions called *kernels*, $K(p_i, p_j) = \phi(p_i)^\top \phi(p_j)$. In this case, the estimation is given by $\hat{\mathbf{x}} = K \cdot \alpha$, where α is the dual representation of weights $\mathbf{w}$ (Smola and Schölkopf, 2004). The kernel matrix can be seen as a similarity matrix between samples (or coefficients), and should reflect the relations between them. Many kernel functions have been proposed in the literature (Smola and Schölkopf, 2004). In the image denoising

case in wavelet domains, we focus on the basic structure of the generalized Radial Basis Functions (RBF) kernel since the relationship among the wavelet coefficients corresponding to spatial neighbors within a subband is local. However, as it will be analyzed in Section 3.2, the kernel will be adapted to incorporate the anisotropic signal relations studied in Section 2.2, see Fig. 2.

3.2 General Constraints on SVR Parameter Space in Image Denoising

As stated above, SVR signal approximation will depend on the penalization parameters, C_i, the insensitivities, ε_i, and the kernel K. In the following, we restrict the range of possible values of these parameters, $\theta = (C_i, \varepsilon_i, K)$, in the particular case of image denoising in wavelet domains:

Penalization factor. In general, the penalization factor of SVRs should be related to the standard deviation of the signal (Cherkassky, 2004). In the denoising problem considered here, the signal variance substantially differs in each wavelet scale. According to this, it is strictly necessary to set a different penalization factor *per* scale, $C_i = C\,k_i$, where k_i is a scale-dependent profile. This profile k_i was obtained by averaging the standard deviation of wavelet coefficients over 100 images from the database used in Section 2. This profile was multiplied by a factor, C, varied in the range $[10, 10^4]$, which did not show a strong impact on the results provided a sufficiently large value. This is consistent with the suggestions reported in Chalimourda et al. (2004) in a more general context. Note that, for instance, in the examples of the next section (Fig. 3), indistinguishable results are obtained for a large enough C. In our experiments, we found that a reasonable prescription for the global factor on the penalization profile is $C \approx 10^3$.

Adaptive insensitivity zone. In general, the insensitivity has to be related to the standard deviation of the noise (Kwok and Tsang, 2003). In transformed domains, the effect of the transform has to be taken into account in order to estimate the corresponding standard deviation. In redundant wavelet representations, this standard deviation is coefficient dependent. Thus it is strictly necessary to introduce a subband-dependent ε_i profile (Camps-Valls et al., 2001; Gómez et al., 2005). The transformed standard deviations can be estimated either (1) empirically from noise samples, or (2) computed from the noise covariance matrix if it is known. In the empirical case, noise samples can be experimentally obtained by applying the noise source to a large enough set of images, and writing the noise as in Eq. 1. In our experiments, we used the natural image database used in Section 2, and we obtained fairly stable results for the profile by considering 100 images. In the case that the noise covariance is known, the corresponding matrix in the selected wavelet domain can be obtained from the noise covariance matrix in the spatial domain, Σ_n, and the transform T (Stark and Woods, 1994). Therefore, the insensitivity profile can be computed as:

$$\varepsilon_i = \tau\, diag(T \cdot \Sigma_n \cdot T^\top)_i^{1/2}. \tag{2}$$

In the case of white noise, $\Sigma_n = \sigma_n^2 \cdot I$, and thus Eq. (2) reduces to:

$$\varepsilon_i = \tau\, \sigma_n\, diag(T \cdot T^\top)_i^{1/2}, \tag{3}$$

where σ_n^2 is the noise variance in the spatial domain, and τ is a scaling factor to be adapted for each particular image and noise combination. The scaling factor, τ, should be in the range

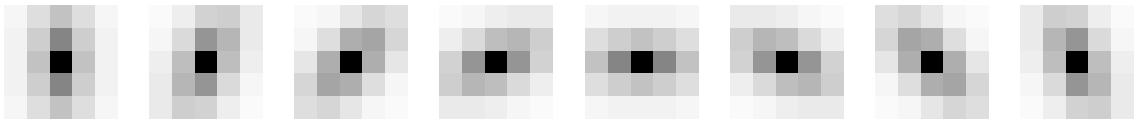


Figure 4: Anisotropic kernel functions used in the support vector regression method for the eight considered orientation subbands.

[0.5, 3] according to the known relationship between the ε-insensitivity zone and the noise standard deviation (Kwok and Tsang, 2003). Note that (2) may cope with colored noise. Considering the off-diagonal elements of the covariance matrix (neglected in (2) and (3)) would give rise to coupling ε-insensitivities among samples. This issue has been already considered and solved in the context of image coding by using an additional non-linear transform and a constant ε in the transformed domain (Camps-Valls et al., 2008). However, in this paper, we restrict ourselves to the approximated diagonal case.

Including signal relations in the kernel. In the kernel methods literature, the use of *prior* knowledge about the problem can be encoded through bagged, cluster, or probabilistic kernels (Jebara et al., 2004; Weston et al., 2004). In our case, we propose to take into account image coefficient relations by analyzing a large (representative) database and taking the (oriented) mutual information among samples as core distance measure. However, using these empirical measures to set the kernels is not straightforward since the kernels have to fulfill Mercer's Theorem (Mercer, 1905). According to this, we propose to use generalized Gaussian kernels. In particular, we fitted anisotropic Laplacian kernels to the MI measures to consider the intraband oriented relations within each subband:

$$K_\alpha(p_i, p_j) \quad = \quad \exp\big(-((p_i - p_j)^\top G(\alpha)^\top \Sigma^{-1} G(\alpha)(p_i - p_j))^{1/2}\big),$$

where $\Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}$, σ_1 and σ_2 are the widths of the kernels, $p_i \in \mathbb{R}^2$ denotes the spatial position of coefficient y_i within a subband, and $G(\alpha)$ is the 2D rotation matrix with rotation angle, α, corresponding to the orientation of each subband (see Fig. 4). Note that these set of oriented kernels describe the signal relationships that emerge from experiments in Section 2 (cf. Fig. 2[top]).

We obtained proper values for the widths σ_1 and σ_2 by fitting the above kernel to the MI measures among coefficients described in Section 2 ($\sigma_1 = 2\sigma_2$, and $\sigma_1 = 4.8$ in spatial coefficient units). The kernel was further normalized in the standard way. Since this width comes from direct measures from images, it describes a fundamental property of natural images so it can be kept fairly constant.

The conclusion of this section is that in the case of image denoising in wavelet domains, an appropriate analysis of the signal variance, the noise variance, and the relations among the wavelet coefficients of the signal can be used to strongly reduce the dimensionality of the SVR parameter space. After this analysis, the only SVR parameter that remains fixed is the global scaling, τ, to be applied to the insensitivity profile.

3.3 Procedure for Automatic SVR Selection

In the more general case, applying SVRs with a given set of parameters, θ, to a noisy image leads to a certain image estimate, $\hat{\mathbf{i}}_\theta = T^{-1} \cdot \hat{\mathbf{x}}_\theta$. From this image estimate, and the convenient additive notation for the noise (Eq. (1)), a noise estimate can be obtained: $\hat{\mathbf{n}}_\theta = \mathbf{i_d} - \hat{\mathbf{i}}_\theta$. In this section we propose a procedure to select the SVR parameters, θ, that better approximates the noise free image, using the available information.

In the more general situation the only available information is the noisy image. However, as stated above, denoising methods usually assume that additional probabilistic information on the signal and noise is available: $p(\mathbf{i})$ and $p(\mathbf{n}|\mathbf{i})$. Note that this knowledge is equivalent to the knowledge of the joint signal and noise distribution since $p(\mathbf{i},\mathbf{n}) = p(\mathbf{n}|\mathbf{i})\,p(\mathbf{i})$.

Let us momentarily assume that this information is available to propose the general procedure to set the SVR parameters. Afterwards, we will relax the requirements by considering an approximation that can be easily applied in practical situations.

In order to enforce solutions that closely follow the (assumed to be known) statistics of signal and noise, we propose to select the SVR that minimizes the k-th order Kullback-Leibler divergence (KLD) (Cover and Tomas, 1991) between the joint PDF of signal and noise, and the joint PDF of the estimated signal and the estimated noise:

$$\theta^* = \arg\ \min_\theta \left\{ D_{KL}\big[\, p(\hat{\mathbf{i}}_\theta, \hat{\mathbf{n}}_\theta) \parallel p(\mathbf{i},\mathbf{n}) \,\big] \right\}. \tag{4}$$

The underlying idea is that the SVR that minimizes the divergence between the above PDFs is the one that better captures the features of the true signal and better removes the degradation.

Although in ideal situations the application of this procedure would obtain the best results in statistical terms, in practical situations the full probabilistic description of the problem is not available. A number of approximations are done in practical situations. For instance, thermal noise in CCD cameras is not independent from the input signal since diffusion increase with the irradiance. However, thermal noise is usually assumed to be independent of the input signal. Additional assumptions as additivity or certain analytical marginal PDF of the noise are also widely used.

In our case, we assume independence between signal and noise:

$$p(\mathbf{i},\mathbf{n}) = p(\mathbf{i})\,p(\mathbf{n}).$$

However, no analytical model for these PDFs is assumed. Under this independence assumption, it is easy to see that Eq. 4 reduces to:

$$\theta^* = \arg\ \min_\theta \left\{ D_{KL}\big[\, p(\hat{\mathbf{i}}_\theta) \parallel p(\mathbf{i}) \,\big] + D_{KL}\big[\, p(\hat{\mathbf{n}}_\theta) \parallel p(\mathbf{n}) \,\big] \right\}. \tag{5}$$

This means that the selected SVR parameters are those that give rise to both signal and noise estimates probabilistically similar to the true signal and noise respectively. Note that this similarity does not require analytical models of the PDFs since it can be computed from histograms (or signal and noise samples).

Of course, the independence assumption does not hold in general, however, as it will be shown in Section 4.2, this is not a critical fact for a good behavior of the method even in non-additive cases in which the noise is clearly signal-dependent. Moreover, the independence assumption simplifies the practical application of the criterion for SVR selection since, for a limited number of samples,

histogram estimations of $p(\mathbf{i})$ and $p(\mathbf{n})$ are far more reliable than histogram estimations of $p(\mathbf{i},\mathbf{n})$, which implies the duplication of the dimensionality (in an already high dimensional situation).

In the examples throughout the paper we restricted ourselves to second order KLD measures due to the lack of samples, yet capturing the second order structure of signal and noise. The optimization in Eq. (5) was carried out by exhaustive search.

3.4 Summary of the Proposed Denoising Method

The proposed denoising method can be summarized as follows. First the noisy image is transformed by a steerable wavelet filter bank. Then, a set of SVRs is applied to the patches of the subbands of the transform. These SVRs use the profiles for the penalization factor and the insensitivity computed from signal and noise samples as described in Section 3.2. The SVRs use the kernel based on MI that captures signal relations in the wavelet domain as described in Section 3.2. While the scaling of the penalization profile and the kernels are kept fixed as indicated in Section 3.2, the scaling of the insensitivity profile is automatically selected following the procedure described in section 3.3.

4. Behavior of the Proposed Method

In this section, we show an illustrative example of how the SVR parameters affect the estimated solution. Moreover we validate the proposed automatic procedure for SVR selection considering examples with different noise sources including non-additive and signal dependent cases.

4.1 Impact of SVR Parameters in Image Denoising

As stated above, the regularization behavior of the SVR depends on $\theta = (C_i, \varepsilon_i, K)$. Here we show the qualitative effect of the global penalization scaling C, the global insensitivity scaling τ, and the kernel width σ assuming a generalized RBF kernel. Figure 5 shows the qualitative effect of SVR estimation as a function of these parameters. Compare the results with the original and noisy subbands shown in Fig. 3.

Increasing the kernel width, σ (vertical direction), introduces too strong relations among coefficients in such a way that spurious energy appears in the reconstruction. Increasing the insensitivity, τ (horizontal direction), a sparser solution is obtained, leading to information loss and thus relevant features of the signal are discarded. On the contrary, a too small insensitivity value gives rise to overfitting, and hence noise is not removed. Small values of the C parameter gives rise to over-regularized estimations. Large enough values of C give rise to similar behavior (see comments in Section 3.2).

Of course, interactions among these parameters occur, and have been studied in other contexts elsewhere (Chalimourda et al., 2004; Cherkassky and Ma, 2003; Cherkassky, 2004). In the image denoising case, the deviation from an *appropriate* solution in combined directions of the parameters gives rise to different solutions that combine the negative effect of the departure in each direction.

The above example suggests that *appropriate* SVRs can certainly recover the underlying structure of the original signal from the noisy observation, which is the rationale of the proposed method.

4.2 Validation of the Automatic Procedure for SVR Selection

In this section, we validate the previous SVR selection procedure in two different ways. Firstly, note that KLD values in the example of Fig. 3 *qualitatively* illustrate the usefulness of the proposed

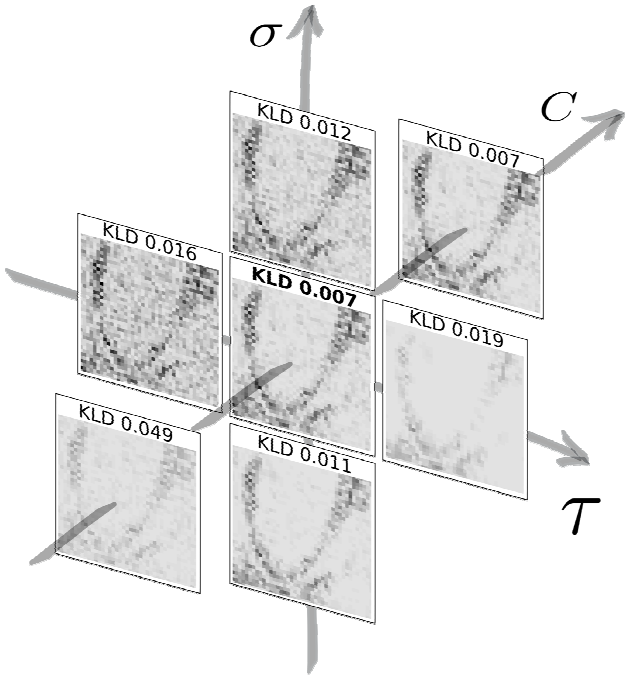


Figure 5: Effect of SVR parameters on the noisy wavelet patch of Fig. 3. The values of the KL-divergence criterion between the estimated and the actual PDFs of noise and signal are given in each case (see text in Section 3.3).

procedure: the minimum divergence solution (central subband patch) gives also a reasonable trade-off between smoothness and detail preservation of the original subband patch.

Secondly, we *quantitatively* show that the SVR that enforces the similarity between the estimated and actual signal and noise joint PDFs (in KLD terms) is not far from the SVR that maximizes the structural similarity between the estimated and the original image. In order to do so, we compare the KLD measures for different SVRs, with the corresponding distortion measured with the Structural SIMilarity (SSIM) index (Wang et al., 2004). The SSIM index is a widely used similarity measure, which is better related to human quality assessment than Euclidean measures, such as MSE or PSNR. Note that while KLD values are available in real situations (provided the noise histogram and a generic natural images histogram are known), distortion measures are not available since the original image is unknown. Consequently, the SSIM results next presented are for mere comparison purposes.

In this experiment, the SVM parameter space is reduced to the scaling factor on the insensitivity profile as recommended in Section 3.2. Accordingly, Fig. 6 shows the KLD and distortion (1-SSIM) results as a function of τ (see Eq. (3)). Curves are shown for different kinds of (Gaussian and non-Gaussian) noise sources corrupting a particular image (details on the noise sources are given in Section 5).

For the Gaussian noise case, two different variances are shown. It is worth noting that (1) the KLD criterion (solid) closely follows the actual distortion curve (dashed), and (2) the minima for low and high noise regime curves are very similar. These facts suggest that, in the Gaussian noise case, the proposed criterion is quite robust and provides consistent results: the higher the noise

(red curves) the higher the ε zone minima. Besides, the linear relation between ε and the noise standard deviation, reported in Kwok and Tsang (2003), is confirmed here: as expected, the scaling factor keeps fairly constant, $\tau \approx 2.5$, for both $\sigma_n^2 = 200$ and $\sigma_n^2 = 400$. Obviously, higher noise levels imply more distorted estimations. For other (non-Gaussian) noise sources, similar results are obtained. For the JPEG and JPEG2000 quantization noise sources, the KLD criterion also matches SSIM performance. For the case of more complex noise sources, such as vertical striping (VS) and Infra Red Imaging System (IRIS) noise, the criterion gives close-to-optimal solutions in SSIM terms. Note that, remarkably, the KLD criterion is better suited to the error minimization when the signal and noise independence assumption holds (Gaussian case). Therefore there is room to further improve the SVR selection criterion. The above results suggest that the proposed SVR selection procedure can be considered as a convenient approximation to distortion minimization (which is not possible in real situations).

5. Denoising Experiments and Discussion

In this section, we evaluate the performance of the proposed method in different scenarios for image denoising. Our algorithm is compared to several wavelet-based denoising methods using standard 256×256 images ('Barbara', 'Boats', 'Lena') with different levels and sources of degradation. In the following, we first give details on implementation issues of the considered algorithms. Then, we analyze their performance for several kinds of noise sources:

- Experiment 1. Additive Gaussian noise of different variances ($\sigma_n^2 = \{200, 400\}$).
- Experiment 2. Coding noise: JPEG and JPEG2000 at different quantization coarseness.
- Experiment 3. Acquisition noise: vertical striping and Infra Red Imaging System (IRIS) noise.

Note that the noise PDF is in general unknown, except for the academic case of Gaussian noise, but the histogram can be computed from samples in all cases.
All results are compared numerically by using the standard (yet not perceptually meaningful) RMSE, and the perceptually meaningful SSIM index (Wang et al., 2004). Moreover, representative examples are shown in every case for visual inspection. For proper visualization, all the results are equalized in the same way by truncating the values outside the $[0, 255]$ range.

5.1 Implementation Details

The algorithms that do not use information about the inter-coefficient relations (Donoho and Johnstone, 1995; Simoncelli, 1999; Figueiredo and Nowak, 2001) are straightforward to implement and have few parameters to tune. All these methods use orthogonal wavelet representations. In our particular implementation, we used 4-scale QMF wavelets from MatlabPyrTools.[3] In every case, we followed authors' prescriptions to choose these parameters for the best performance:

- *Hard Thresholding (HT)*. The key parameter is the threshold value λ. We use the noise variance to set the threshold, $\lambda = 3\sigma_n$, as suggested in Donoho and Johnstone (1995).

3. See `http://www.cns.nyu.edu/~eero/software.php`.

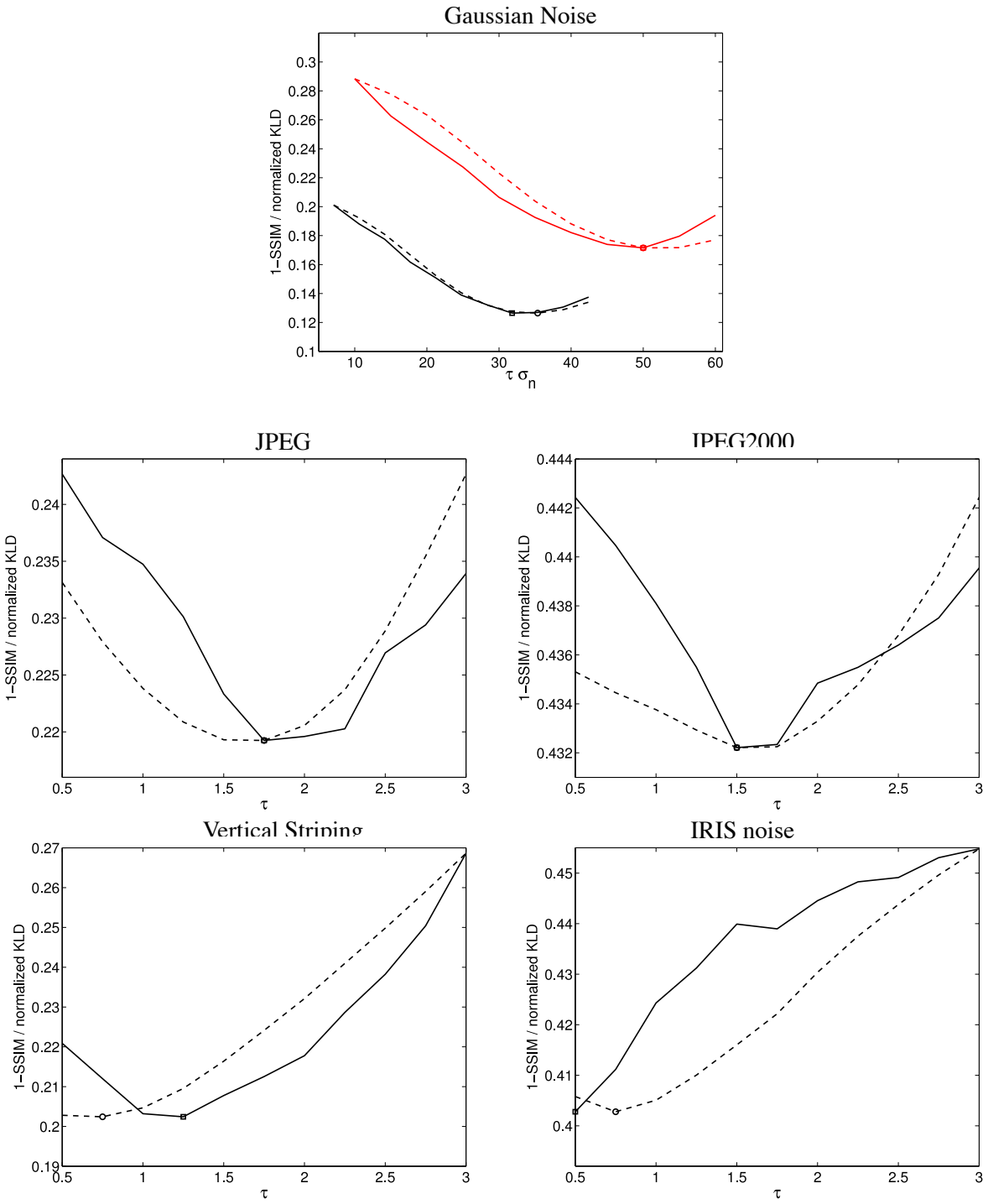


Figure 6: Validation of the proposed KLD criterion to adjust SVR parameter ε (or equivalently τ, see text). In every distortion case, solid lines represent the KLD criterion and dashed lines represent the distortion (1-SSIM). For proper visualization, KLD curves were normalized to fall in the same range as the distortion. In the Gaussian noise case, two different noise variances are considered: $\sigma_n^2 = 200$ (black lines) and $\sigma_n^2 = 400$ (red lines). As can be seen, the minima of the KL distance (squares) are always in the same region as the minima of the distortion (circles), thus giving rise to similar SSIM performance.

- *Soft Thresholding (ST)*. In our implementation, the threshold in each subband is derived from the standard deviation of the noise, σ_n, using optimized values to minimize the mean square error (MSE) in a set of 100 natural images. Threshold values were optimized for the σ_n^2 in the range [0,1600].

- *Bayesian Laplacian (BL)*. In this case, the parameters of the Laplacian distribution (s and p in Simoncelli, 1999) for the marginal PDFs in each subband are estimated by maximum likelihood (ML), as suggested by the author.

- *Bayesian Gaussian (BG)*. The threshold value was set according to the function of noise variance provided in Figueiredo and Nowak (2001).

On the other hand, in the case of the Gaussian Scale Mixture (GSM) (Portilla et al., 2003), which does consider inter-coefficient relations, we used the implementation provided by the authors.[4] We have used (1) the same representation as in the proposed method (4-scale, 8-orientation steerable pyramid), and (2) we also provided the average noise power spectral density (PSD) to achieve the best possible performance of the GSM method.

Details of the proposed SVR method are included in previous Section 3.2. A Matlab implementation of the algorithm is available online.[5] Since the C_i and ε_i profiles are computed off-line, the computational cost of the proposed method is mainly constrained by the SVR training. In our current implementation, we used the IRWLS algorithm in Matlab (Pérez-Cruz et al., 2000) in order to drop the bias term and incorporate the insensitivity and penalization profiles easily. These modifications are not trivial in faster implementations (Huang and Kecman, 2004; Kecman et al., 2004). As a result, our Matlab implementation takes about 30 seconds[6] for each image/noise estimation for a set of SVR parameters. Three strategies can be carried out for speeding up the optimization: (1) using faster SVR implementations (Platt, 1999; Chang and Lin, 2001; Tsang et al., 2005), (2) alternative procedures to exhaustive search on convex error surfaces (Torczon, 1997; Lewis and Torczon, 2002; Vishwanathan et al., 2006), and (3) restricting the dimension of the parameter space (as done in Section 3.2).

5.2 Experiment 1. Additive Gaussian Noise

Table 1 shows the distortion results for the three considered images and the two noise variances, $\sigma_n^2 = 200$ and $\sigma_n^2 = 400$.. The best SSIM values in each case are highlighted. In every case, we also provide the SVR^{opt} result, which is the best result the proposed method can get in SSIM terms. This is useful to assess the performance of the proposed divergence-based criterion and to give an upper bound of method's performance. Results show that our algorithm performs better than the methods that neglect signal relations (HT, ST, BG and BL), and obtains similar (yet slightly lower) numerical results than the one which incorporates them (GSM). It is not surprising that the GSM method achieves the best performance in this case, since it is analytically suited to deal with Gaussian noise. The SVR performance is consistent through all images and noise variances, thus suggesting that the guiding criterion is robust. Finally, it must be noted that, in the most difficult case of $\sigma_n^2 = 400$, GSM and SVR offer more similar results, and clearly outperform the rest of the methods.

4. See `http://decsai.ugr.es/~javier/denoise/`.
5. See `http://www.uv.es/vista/vistavalencia/denoising_SVR/`.
6. Computations were carried out in a 2.8GHz processor with 4GB RAM.

Method	'Barbara' SSIM	'Barbara' RMSE	'Boats' SSIM	'Boats' RMSE	'Lena' SSIM	'Lena' RMSE
HT	0.77	16.48	0.76	13.62	0.73	18.97
ST	0.78	14.37	0.79	10.26	0.74	12.59
BG	0.80	14.14	0.79	11.70	0.76	12.75
BL	0.81	12.95	0.83	8.30	0.78	11.66
GSM	**0.90**	8.94	**0.87**	8.94	**0.83**	13.61
SVR	0.87	10.11	0.84	10.16	0.81	12.54
SVR^{opt}	0.87	10.11	0.85	10.36	0.82	12.30

Method	'Barbara' SSIM	'Barbara' RMSE	'Boats' SSIM	'Boats' RMSE	'Lena' SSIM	'Lena' RMSE
HT	0.67	24.52	0.68	20.15	0.67	20.22
ST	0.69	19.04	0.71	16.16	0.66	19.72
BG	0.70	20.40	0.70	19.17	0.67	19.26
BL	0.73	16.52	0.77	10.26	0.67	18.45
GSM	**0.86**	11.02	0.80	17.40	**0.79**	15.95
SVR	0.83	13.13	**0.81**	10.73	0.78	14.50
SVR^{opt}	0.83	13.13	0.81	10.73	0.78	14.06

Table 1: Results for the Gaussian noise: distortions for different images and methods are given at $\sigma_n^2 = 200$ (top) and $\sigma_n^2 = 400$ (bottom).

Figure 7 shows representative visual results in the challenging situation of $\sigma_n^2 = 400$. It can be noticed that thresholding methods (HT, ST) and Bayesian generalizations not including signal relations in the model (BG, BL) show poor performance, producing images either grained or corrupted by too salient wavelet functions. Even though SVR yields slightly lower numerical scores than GSM, global visual performance is comparable.

5.3 Experiment 2. Coding Noise: JPEG and JPEG2000

In this section, we focus on restoring grayscale images after JPEG or JPEG2000 compression, which induces non-Gaussian noise: it produces heavy tailed marginal error PDFs in the spatial domain with non-negligible relations among the pixels (see comments in Section 5.5). Quantization noise is an illustrative example of how the proposed method can cope with non-Gaussian, colored and signal-dependent noise. In order to obtain the necessary samples to build the noise histograms, we used 100 images from the database described in Section 2 encoded by JPEG and JPEG2000. In the first case, the Matlab implementation of the JPEG algorithm with quality factors $Q = 9$ (small distortion) and $Q = 7$ (large distortion) was used. In the second case, scalar quantization of the QMF wavelet domain using standard JPEG2000 bit allocation tables (Taubman and Marcellin, 2001) was used. Different values of quantization coarseness, that will be referred to as Δ_1 (small distortion) and Δ_2 (large distortion) were applied.

Table 2 shows the quantitative results for all considered methods for the three images at different quantization levels. It can be noticed that again the SVR method outperforms the thresholding methods (HT, ST) and those not including signal relations in the model (BG, BL). SVR yields similar numerical scores than GSM in JPEG (Fig. 8). However, in JPEG2000 better numerical (Table 2 [bottom]) and visual (Fig. 9) results are obtained with SVR. In general, high frequency details are better preserved by our method, while GSM yields over smoothed solutions, particularly in JPEG2000.

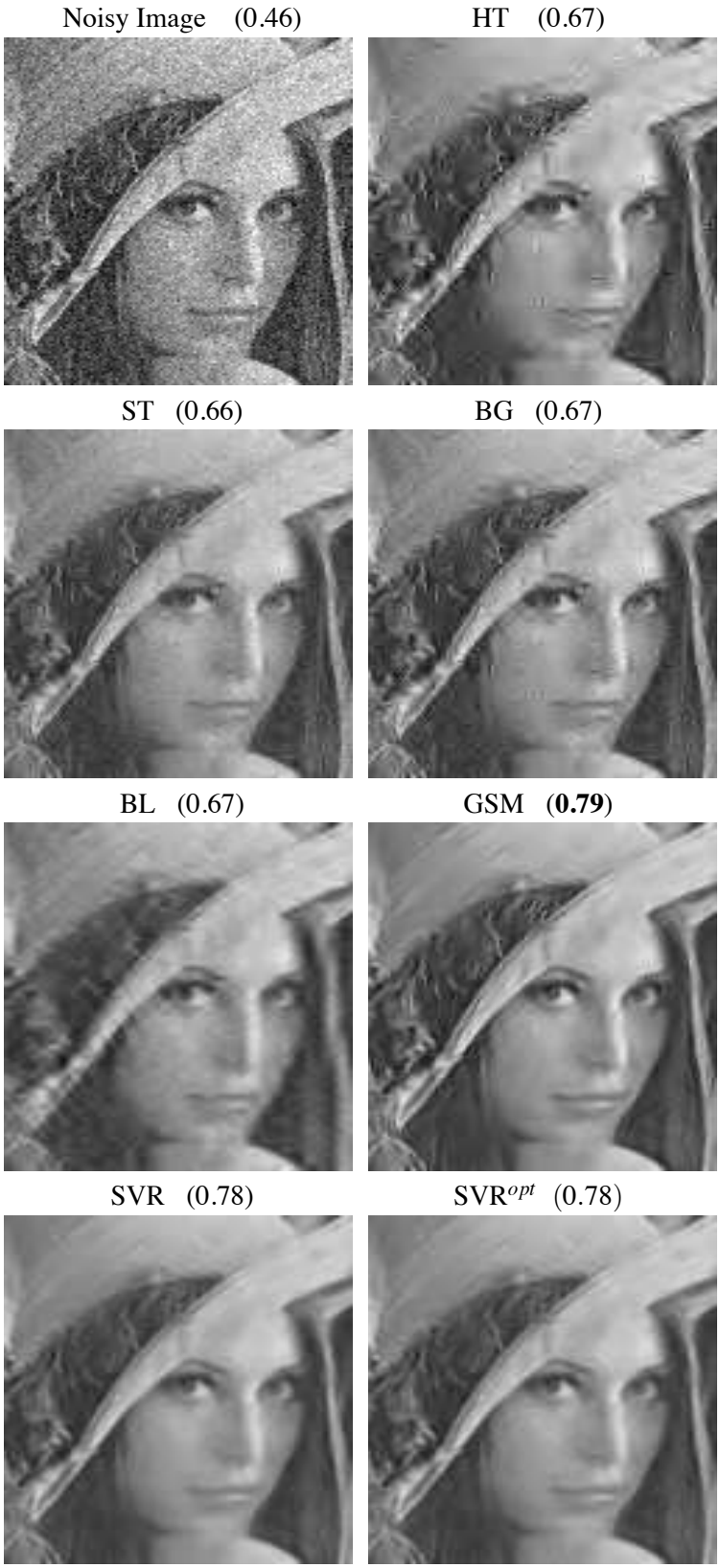


Figure 7: Visual results for the 'Lena' image corrupted with Gaussian noise, $\sigma_n^2 = 400$. SSIM values are given in parentheses.

Figure 8: Visual results for the 'Barbara' image with JPEG quantization noise ($Q = 7$). SSIM values are given in parentheses.

Figure 9: Visual results for the 'Barbara' image with coarse quantization JPEG2000 noise. SSIM values are given in parentheses.

JPEG	$Q=9$						$Q=7$					
	'Barbara'		'Boats'		'Lena'		'Barbara'		'Boats'		'Lena'	
Method	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE
HT	0.70	20.05	0.75	13.07	0.70	18.40	0.65	22.11	0.71	16.34	0.65	24.99
ST	0.73	17.51	0.78	11.59	0.73	15.13	0.68	19.71	0.75	12.72	0.68	18.77
BG	0.72	18.76	0.77	12.30	0.72	16.27	0.66	21.57	0.74	13.32	0.67	21.05
BL	0.71	20.37	0.77	13.43	0.73	16.52	0.64	21.67	0.74	14.70	0.69	17.65
GSM	0.77	15.50	**0.80**	11.15	**0.75**	13.66	**0.71**	18.56	**0.77**	12.18	**0.71**	17.45
SVR	**0.78**	14.89	0.78	12.13	0.74	13.22	**0.71**	18.42	0.76	12.84	**0.71**	15.68
SVR^{opt}	0.78	14.89	0.80	11.35	0.75	13.97	0.73	18.28	0.76	12.89	0.71	15.72
JPEG2000	Δ_2						Δ_1					
	'Barbara'		'Boats'		'Lena'		'Barbara'		'Boats'		'Lena'	
Method	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE
HT	0.54	30.81	0.55	26.23	0.51	32.66	0.67	24.82	0.59	25.18	0.56	28.25
ST	0.55	28.83	0.55	25.15	0.51	31.24	0.68	22.52	0.60	23.69	0.56	27.47
BG	0.54	30.37	0.55	26.08	0.51	32.45	0.67	24.16	0.59	24.92	0.56	28.10
BL	0.54	30.30	0.55	25.87	0.51	29.05	0.67	24.35	0.59	24.79	0.56	28.12
GSM	0.55	28.47	**0.57**	20.92	**0.52**	25.84	0.68	20.54	**0.64**	17.94	0.58	23.64
SVR	**0.57**	25.31	**0.57**	21.88	**0.52**	29.32	**0.71**	17.23	**0.64**	18.27	**0.59**	21.55
SVR^{opt}	0.57	25.31	0.57	21.74	0.52	25.35	0.72	17.04	0.64	18.27	0.59	21.55

Table 2: Results for the coding noise: distortions at different quality levels of JPEG ($Q = \{9,7\}$) and JPEG2000 (coarseness Δ_1 and Δ_2) are given for different images and methods.

5.4 Experiment 3. Acquisition Noise: Vertical Striping and IRIS Noise

Real imaging systems introduce complex forms of noise depending on the acquisition process, so assuming a particular PDF for all cases is far from being realistic. For instance, variation of the intensity between neighboring elements of the CCD typically leads to vertical striping noise in pushbroom sensors (Mouroulis et al., 2000; Barducci and Pippi, 2001). Other typical acquisition noise source is observed in infrared imaging cameras, which is a complex mixture of different noise sources. In this section, we pay attention to these two particular non-Gaussian realistic acquisition noises through controlled experiments:

1. *Vertical striping noise*. We simulated this noise by modifying 4% of the image columns selected randomly. The luminance of the selected columns was modified by a random factor following a uniform distribution between 0.8 and 1. Spatial coherence was forced by attaching groups of contiguous 5 to 10 strips.

2. *InfraRed Imaging System (IRIS) noise*. Inspired in the observed characteristics of a representative number of acquired images by a commercial IR camera, the noise was modeled by a combination of four noise sources: low-variance Gaussian noise ($\sigma_n^2 \approx 50$), 'salt-and-pepper' noise (with a percentage of corrupted pixels about 0.05%), some spatially coherent missing pixels (black patches), and interlaced lines all over the image.

In both cases, we computed the contrast noise PDF, $p(\mathbf{n})$, from 100 noisy images. In the next Section 5.5, the non-Gaussian nature of these acquisition noise PDFs is shown.
Table 3 shows the obtained numerical results for all images and both acquisition noise sources. In both complex scenarios, the proposed SVR-based method outperforms GSM and the rest of

Method	'Barbara'		'Boats'		'Lena'	
	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE
HT	0.73	17.43	0.73	15.99	0.69	18.07
ST	0.77	15.71	0.78	14.04	0.75	14.08
BG	0.76	16.01	0.76	14.75	0.73	15.14
BL	0.77	16.56	0.81	14.96	0.77	14.64
GSM	0.79	14.83	0.79	14.36	0.75	14.45
SVR	**0.80**	15.66	**0.80**	13.47	**0.79**	13.18
SVR^{opt}	0.80	15.45	0.82	14.25	0.80	13.31
HT	0.50	30.80	0.58	28.70	0.56	28.81
ST	0.55	27.02	0.64	23.48	0.60	24.40
BG	0.54	28.40	0.62	25.44	0.59	26.20
BL	0.50	28.74	0.60	21.77	0.55	24.08
GSM	0.53	30.51	0.64	25.92	0.61	30.99
SVR	**0.59**	31.07	**0.67**	21.44	**0.66**	31.44
SVR^{opt}	0.60	30.71	0.70	24.56	0.66	32.05

Table 3: Acquisition noise: vertical striping (top) and IRIS noise (bottom). Distortions for different images and methods.

methods numerically. A noticeable gain in SSIM is observed, which is confirmed when looking at the restored images in Figs. 10 and 11. It is worth noting that in the vertical striping noise (Fig. 10), SVR yields a sharper (and more realistic) reconstruction while GSM produces an over-blurred solution. In the case of the IRIS noise, only SVR removes the interlacing noise contribution, producing better visual results. Including the average PSD information in GSM, as we do in the experiments, improves its performance. However, it is not enough to remove the interlacing artifact due to the particular nature of IRIS noise. IRIS noise is difficult because the PSD and variance of each particular realization of the noise may substantially differ from the (estimated) averages. On the contrary, the proposed SVR method uses an adaptive cost function learned from the noisy image. Here, nevertheless, the upper bound of performance is not met, suggesting that there is still room for improving the selection criterion proposed, possibly considering the joint density.

5.5 Analysis of the Residuals

Further qualitative insight in the obtained solutions can be achieved by comparing the estimated and actual PDFs of signal and noise with the different methods and noise sources. Since we are restricting ourselves to second order KLD criterion, this comparison reduces to assess the difference between 2D histograms (in the spatial domain).

It is widely known that the PDF of pairs of neighbor pixels in natural images is an oriented ellipsoid reflecting the strong correlation among luminance values in the spatial domain (Clarke, 1985). The corresponding restored images (even for the worse performing algorithms) also display such strong local correlation. Therefore, no relevant conclusion is gained by direct inspection of these histograms (results not shown). On the contrary, the 2D histograms of the noise are more

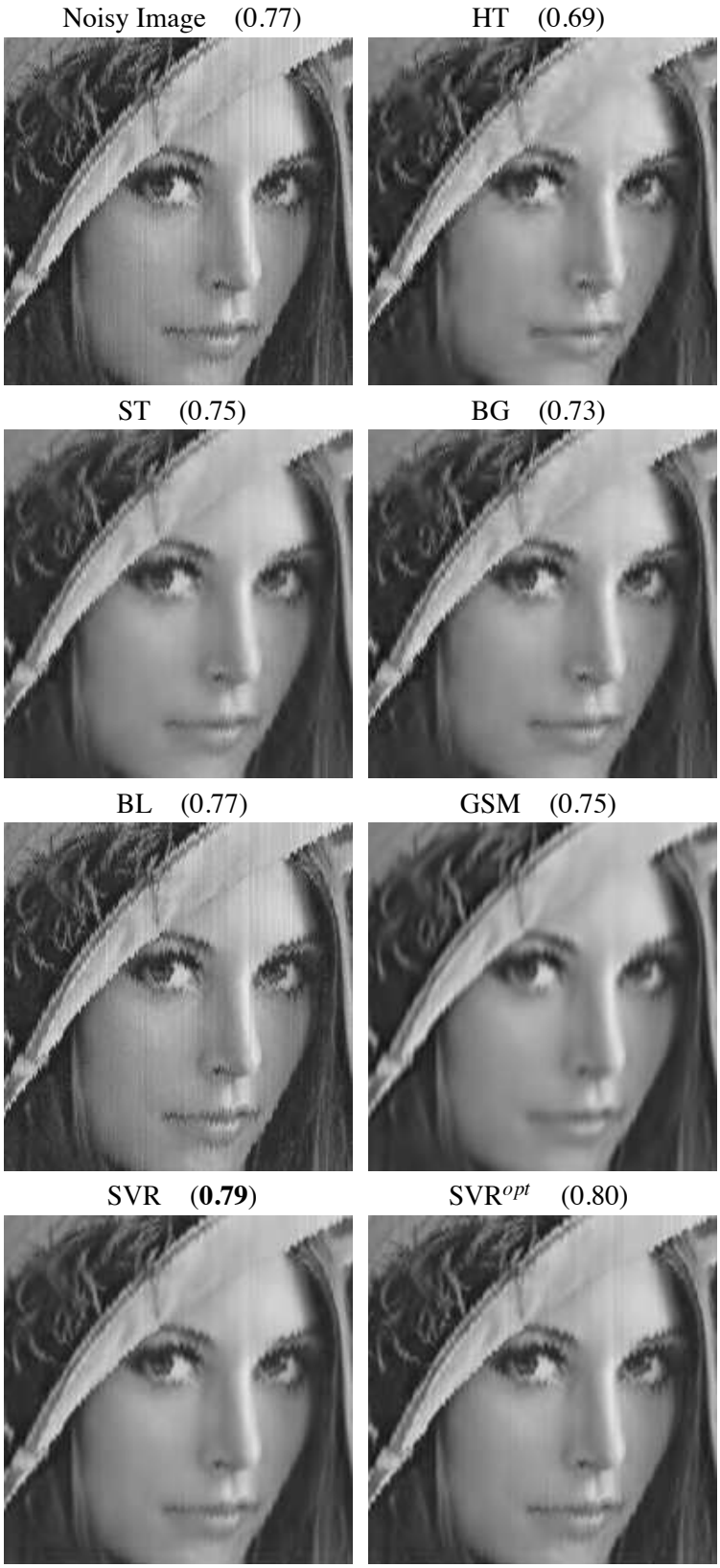


Figure 10: Visual results for the 'Lena' image with vertical striping noise. SSIM values are given in parentheses.

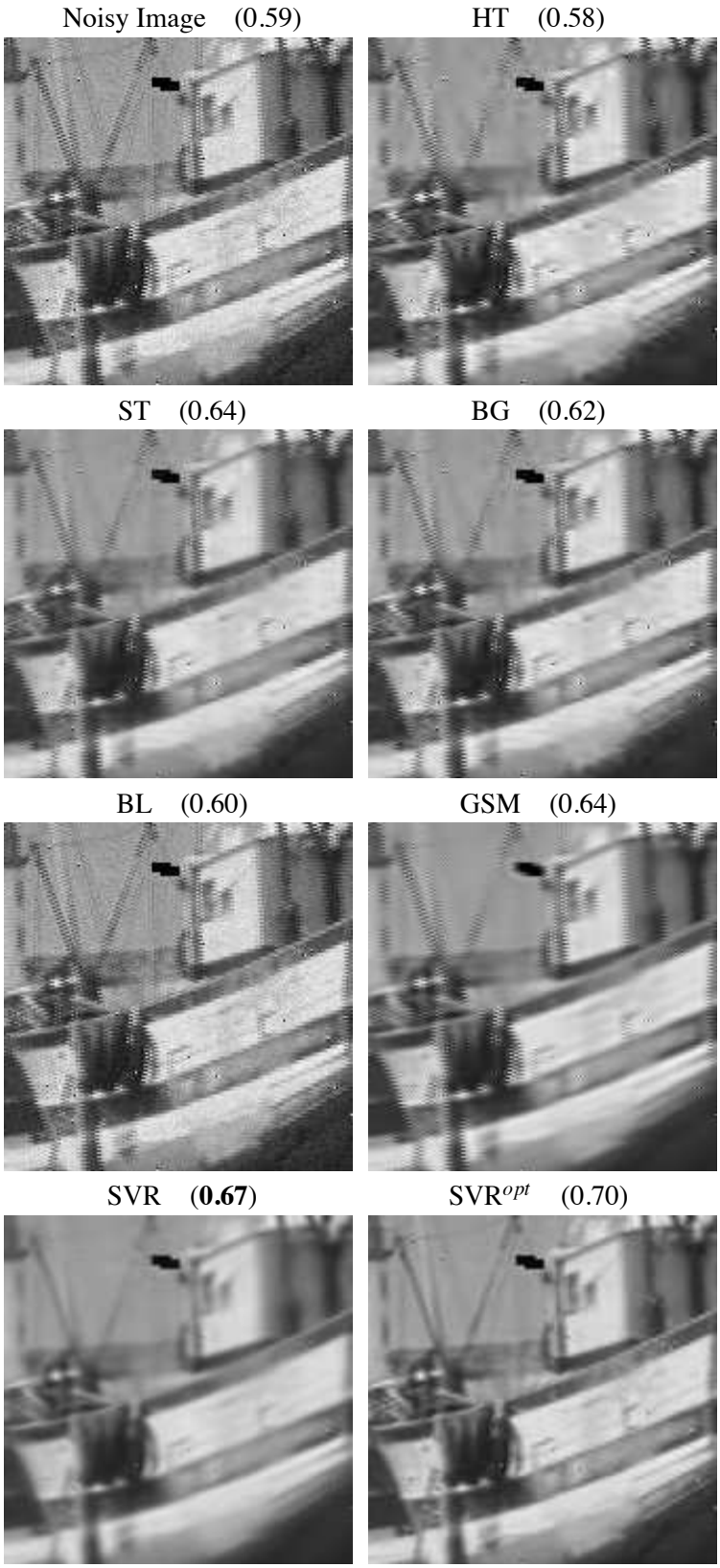


Figure 11: Visual results for the 'Boats' image with IRIS noise. SSIM values are given in parentheses.

suitable for direct inspection because (1) actual noise histograms are quite different for the different noise sources, and (2) the estimated histograms strongly depend on the denoising method.

Figure 4 represents the distribution of the actual and estimated noise PDFs by all the considered methods in the spatial domain. It can be noticed that, for the Gaussian noise, all methods reproduce quite well the shape and extent of the PDF, as expected for the parametric models, which use a proper Gaussian noise model. Note that the SVR method also succeeds in approximating the energy of the noise even without using the Gaussian assumption explicitly.

For non-Gaussian noise sources, the behavior of the methods markedly differ. For instance, the quantization noise induced by JPEG/JPEG2000 follows a non-Gaussian, oriented joint distribution (the central dark area is actually an oriented ellipsoid), indicating correlation among noise samples. In the case of JPEG, this central ellipsoid is better reproduced by hard thresholding and the proposed SVR method. The other methods slightly underestimate the variance of the noise. For the case of JPEG2000, methods not considering signal relations dramatically underestimate the noise variance. In the case of more complex noise sources, such as vertical striping or IRIS, none of the methods reproduce the low probability structure (light gray regions). However, the central peak is poorly reproduced by marginal methods, either overestimating (HT, ST, BG) or underestimating (BL) the width. On the contrary, GSM and SVR give more reasonable width estimation. To conclude, methods assuming an (inadequate) Gaussian noise model do not match, in general, the noise distribution, so they should be reformulated for each particular noise source, which may be complicated or even impossible. GSM constitutes an exception to this statement, since results suggest that the quality of the signal model compensates the unsuitability of the noise model. On the contrary, this is not necessary for the proposed method, which only needs examples of noisy images to *learn* from.

6. Conclusions

In this work, we proposed an alternative non-parametric way to take into account the relations among natural image wavelet coefficients for denoising: we used SVR in the wavelet domain to enforce these relations in the estimated signal. The specific signal relations, which proved to be more relevant in intraband coefficients, are encoded in an anisotropic kernel based on mutual information computed from a representative image database. An adaptive SVR with different cost function *per* subband was developed: the subband-dependent ε_i and C_i are modeled by analyzing the particular signal and noise variances in a representative image database. By following general recommendations for the design of the kernel, ε_i and C_i, and adapting them to the particular image denoising problem, we restricted the class of appropriate SVRs. A KLD-based criterion was proposed to automatically select the SVR that best recovers the relevant wavelet coefficient relations of the true signal. The criterion was quite consistent but there is still room for improvement, specially in the case of complex noise sources.

Results show that the performance of the proposed non-parametric method is (1) better than conventional wavelet methods that assume coefficient independence, (2) similar to state-of-the-art methods that do explicitly include these relations when the noise source is Gaussian, and (3) numerically and visually better results are obtained when more complex realistic noise sources are considered. Therefore, the proposed SVR approach can be seen as a more flexible (model-free) alternative to the explicit description of coefficient relations. The important thing here is that no reformulation is needed for dealing with any other kinds of noise. Moreover, these results are an

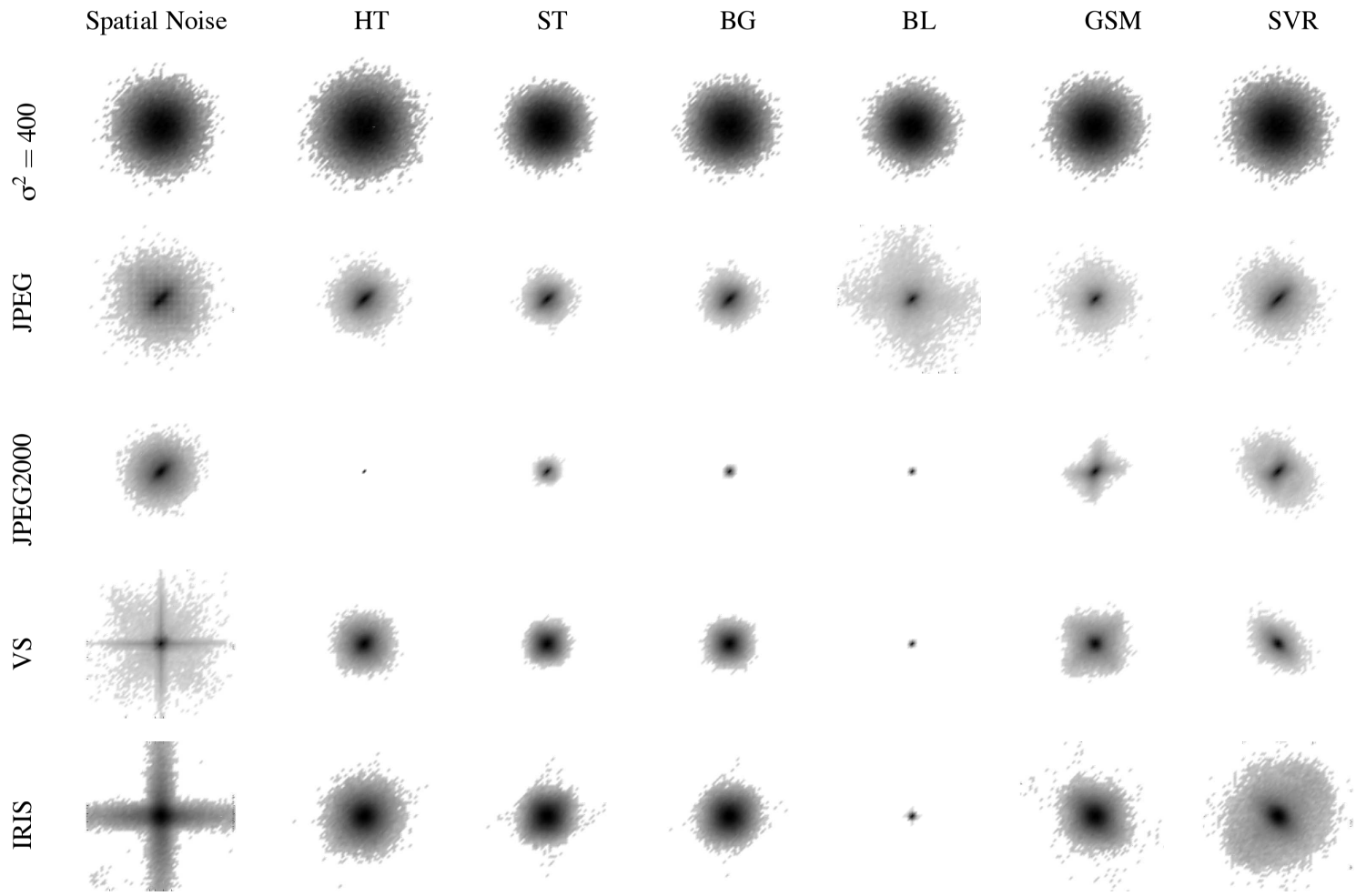


Table 4: 2D histograms of the residuals in the spatial domain for all methods and noise sources (darker regions indicate higher probability). In all cases, we considered a pixel and its right-hand neighbor (one pixel shift). All the histogram values have been exponentiated to 0.25 for better visualization.

additional indication that relation between local frequency coefficients is a salient natural image feature that should not be neglected in denoising applications.

Future work is tied to the incorporation of new information in the kernels: here we focused on the consideration of signal relations in the kernel, but the particular structure of the noise could be eventually incorporated. Note that the denoising procedure is quite general and admits any kind of non-parametric regression machine, such as Gaussian Processes.

Acknowledgments

This work was partially supported by projects CICYT-FEDER TEC2009-13696, AYA2008-05965-C04-03, and CSD2007-00018. Valero Laparra acknowledges the support of the Ph.D grant BES-2007-16125. The authors thank the reviewers for thorough and constructive comments on the submitted manuscript.

References

H.C. Andrews and B.R. Hunt. *Digital Image Restoration*. Prentice Hall, NY, 1977.

M. Banham and A. Katsaggelos. Digital image restoration. *IEEE Signal Processing Magazine*, 14: 24–41, 1997.

A. Barducci and I. Pippi. Analysis and rejection of systematic disturbances in hyperspectral remotely sensed images of the Earth. *Applied Optics*, 40:1464–1477, 2001.

A. J. Bell and T. J. Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338, 1997.

M. Bertero, T.A. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76(8):869–889, 1988.

R. W. Buccigrossi and E. P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. *IEEE Transactions on Image Processing*, 8(12):1688–1701, 1999.

P. J. Burt and E. H. Adelson The Laplacian Pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.

G. Camps-Valls, E. Soria-Olivas, J. Pérez-Ruixo, A. Artés-Rodríguez, F. Pérez-Cruz, and A. Figueiras-Vidal. A profile-dependent kernel-based regression for cyclosporine concentration prediction. In *Neural Information Processing Systems (NIPS) – Workshop on New Directions in Kernel-Based Learning Methods*, Vancouver, Canada, December 2001.

G. Camps-Valls, J. Gutiérrez, G. Gómez, and J. Malo. On the suitable domain for SVM training in image coding. *Journal of Machine Learning Research*, 9:49–66, 2008.

A Chalimourda, B Schölkopf, and Alex J. Smola. Experimentally optimal ν in support vector regression for different noise models and parameter settings. *Neural Networks*, 17(1):127–141, 2004.

C. Chih Chang and Chih J. Lin. *libSVM: A Library for Support Vector Machines,* (http://www.csie.ntu.edu.tw/~cjlin/libsvm/), 2001.

H. Cheng, J.W. Tian, J. Liu, and Q.Z. Yu. Wavelet domain image denoising via SVR. *IEE Electronics Letters*, 40, 2004.

V. Cherkassky. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks*, 17(1):113–126, 2004.

V. Cherkassky and Y. Ma. Comparison of model selection for regression. *Neural Comput.*, 15(7): 1691–1714, 2003.

R.J. Clarke. *Transform Coding of Images*. Academic Press, New York, 1985.

R. R. Coifman and D. L. Donoho. Translation-invariant de-noising. In A. Antoniadis and G. Oppenheim, editors, *Wavelets and Statistics*. *Lecture Notes in Statistics*, volume 103, pages 125–150. Springer, Berlin, Department of Statistics, 1995.

T.M. Cover and J.A. Tomas. *Elements of Information Theory*. John Wiley & Sons, New York, 1991.

K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

D. L. Donoho and I. M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90:1200–1224, 1995.

D. Field. Relations between the Statistics of Natural Images and the Response Properties of Cortical Cells. *Journal of the Optical Society of America A*, 4(12):2379–2394, 1987.

M. Figueiredo and R. Nowak. Wavelet-based image estimation: an empirical Bayes approach using Jeffrey's noninformative prior. *IEEE Transactions on Image Processing*, 10(9):1322–1331, 2001.

W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

G. Gómez, G. Camps-Valls, J. Gutiérrez, and J. Malo. Perceptual adaptive insensitivity for support vector machine image coding. *IEEE Transactions on Neural Networks*, 16(6):1574–1581, 2005.

B. Goossens, and Pizurica, A. and Philips, W. Removal of correlated noise by modeling the signal of interest in the wavelet domain. *IEEE Transactions on Image Processing*, 18(6):1153–1165, 2009.

J. Gutiérrez, F. Ferri, and J. Malo. Regularization operators for natural images based on nonlinear perception models. *IEEE Transactions on Image Processing*, 15(1):189–200, 2006.

T.-M. Huang and V. Kecman. Bias term *b* in SVMs again. In *12th European Symposium on Artificial Neural Network, ESANN 2004*, pages 441–448, Bruges, Belgium, April 2004.

A. Hyvärinen. Sparse code shrinkage: denoising of nongaussian data by maximum likelihood estimation. *Neural Computation*, 11(7):1739–1768, 1999.

A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, New York, 2001.

T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.

D. Kai Tick Chow and T. Lee. Image approximation and smoothing by support vector regression. In *International Joint Conference on Neural Networks, IJCNN'01*., volume 4, pages 2427–2432, Washington, DC, USA, 2001.

V. Kecman, T. Huang, and M. Vogt. Iterative single data algorithm for training kernel machines from huge data sets: Theory. In *Performance, Support Vector Machines: Theory and Applications, Springer-Verlag, Studies in Fuzziness and Soft Computing*, p 255–274, 2004.

C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image denoising and representation. *Intl Journal of Computer Vision*, 16(2):349–366, Feb 2007.

N. Kingsbury. Rotation-invariant local feature matching with complex wavelets. In *Proc. European Conference on Signal Processing (EUSIPCO)*, Florence, Italy, 2006.

J. T. Kwok and I. W. Tsang. Linear dependency between ε and the input noise in ε-Support Vector Regression. *IEEE Transactions on Neural Networks*, pages 544–553, May 2003.

V. Laparra, J. Muñoz-Marí, and J. Malo. Divisive Normalization Image Quality Metric Revisited. *Accepted for publication in Journal of the Optical Society of America A*, 2010.

R. Michael Lewis and V. Torczon. A globally convergent augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds. *SIAM J. on Optimization*, 12(4):1075–1089, 2002.

J. Liu and P. Moulin. Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients. *IEEE Transactions on Image Processing*, 10:1647–1658, 2001.

J. Malo, I. Epifanio, R. Navarro, and E. Simoncelli. Non-linear image representation for efficient perceptual coding. *IEEE Transactions on Image Processing*, 15(1):68–80, 2006.

J. Malo and J. Gutiérrez. V1 non-linear properties emerge from local-to-global non-linear ICA. *Network: Computation in Neural Systems*, 17(1):85–102, 2006.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London*, CCIX(A456):215–228, May 1905.

P. Mouroulis, R. O. Green, and T. G. Chrien. Design of pushbroom imaging spectrometers for optimum recovery of spectroscopic and spatial information. *Applied Optics*, 39:2210–2220, 2000.

A. Navia-Vázquez, and F. Pérez-Cruz, and A. Artés-Rodríguez, and A. Figueiras-Vidal. Weighted least squares training of support vector classifiers leading to compact and adaptive schemes *IEEE Transactions on Neural Networks*, 12:1047–1059, 2001.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

F. Pérez-Cruz, A. Navia-Vázquez, P. Alarcón-Diana, and A. Artés-Rodríguez. An IRWLS procedure for SVR. In *European Signal Processing Conference (EUSIPCO)*, Tampere, Finland, Sept 2000.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal on Computer Vision*, 40(1):49–71, 2000.

J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli. Image denoising using a scale mixture of Gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003.

L. Siwei and E. Simoncelli. Statistical modeling of images with fields of GSMs. In *Proc. NIPS06'*. MIT Press, May 2007.

E. Simoncelli. Bayesian denoising of visual images in the wavelet domain. In *Bayesian Inference in Wavelet Based Models*, pages 291–308. Springer-Verlag, New York, 1999.

E. Simoncelli and W. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *Proc 2nd IEEE Int'l Conference on Image Processing*, 1995.

E. P. Simoncelli. Statistical models for images: Compression, restoration and synthesis. In *31st Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA*, 1997.

A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

H. Stark and J. Woods. *Probability, Random Processes, and Estimation Theory for Engineers*. Prentice Hall, NJ, 1994.

H. Takeda, S. Farsiu, and P. Milanfar. Kernel regression for image processing and reconstruction. *IEEE Transactions on Image Processing*, 16(2):349–366, Feb 2007.

D. S. Taubman and M. W. Marcellin. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, Boston, 2001.

V. Torczon. On the convergence of pattern search algorithms. *SIAM J. on Optimization*, 7(1):1–25, 1997.

I. W. Tsang, J. T. Kwok, and P. Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.

B. van Ginneken and A. Mendrik. Image denoising with k-nearest neighbor and support vector regression. In *The 18th International Conference on Pattern Recognition, ICPR'06*, volume 3, pages 603–606, Hong Kong, 2006.

S. V. N. Vishwanathan, Nicol N. Schraudolph, and Alex J. Smola. Step size adaptation in reproducing kernel hilbert space. *Journal of Machine Learning Research*, 7:1107–1133, 2006.

Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

J. Weston, C. Leslie, E. Ie, D. Zhou, A. Elisseeff, and W. Stafford Noble. Semi-supervised protein classification using cluster kernels. *Bioinformatics*, 21(15):3241–3247, 2004.

Journal of Machine Learning Research 11 (2010) 905-934 Submitted 12/08; Revised 8/09; Published 2/10

On Learning with Integral Operators

Lorenzo Rosasco[*] LROSASCO@MIT.EDU
Center for Biological and Computational Learning
Massachusetts Institute of Technology
Cambridge, MA, USA

Mikhail Belkin MBELKIN@CSE.OHIO-STATE.EDU
Department of Computer Science and Engineering
Ohio State University
Columbus, OH, USA

Ernesto De Vito[†] DEVITO@DIMA.UNIGE.IT
Dipartimento di Scienze per l'Architettura
Università di Genova, Italy

Editor: Sridhar Mahadevan

Abstract

A large number of learning algorithms, for example, spectral clustering, kernel Principal Components Analysis and many manifold methods are based on estimating eigenvalues and eigenfunctions of operators defined by a similarity function or a kernel, given empirical data. Thus for the analysis of algorithms, it is an important problem to be able to assess the quality of such approximations. The contribution of our paper is two-fold:
1. We use a technique based on a concentration inequality for Hilbert spaces to provide new much simplified proofs for a number of results in spectral approximation.
2. Using these methods we provide several new results for estimating spectral properties of the graph Laplacian operator extending and strengthening results from von Luxburg et al. (2008).

Keywords: spectral convergence, empirical operators, learning integral operators, perturbation methods

1. Introduction

A broad variety of methods for machine learning and data analysis from Principal Components Analysis (PCA) to Kernel PCA, Laplacian-based spectral clustering and manifold methods, rely on estimating eigenvalues and eigenvectors of certain data-dependent matrices. In many cases these matrices can be interpreted as empirical versions of underlying integral operators or closely related objects, such as continuous Laplacian operators. Thus establishing connections between empirical operators and their continuous counterparts is essential to understanding these algorithms. In this paper, we propose a method for analyzing empirical operators based on concentration inequalities in Hilbert spaces. This technique together with perturbation theory results allows us to derive a number of results on spectral convergence in an exceptionally simple way. We note that the approach using concentration inequalities in a Hilbert space has already been proved useful for analyzing

∗. Also at Dipartimento di Informatica e Scienze dell'Informazione, Università di Genova, Italy.
†. Also at INFN, Sezione di Genova, Italy.

supervised kernel algorithms, see De Vito et al. (2005b), Yao et al. (2007), Bauer et al. (2007) and Smale and Zhou (2007). Here we develop on this approach to provide a detailed and comprehensive study of perturbation results for empirical estimates of integral operators as well as empirical graph Laplacians.

In recent years several works started considering these connections. The first study of this problem appeared in Koltchinskii and Giné (2000) and Koltchinskii (1998), where the authors consider integral operators defined by a kernel. In Koltchinskii and Giné (2000) the authors study the relation between the spectrum of an integral operator with respect to a probability distribution and its (modified) empirical counterpart in the framework of U-statistics. In particular they prove that the ℓ_2 distance between the two (ordered) spectra goes to zero under the assumption that the kernel is symmetric and square integrable. Moreover, under some stronger conditions, rates of convergence and distributional limit theorems are obtained. The results are based on an inequality due to Lidskii and to Wielandt for finite dimensional matrices and the Marcinkiewicz law of large numbers. In Koltchinskii (1998) similar results were obtained for convergence of eigenfunctions and, using the triangle inequality, for spectral projections. These investigations were continued in Mendelson and Pajor (2005) and Mendelson and Pajor (2006), where it was shown that, under the assumption that the kernel is of positive type, the problem of eigenvalue convergence reduces to the study of how the random operator $\frac{1}{n}\sum_{i=1}^{n} X_i \otimes X_i$ deviates from its average $\mathbb{E}[X \otimes X]$, with respect to the operator norm, where $X, X_1, \ldots, X_n$ are i.i.d ℓ_2 random vectors. The result is based on a symmetrization technique and on the control of a suitable Radamacher complexity.
The above studies are related to the problem of consistency of kernel PCA considered in Shawe-Taylor et al. (2002) and Shawe-Taylor et al. (2005) and refined in Zwald et al. (2004) and Zwald and Blanchard (2006). In particular, Shawe-Taylor et al. (2002) and Shawe-Taylor et al. (2005) study the deviation of the sum of the all but the largest k eigenvalues of the empirical matrix to its mean using McDiarmid inequality. The above result is improved in Zwald et al. (2004) where fast rates are provided by means of a localized Rademacher complexities approach. The results in Zwald and Blanchard (2006) are a development of the results in Koltchinskii (1998). Using a new perturbation result the authors study directly the convergence of the whole subspace spanned by the first k eigenvectors and are able to show that only the gap between the k and $k+1$ eigenvalue affects the estimate. All the above results hold for symmetric, positive definite kernels.

A second related series of works considered convergence of the graph Laplacian in various settings , see for example, Belkin (2003), Lafon (2004), Belkin and Niyogi (2005), Hein et al. (2005), Hein (2006), Singer (2006) and Giné and Koltchinskii (2006). These papers discuss convergence of the graph Laplacian directly to the Laplace-Beltrami operator. Convergence of the normalized graph Laplacian applied to a fixed smooth function on the manifold is discussed in Hein et al. (2005), Singer (2006) and Lafon (2004). Results showing uniform convergence over some suitable class of test functions are presented in Hein (2006) and Giné and Koltchinskii (2006). Finally, convergence of eigenvalues and eigenfunctions for the case of the uniform distribution was shown in Belkin and Niyogi (2007).

Unlike these works, where the kernel function is chosen adaptively depending on the number of points, we will be primarily interested in convergence of the graph Laplacian to its continuous (population) counterpart for a *fixed* weight function. Von Luxburg et al. (2004) study the convergence of the second eigenvalue which is relevant in spectral clustering problems. These results are extended in von Luxburg et al. (2008), where operators are defined on the space of continuous functions. The analysis is performed in the context of perturbation theory in Banach spaces and bounds

on individual eigenfunctions are derived. The problem of out-of-sample extension is considered via a Nyström approximation argument. By working in Banach spaces the authors have only mild requirements for the weight function defining the graph Laplacian, at the price of having to do a fairly complicated analysis.

Our contribution is twofold. In the first part of the paper, we assume that the kernel K is symmetric and positive definite. We start considering the problem of out-of-sample extension of the kernel matrix and discuss a singular value decomposition perspective on Nyström-like extensions. More precisely, we show that a finite rank (extension) operator acting on the Reproducing Kernel Hilbert space $\mathcal{H}$ defined by K can be naturally associated with the empirical kernel matrix: the two operators have same eigenvalues and related eigenvectors/eigenfunctions. The kernel matrix and its extension can be seen as compositions of suitable restriction and extension operators that are explicitly defined by the kernel. A similar result holds true for the asymptotic integral operator, whose restriction to $\mathcal{H}$ is a Hilbert-Schmidt operator. We can use concentration inequalities for operator valued random variables and perturbation results to derive concentration results for eigenvalues (taking into account the multiplicity), as well as for the sums of eigenvalues. Moreover, using a perturbation result for spectral projections, we derive finite sample bounds for the deviation between the spectral projection associated with the k largest eigenvalues. We recover several known results with simplified proofs, and derive new results.

In the second part of the paper, we study the convergence of the asymmetric normalized graph Laplacian to its continuous counterpart. To this aim we consider a fixed positive symmetric weight function satisfying some smoothness conditions. These assumptions allow us to introduce a suitable intermediate Reproducing Kernel Hilbert space $\mathcal{H}$, which is, in fact, a Sobolev space. We describe explicitly restriction and extension operators and introduce a finite rank operator with spectral properties related to those of the graph Laplacian. Again we consider the law of large numbers for operator-valued random variables to derive concentration results for empirical operators. We study behavior of eigenvalues as well as the deviation of the corresponding spectral projections with respect to the Hilbert-Schmidt norm. To obtain explicit estimates for spectral projections we generalize the perturbation result in Zwald and Blanchard (2006) to deal with non-self-adjoint operators. From a technical point the main difficulty in studying the asymmetric graph Laplacian is that we no longer assume the weight function to be positive definite so that there is no longer a natural Reproducing Kernel Hilbert space space associated with it. In this case we have to deal with non-self-adjoint operators and the functional analysis becomes more involved. Comparing to von Luxburg et al. (2008), we note that the RKHS $\mathcal{H}$ replaces the Banach space of continuous functions. Assuming some regularity assumption on the weight functions we can exploit the Hilbert space structure to obtain more explicit results. Among other things, we derive explicit convergence rates for a large class of weight functions. Finally we note that for the case of positive definite weight functions results similar to those presented here have been independently derived by Smale and Zhou (2009).

The paper is organized as follows. We start by introducing the necessary mathematical objects in Section 2. We recall some facts about the properties of linear operators in Hilbert spaces, such as their spectral theory and some perturbation results, and discuss some concentration inequalities in Hilbert spaces. This technical summary section aims at making this paper self-contained and provide the reader with a (hopefully useful) overview of the needed tools and results. In Section 3, we study the spectral properties of kernel matrices generated from random data. We study concentration of operators obtained by an out-of-sample extension using the kernel function and obtain

considerably simplified derivations of several existing results on eigenvalues and eigenfunctions. We expect that these techniques will be useful in analyzing algorithms requiring spectral convergence. In fact, in Section 4, we apply these methods to prove convergence of eigenvalues and eigenvectors of the asymmetric graph Laplacian defined by a fixed weight function. We refine the results in von Luxburg et al. (2008), which, to the best of our knowledge, is the only other paper considering this problem so far.

2. Notation and Preliminaries

In this section we will discuss various preliminary results necessary for the further development.

2.1 Operator Theory

We first recall some basic notions in operator theory (see, for example, Lang, 1993). In the following we let $A : \mathcal{H} \to \mathcal{H}$ be a (linear) bounded operator, where $\mathcal{H}$ is a complex (separable) Hilbert space with scalar product[1] (norm) $\langle \cdot, \cdot \rangle$ ($\|\cdot\|$) and $(e_j)_{j\geq 1}$ a Hilbert basis in $\mathcal{H}$. We often use the notation $j \geq 1$ to denote a sequence or a sum from 1 to p where p can be infinite. The set of bounded operators on $\mathcal{H}$ is a Banach space with respect to the operator norm $\|A\|_{\mathcal{H},\mathcal{H}} = \|A\| = \sup_{\|f\|=1} \|Af\|$. If A is a bounded operator, we let A^* be its adjoint, which is a bounded operator with $\|A^*\| = \|A\|$.
A bounded operator A is Hilbert-Schmidt if $\sum_{j\geq 1} \|Ae_j\|^2 < \infty$ for some (any) Hilbert basis $(e_j)_{j\geq 1}$. The space of Hilbert-Schmidt operators is also a Hilbert space (a fact which will be a key in our development) endowed with the scalar product $\langle A, B \rangle_{HS} = \sum_j \langle Ae_j, Be_j \rangle$ and we denote by $\|\cdot\|_{HS}$ the corresponding norm. In particular, Hilbert-Schmidt operators are compact.

A closely related notion is that of a *trace class* operator. We say that a bounded operator A is trace class, if $\sum_{j\geq 1} \left\langle \sqrt{A^*A} e_j, e_j \right\rangle < \infty$ for some (any) Hilbert basis $(e_j)_{j\geq 1}$ (where $\sqrt{A^*A}$ is the square root of the positive operator A^*A defined by spectral theorem (see, for example, Lang, 1993). In particular, $\mathrm{Tr}(A) = \sum_{j\geq 1} \langle Ae_j, e_j \rangle < \infty$ and $\mathrm{Tr}(A)$ is called the trace of A. The space of trace class operators is a Banach space endowed with the norm $\|A\|_{TC} = \mathrm{Tr}(\sqrt{A^*A})$. Trace class operators are also Hilbert Schmidt (hence compact). The following inequalities relate the different operator norms:

$$\|A\| \leq \|A\|_{HS} \leq \|A\|_{TC}.$$

It can also be shown that for any Hilbert-Schmidt operator A and bounded operator B we have

$$\begin{aligned} \|AB\|_{HS} &\leq \|A\|_{HS}\|B\|, \\ \|BA\|_{HS} &\leq \|B\|\|A\|_{HS}. \end{aligned} \tag{1}$$

Remark 1 *If the context is clear we will simply denote the norm and the scalar product by $\|\cdot\|$ and $\langle \cdot, \cdot \rangle$ respectively. However, we will add a subscript when comparing norms in different spaces. When A is a bounded operator, $\|A\|$ always denotes the operator norm.*

2.2 Spectral Theory for Compact Operators

Recall that the spectrum of a matrix K can be defined as the set of eigenvalues $\lambda \in \mathbb{C}$, s.t. $\det(K - \lambda I) = 0$, or, equivalently, such that $\lambda I - K$ does not have a (bounded) inverse. This definition can be

1. We choose the convention for which the scalar product is linear in the first argument.

generalized to operators. Let $A : \mathcal{H} \to \mathcal{H}$ be a bounded operator, we say that $\lambda \in \mathbb{C}$ belongs to the spectrum $\sigma(A)$, if $(A - \lambda I)$ does not have a bounded inverse. For any $\lambda \notin \sigma(A)$, $R(\lambda) = (A - \lambda I)^{-1}$ is the *resolvent operator*, which is by definition a bounded operator. If A is a compact operator, then $\sigma(A) \setminus \{0\}$ consists of a countable family of isolated points with finite multiplicity $|\lambda_1| \geq |\lambda_2| \geq \cdots$ and either $\sigma(A)$ is finite or $\lim_{n\to\infty} \lambda_n = 0$ (see, for example, Lang, 1993).

If the bounded operator A is self-adjoint ($A = A^*$, analogous to a hermitian matrix in the finite-dimensional case), the eigenvalues are real. Each eigenvalue λ has an associated *eigenspace* which is the closed subspace of all eigenvectors with eigenvalue λ. A key result, known as the *Spectral Theorem*, ensures that

$$A = \sum_{i=1}^{\infty} \lambda_i P_{\lambda_i},$$

where P_λ is the *orthogonal projection operator* onto the eigenspace associated with λ. Moreover, it can be shown that the projection P_λ can be written explicitly in terms of the resolvent operator. Specifically, we have the following remarkable equality:

$$P_\lambda = \frac{1}{2\pi i} \int_\Gamma (\gamma I - A)^{-1} d\gamma,$$

where the integral can be taken over any closed simple rectifiable curve $\Gamma \subset \mathbb{C}$ (with positive direction) containing λ and no other eigenvalue. We note that while an integral of an operator-valued function may seem unfamiliar, it is defined along the same lines as an integral of an ordinary real-valued function. Despite the initial technicality, the above equation allows for far simpler analysis of eigenprojections than other seemingly more direct methods.

This analysis can be extended to operators, which are not self-adjoint, to obtain a decomposition parallel to the Jordan canonical form for matrices. To avoid overloading this section, we postpone the precise technical statements for that case to the Appendix B.

Remark 2 *Though in manifold and spectral learning we typically work with real valued functions, in this paper we will consider complex vector spaces to be able to use certain results from the spectral theory of (possibly non self-adjoint) operators. However, if the reproducing kernel and the weight function are both real valued, as usually is the case in machine learning, we will show that all functions of interest are real valued as well.*

2.3 Reproducing Kernel Hilbert Space (RKHS)

Let X be a subset of $\mathbb{R}^d$. A *Reproducing Kernel Hilbert space* is a Hilbert space $\mathcal{H}$ of functions $f : X \to \mathbb{C}$, such that all the evaluation functionals are bounded, that is

$$f(x) \leq C_x \|f\| \qquad \text{for some constant } C_x.$$

It can be shown that there is a unique conjugate symmetric positive definite kernel function $K : X \times X \to \mathbb{C}$, called *reproducing kernel*, associated with $\mathcal{H}$ and the following reproducing property holds

$$f(x) = \langle f, K_x \rangle,$$

where $K_x := K(\cdot, x)$. It is also well known (Aronszajn, 1950) that any conjugate symmetric positive definite kernel K uniquely defines a reproducing kernel Hilbert space whose reproducing kernel is

K. We will assume that the kernel is continuous and bounded, and we set

$$\kappa = \sup_{x \in X} K(x,x).$$

As a consequence, the elements of $\mathcal{H}$ are bounded continuous functions, the space $\mathcal{H}$ is separable and is compactly embedded in $\mathcal{C}(X)$ with the compact-open topology (Aronszajn, 1950).

Remark 3 *The set X can be taken to be any locally compact separable metric space and the assumption about continuity can be weakened. However, the above setting will simplify some technical considerations, in particular in Section* 4.2 *where Sobolev spaces are considered.*

2.4 Concentration Inequalities in Hilbert spaces

We recall that if $\xi_1, \ldots, \xi_n$ are independent (real-valued) random variables with zero mean and such that $|\xi_i| \leq C, \quad i = 1, \ldots, n$, then Hoeffding inequality ensures that $\forall \varepsilon > 0$,

$$\mathrm{P}\left[\left|\frac{1}{n}\sum_{i=1}^{n} \xi_i\right| \geq \varepsilon\right] \leq 2e^{-\frac{n\varepsilon^2}{2C^2}}.$$

If we set $\tau = \frac{n\varepsilon^2}{2C^2}$ then we can express the above inequality saying that with probability at least (with confidence) $1 - 2e^{-\tau}$,

$$\left|\frac{1}{n}\sum_{i=1}^{n} \xi_i\right| \leq \frac{C\sqrt{2\tau}}{\sqrt{n}}. \tag{2}$$

Similarly if $\xi_1, \ldots, \xi_n$ are zero mean independent random variables with values in a separable complex Hilbert space and such that $\|\xi_i\| \leq C, \quad i = 1, \ldots, n$, then the same inequality holds with the absolute value replaced by the norm in the Hilbert space, that is, the following bound

$$\left\|\frac{1}{n}\sum_{i=1}^{n} \xi_i\right\| \leq \frac{C\sqrt{2\tau}}{\sqrt{n}} \tag{3}$$

holds true with probability at least $1 - 2e^{-\tau}$ (Pinelis, 1992).

Remark 4 *In the cited reference the concentration inequality* (3) *is stated for real Hilbert spaces. However, a complex Hilbert space $\mathcal{H}$can be viewed as a real vector space with the scalar product given by $\langle f, g\rangle_{\mathcal{H}_\mathbb{R}} = (\langle f, g\rangle_{\mathcal{H}} + \langle g, f\rangle_{\mathcal{H}})/2$, so that $\|f\|_{\mathcal{H}_\mathbb{R}} = \|f\|_{\mathcal{H}}$. This last equality implies that* (3) *holds also for complex Hilbert spaces.*

2.5 Perturbation Theory

First we recall some results on perturbation of eigenvalues and eigenfunctions. About eigenvalues, we need to recall the notion of *extended enumeration* of discrete eigenvalues. We adapt the definition of Kato (1987), which is given for an arbitrary self-adjoint operator, to the compact operators. Let $A : \mathcal{H} \to \mathcal{H}$ be a compact operator, an extended enumeration is a sequence of real numbers where every nonzero eigenvalue of A appears as many times as its multiplicity and the other values (if any) are zero. An enumeration is an extended numeration where any element of the sequence is an

isolated eigenvalue with finite multiplicity. If the sequence is infinite, this last condition is equivalent to the fact that any element is nonzero.

The following result due to Kato (1987) is an extension to infinite dimensional operators of an inequality due to Lidskii for finite rank operator.

Theorem 5 (Kato 1987) *Let $\mathcal{H}$ be a separable Hilbert space with A, B self-adjoint compact operators. Let $(\gamma_j)_{j\geq 1}$, be an enumeration of discrete eigenvalues of $B-A$, then there exist extended enumerations $(\beta_j)_{j\geq 1}$ and $(\alpha_j)_{j\geq 1}$ of discrete eigenvalues of B and A respectively such that,*

$$\sum_{j\geq 1} \phi(\beta_j - \alpha_j) \leq \sum_{j\geq 1} \phi(\gamma_j),$$

where ϕ is any nonnegative convex function with $\phi(0) = 0$.

By choosing $\phi(t) = |t|^p$, $p \geq 1$, the above inequality becomes

$$\sum_{j\geq 1} |\beta_j - \alpha_j|^p \leq \sum_{j\geq 1} |\gamma_j|^p.$$

Letting $p = 2$ and recalling that $\|B-A\|_{HS}^2 = \sum_{j\geq 1} |\gamma_j|^2$, it follows that

$$\sum_{j\geq 1} |\beta_j - \alpha_j|^2 \leq \|B-A\|_{HS}^2.$$

Moreover, since $\lim_{p\to\infty} (\sum_{j\geq 1} |\gamma_j|^p)^{1/p} = \sup_{j\geq 1} |\gamma_j| = \|B-A\|$, we have that

$$\sup_{j\geq 1} |\beta_j - \alpha_j| \leq \|B-A\|.$$

Given an integer N, let m_N be the sum of the multiplicities of the first N nonzero top eigenvalues of A, it is possible to prove that the sequences $(\alpha_j)_{j\geq 1}$ and $(\beta_j)_{j\geq 1}$ in the above proposition can be chosen in such a way that

$$\begin{aligned} &\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_{m_N} > \alpha_j && j > m_N, \\ &\beta_1 \geq \beta_2 \geq \ldots \geq \beta_{m_N} \geq \beta_j && j > m_N. \end{aligned}$$

However in general we need to consider extended enumerations, which are not necessarily decreasing sequence, in order to take into account the kernel spaces of A and B, which are potentially infinite dimensional vector spaces (also see the remark after Theorem II in Kato 1987).

To control the spectral projections associated with one or more eigenvalues we need the following perturbation result due to Zwald and Blanchard (2006) (see also Theorem 20 in Section 4.3). Let A be a positive compact operator such that the number of eigenvalues is infinite. Given $N \in \mathbb{N}$, let P_N^A be the orthogonal projection on the eigenvectors corresponding to the top N distinct eigenvalues $\alpha_1 > \ldots > \alpha_N$ and α_{N+1} the next one.

Proposition 6 *Let A be a compact positive operator. Given an integer N, if B is another compact positive operator such that $\|A-B\| \leq \frac{\alpha_N - \alpha_{N+1}}{4}$, then*

$$\|P_D^B - P_N^A\| \leq \frac{2}{\alpha_N - \alpha_{N+1}} \|A-B\|$$

where the integer D is such that the dimension of the range of P_D^B is equal to the dimension of the range of P_N^A. If A and B are Hilbert-Schmidt, in the above bound the operator norm can be replaced by the Hilbert-Schmidt norm.

We note that a bound on the projections associated with simple eigenvalues implies that the corresponding eigenvectors are close since, if u and v are taken to be normalized and such that $\langle u,v\rangle > 0$, then the following inequality holds

$$\|P_u - P_v\|_{HS}^2 = 2(1-\langle u,v\rangle^2) \geq 2(1-\langle u,v\rangle) = \|u-v\|_{\mathcal{H}}^2.$$

3. Integral Operators Defined by a Reproducing Kernel

Let X be a subset of $\mathbb{R}^d$ and $K : X \times X \to \mathbb{C}$ be a reproducing kernel satisfying the assumptions stated in Section 2.3. Let ρ be a probability measure on X and denote by $L^2(X,\rho)$ the space of square integrable (complex) functions with norm $\|f\|_\rho^2 = \langle f,f\rangle_\rho = \int_X |f(x)|^2 d\rho(x)$. Since $K(x,x) \leq \kappa$ by assumption, the corresponding integral operator $L_K : L^2(X,\rho) \to L^2(X,\rho)$

$$(L_K f)(x) = \int_X K(x,s) f(s) d\rho(s)$$

is a bounded operator.

Suppose we are now given a set of points $\mathbf{x} = (x_1,\ldots,x_n)$ sampled i.i.d. according to ρ. Many problems in statistical data analysis and machine learning deal with the empirical kernel $n \times n$-matrix $\mathbf{K}$ given by $\mathbf{K}_{ij} = \frac{1}{n} K(x_i,x_j)$. The question we want to discuss is to which extent we can use the kernel matrix $\mathbf{K}$ (and the corresponding eigenvalues, eigenvectors) to estimate L_K (and the corresponding eigenvalues, eigenfunctions). Answering this question is important as it guarantees that the computable empirical proxy is sufficiently close to the ideal infinite sample limit.
The first difficulty in relating L_K and $\mathbf{K}$ is that they operate on different spaces. By default, L_K is an operator on $L^2(X,\rho)$, while $\mathbf{K}$ is a finite dimensional matrix. To overcome this difficulty we let $\mathcal{H}$ be the RKHS associated with K and define the operators $T_{\mathcal{H}}, T_n : \mathcal{H} \to \mathcal{H}$ given by

$$T_{\mathcal{H}} = \int_X \langle \cdot, K_x\rangle K_x d\rho(x), \tag{4}$$

$$T_n = \frac{1}{n}\sum_{i=1}^n \langle \cdot, K_{x_i}\rangle K_{x_i}. \tag{5}$$

Note that $T_{\mathcal{H}}$ is the integral operator with kernel K with range and domain $\mathcal{H}$ rather than in $L^2(X,\rho)$. The reason for writing it in this seemingly complicated form is to make the parallel with (5) clear. To justify the "extension operator" in (5), consider the natural "restriction operator",[2] $R_n : \mathcal{H} \to \mathbb{C}^n$, $R_n f = (f(x_1),\ldots,f(x_n))$. It is not hard to check that the adjoint operator $R_n^* : \mathbb{C}^n \to \mathcal{H}$ can be written as $R_n^*(y_1,\ldots,y_n) = \frac{1}{n}\sum_{i=1}^n y_i K_{x_i}$. Indeed, we see that

$$\begin{aligned}\langle R_n^*(y_1,\ldots,y_n), f\rangle_{\mathcal{H}} &= \langle (y_1,\ldots,y_n), R_n f\rangle_{\mathbb{C}^n} \\ &= \frac{1}{n}\sum_{i=1}^n y_i \overline{f(x_i)} = \frac{1}{n}\sum_{i=1}^n y_i \langle K_{x_i}, f\rangle_{\mathcal{H}},\end{aligned}$$

2. R_n is also called sampling or evaluation operator. We prefer to call it the *restriction operator* since $R_n f$ is the restriction of the function $f : X \to \mathbb{R}$ to the set of points $\{x_1,\ldots,x_n\}$.

where $\mathbb{C}^n$ is endowed with $1/n$ times the canonical scalar product. Thus, we observe that $T_n = R_n^* R_n$ is the composition of the restriction operator and its adjoint. On the other hand for the matrix $\mathbf{K}$ we have that $\mathbf{K} = R_n R_n^*$, regarded as operator on $\mathbb{C}^n$. Similarly, if $R_{\mathcal{H}}$ denotes the inclusion $\mathcal{H} \hookrightarrow L^2(X,\rho)$, $T_{\mathcal{H}} = R_{\mathcal{H}}^* R_{\mathcal{H}}$ and $L_K = R_{\mathcal{H}} R_{\mathcal{H}}^*$.

In the next subsection, we discuss a parallel with the Singular Value Decomposition for matrices and show that $T_{\mathcal{H}}$ and L_K have the same eigenvalues (possibly, up to some zero eigenvalues) and the corresponding eigenfunctions are closely related. A similar relation holds for T_n and $\mathbf{K}$. Thus to establish a connection between the spectral properties of $\mathbf{K}$ and L_K, it is sufficient to bound the difference $T_{\mathcal{H}} - T_n$, which is done in the following theorem (De Vito et al., 2005b). While the proof can be found in De Vito et al. (2005b), we provide it for completeness and to emphasize its simplicity.

Theorem 7 *The operators $T_{\mathcal{H}}$ and T_n are Hilbert-Schmidt. Under the above assumption with confidence $1-2e^{-\tau}$*

$$\|T_{\mathcal{H}} - T_n\|_{HS} \leq \frac{2\sqrt{2}\kappa\sqrt{\tau}}{\sqrt{n}}.$$

Proof We introduce a sequence $(\xi_i)_{i=1}^n$ of random variables in the Hilbert space of Hilbert-Schmidt operators by

$$\xi_i = \langle \cdot, K_{x_i} \rangle K_{x_i} - T_{\mathcal{H}}.$$

From (4) follows that $E(\xi_i) = 0$. By a direct computation we have that $\|\langle \cdot, K_x \rangle K_x\|_{HS}^2 = \|K_x\|^4 \leq \kappa^2$. Hence, using (4), $\|T_{\mathcal{H}}\|_{HS} \leq \kappa$ and

$$\|\xi_i\|_{HS} \leq 2\kappa, \qquad i = 1, \ldots, n.$$

From inequality (3) we have with probability $1-2e^{-\tau}$

$$\|\frac{1}{n}\sum_{i=1}^n \xi_i\|_{HS} = \|T_{\mathcal{H}} - T_n\|_{HS} \leq \frac{2\sqrt{2}\kappa\sqrt{\tau}}{\sqrt{n}},$$

which establishes the result. ■

As a direct consequence of Theorem 7 we obtain several concentration inequalities for eigenvalues and eigenfunctions. These results will be discussed in subsection 3.2 and they are based on an interpretation of the Nyström extension in terms of Singular Value Decomposition of the empirical operator and its mean, as explained in the following subsection.

3.1 Extension Operators

We will now briefly revisit the Nyström extension and clarify some connections to the Singular Value Decomposition (SVD) for operators. Recall that applying SVD to a $p \times m$ matrix A produces a *singular system* consisting of singular (strictly positive) values $(\sigma_j)_{j=1}^k$ with k being the rank of A, vectors $(u_j)_{j=1}^m \in \mathbb{C}^m$ and $(v_j)_{j=1}^p \in \mathbb{C}^p$ such that they form orthonormal bases of $\mathbb{C}^m$ and $\mathbb{C}^p$ respectively, and

$$\begin{cases} A^* A u_j = \sigma_j u_j & j = 1, \ldots k \\ A^* A u_j = 0 & j = k+1, \ldots, m \\ A A^* v_j = \sigma_j v_j & j = 1, \ldots k \\ A A^* v_j = 0 & j = k+1, \ldots, p. \end{cases}$$

It is not hard to see that the matrix A can be written as $A = V\Sigma^{1/2}U^*$, where U and V are matrices obtained by "stacking" u's and v's in the columns, and Σ is a $p \times m$ matrix having the singular values σ_i on the first k-entries on the diagonal (and zero outside), so that

$$\begin{cases} Au_j = \sqrt{\sigma_j} v_j & j = 1, \dots k \\ Au_j = 0 & j = k+1, \dots, m \\ A^* v_j = \sqrt{\sigma_j} u_j & j = 1, \dots k \\ A^* v_j = 0 & j = k+1, \dots, p, \end{cases}$$

which is the formulation we will use in this paper. The same formalism applies more generally to operators and allows us to connect the spectral properties of L_K and $T_{\mathcal{H}}$ as well as the matrix $\mathbf{K}$ and the operator T_n. The basic idea is that each of these pairs (as shown in the previous subsection) corresponds to a singular system and thus share eigenvalues (up to some zero eigenvalues) and have eigenvectors related by a simple equation. Indeed the following result can be obtained considering the SVD decomposition associated with $R_{\mathcal{H}}$ (and Proposition 9 considering the SVD decomposition associated with R_n). The proof of the following proposition can be deduced from the results in De Vito et al. (2005b) and De Vito et al. (2006).[3]

Proposition 8 *The following facts hold true.*

1. *The operators L_K and $T_{\mathcal{H}}$ are positive, self-adjoint and trace class. In particular both $\sigma(L_K)$ and $\sigma(T_{\mathcal{H}})$ are contained in $[0, \kappa]$.*

2. *The spectra of L_K and $T_{\mathcal{H}}$ are the same, possibly up to the zero. If σ is a nonzero eigenvalue and u, v are the associated eigenfunctions of L_K and $T_{\mathcal{H}}$ (normalized to norm 1 in $L^2(X, \rho)$ and $\mathcal{H}$) respectively, then*

$$\begin{aligned} u(x) &= \frac{1}{\sqrt{\sigma_j}} v(x) \qquad \text{for } \rho\text{-almost all } x \in X, \\ v(x) &= \frac{1}{\sqrt{\sigma_j}} \int_X K(x,s) u(s) d\rho(s) \qquad \text{for all } x \in X. \end{aligned}$$

3. *The following decompositions hold:*

$$\begin{aligned} L_K g &= \sum_{j \geq 1} \sigma_j \langle g, u_j \rangle_\rho \, u_j \qquad g \in L^2(X, \rho), \\ T_{\mathcal{H}} f &= \sum_{j \geq 1} \sigma_j \langle f, v_j \rangle \, v_j \qquad f \in \mathcal{H}, \end{aligned}$$

where the eigenfunctions $(u_j)_{j \geq 1}$ of L_K form an orthonormal basis of $\ker L_K{}^{\perp}$ and the eigenfunctions $(v_j)_{j \geq 1}$ of $T_{\mathcal{H}}$ form an orthonormal basis for $\ker(T_{\mathcal{H}})^{\perp}$.

If K is real-valued, both the families $(u_j)_{j \geq 1}$ and $(v_j)_{j \geq 1}$ can be chosen as real valued functions.

3. In De Vito et al. (2005b) and De Vito et al. (2006) the results are stated for real kernels, however the proof does not change if K is complex valued. Moreover, if K is real and L_K is regarded as integral operator on the space of square integrable complex functions, one can easily check that the eigenvalues are positive and, if u is an eigenfunction with eigenvalue $\sigma \geq 0$, then the complex conjugate $\overline{u}$ is also an eigenfunction with the same eigenvalue, so that it is always possible to choose all the eigenfunctions to be real valued.

Note that the RKHS $\mathcal{H}$ does not depend on the measure ρ. If the support of the measure ρ is only a subset of X (e.g., a finite set of points or a submanifold), then functions in $L^2(X,\rho)$ are only defined on the support of ρ whereas functions in $\mathcal{H}$ are defined on the whole space is X. The eigenfunctions of L_K and $T_{\mathcal{H}}$ coincide (up-to a scaling factor) on the support of the measure, and v is an *extension*[4] of u outside of the support of ρ. Moreover, the extension/restriction operations preserve both the normalization and orthogonality of the eigenfunctions. In a slightly different context Coifman and Lafon (2006) showed the connection between the Nyström method and the set of eigenfunctions of L_K, which are called *geometric harmonics*. The main difference between our result and the cited paper is that we consider all the eigenfunctions, whereas Coifman and Lafon introduce a threshold on the spectrum to ensure stability since they do not consider a sampling procedure.

An analogous result relates the matrix $\mathbf{K}$ and the operator T_n .

Proposition 9 *The following facts hold.*

1. *The operator T_n is of finite rank, self-adjoint and positive, whereas the matrix $\mathbf{K}$ is conjugate symmetric and semi-positive definite. In particular the spectrum $\sigma(T_n)$ has only finitely many nonzero elements and is contained in $[0,\kappa]$.*

2. *The spectra of $\mathbf{K}$ and T_n are the same up to the zero, that is, $\sigma(\mathbf{K})\setminus\{0\} = \sigma(T_n)\setminus\{0\}$. Moreover, if $\hat{\sigma}$ is a nonzero eigenvalue and $\hat{u},\hat{v}$ are the corresponding eigenvector and eigenfunction of $\mathbf{K}$ and T_n (normalized to norm 1 in $\mathbb{C}^n$ and $\mathcal{H}$) respectively, then*

$$\begin{aligned} \hat{u} &= \frac{1}{\sqrt{\hat{\sigma}}}(\hat{v}(x_1),\ldots,\hat{v}(x_n)), \\ \hat{v} &= \frac{1}{\sqrt{\hat{\sigma}}}\left(\frac{1}{n}\sum_{i=1}^{n} K_{x_i}\hat{u}^i\right), \end{aligned}$$

   where $\hat{u}^i$ is the $i-$th component of the eigenvector $\hat{u}$.

3. *The following decompositions hold:*

$$\begin{aligned} \mathbf{K}w &= \sum_{j=1}^{k} \hat{\sigma}_j \langle w,\hat{u}_j\rangle \hat{u}_j \qquad w\in\mathbb{C}^n, \\ T_n f &= \sum_{j=1}^{k} \hat{\sigma}_j \langle f,\hat{v}_j\rangle_{\mathcal{H}} \hat{v}_j \qquad f\in\mathcal{H}, \end{aligned}$$

   where k is the rank of K and both sums run over the nonzero eigenvalues, the family $(\hat{u}_j)_{j\geq 1}$ is an orthonormal basis for $\ker\{\mathbf{K}\}^{\perp}\subset\mathbb{C}^n$ and the family $(\hat{v}_j)_{j\geq 1}$ of T_n forms an orthonormal basis for the space $\ker(T_n)^{\perp}\subset\mathcal{H}$, where

$$\ker(T_n) = \{f\in\mathcal{H} \mid f(x_i)=0\ \forall i=1,\ldots,n\}.$$

If K is real-valued, both the families $(\hat{u}_j)_{j\geq 1}$ and $(\hat{v}_j)_{j\geq 1}$ can be chosen as real valued vectors and functions, respectively.

4. However, the extension is *trivial* in the points $x\in X$ where $K(x,x)=0$, as it happens if the kernel is compactly supported.

Note that in this section L_K, T and T_n are self-adjoint operators and $\mathbf{K}$ is a conjugate symmetric matrix. If K is real, we can directly work with real Hilbert spaces. However since we need complex vector spaces in Section 4 for consistency we stated the above results for complex reproducing kernels.

3.2 Bounds on Eigenvalues and Spectral Projections

Using Theorem 7, we are able to bound the ℓ_2-distance between the spectrum of L_K and the spectrum of $\mathbf{K}$.

Proposition 10 *There exist an extended enumeration $(\sigma_j)_{j\geq 1}$ of discrete eigenvalues for L_K and an extended enumeration $(\hat{\sigma}_j)_{j\geq 1}$ of discrete eigenvalues for $\mathbf{K}$ such that*

$$\sum_{j\geq 1}(\sigma_j-\hat{\sigma}_j)^2 \leq \frac{8\kappa^2\tau}{n},$$

with confidence greater than $1-2e^{-\tau}$. In particular $\sup_{j\geq 1}|\sigma_j-\hat{\sigma}_j| \leq \frac{2\sqrt{2}\kappa\sqrt{\tau}}{\sqrt{n}}$.

Proof By Proposition 8, an extended enumeration $(\sigma_j)_{j\geq 1}$ of discrete eigenvalues for L_K is also an extended enumeration $(\sigma_j)_{j\geq 1}$ of discrete eigenvalues for $T_{\mathcal{H}}$, and a similar relation holds for $\mathbf{K}$ and T_n by Proposition 9. Theorem 5 with $A=T_n$ and $B=T_{\mathcal{H}}$ gives that

$$\sum_{j\geq 1}(\sigma_j-\hat{\sigma}_j)^2 \leq \|T_{\mathcal{H}}-T_n\|_{HS}^2$$

for a suitable extended enumerations $(\sigma_j)_{j\geq 1}$, $(\hat{\sigma}_j)_{j\geq 1}$ of discrete eigenvalues for T and T_n, respectively. Theorem 7 provides us with the claimed bound. ■

Theorem 4.2 and the following corollaries of Koltchinskii and Giné (2000) provide the same convergence rate (in expectation) under a different setting (the kernel K is only symmetric, but with some assumption on the decay of the eigenvalues of L_K).

The following result can be deduced by Theorem 5 with $p=1$ and Theorem 7, however a simpler direct proof is given below.

Proposition 11 *Under the assumption of Proposition 10 with confidence $1-2e^{-\tau}$*

$$|\sum_{j\geq 1}(\sigma_j-\hat{\sigma}_j)| = |\operatorname{Tr}(T_{\mathcal{H}})-\operatorname{Tr}(T_n)| \leq \frac{2\sqrt{2}\kappa\sqrt{\tau}}{\sqrt{n}}.$$

Proof Note that

$$\operatorname{Tr}(T_n)=\frac{1}{n}\sum_{i=1}^n K(x_i,x_i), \qquad \text{and} \qquad \operatorname{Tr}(T_{\mathcal{H}})=\int_X K(x,x)d\rho(x).$$

Then we can define a sequence $(\xi_i)_{i=1}{}^n$ of real-valued random variables by $\xi_i=K(x_i,x_i)-\operatorname{Tr}(T_{\mathcal{H}})$. Clearly $\mathbb{E}[\xi_i]=0$ and $|\xi_i|\leq 2\kappa$, $i=1,\ldots,n$ so that Hoeffding inequality (2) yields with confidence $1-2e^{-\tau}$

$$\left|\frac{1}{n}\sum_{i=1}^n \xi_i\right| = |\operatorname{Tr}(T_{\mathcal{H}})-\operatorname{Tr}(T_n)| \leq \frac{2\sqrt{2}\kappa\sqrt{\tau}}{\sqrt{n}}.$$

■

From Proposition 6 and Theorem 7 we directly derive the probabilistic bound on the eigenprojections given by Zwald and Blanchard (2006)—their proof is based on bounded difference inequality for real random variables—see also De Vito et al. (2005a). For the sake of simplicity, in the following result we assume that the number of eigenvalues of L_K is infinite.

Theorem 12 *Given an integer N, let m be the sum of the multiplicities of the first N distinct eigenvalues of L_K, so that*

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m > \sigma_{m+1},$$

and P_N be the orthogonal projection from $L^2(X,\rho)$ onto the span of the corresponding eigenfunctions. Let k be the rank of $\mathbf{K}$, and $\hat{u}_1 \ldots, \hat{u}_k$ the eigenvectors corresponding to the nonzero eigenvalues of $\mathbf{K}$ in a decreasing order. Denote by $\hat{v}_1 \ldots, \hat{v}_k \in \mathcal{H} \subset L^2(X,\rho)$ the corresponding Nyström extension given by item 2 of Proposition 9.

Given $\tau > 0$, if the number n of examples satisfies

$$n > \frac{128\,\kappa^2\tau}{(\sigma_m - \sigma_{m+1})^2},$$

then

$$\sum_{j=1}^{m} \|(I - P_N)\hat{v}_j\|_\rho^2 + \sum_{j=m+1}^{k} \|P_N \hat{v}_j\|_\rho^2 \leq \frac{32\kappa^2\tau}{(\sigma_m - \sigma_{m+1})^2\, n}\,, \tag{6}$$

with probability greater than $1 - 2e^{-\tau}$.

Proof Let $(u_j)_{j\geq 1}$ be an orthonormal family of eigenfunctions of L_K with strictly positive eigenvalues. Without loss of generality, we can assume that $u_1, \ldots, u_m$ are the eigenfunctions with eigenvalues $\sigma_1, \sigma_2, \ldots, \sigma_m$. Let $(v_j)_{j\geq 1}$ the corresponding family of eigenfunctions of $T_{\mathcal{H}}$ given by Proposition 8 and complete to an orthonormal basis of $\mathcal{H}$. Complete also the family $\hat{v}_1 \ldots, \hat{v}_k$ to an other orthonormal basis of $\mathcal{H}$.

Apply Proposition 6 with $A = T_{\mathcal{H}}$ and $B = T_n$ by taking into account Theorem 7. With high probability

$$\|P^{T_n} - P^{T_{\mathcal{H}}}\|_{\mathrm{HS}}^2 \leq \frac{8\kappa^2\tau}{(\sigma_m - \sigma_{m+1})^2\, n} \leq \frac{a_m - a_{m+1}}{2},$$

where

$$P^{T_{\mathcal{H}}} = \sum_{j=1}^{m} \langle f, v_j \rangle_{\mathcal{H}} v_j \qquad P^{T_n} = \sum_{j=1}^{m} \langle f, \hat{v}_j \rangle_{\mathcal{H}} \hat{v}_j$$

and the last bound follows from the condition on n. In particular, $\hat{\sigma}_m > \hat{\sigma}_{m+1}$.

Since both $(v_i)_{j\geq 1}$ and $(\hat{v}_i)_{j\geq 1}$ are orthonormal bases for $\mathcal{H}$

$$\begin{aligned}
\|P^{T_n} - P^{T_{\mathcal{H}}}\|_{\mathrm{HS}}^2 &= \sum_{i,j\geq 1} |\langle P^{T_n} v_i - P^{T_{\mathcal{H}}} v_i, \hat{v}_j \rangle_{\mathcal{H}}|^2 \\
&= \sum_{j=1}^{m} \sum_{i\geq m+1} |\langle v_i, \hat{v}_j \rangle_{\mathcal{H}}|^2 + \sum_{j\geq m+1} \sum_{i=1}^{m} |\langle v_i, \hat{v}_j \rangle_{\mathcal{H}}|^2 \\
&\geq \sum_{j=1}^{m} \sum_{\substack{i\geq m+1 \\ T_{\mathcal{H}} v_i \neq 0}} |\langle v_i, \hat{v}_j \rangle_{\mathcal{H}}|^2 + \sum_{j\geq m+1}^{k} \sum_{i=1}^{m} |\langle v_i, \hat{v}_j \rangle_{\mathcal{H}}|^2.
\end{aligned}$$

Since the sum on i is with respect to the eigenfunctions of $T_{\mathcal{H}}$ with nonzero eigenvalue, the Mercer theorem implies that $\langle v_i, \hat{v}_j \rangle_{\mathcal{H}} = \langle u_i, \hat{v}_j \rangle_\rho$. Finally, observe that

$$\sum_{i=1}^{m} |\langle u_i, \hat{v}_j \rangle_\rho|^2 = \|P_N \hat{v}_j\|_\rho^2$$
$$\sum_{\substack{i \geq m+1 \\ T_{\mathcal{H}} v_i \neq 0}} |\langle u_i, \hat{v}_j \rangle_\rho|^2 = \sum_{\substack{i \geq m+1 \\ L_K u_i \neq 0}} |\langle u_i, \hat{v}_j \rangle_\rho|^2 = \|(I - P_N)\hat{v}_j\|_\rho^2$$

where we used that $\ker T_{\mathcal{H}} \subset \ker T_n$, so that $\hat{v}_j \in \ker L_K{}^\perp$ with probability 1. ■

The first term on the left side of inequality (6) is the projection of the vector space spanned by the Nyström extension of the first top eigenvectors of the empirical matrix $\mathbf{K}$ onto the orthogonal of the vector space $\mathcal{M}_N$ spanned by the first top eigenfunctions of the integral operator L_K, the second term is the projection of the vector space spanned by the Nyström extensions of the other eigenvectors of $\mathbf{K}$ onto $\mathcal{M}_N$. Both differences are in $L^2(X, \rho)$ norm. A similar result is given in Zwald and Blanchard (2006), however, the role of the Nyström extensions is not considered—they study only the operators $T_{\mathcal{H}}$ and T_n (with our notation). Another result similar to ours is independently given in Smale and Zhou (2009), where the authors considered a single eigenfunction with multiplicity 1.

4. Asymmetric Graph Laplacian

In this section we will consider the case of the so-called asymmetric normalized graph Laplacian, which is the identity matrix minus the transition matrix for the natural random walk on a graph. In such a random walk, the probability of leaving a vertex along a given edge is proportional to the weight of that edge. As before, we will be interested in a specific class of graphs (matrices) associated with data.

Let $W : X \times X \to \mathbb{R}$ be a symmetric continuous weight function such that

$$0 < c \leq W(x,s) \leq C \qquad x, s \in X. \tag{7}$$

Note that we will not require W to be positive definite, but positive.

A set of data points $\mathbf{x} = (x_1, \ldots, x_n) \in X$ defines a weighted undirected graph with the weight matrix $\mathbf{W}$ given by $\mathbf{W}_{ij} = \frac{1}{n} W(x_i, x_j)$. The (asymmetric) normalized graph Laplacian $\mathbf{L} : \mathbb{C}^n \to \mathbb{C}^n$ is an $n \times n$ matrix given by

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W},$$

where the *degree* matrix $\mathbf{D}$ is diagonal with

$$\mathbf{D}_{ii} = \frac{1}{n} \sum_{j=1}^{n} W(x_i, x_j) = \sum_{j=1}^{n} \mathbf{W}_{ij},$$

which is invertible since $\mathbf{D} \geq cI$ by (7).

As in the previous section, X is a subset of $\mathbb{R}^d$ endowed with a probability measure ρ and $L^2(X, \rho)$ is the space of square integrable complex functions with respect to ρ.

Let $m(x) = \int_X W(x,s) d\rho(s)$ be the *degree function*, bound (7) implies that the operator $\mathbb{L} : L^2(X, \rho) \to L^2(X, \rho)$

$$(\mathbb{L}f)(x) = f(x) - \int_X \frac{W(x,s) f(s)}{m(x)} d\rho(s),$$

is well defined and continuous. The fact that W is bounded away from zero is essential to control the behavior of the degree function m, however it might be possible to replace this condition with the requirement that $m(x) \geq c$, to consider localized weight functions.

We see that when a set $\mathbf{x} = (x_1, \ldots, x_n) \in X$ is sampled i.i.d. according to ρ, the matrix $\mathbf{L}$ is an empirical version of the operator $\mathbb{L}$. We will view $\mathbf{L}$ as a perturbation of $\mathbb{L}$ due to finite sampling and will extend the approach developed in this paper to relate their spectral properties. Note that the methods described in the previous section are not directly applicable in this setting since W is not necessarily positive definite, so there is no RKHS associated with it. Moreover, even if W is positive definite, $\mathbb{L}$ involves division by a function, and may not be a map from the RKHS to itself. To overcome this difficulty in our theoretical analysis, we will rely on an auxiliary RKHS $\mathcal{H}$ with reproducing kernel K. Interestingly enough, this space will play no role from the algorithmic point of view, but only enters the theoretical analysis.

To state the properties of $\mathcal{H}$ we define the following functions

$$\begin{aligned} K_x &: X \to \mathbb{C} \qquad K_x(t) = K(t,x), \\ W_x &: X \to \mathbb{R} \qquad W_x(t) = W(t,x), \\ m_n &: X \to \mathbb{R} \qquad m_n = \frac{1}{n}\sum_{i=1}^{n} W_{x_i}, \end{aligned}$$

where m_n is the empirical counterpart of the function m and, in particular, $m_n(x_i) = \mathbf{D}_{ii}$.

To proceed we need the following assumption, which postulates that the functions $w_x, W_x/m, W_x/m_n$ belong to $\mathcal{H}$. However it is important to note that for W sufficiently smooth (as we expect it to be in most applications) these conditions can be satisfied by choosing $\mathcal{H}$ to be a Sobolev space of sufficiently high degree. This is made precise in the Section 4.2 (see Assumption 2).

Assumption 1 *Given a continuous weight function W satisfying* (7)*, we assume there exists a RKHS $\mathcal{H}$ with bounded continuous kernel K such that*

$$\begin{aligned} &W_x,\ \frac{1}{m}W_x,\ \frac{1}{m_n}W_x \in \mathcal{H} \\ &\|\frac{1}{m}W_x\|_{\mathcal{H}} \leq C, \end{aligned}$$

for all $x \in X$.

Assumption 1 allows to define the following bounded operators $\mathbb{L}_{\mathcal{H}}, A_{\mathcal{H}} : \mathcal{H} \to \mathcal{H}$

$$\begin{aligned} A_{\mathcal{H}} &= \int_X \langle \cdot, K_x \rangle_{\mathcal{H}} \frac{1}{m} W_x \, d\rho(x), \\ \mathbb{L}_{\mathcal{H}} &= I - A_{\mathcal{H}} \end{aligned}$$

and their empirical counterparts $\mathbb{L}_n, A_n : \mathcal{H} \to \mathcal{H}$

$$\begin{aligned} A_n &= \frac{1}{n}\sum_{i=1}^{n} \langle \cdot, K_{x_i} \rangle_{\mathcal{H}} \frac{1}{m_n} W_{x_i}, \\ \mathbb{L}_n &= I - A_n. \end{aligned}$$

Next result will show that $\mathbb{L}_{\mathcal{H}}$, $A_{\mathcal{H}}$ and $\mathbb{L}$ have related eigenvalues and eigenfunctions and that eigenvalues and eigenfunctions (eigenvectors) of $\mathbb{L}_n, A_n$ and $\mathbf{L}$ are also closely related. In particular

we will see in the following that to relate the spectral properties of $\mathbb{L}$ and $\mathbf{L}$ it suffices to control the deviation $A_{\mathcal{H}} - A_n$. However, before doing this, we make the above statements precise in the following subsection.

4.1 Extension Operators

In analogy to Section 3.1 we consider the relation between the operators we want to study and their extensions. We define the restriction operator $R_n : \mathcal{H} \to \mathbb{C}^n$ and the extension operator $E_n : \mathbb{C}^n \to \mathcal{H}$ as

$$\begin{aligned} R_n f &= (f(x_1), \ldots, f(x_n)) \qquad && f \in \mathcal{H}, \\ E_n(y_1, \ldots, y_n) &= \frac{1}{n} \sum_{i=1}^{n} y_i \frac{1}{m_n} W_{x_i} \qquad && (y_1, \ldots, y_n) \in \mathbb{C}^n. \end{aligned}$$

Clearly the extension operator is no longer the adjoint of R_n but the connection among the operators $\mathbf{L}$ to $\mathbb{L}_n$ and A_n can still be clarified by means of R_n and E_n. Indeed it is easy to check that $A_n = E_n R_n$ and $\mathbf{D}^{-1}\mathbf{W} = R_n E_n$. Similarly the infinite sample restrictions and extension operators can be defined to relate the operators $\mathbb{L}$, $A_{\mathcal{H}}$ and $\mathbb{L}_{\mathcal{H}}$. The next proposition considers such a connection.

Proposition 13 *The following facts hold true.*

1. *The operator $A_{\mathcal{H}}$ is Hilbert-Schmidt, the operators $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ are bounded and have positive eigenvalues.*

2. *Given $\sigma \in [0, +\infty[$, $\sigma \in \sigma(\mathbb{L}_{\mathcal{H}})$ if and only if $1 - \sigma \in \sigma(A_{\mathcal{H}})$, with the same eigenfunction.*

3. *The spectra of $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ are the same up to the eigenvalue 1. If $\sigma \neq 1$ is an eigenvalue and u, v associated eigenfunctions of $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ respectively, then*

$$\begin{aligned} u(x) &= v(x) \qquad \text{for almost all } x \in X, \\ v(x) &= \frac{1}{1-\sigma} \int_X \frac{W(x,t)}{m(x)} u(t)\, d\rho(t) \qquad \text{for all } x \in X. \end{aligned}$$

4. *Finally the following decompositions hold*

$$\mathbb{L} = \sum_{\substack{j \geq 1 \\ \sigma_j \neq 1}} \sigma_j P_j + P_0, \tag{8}$$

$$\mathbb{L}_{\mathcal{H}} = I - \sum_{\substack{j \geq 1 \\ \sigma_j \neq 1}} (1 - \sigma_j) Q_j + D, \tag{9}$$

where the projections Q_j, P_j are the spectral projections of $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ associated with the eigenvalue σ_j, P_0 is the spectral projection of $\mathbb{L}$ associated with the eigenvalue 1, and D is a quasi-nilpotent operator such that $\ker D = \ker(I - \mathbb{L}_{\mathcal{H}})$ and $Q_j D = D Q_j = 0$ for all $j \geq 1$.

Furthermore, all the eigenfunctions can be chosen as real-valued.

The proof of the above result is long and quite technical and can be found in Appendix A. Note that, with respect to Proposition 9, neither the normalization nor the orthogonality is preserved by the extension/restriction operations, so that we are free to normalize v with the factor $1/(1-\sigma)$, instead of $1/\sqrt{1-\sigma}$ as in Proposition 8. One can easily show that, if $u_1,\ldots,u_m$ is a linearly independent family of eigenfunctions of $\mathbb{L}$ with eigenvalues $\sigma_1,\ldots,\sigma_m \neq 1$, then the extension $v_1,\ldots,v_m$ is a linearly independent family of eigenfunctions of $\mathbb{L}_{\mathcal{H}}$ with eigenvalues $\sigma_1,\ldots,\sigma_m \neq 1$. Finally, we stress that in item 4 both series converge in the strong operator topology, however, though $\sum_{j\geq 1} P_i = I - P_0$, it is not true that $\sum_{j\geq 1} Q_i$ converges to $I - Q_0$, where Q_0 is the spectral projection of $\mathbb{L}_{\mathcal{H}}$ associated with the eigenvalue 1. This is the reason why we need to write the decomposition of $\mathbb{L}_{\mathcal{H}}$ as in (9) instead of (8). An analogous result allows us to relate $\mathbf{L}$ to $\mathbb{L}_n$ and A_n.

Proposition 14 *The following facts hold true.*

1. *The operator A_n is Hilbert-Schmidt, the matrix $\mathbf{L}$ and the operator $\mathbb{L}_n$ have positive eigenvalues.*

2. *Given $\sigma \in [0,+\infty[$, $\sigma \in \sigma(\mathbb{L}_n)$ if and only if $1-\sigma \in \sigma(A_n)$, with the same eigenfunction.*

3. *The spectra of $\mathbf{L}$ and $\mathbb{L}_n$ are the same up to the eigenvalue 1, moreover if $\hat{\sigma} \neq 1$ is an eigenvalue and the $\hat{u}, \hat{v}$ eigenvector and eigenfunction of $\mathbf{L}$ and $\mathbb{L}_n$, then*

$$\begin{aligned} \hat{u} &= (\hat{v}(x_1),\ldots,\hat{v}(x_1)), \\ \hat{v}(x) &= \frac{1}{1-\hat{\sigma}}\frac{1}{n}\sum_{i=1}^{n}\frac{W(x,x_i)}{m_n(x)}\hat{u}^i \end{aligned}$$

   where $\hat{u}^i$ is the $i-th$ component of the eigenvector $\hat{u}$.

4. *Finally the following decompositions hold*

$$\begin{aligned} \mathbf{L} &= \sum_{\substack{j\geq 1 \\ \hat{\sigma}_j \neq 1}} \hat{\sigma}_j \hat{P}_j + \hat{P}_0, \\ \mathbb{L}_n &= \sum_{\substack{j\geq 1 \\ \hat{\sigma}_j \neq 1}} \hat{\sigma}_j \hat{Q}_j + \hat{Q}_0 + \hat{D}, \end{aligned}$$

   where the projections Q_j, P_j are the spectral projections of $\mathbf{L}$ and $\mathbb{L}_n$ associated with the eigenvalue σ_j, $\hat{P}_0$ and $\hat{Q}_0$ are the spectral projections of $\mathbf{L}$ and $\mathbb{L}_n$ associated with the eigenvalue 1, and $\hat{D}$ is a quasi-nilpotent operator such that $\ker \hat{D} = \ker(I - \mathbb{L}_n)$ and $\hat{Q}_j \hat{D} = \hat{D}\hat{Q}_j = 0$ for all j with $\hat{\sigma}_j \neq 1$.

The last decomposition is parallel to the Jordan canonical form for (non-symmetric) matrices. Notice that, since the sum is finite, $\sum_{\substack{j\geq 1 \\ \hat{\sigma}_j \neq 1}} \hat{Q}_j + \hat{Q}_0 = I$.

4.2 Graph Laplacian Convergence for Smooth Weight Functions

If the weight function W is sufficiently differentiable, we can choose the RKHS $\mathcal{H}$ to be a suitable Sobolev space. For the sake of simplicity, we assume that X is a bounded open subset of $\mathbb{R}^d$ with a

nice boundary.[5] Given $s \in \mathbb{N}$, the Sobolev space $\mathcal{H}^s = \mathcal{H}^s(X)$ is

$$\mathcal{H}^s = \{f \in L^2(X,dx) \mid D^\alpha f \in L^2(X,dx)\ \forall |\alpha| = s\},$$

where $D^\alpha f$ is the (weak) derivative of f with respect to the multi-index $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}^d$, $|\alpha| = \alpha_1 + \cdots + \alpha_d$ and $L^2(X,dx)$ is the space of square integrable complex functions with respect to the Lebesgue measure (Burenkov, 1998). The space $\mathcal{H}^s$ is a separable Hilbert space with respect to the scalar product

$$\langle f, g \rangle_{\mathcal{H}^s} = \langle f, g \rangle_{L^2(X,dx)} + \sum_{|\alpha|=s} \langle D^\alpha f, D^\alpha g \rangle_{L^2(X,dx)}.$$

Let $C_b^s(X)$ be the set of continuous bounded functions such that all the (standard) derivatives of order s exists and are continuous bounded functions. The space $C_b^s(X)$ is a Banach space with respect to the norm

$$\|f\|_{C_b^s} = \sup_{x \in X} |f(x)| + \sum_{|\alpha|=s} \sup_{x \in X} |(D^\alpha f)(x)|.$$

Since X is bounded, it is clear that $C_b^s(X) \subset \mathcal{H}^s$ and $\|f\|_{\mathcal{H}^s} \leq d \|f\|_{C_b^s}$, where d is a suitable constant depending only on s. A sort of converse also holds, which will be crucial in our approach, see Corollary 21 of Burenkov (1998). Let $l, m \in \mathbb{N}$ such that $l - m > \frac{d}{2}$, then

$$\mathcal{H}^l \subset C_b^m(X) \qquad \|f\|_{C_b^m} \leq d' \|f\|_{\mathcal{H}^l} \tag{10}$$

where d' is a constant depending only on l and m.

From Eq (10) with $l = s$ and $m = 0$, we see that the Sobolev space $\mathcal{H}^s$, where $s = \lfloor d/2 \rfloor + 1$, is a RKHS with a continuous[6] real valued bounded kernel K^s.

We are ready to state our assumption on the weight function, which implies Assumption 1.

Assumption 2 *We assume that $W : X \times X \to \mathbb{R}$ is a positive, symmetric function such that*

$$W(x,t) \geq c > 0 \qquad \forall x, t \in X, \tag{11}$$

$$W \in C_b^{d+1}(X \times X). \tag{12}$$

As we will see, condition (12) quantifies the regularity of W we need to use Sobolev spaces as RKHS and, as usual, it critically depends on the dimension of the input space, see also Remark 19 below. By inspecting our proofs, (12) can be replaced by the more technical condition $W \in \mathcal{H}^{d+1}(X \times X)$.

As a consequence of Assumption 2, we are able to control the deviation of $\mathbb{L}_n$ from $\mathbb{L}_{\mathcal{H}}$.

Theorem 15 *Under the conditions of Assumption 2, with confidence $1 - 2e^{-\tau}$ we have*

$$\|\mathbb{L}_n - \mathbb{L}_{\mathcal{H}}\|_{HS} = \|A_{\mathcal{H}} - A_n\|_{HS} \leq C \frac{\sqrt{\tau}}{\sqrt{n}}, \tag{13}$$

where $\|\cdot\|_{HS}$ is the Hilbert-Schimdt norm of an operator in the Sobolev space $\mathcal{H}^s$ with $s = \lfloor d/2 \rfloor + 1$, and C is a suitable constant.

5. The conditions, like quasi-resolved boundary open set, are very technical and we refer to Burenkov (1998) for the precise assumptions, see Section 4.3 of the cited reference.

6. The kernel K^s is continuous on $X \times X$ since the embedding of $\mathcal{H}^s$ into $C_b(X)$ is compact, see Schwartz (1964).

To prove this result we need some preliminary lemmas. In the following C will be a constant that could change from one statement to the other. The first result shows that Assumption 2 implies Assumption 1 with $\mathcal{H} = \mathcal{H}^s$ and that the multiplicative operators defined by the degree function, or its empirical estimate, are bounded.

Lemma 16 *There exists a suitable constant $C > 0$ such that*

1. *for all $x \in X$, $W_x, \frac{1}{m}W_x, \frac{1}{m_n}W_x \in \mathcal{H}^{d+1} \subset \mathcal{H}^s$ and $\|\frac{1}{m}W_x\|_{\mathcal{H}^s} \leq C$;*
2. *the multiplicative operators $M, M_n : \mathcal{H}^s \to \mathcal{H}^s$ defined by*

$$Mf = mf, \qquad M_n f = m_n f, \qquad\qquad f \in \mathcal{H}^s$$

are bounded invertible operators satisfying

$$\|M\|_{\mathcal{H}^s,\mathcal{H}^s}, \|M^{-1}\|_{\mathcal{H}^s,\mathcal{H}^s}, \|M_n\|_{\mathcal{H}^s,\mathcal{H}^s}, \|M_n^{-1}\|_{\mathcal{H}^s,\mathcal{H}^s} \leq C,$$
$$\|M - M_n\|_{\mathcal{H}^s,\mathcal{H}^s} \leq C\|m - m_n\|_{\mathcal{H}^{d+1}},$$

where $\|\cdot\|_{\mathcal{H}^s,\mathcal{H}^s}$ is the operator norm of an operator in the Sobolev space $\mathcal{H}^s$.

Proof Let $C_1 = \|W\|_{C_b^{d+1}(X\times X)}$. Given $x \in X$, clearly $W_x \in C_b^{d+1}(X)$ and, by standard results of differential calculus, both m and $m_n \in C_b^{d+1}(X)$ with

$$\|W_x\|_{C_b^{d+1}(X)}, \|m\|_{C_b^{d+1}(X)}, \|m_n\|_{C_b^{d+1}(X)} \leq C_1.$$

Leibniz rule for the quotient with bound (11) gives that $\frac{1}{m}$ and $\frac{1}{m_n} \in C_b^{d+1}(X)$ with

$$\|\frac{1}{m}\|_{C_b^{d+1}(X)}, \|\frac{1}{m_n}\|_{C_b^{d+1}(X)} \leq C_2,$$

where C_2 is independent both on n and on the sample $(x_1, \ldots, x_n)$. Claim in item 1 is a consequence of the fact that pointwise multiplication is a continuous bilinear map on $C_b^{d+1}(X)$, and $C_b^{d+1}(X) \subset \mathcal{H}^{d+1} \subset \mathcal{H}^s$ with

$$\|f\|_{\mathcal{H}^s} \leq C_3\|f\|_{\mathcal{H}^{d+1}} \leq C_4\|f\|_{C_b^{d+1}(X)}.$$

We claim that if $g \in C_b^{d+1}(X)$ and $f \in \mathcal{H}^s$, then $gf \in \mathcal{H}^s$ and

$$\|gf\|_{\mathcal{H}^s} \leq \|g\|_{\mathcal{H}^{d+1}}\|f\|_{\mathcal{H}^s}.$$

Indeed, Lemma 15 of Section 4 of Burenkov (1998) with $p = p_2 = 2$, $p_1 = \infty$, $l = s$ and $n = d$ ensures that

$$\|gf\|_{\mathcal{H}^s} \leq \|g\|_{C_b^s(X)}\|f\|_{\mathcal{H}^s}.$$

Eq. (10) with $m = s$ and $l = d + 1 > d/2 + s = d/2 + [d/2] + 1$ provides us with the claimed bound.

The content of item 2. is a consequence of the above result with $g = m, m_n, \frac{1}{m}, \frac{1}{m_n}$, and $m - m_n$, respectively, satisfying $\|g\|_{\mathcal{H}^{d+1}} \leq C_4 \max\{C_1, C_2\} = C_5$.
The constant C will be the maximum among the constants C_i. ■

Next lemma shows that the integral operator of kernel W and its empirical counterpart are Hilbert-Schmidt operators and it bounds their difference.

Lemma 17 *The operators $L_{W,\mathcal{H}}, L_{W,n} : \mathcal{H}^s \to \mathcal{H}^s$ defined by*

$$\begin{aligned} L_{W,\mathcal{H}} &= \int_X \langle \cdot, K_x^s \rangle_{\mathcal{H}^s} W_x d\rho(x), \\ L_{W,n} &= \frac{1}{n} \sum_{i=1}^n \langle \cdot, K_{x_i}^s \rangle_{\mathcal{H}^s} W_{x_i}, \end{aligned}$$

are Hilbert-Schmidt. Furthermore, with confidence $1 - 2e^{-\tau}$

$$\|L_{W,\mathcal{H}} - L_{W,n}\|_{HS} \leq C \frac{\sqrt{\tau}}{\sqrt{n}}.$$

for a suitable constant C.

Proof Note that $\|\langle \cdot, K_{x_i}^s \rangle_{\mathcal{H}^s} W_{x_i}\|_{HS} = \|K_{x_i}^s\|_{\mathcal{H}^s} \|W_{x_i}\|_{\mathcal{H}^s} \leq C_1$ for a suitable constant C_1, which is finite by item 1 of Lemma 16 and the fact that K^s is bounded. Hence $L_{W,n}, L_{W,\mathcal{H}}$ are Hilbert Schmidt operators on $\mathcal{H}^s$. The random variables $(\xi_i)_{i=1}^n$ defined by $\xi_i = \langle \cdot, K_{x_i}^s \rangle_{\mathcal{H}^s} W_{x_i} - L_{W,\mathcal{H}}$, taking value in the Hilbert space of Hilbert-Schmidt operators, are zero mean and bounded. Applying (3) we have with confidence $1 - 2e^{-\tau}$

$$\|L_{W,\mathcal{H}} - L_{W,n}\|_{HS} \leq C \frac{\sqrt{\tau}}{\sqrt{n}}, \tag{14}$$

for a suitable constant C. ■

We then consider multiplication operators defined by the degree functions.

Lemma 18 *With confidence $1 - 2e^{-\tau}$*

$$\|M - M_n\|_{\mathcal{H}^s, \mathcal{H}^s} \leq C \frac{\sqrt{\tau}}{\sqrt{n}}.$$

for a suitable constant C.

Proof Item 2 of Lemma 16 ensures that M and M_n are bounded operators on $\mathcal{H}^s$ with $\|M - M_n\|_{\mathcal{H}^s, \mathcal{H}^s} \leq C_1 \|m - m_n\|_{\mathcal{H}^{d+1}}$.
The random variables $(\xi_i)_{i=1}^n$, defined by $\xi_i = W_{x_i} - m \in \mathcal{H}^{d+1}$ are zero mean and bounded. Applying (3) we have with high probability

$$\|m - m_n\|_{\mathcal{H}^{d+1}} \leq \frac{C_2 \sqrt{\tau}}{\sqrt{n}},$$

so that the claim is proved with a suitable choice for C. ■

Remark 19 *In the above lemma we need to control $m - m_n$ in a suitable Hilbert space in order to use Hoeffding inequality* (3)*. Lemma* 15 *of Burenkov (1998) ensures that $\|M - M_n\|_{\mathcal{H}^s, \mathcal{H}^s}$ is bounded by $\|m - m_n\|_{C_b^s(X)}$. In order to control it with a Sobolev norm by means of* (10)*, we need to require that $m - m_n \in \mathcal{H}^l$ with $l > s + d/2$. Furthermore, the requirement that $\mathcal{H}^s$ is a RKHS with continuous bounded kernel implies that $s > d/2$ so that $l > d$. Hence a natural requirement on the weight function is that $W_x \in \mathcal{H}^l(X)$, which is closely related to Assumption 2 with the minimal choice $l = d + 1$.*

Finally, we can combine the above two lemmas to get the proof of Theorem 15.

Proof [Proof of Theorem 15] By Lemma 16, both M and M_n are invertible operators and

$$A_{\mathcal{H}} = M^{-1} L_{W,\mathcal{H}}, \qquad A_n = M_n^{-1} L_{W,n},$$

so that we can consider the following decomposition

$$\begin{aligned} A_n - A_{\mathcal{H}} &= M_n^{-1} L_{W,n} - M^{-1} L_{W,\mathcal{H}} \\ &= (M_n^{-1} - M^{-1}) L_{W,\mathcal{H}} + M_n^{-1}(L_{W,n} - L_{W,\mathcal{H}}) \\ &= M_n^{-1}(M - M_n) M^{-1} L_{W,\mathcal{H}} + \\ &\quad + M_n^{-1}(L_{W,n} - L_{W,\mathcal{H}}). \end{aligned} \tag{15}$$

By taking the Hilbert-Schmidt norm of the above expression and using (1) with the bounds provided by Lemma 16, we get

$$\|A_n - A_{\mathcal{H}}\|_{HS} \leq C^2 \|M - M_n\|_{\mathcal{H}^S, \mathcal{H}^S} \|L_{W,\mathcal{H}}\|_{HS} + C\|L_{W,n} - L_{W,\mathcal{H}}\|_{HS}.$$

The concentration inequalities (17) and (18) give (13), possibly redefining the constant C. ■

In the next section we discuss the implications of the above result in terms of concentration of eigenvalues and spectral projections.

4.3 Bounds on Eigenvalues and Spectral Projections

Since the operators are no longer self-adjoint the perturbation results in Section 3.2 cannot be used. See Appendix B for a short review about spectral theory for compact (not necessarily self-adjoint) operators. The following theorem is an adaptation of results in Anselone (1971), compare with Theorem 4.21.

Theorem 20 *Let A be a compact operator. Given a finite set Λ of non-zero eigenvalues of A, let Γ be any simple rectifiable closed curve (having positive direction) with Λ inside and $\sigma(A) \setminus \Lambda$ outside. Let P be the spectral projection associated with Λ, that is,*

$$P = \frac{1}{2\pi i} \int_\Gamma (\lambda - A)^{-1} \, d\lambda,$$

and define

$$\delta^{-1} = \sup_{\lambda \in \Gamma} \|(\lambda - A)^{-1}\|.$$

Let B be another compact operator such that

$$\|B - A\| \leq \frac{\delta^2}{\delta + \ell(\Gamma)/2\pi} < \delta,$$

where $\ell(\Gamma)$ is the length of Γ, then the following facts hold true.

1. *The curve Γ is a subset of the resolvent set of B enclosing a finite set $\widehat{\Lambda}$ of non-zero eigenvalues of B;*

2. *Denoting by $\widehat{P}$ the spectral projection of B associated with $\widehat{\Lambda}$, then*

$$\|\widehat{P}-P\| \leq \frac{\ell(\Gamma)}{2\pi\delta}\frac{\|B-A\|}{\delta-\|B-A\|};$$

3. *The dimension of the range of P is equal to the dimension of the range of $\hat{P}$.*

Moreover, if $B-A$ is a Hilbert-Schmidt operator, then

$$\|\widehat{P}-P\|_{HS} \leq \frac{\ell(\Gamma)}{2\pi\delta}\frac{\|B-A\|_{HS}}{\delta-\|B-A\|}.$$

We postpone the proof of the above result to Appendix A.

We note that, if A is self-adjoint, then the spectral theorem ensures that

$$\delta = \min_{\lambda\in\Gamma,\sigma\in\Lambda}|\lambda-\sigma|.$$

The above theorem together with the results obtained in the previous section allows to derive several results.

Proposition 21 *If Assumption 2 holds, let σ be an eigenvalue of $\mathbb{L}$, $\sigma\neq 1$, with multiplicity m. For any $\varepsilon>0$ and $\tau>0$, there exists an integer n_0 and a positive constant C such that, if the number of examples is greater than n_0, with probability greater than $1-2e^{-\tau}$,*

1. *there are $\hat{\sigma}_1,\ldots,\hat{\sigma}_m$ (possibly repeated) eigenvalues of the matrix $\mathbf{L}$ satisfying*

$$|\hat{\sigma}_i-\sigma|\leq\varepsilon \qquad \text{for all } i=1,\ldots,m.$$

2. *for any normalized eigenvector $\hat{u}\in\mathbb{C}^n$ of $\mathbf{L}$ with eigenvalue $\hat{\sigma}_i$ for some $i=1,\ldots,m$, there exists an eigenfunction $u\in\mathcal{H}^s\subset L^2(X,\rho)$ of $\mathbb{L}$ with eigenvalue σ, satisfying*

$$\|\hat{v}-u\|_{\mathcal{H}^s}\leq C\frac{\sqrt{\tau}}{\sqrt{n}},$$

where $\hat{v}$ is the Nyström extension of the vector $\hat{u}$ given in item 3 of Proposition 14.

If $\mathbb{L}_{\mathcal{H}}$ is self-adjoint, then $n_0\geq 4\frac{C^2\tau}{\varepsilon^2}$ provided that $\varepsilon<\min_{\sigma'\in\sigma(\mathbb{L}_{\mathcal{H}}),\sigma'\neq\sigma}|\sigma'-\sigma|$.

Proof Theorem 15 gives that, with probability greater than $1-2e^{-\tau}$,

$$\|A_n-A_{\mathcal{H}}\|\leq\|A_n-A_{\mathcal{H}}\|_{HS}\leq\frac{C_1\sqrt{\tau}}{\sqrt{n}}\leq\frac{\delta^2}{\delta+\varepsilon}. \tag{16}$$

for all $n\geq n_0$, where C_1 is a suitable constant, $n_0\in\mathbb{N}$ is such that $\frac{C_1\sqrt{\tau}}{\sqrt{n_0}}\leq\frac{\delta^2}{\delta+\varepsilon}$ and $\delta^{-1}=\sup_{\lambda\in\Gamma}\|(\lambda-A_{\mathcal{H}})^{-1}\|$. Under these conditions, we apply Theorem 20 with $A=A_{\mathcal{H}}$, $B=A_n$ and $\Gamma=\{\lambda\in\mathbb{C}\mid|\lambda-(1-\sigma)|=\varepsilon\}$, so that $\ell(\Gamma)=2\pi\varepsilon$. Since $A_{\mathcal{H}}$ is compact and assuming ε small enough, we have that $\Lambda=\{1-\sigma\}$.

Item 1 of Theorem 20 with Proposition 14 ensures that $\widehat{\Lambda}=\{1-\hat{\sigma}_1,\ldots,1-\hat{\sigma}_m\}$, so that $|\hat{\sigma}_i-\sigma|<\varepsilon$ for all $i=1,\ldots,m$. Let now $\hat{u}\in\mathbb{C}^n$ be a normalized vector such that $\mathbf{L}\hat{u}=\hat{\sigma}_i\hat{u}$ for some $i=1,\ldots,m$.

Proposition 14 ensures that $\hat{v}$ is an eigenfunction of A_n with eigenvalue $1-\hat{\sigma}$, so that $\hat{Q}\hat{v} = \hat{v}$ where $\hat{Q}$ is the spectral projection of A_n associated with $\hat{\Lambda}$. Let Q be the spectral projection of $A_{\mathcal{H}}$ associated with $1-\sigma$ and define $u = Q\hat{v} \in \mathcal{H}^s$. By definition of Q, $A_{\mathcal{H}}u = (1-\sigma)u$. Since $\mathcal{H}^s \subset L^2(X,\rho)$, Proposition 13 ensures that $\mathbb{L}u = \sigma u$. Item 2 of Theorem 20 gives that

$$\begin{aligned}\|\hat{v}-u\|_{\mathcal{H}^s} = \|\hat{Q}\hat{v}-Q\hat{v}\|_{\mathcal{H}^s} &\leq \|\hat{Q}-Q\|_{\mathcal{H}^s,\mathcal{H}^s}\|\hat{v}\|_{\mathcal{H}^s} \leq \|\hat{v}\|_{\mathcal{H}^s}\frac{\varepsilon}{\delta}\frac{\|A_n - A_{\mathcal{H}}\|_{HS}}{\delta - \|A_n - A_{\mathcal{H}}\|_{HS}} \\ &\leq \frac{C_2}{1-(\sigma+\varepsilon)}C_1\frac{\delta+\varepsilon}{\delta^2}\frac{\sqrt{\tau}}{\sqrt{n}},\end{aligned}$$

where we use (16), the fact that $\|A_n - A_{\mathcal{H}}\| \leq \frac{\delta^2}{\delta+\varepsilon}$ and

$$\|\hat{v}\|_{\mathcal{H}^s} \leq \frac{1}{1-\hat{\sigma}}\sup_{x\in X}\|\frac{1}{m_n}W_x\|_{\mathcal{H}^s} = \frac{C_2}{1-\hat{\sigma}} \leq \frac{C_2}{1-(\sigma+\varepsilon)}$$

with C_2 being the constant in item 1 of Lemma 16.
If $A_{\mathcal{H}}$ is self-adjoint, the spectral theorem ensures that $\delta = \varepsilon$, so that $n_0 \geq 4\frac{C^2\tau}{\varepsilon^2}$. ■

Next we consider convergence of the spectral projections of $A_{\mathcal{H}}$ and A_n associated with the first N-eigenvalues. For sake of simplicity, we assume that the cardinality of $\sigma(A_{\mathcal{H}})$ is infinite.

Proposition 22 *Consider the first N eigenvalues of $A_{\mathcal{H}}$. There exist an integer n_0 and a constant $C > 0$, depending on N and $\sigma(A_{\mathcal{H}})$, such that, with confidence $1-2e^{-\tau}$ and for any $n \geq n_0$,*

$$\|P_N - \hat{P}_D\|_{HS} \leq \frac{C\sqrt{\tau}}{\sqrt{n}},$$

where $P_N, \hat{P}_D$ are the eigenprojections corresponding to the first N eigenvalues of $A_{\mathcal{H}}$ and D eigenvalues of A_n, and D is such that the sum of the multiplicity of the first D eigenvalues of A_n is equal to the sum of the multiplicity of the first N eigenvalues of $A_{\mathcal{H}}$.

Proof The proof is close to the one of the previous proposition. We apply Theorem 20 with $A = A_{\mathcal{H}}$, $B = A_n$ and the curve Γ equal to the boundary of the rectangle

$$\{\lambda \in \mathbb{C} \mid \frac{\alpha_N + \alpha_{N+1}}{2} \leq \Re e(\lambda) \leq \|A\| + 2,\ |\Im m(\lambda)| \leq 1\},$$

where α_N is the N-largest eigenvalue of $A_{\mathcal{H}}$ and α_{N+1} the $N+1$-largest eigenvalue of $A_{\mathcal{H}}$. Clearly Γ encloses the first N largest eigenvalues of $A_{\mathcal{H}}$, but no other points of $\sigma(A)$. Define $\delta^{-1} = \sup_{\lambda\in\Gamma}\|(\lambda - A_{\mathcal{H}})^{-1}\|$ and $n_0 \in \mathbb{N}$ such that

$$\frac{C_1\sqrt{\tau}}{\sqrt{n_0}} \leq \frac{\delta^2}{\delta + \ell(\Gamma)/2\pi} \qquad \text{and} \qquad \frac{C_1\sqrt{\tau}}{\sqrt{n_0}} < 1,$$

where C_1 is the constant in Theorem 15. As in the above corollary, with probability greater than $1-2e^{-\tau}$, for all $n \geq n_0$

$$\|A_n - A_{\mathcal{H}}\| \leq \frac{\delta^2}{\delta + \ell(\Gamma)/2\pi} \qquad \text{and} \qquad \|A_n - A_{\mathcal{H}}\| < 1.$$

The last inequality ensures that the largest eigenvalues of A_n is smaller than $1 + \|A_{\mathcal{H}}\|$, so that by Theorem 20, the curve Γ encloses the first D-eigenvalues of A_n, where D is equal to the sum of the multiplicity of the first N eigenvalues of $A_{\mathcal{H}}$. The proof is finished letting $C = \frac{\delta + \ell(\Gamma)/2\pi}{\delta^2} C_1$. ■

Acknowledgments

We would like to thank the referees for many constructive suggestions and comments. E.D.V. and L.R. have been partially supported by the FIRB project RBIN04PARL and by the EU Integrated Project Health-e-Child IST-2004-027749. M. B. is partially supported by the NSF Early Career Award 0643916.

Appendix A. Some Proofs

We start giving the proof of Proposition 13.

Proof [of Proposition 13]

We first need a preliminary fact. The function m is bounded from above and below by a positive constant by (7), so that the measure $\rho_W = m\rho$, having density m w.r.t. ρ, is equivalent[7] to ρ. This implies that the spaces $L^2(X,\rho)$ and $L^2(X,\rho_W)$ are the same vector space and the corresponding norm are equivalent. In this proof, we regard $\mathbb{L}$ as an operator from $L^2(X,\rho_W)$ into $L^2(X,\rho_W)$, observing that its eigenvalues and eigenfunctions are the same as the eigenvalues and eigenfunctions of $\mathbb{L}$, viewed as an operator from $L^2(X,\rho)$ into $L^2(X,\rho)$—however, functions that are orthogonal in $L^2(X,\rho_W)$ in general are not orthogonal in $L^2(X,\rho)$.

Note that the operator $I_K : \mathcal{H} \to L^2(X,\rho_W)$ defined by $I_K f(x) = \langle f, K_x \rangle$ is linear and Hilbert-Schmidt since

$$\begin{aligned}\|I_K\|_{HS}^2 &= \sum_{j\geq 1} \|I_K e_j\|_{\rho_W}^2 = \int_X \sum_{j\geq 1} \langle K_x, e_j \rangle^2 \, d\rho_W(x) \\ &= \int_X K(x,x) m(x) \, d\rho(x) \leq \kappa \|m\|_\infty,\end{aligned}$$

where $\kappa = \sup_{x\in X} K(x,x)$. The operator $I_W^* : L^2(X,\rho_W) \to \mathcal{H}$ defined by

$$I_W^* f = \int_X \frac{1}{m} W_x f(x) \, d\rho(x)$$

is linear and bounded since, by Assumption 1

$$\|\int_X \frac{1}{m} W_x f(x) \, d\rho(x)\|_{\mathcal{H}} \leq \int_X \|\frac{1}{m} W_x\|_{\mathcal{H}} |f(x)| \, d\rho(x) \leq C\|f\|_\rho \leq \frac{C}{c} \|f\|_{\rho_W}.$$

A direct computation shows that

$$I_W^* I_K = A_{\mathcal{H}} = I - \mathbb{L}_{\mathcal{H}}, \qquad \sigma(A_{\mathcal{H}}) = 1 - \sigma(\mathbb{L}_{\mathcal{H}})$$

and

$$I_K I_W^* = I - \mathbb{L}, \qquad \sigma(I_K I_W^*) = 1 - \sigma(\mathbb{L}).$$

7. Two measures are equivalent if they have the same null sets.

Both $I_W^* I_K$ and $I_K I_W^*$ are Hilbert-Schmidt operators since they are composition of a bounded operator and Hilbert-Schmidt operator. Moreover, let $\sigma \neq 1$ and $v \in \mathcal{H}$ with $v \neq 0$ such that $\mathbb{L}_{\mathcal{H}} v = \sigma v$. Letting $u = I_K v$, then

$$\mathbb{L}u = (I - I_K I_W^*) I_K v = I_K \mathbb{L}_{\mathcal{H}} v = \sigma u \quad \text{and} \quad I_W^* u = I_W^* I_K v = (1-\sigma) v \neq 0,$$

so that $u \neq 0$ and u is an eigenfunction of $\mathbb{L}$ with eigenvalue σ. Similarly we can prove that if $\sigma \neq 1$ and $u \in L^2(X, \rho), u \neq 0$ is such that $\mathbb{L}u = \sigma u$, then $v = \frac{1}{1-\sigma} I_W^* u$ is different from zero and is an eigenfunction of $\mathbb{L}_{\mathcal{H}}$ with eigenvalue σ.

We now show that $\mathbb{L}$ is a positive operator on $L^2(X, \rho_W)$, so that both $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ have positive eigenvalues. Indeed, let $f \in L^2(X, \rho_W)$,

$$\begin{aligned}
\langle \mathbb{L} f, f \rangle_{\rho_W} &= \int_X |f(x)|^2 m(x) d\rho(x) - \int_X \left(\int_X \frac{W(x,s)}{m(x)} f(s) d\rho(s) \right) \overline{f(x)} m(x) d\rho(x) \\
&= \frac{1}{2} \int_X \int_X \left[|f(x)|^2 W(x,s) - 2W(x,s) f(x) \overline{f(s)} + |f(s)|^2 W(x,s) \right] d\rho(s) d\rho(x) \\
&= \frac{1}{2} \int_X \int_X W(x,s) |f(x) - f(s)|^2 d\rho(s) d\rho(x) \geq 0,
\end{aligned}$$

where we used that $W(x,s) = W(s,x) > 0$ and $m(x) = \int_X W(x,s)\, d\rho(s)$. Since W is real valued, it follows that we can always choose the eigenfunctions of $\mathbb{L}$ as real valued, and, as a consequence, the eigenfunctions of $\mathbb{L}_{\mathcal{H}}$.

Finally we prove that both $\mathbb{L}$ and $\mathbb{L}_{\mathcal{H}}$ admit a decomposition in terms of spectral projections— we stress that the spectral projection of $\mathbb{L}$ is orthogonal in $L^2(X, \rho_W)$, but not in $L^2(X, \rho)$.
Since $\mathbb{L}$ is a positive operator on $L^2(X, \rho_W)$, it is self-adjoint, as well as $I_K I_W^*$, hence the spectral theorem gives

$$I_K I_W^* = \sum_{j \geq 1} (1 - \sigma_j) P_j$$

where for all j, $P_j : L^2(X, \rho_W) \to L^2(X, \rho_W)$ is the spectral projection of $I_K I_W^*$ associated to the eigenvalue $1 - \sigma_j \neq 0$. Moreover note that P_j is also the spectral projection of $\mathbb{L}$ associated to the eigenvalue $\sigma_j \neq 1$. By definition P_j satisfies:

$$\begin{aligned}
P_j^2 &= P_j, \\
P_j^* &= P_j \qquad \text{the adjoint is in } L^2(X, \rho_W), \\
P_j P_i &= 0, \quad i \neq j, \\
P_j P_{ker(I_K I_W^*)} &= 0, \\
\sum_{j \geq 1} P_j &= I - P_{ker(I_K I_W^*)} = I - P_0
\end{aligned}$$

where $P_{ker(I_K I_W^*)}$ is the projection on the kernel of $I_K I_W^*$, that is, the projection P_0. Moreover the sum in the last equation converges in the strong operator topology. In particular we have

$$I_K I_W^* P_j = P_j I_K I_W^* = (1 - \sigma_j) P_j,$$

so that

$$\mathbb{L} = I - I_K I_W^* = \sum_{j \geq 1} \sigma_j P_j + P_0.$$

Let $Q_j : \mathcal{H} \to \mathcal{H}$ be defined by

$$Q_j = \frac{1}{1-\sigma_j} I_W^* P_j I_K.$$

Then from the properties of the projections P_j we have,

$$\begin{aligned} Q_j^2 &= \frac{1}{(1-\sigma_j)^2} I_W^* P_j I_K I_W^* P_j I_K = \frac{1}{1-\sigma_j} I_W^* P_j P_j I_K = Q_j, \\ Q_j Q_i &= \frac{1}{(1-\sigma_j)(1-\sigma_i)} I_W^* P_j I_K I_W^* P_i I_K = \frac{1}{1-\sigma_i} I_W^* P_j P_i I_K = 0, \quad i \neq j. \end{aligned}$$

Moreover,

$$\sum_{j\geq 1}(1-\sigma_j)Q_j = \sum_{j\geq 1}(1-\sigma_j)\frac{1}{1-\sigma_j} I_W^* P_j I_K = I_W^*(\sum_{j\geq 1} P_j) I_K = I_W^* I_K - I_W^* P_{ker(I_K I_W^*)} I_K$$

so that

$$I_K I_W^* = \sum_{j\geq 1}(1-\sigma_j)Q_j + I_W^* P_{ker(I_K I_W^*)} I_K,$$

where again all the sums are to be intended as converging in the strong operator topology. If we let $D = I_W^* P_{ker(I_K I_W^*)} I_K$ then

$$Q_j D = \frac{1}{1-\sigma_j} I_W^* P_j I_K I_W^* P_{ker(I_K I_W^*)} = I_W^* P_j P_{ker(I_K I_W^*)} = 0,$$

and, similarly $DQ_j = 0$. By construction $\sigma(D) = 0$, that is, D is a quasi-nilpotent operator. Equation (9) is now clear as well as the fact that $\ker D = \ker(I - \mathbb{L}_{\mathcal{H}})$. ■

Proof [Proof of Proposition 14] The proof is the same as the above proposition by replacing ρ with the empirical measure $\frac{1}{n}\sum_{i=1}^n \delta_{x_i}$. ■

Next we prove Theorem 20.

Proof [Proof of Theorem 20] We recall the following basic result. Let S and T two bounded operators acting on $\mathcal{H}$ and defined $C = I - ST$. If $\|C\| < 1$, then T has a bounded inverse and

$$T^{-1} - S = (I - C)^{-1} CS$$

where we note that $\|I - C\|^{-1} \leq \frac{1}{1-\|C\|}$ since $\|C\| < 1$, see Proposition 1.2 of Anselone (1971).

Let A and B two compact operators. Let Γ be a compact subset of the resolvent set of A and define

$$\delta^{-1} = \sup_{\lambda\in\Gamma} \|(\lambda - A)^{-1}\|,$$

which is finite since Γ is compact and the resolvent operator $(\lambda - A)^{-1}$ is norm continuous (see, for example, Anselone, 1971). Assume that

$$\|B - A\| < \delta,$$

then for any $\lambda \in \Gamma$

$$\|(\lambda - A)^{-1}(B-A)\| \leq \|(\lambda - A)^{-1}\|\|B-A\| \leq \delta^{-1}\|B-A\| < 1.$$

Hence we can apply the above result with $S = (\lambda - A)^{-1}, T = (\lambda - B)$, since

$$C = I - (\lambda - A)^{-1}(\lambda - B) = (\lambda - A)^{-1}(B-A).$$

It follows that $(\lambda - B)$ has a bounded inverse and

$$(\lambda - B)^{-1} - (\lambda - A)^{-1} = (I - (\lambda - A)^{-1}(B-A))^{-1}(\lambda - A)^{-1}(B-A)(\lambda - A)^{-1}.$$

In particular, Γ is a subset of the resolvent set of B and, if $B - A$ is a Hilbert-Schmidt operator, so is $(\lambda - B)^{-1} - (\lambda - A)^{-1}$.

Let Λ be a finite set of non-zero eigenvalues. Let Γ be any simple closed curve with Λ inside and $\sigma(A) \setminus \Lambda$ outside. Let P be the spectral projection associated with Λ, then

$$P = \frac{1}{2\pi i}\int_\Gamma (\lambda - A)^{-1}\, d\lambda.$$

Applying the above result, it follows that Γ is a subset of the resolvent set of B and we let $\widehat{\Lambda}$ be the subset of $\sigma(B)$ inside Γ and $\widehat{P}$ the corresponding spectral projection, then

$$\begin{aligned}\widehat{P} - P &= \frac{1}{2\pi i}\int_\Gamma (\lambda - B)^{-1} - (\lambda - A)^{-1}\, d\lambda \\ &= \frac{1}{2\pi i}\int_\Gamma (I - (\lambda - A)^{-1}(B-A))^{-1}(\lambda - A)^{-1}(B-A)(\lambda - A)^{-1}\, d\lambda.\end{aligned}$$

It follows that

$$\|\widehat{P} - P\| \leq \frac{\ell(\Gamma)}{2\pi}\frac{\delta^{-2}\|B-A\|}{1 - \delta^{-1}\|B-A\|} = \frac{\ell(\Gamma)}{2\pi\delta}\frac{\|B-A\|}{\delta - \|B-A\|}.$$

In particular if $\|B-A\| \leq \frac{\delta^2}{\delta + \ell(\Gamma)/2\pi} < \delta$, $\|\widehat{P} - P\| \leq 1$ so that the dimension of the range of P is equal to the dimension of the range of $\hat{P}$. It follows that $\widehat{\Lambda}$ is not empty.
If $B - A$ is a Hilbert-Schmidt operator, we can replace the operator norm with the Hilbert-Schmidt norm, and the corresponding inequality is a consequence of the fact that the Hilbert-Schmidt operator are an ideal. ■

Appendix B. Spectral Theorem for Non-self-adjoint Compact Operators

Let $A : \mathcal{H} \to \mathcal{H}$ be a compact operator. The spectrum $\sigma(A)$ of A is defined as the set of complex number such that the operator$(A - \lambda I)$ does not admit a bounded inverse, whereas the complement of $\sigma(A)$ is called the resolvent set and denoted by $\rho(A)$. For any $\lambda \in \rho(A)$, $R(\lambda) = (A - \lambda I)^{-1}$ is the resolvent operator, which is by definition a bounded operator. We recall the main results about the spectrum of a compact operator (Kato, 1966)

Proposition 23 *The spectrum of a compact operator A is a countable compact subset of* $\mathbb{C}$ *with no accumulation point different from zero, that is,*

$$\sigma(A) \setminus \{0\} = \{\lambda_i \mid i \geq 1,\ \lambda_i \neq \lambda_j \text{ if } i \neq j\}$$

with $\lim_{i\to\infty} \lambda_i = 0$ *if the cardinality of* $\sigma(A)$ *is infinite. For any* $i \geq 1$, λ_i *is an eigenvalue of A, that is, there exists a nonzero vector* $u \in \mathcal{H}$ *such that* $Au = \lambda_i u$. *Let* Γ_i *be a rectifiable closed simple curve (with positive direction) enclosing* λ_i, *but no other points of* $\sigma(A)$, *then the operator defined by*

$$P_{\lambda_i} = \frac{1}{2\pi i} \int_{\Gamma_i} (\lambda I - A)^{-1} d\lambda$$

satisfies

$$P_{\lambda_i} P_{\lambda_j} = \delta_{ij} P_{\lambda_i} \qquad \text{and} \qquad (A - \lambda_i) P_{\lambda_i} = D_{\lambda_i} \qquad \text{for all } i, j \geq 1,$$

where D_{λ_i} *is a nilpotent operator such that* $P_{\lambda_i} D_{\lambda_i} = D_{\lambda_i} P_{\lambda_i} = D_{\lambda_i}$. *In particular the dimension of the range of* P_{λ_i} *is always finite.*

We notice that P_{λ_i} is a projection onto a finite dimensional space $\mathcal{H}_{\lambda_i}$, which is left invariant by A. A nonzero vector u belongs to $\mathcal{H}_{\lambda_i}$ if and only if there exists an integer $m \leq \dim \mathcal{H}_{\lambda_i}$ such that $(A - \lambda)^m u = 0$, that is, u is a generalized eigenvector of A. However, if A is symmetric, for all $i \geq 1$, $\lambda_i \in \mathbb{R}$, P_{λ_i} is an orthogonal projection and $D_{\lambda_i} = 0$ and it holds that

$$A = \sum_{i \geq 1} \lambda_i P_{\lambda_i}$$

where the series converges in operator norm. Moreover, if $\mathcal{H}$ is infinite dimensional, $\lambda = 0$ is always in $\sigma(A)$, but it can be or not an eigenvalue of A.

If A be a compact operator with $\sigma(A) \subset [0, \|A\|]$, we introduce the following notation. Denoted by p_A the cardinality of $\sigma(A) \setminus \{0\}$ and given an integer $1 \leq N \leq p_A$, let $\lambda_1 > \lambda_2 > \ldots, \lambda_N > 0$ be the first N nonzero eigenvalues of A, sorted in a decreasing way. We denote by P_N^A the spectral projection onto all the generalized eigenvectors corresponding to the eigenvalues $\lambda_1, \ldots, \lambda_N$. The range of P_N^A is a finite-dimensional vector space, whose dimension is the sum of the algebraic multiplicity of the first N eigenvalues. Moreover

$$P_N^A = \sum_{j=1}^{N} P_{\lambda_j} = \frac{1}{2\pi i} \int_{\Gamma} (\lambda I - A)^{-1} d\lambda$$

where Γ is a rectifiable closed simple curve (with positive direction) enclosing $\lambda_1, \ldots, \lambda_N$, but no other points of $\sigma(A)$.

References

P. M. Anselone. *Collectively Compact Operator Approximation Theory and Applications to Integral Equations*. Prentice-Hall Inc., Englewood Cliffs, N. J., 1971. With an appendix by Joel Davis, Prentice-Hall Series in Automatic Computation.

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *J. Complexity*, 23(1):52–72, 2007.

M. Belkin. *Problems of Learning on Manifolds*. PhD thesis, University of Chicago, USA, 2003.

M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Proceedings of the 18th Conference on Learning Theory (COLT)*, pages 486–500, 2005.

M. Belkin and P. Niyogi. Convergence of Laplacian eigenmaps. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 129–136. MIT Press, Cambridge, MA, 2007.

V.I. Burenkov. *Sobolev Spaces on Domains*. B. G. Teubuer, Stuttgart-Leipzig, 1998.

R.R. Coifman and S. Lafon. Geometric harmonics: a novel tool for multiscale out-of-sample extension of empirical functions. *Appl. Comput. Harmon. Anal.*, 21(1):31–52, 2006. ISSN 1063-5203.

E. De Vito, A. Caponnetto, and L. Rosasco. Model selection for regularized least-squares algorithm in learning theory. *Foundations of Computational Mathematics*, 5(1):59–85, February 2005a.

E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, and F. Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6:883–904, May 2005b.

E. De Vito, L. Rosasco, and A. Caponnetto. Discretization error analysis for Tikhonov regularization. *Anal. Appl. (Singap.)*, 4(1):81–99, 2006.

E. Giné and V. Koltchinskii. Empirical graph Laplacian approximation of Laplace-Beltrami operators: Large sample results. *High Dimensional Probability*, 51:238–259, 2006.

M. Hein. Uniform convergence of adaptive graph-based regularization. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT)*, pages 50–64, New York, 2006. Springer.

M. Hein, J-Y. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proceedings of the 18th Conference on Learning Theory (COLT)*, pages 470–485, 2005. Student Paper Award.

T. Kato. *Perturbation Theory for Linear Operators*. Springer, Berlin, 1966.

T. Kato. Variation of discrete spectra. *Commun. Math. Phys.*, III:501–504, 1987.

V. Koltchinskii. Asymptotics of spectral projections of some random matrices approximating integral operators. *Progress in Probabilty*, 43, 1998.

V. Koltchinskii and E. Giné. Random matrix approximation of spectra of integral operators. *Bernoulli*, 6:113–167, 2000.

S. Lafon. *Diffusion Maps and Geometric Harmonics*. PhD thesis, Yale University, USA, 2004.

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

S. Mendelson and A. Pajor. Ellipsoid approximation with random vectors. In *Proceedings of the 18th Annual Conference on Learning Theory (COLT)*, pages 429–433, New York, 2005. Springer.

S. Mendelson and A. Pajor. On singular values of matrices with independent rows. *Bernoulli*, 12(5): 761–773, 2006.

I. Pinelis. An approach to inequalities for the distributions of infinite-dimensional martingales. *Probability in Banach Spaces, 8, Proceedings of the 8th International Conference*, pages 128–134, 1992.

L. Schwartz. Sous-espaces hilbertiens d'espaces vectoriels topologiques et noyaux associés (noyaux reproduisants). *J. Analyse Math.*, 13:115–256, 1964. ISSN 0021-7670.

J. Shawe-Taylor, N. Cristianini, and J. Kandola. On the concentration of spectral properties. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 511–517, Cambridge, MA, 2002. MIT Press.

J. Shawe-Taylor, C. Williams, N. Cristianini, and J. Kandola. On the eigenspectrum of the Gram matrix and the generalization error of kernel-PCA. *IEEE Trans. Inform. Theory*, 51(7):2510–2522, 2005. ISSN 0018-9448.

A. Singer. From graph to manifold Laplacian: The convergence rate. *Appl. Comput. Harmon. Anal.*,, 21:128–134, 2006.

S. Smale and D.X. Zhou. Learning theory estimates via integral operators and their approximations. *Constr. Approx.*, 26(2):153–172, 2007. ISSN 0176-4276.

S. Smale and D.X. Zhou. Geometry of probability spaces. *Constr. Approx.*, 30(3):311–323, 2009.

U. Von Luxburg, O. Bousquet, and M. Belkin. On the convergence of spectral clustering on random samples: the normalized case. In *Proceedings of the 17th Annual Conference on Learning Theory (COLT 2004*, pages 457–471. Springer, 2004.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Ann. Statist.*, 36 (2):555–586, 2008.

Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constr. Approx.*, 26(2):289–315, 2007.

L. Zwald and G. Blanchard. On the convergence of eigenspaces in kernel principal component analysis. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1649–1656. MIT Press, Cambridge, MA, 2006.

L. Zwald, O. Bousquet, and G. Blanchard. Statistical properties of kernel principal component analysis. In *Learning theory*, volume 3120 of *Lecture Notes in Comput. Sci.*, pages 594–608. Springer, Berlin, 2004.

Journal of Machine Learning Research 11 (2010) 935-953 Submitted 4/09; Published 2/10

On Spectral Learning

Andreas Argyriou A.ARGYRIOU@CS.UCL.AC.UK
Department of Computer Science
University College London
Gower Street, London, WC1E 6BT, UK

Charles A. Micchelli
Department of Mathematics and Statistics
State University of New York
The University at Albany
Albany, New York 12222, USA

Massimiliano Pontil M.PONTIL@CS.UCL.AC.UK
Department of Computer Science
University College London
Gower Street, London, WC1E 6BT, UK

Editor: Alexander Smola

Abstract

In this paper, we study the problem of learning a matrix W from a set of linear measurements. Our formulation consists in solving an optimization problem which involves regularization with a spectral penalty term. That is, the penalty term is a function of the spectrum of the covariance of W. Instances of this problem in machine learning include multi-task learning, collaborative filtering and multi-view learning, among others. Our goal is to elucidate the form of the optimal solution of spectral learning. The theory of spectral learning relies on the von Neumann characterization of orthogonally invariant norms and their association with symmetric gauge functions. Using this tool we formulate a representer theorem for spectral regularization and specify it to several useful example, such as Schatten $p-$norms, trace norm and spectral norm, which should proved useful in applications.

Keywords: kernel methods, matrix learning, minimal norm interpolation, multi-task learning, orthogonally invariant norms, regularization

1. Introduction

In this paper, we study the problem of learning a matrix from a set of linear measurements. Our formulation consists in solving for the matrix

$$\hat{W} = \operatorname{argmin}\{E(I(W), y) + \gamma\Omega(W) : W \in M_{d,n}\}, \tag{1}$$

where $M_{d,n}$ is the set of $d \times n$ real matrices, y an $m-$dimensional real vector of observations and $I : M_{d,n} \to \mathbb{R}^m$ a linear operator, whose components are given by the Frobenius inner product between the matrix W and prescribed data matrices. The objective function in (1) combines a data term, $E(I(W), y)$, which measures the fit of W to available training data and a penalty term or regularizer, $\Omega(W)$. The positive constant γ controls the trade-off between the two terms and may be chosen

by prior information on the noise underlying the data. A typical example for the data term is $E(I(W),y) = \|I(W) - y\|_2^2$, where the subscript indicates the Euclidean norm.

In the design of learning algorithms from the point of view of regularization the choice of the penalty term is essential. To obtain insights into this issue, we shall investigate in this paper the form of matrices which solve the variational problem (1) when the penalty term is an *orthogonally invariant norm* (OI-norm). This means, for any pair of orthogonal matrices U and V, that

$$\Omega(UWV) = \Omega(W).$$

There are many important examples of OI-norms. Among them, the family of Schatten $p-$norms, $1 \leq p \leq \infty$, namely the ℓ_p-norm of the singular values of a matrix, are especially useful.

Our main motivation for studying the optimization problem (1) arises from its application to multi-task learning, see Argyriou et al. (2007a) and references therein. In this context, the matrix columns are interpreted as the parameters of different regression or classification tasks and the regularizer Ω is chosen in order to favor certain kinds of dependencies across the tasks. The operator I consists of inner products formed from the inputs of each task and the error term $E(I(W),y)$ is the sum of losses on the individual tasks. Collaborative filtering (Srebro et al., 2005) provides another interesting instance of problem (1), in which the operator I is formed from a subset of the matrix elements. Further examples in which OI-norm have been used include multi-class classification (Amit et al., 2007), multi-view learning (Cavallanti et al., 2008) and similarity learning (Maurer, 2008a).

A recent trend in regularization methods in machine learning is to use matrix regularizers which are orthogonally invariant (Argyriou et al., 2007a,b; Abernethy et al., 2009; Srebro et al., 2005). In particular, an important case is the Schatten one norm of W, which is often referred to as the *trace norm*. The general idea behind this methodology is that a small trace norm favors low-rank solution matrices to (1). This means that the tasks (the columns of W) are related in that they all lie in a low-dimensional subspace of $\mathbb{R}^d$. Indeed, if we choose the regularizer to be the rank of a matrix, we obtained a non-convex NP-hard problem. However, the trace norm provides a convex relaxation of this problem, which has been justified in various ways (see, for example, Fazel et al., 2001; Candès and Recht, 2008).

The main purpose of this paper is to characterize the form of the solutions to problem (1). Specifically, we provide what in machine learning is known as a *representer theorem*. Namely we show, for a wide variety of OI-norm regularizers, that it is possible to compute the inner product $\langle \hat{W}, X \rangle$ only in terms of the $m \times m$ Gram matrix I^*I and $I(X)$. A representer theorem is appealing from a practical point of view, because it ensures that the cost of solving the optimization problem (1) depends on the size m of the training sample, which can be much smaller than the number of elements of the matrix W. For example, in multi-task learning, the number of rows in the matrix W may be much larger than the number of data per task. More fundamentally, the task vectors (the columns of matrix W) may be elements of a reproducing kernel Hilbert Space.

Our point of view in developing these theorems is through the study of the *minimal norm interpolation* problem

$$\min\{\|W\| : I(W) = \hat{y},\ W \in M_{d,n}\}.$$

The reason for this is that the solution $\hat{W}$ of problem (1) also solves the above problem for an appropriately chosen $\hat{y} \in \mathbb{R}^m$. Specifically, this is the case if we choose $\hat{y} = I(\hat{W})$. In the development of these results, tools from convex analysis are needed. In particular, a key tool that we use in this

paper is a classical result of von Neumann (1962), which characterizes OI-norms in terms of the notion of *symmetric gauge function*; see also Lewis (1995) for a discussion of the von Neumann theorem in the context of convex analysis. We record some of these facts which we need in Section 4.

The paper is organized in the following manner. In Section 2 we introduce our notation and describe the connection between minimal norm interpolation and regularization. In Section 3 we describe the relationship between any solution of (1) and any solution of a dual problem, which involves a number of variables equal to the training set size. In Section 4 we specify this result to the class of OI-norms. In particular, we describe a special case of such norms, which contains the Schatten $p-$norms, and derive a linear representer theorem for this case. As we shall see, this computation in general involves a nonlinear function and a singular value decomposition of an appropriate matrix.

2. Background

Before proceeding, we introduce some of the notation used in the paper and review some basic facts.

2.1 Notation

We use $\mathbb{N}_d$ as a shorthand for the set of integers $\{1,\ldots,d\}$, $\mathbb{R}^d$ for the linear space of vectors with d real components and $M_{d,n}$ for the linear space of $d\times n$ real matrices. For any vector $a\in\mathbb{R}^d$ we use a_i to denote its i-th component and for any matrix $W\in M_{d,n}$ we use w_t to denote the t-th column of W, for $t\in\mathbb{N}_n$. For a vector $\lambda\in\mathbb{R}^d$, we let $\mathrm{Diag}(\lambda)$ or $\mathrm{Diag}(\lambda_i)_{i\in\mathbb{N}_d}$ to denote the $d\times d$ diagonal matrix having the elements of λ on the diagonal. We denote the trace of matrix W by $\mathrm{tr}(W)$. We use $\mathbf{S}^d$ to denote the set of $d\times d$ real symmetric matrices and $\mathbf{S}^d_+$ and $\mathbf{S}^d_{++}$ to denote the subsets of positive semidefinite and positive definite ones, respectively. We use $\succ$ and $\succeq$ for the positive definite and positive semidefinite partial orderings on $\mathbf{S}^d$, respectively. We also let O_d be the set of $d\times d$ orthogonal matrices and $\mathcal{P}_d$ the set of $d\times d$ permutation matrices. Finally, in this paper, the notation $\langle\cdot,\cdot\rangle$ denotes the standard inner products on $\mathbb{R}^d$ and $M_{d,n}$, that is, $\langle a,b\rangle=\sum_{i\in\mathbb{N}_d}a_ib_i$ for any vectors $a,b\in\mathbb{R}^d$ and $\langle W,V\rangle=\mathrm{tr}(W^\top V)$ for any matrices $W,V\in M_{d,n}$.

2.2 Regularization and Interpolation with Matrices

Let us first describe the type of optimization problems of interest in this paper. Our motivation comes from recent work in machine learning which deals with the problem of multi-task learning. Beyond these practical concerns, the matrix optimization problems we consider here have the property that the *matrix structure* is important.

We shall consider *regularization* problems of the type

$$\min\left\{E(I(W),y)+\gamma\,\Omega(W):W\in M_{d,n}\right\}, \tag{2}$$

where $E:\mathbb{R}^m\times\mathbb{R}^m\to\mathbb{R}$ is called a *loss function*, $\Omega:M_{d,n}\to\mathbb{R}$ a *regularizer*, $\gamma>0$ the *regularization parameter*, $I:M_{d,n}\to\mathbb{R}^m$ is a *linear operator* and $y\in\mathbb{R}^m$. Associated to the above regularization problem is the *interpolation problem*

$$\min\left\{\Omega(W):W\in M_{d,n},\ I(W)=y\right\}. \tag{3}$$

Unless otherwise stated, we always assume that the minima in problems (2) and (3) are attained.

Regularization enables one to trade off interpolation of the data against smoothness or simplicity of the model, whereas interpolation frequently suffers from *overfitting*. Note that the family of the former problems encompasses the latter ones. Indeed, an interpolation problem can be simply obtained in the limit as the regularization parameter γ goes to zero (see, for example, Micchelli and Pinkus, 1994).

For example, a special case of matrix regularization problems of the type (2) is obtained with the choice

$$I(W) = (\langle w_t, x_{ti} \rangle : t \in \mathbb{N}_n, i \in \mathbb{N}_{m_t}) \,,$$

where the x_{ti} are given input vectors in $\mathbb{R}^d$. This occurs, for example, in multi-task learning and problems closely related to it (Abernethy et al., 2009; Argyriou et al., 2007a,b; Candès and Recht, 2008; Cavallanti et al., 2008; Izenman, 1975; Maurer, 2006a,a; Srebro et al., 2005; Yuan et al., 2007, etc.). In learning multiple tasks jointly, each task may be represented by a vector of regression parameters which corresponds to the column w_t in our notation. There are n tasks and m_t data examples $\{(x_{ti}, y_{ti}) : i \in \mathbb{N}_{m_t}\}$ for the t-th task.

In multi-task learning, the error term E in (2) expresses the objective that the regression vector for each task should fit well the data for this particular task. The choice of the regularizer Ω is important in that it captures certain relationships between the tasks. For example, one such regularizer, considered in Evgeniou et al. (2005), is a specific quadratic form in W, namely

$$\Omega(W) = \sum_{s,t \in \mathbb{N}_n} \langle w_s, E_{st} w_t \rangle,$$

where the matrices $E_{st} \in \mathbf{S}^d$ are chosen to model cross-tasks interactions.

Another common choice for the regularizer is the *trace norm*, which is defined to be the sum of the singular values of a matrix,

$$\Omega(W) = \sum_{j \in \mathbb{N}_r} \sigma_j(W),$$

where $r = \min(d, m)$. Equivalently this regularizer can be expressed as $\Omega(W) = \mathrm{tr}(W^\top W)^{\frac{1}{2}}$. Regularization with the trace norm learns the tasks as one joint optimization problem, by favoring matrices with low rank (Argyriou et al., 2007a). In other words, the vectors w_t are related in that they are *all* linear combinations of a *small* set of basis vectors. It has been demonstrated that this approach allows for accurate estimation of related tasks even when there are only a *few* data points available for each task.

In general, the linear operator I can be written in the form

$$I(W) = (\langle W, X_i \rangle : i \in \mathbb{N}_m) \,, \tag{4}$$

where the inputs matrices X_i are in $M_{d,n}$. Recall that the *adjoint* operator, $I^* : \mathbb{R}^m \to M_{d,n}$, is defined by the property that

$$\langle I^*(c), W \rangle = \langle c, I(W) \rangle \,,$$

for all $c \in \mathbb{R}^m, W \in M_{d,n}$. Therefore, it follows that I^* is given at $c \in \mathbb{R}^m$, by

$$I^*(c) = \sum_{i \in \mathbb{N}_m} c_i X_i.$$

We denote by $\mathcal{R}(I)$ and $\mathcal{N}(I)$ the range and the null space of the operator I, respectively.

In this paper, we are interested in studying the form of the solution to matrix problems (2) or (3). For certain families of regularizers, the solutions can be expressed in terms of the given inputs X_i in (4). Such facts are known in machine learning as *representer theorems*, see Argyriou et al. (2009) and reference therein.

The line of attack we shall follow in this paper will go through *interpolation*. That is, our main concern will be to obtain representer theorems which hold for problems like (3). This in turn will imply representer theorems for the associated regularization problems. This is justified by the next lemma.

Lemma 1 *Let $E : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$, a linear operator $I : M_{d,n} \to \mathbb{R}^m$, $\Omega : M_{d,n} \to \mathbb{R}$, $\gamma > 0$ such that the problems* (2) *and* (3) *admit a minimizer for every $y \in \mathbb{R}^m$. Then for every $y \in \mathbb{R}^m$ there exists $\hat{y} \in \mathbb{R}^m$ such that any solution of the interpolation problem* (3) *with $y = \hat{y}$ is a solution of the regularization problem* (2).

Proof If $\hat{W}$ solves (2), we may define $\hat{y} := I(\hat{W})$. It then readily follows that any solution of (3) with $\hat{y}$ in place of y is a solution of (2). ■

For some other results relating optimality conditions for regularization and interpolation problems, see Argyriou et al. (2009). We shall return to this issue in Section 4, where we study representer theorems of a particular type for regularizers which are OI-norms.

3. Duality and Minimal Norm Interpolation

In this section, we turn our attention to the study of the interpolation problem (3) when the function Ω is a *norm* on $M_{d,n}$. That is we prescribe a linear operator $I : M_{d,n} \to \mathbb{R}^m$, a vector $y \in \mathcal{R}(I) \setminus \{0\}$ and study the minimal norm interpolation problem

$$\phi := \min\{\|W\| : I(W) = y, W \in M_{d,n}\}. \tag{5}$$

The approach we take to analyze problem (5) makes use of a dual problem. To identify it, we recall the definition of the dual norm, given by

$$\|X\|_D = \max\{\langle X, W\rangle : W \in M_{d,n}, \|W\| \leq 1\}.$$

Consequently, it follows, for every $X, W \in M_{d,n}$, that

$$|\langle X, W\rangle| \leq \|X\|_D \|W\|. \tag{6}$$

Associated with this inequality is the notion of the *peak set* of the norm $\|\cdot\|$ at X, namely

$$\mathscr{D}\|X\| = \{W : \langle X, W\rangle = \|X\|_D, : W \in M_{d,n}, \|W\| = 1\}.$$

Note that, for each $X \in M_{d,n}\backslash\{0\}$ the peak set $\mathscr{D}\|X\|$ is a nonempty compact convex set which contains all $W \in M_{d,n}\backslash\{0\}$ that make the bound in (6) tight.

As we shall see in the theorem below, the dual norm leads to the following *dual problem*

$$\theta := \min\left\{\|I^*(c)\|_D : c \in \mathcal{R}(I),\ \langle c, y\rangle = 1\right\}. \tag{7}$$

Let us first observe that both the primal and dual problem have solutions. In the primal problem we minimize a norm which is a function which grows at infinity and, so, the existence of a solution is

assured. Similarly, the quantity $\|I^*(c)\|_D$ which is minimized in the dual problem is also norm on $c \in \mathcal{R}(I)$.

The main result of this section establishes the relationship between the solutions of the primal problem (5) and those of the dual problem (7).

Theorem 2 *A vector $\hat{c} \in \mathbb{R}^m$ solves the dual problem* (7) *if and only if there exists $\hat{W} \in \theta^{-1}\mathscr{D}\|I^*(\hat{c})\|$ such that $I(\hat{W}) = y$. Moreover, in this case $\hat{W}$ solves the primal problem* (5) *and $\phi\theta = 1$. Conversely, for every $\hat{W}$ solving the primal problem* (5) *and* any *solution $\hat{c}$ of the dual problem* (7)*, it holds that $\hat{W} \in \theta^{-1}\mathscr{D}\|I^*(\hat{c})\|$.*

Before we proceed with a proof, let us explain the rationale behind this result. The number of free parameters in the dual problem is at most $m-1$, while the primal problem involves $dn-m$ parameters. Typically, in applications, dn is much larger than m. Recalling the connection to multi-task learning in Section 2, this means that d is much larger than the number of data per task, $\frac{m}{n}$. Therefore, from the perspective of this parameter count, solving the dual problem may be advantageous. More importantly, any solution of the dual problem will provide us with a solution of the primal problem and conditions on the latter are obtained from a study of the peak set $\mathscr{D}\|I^*(\hat{c})\|$. For example, as we shall see in Section 4, in the case of OI-norms, this fact will be facilitated by fundamental matrix inequalities.

Proof of Theorem 2 First let us establish that

$$\frac{1}{\theta} \leq \phi. \tag{8}$$

To this end, consider any $c \in \mathbb{R}^m$ with $\langle c, y\rangle = 1$ and $W \in M_{d,n}$ with $I(W) = y$. Then

$$1 = \langle c, y\rangle = \langle c, I(W)\rangle = \langle I^*(c), W\rangle \leq \|I^*(c)\|_D \|W\|. \tag{9}$$

From this inequality we get the desired claim. To prove the reverse inequality in (8), we let $\hat{c} \in \mathcal{R}(I)$ be a solution of the dual problem (7) and conclude, for any $b \in \mathcal{R}(I)$ such that $\langle b, y\rangle = 0$, that

$$\lim_{\varepsilon \to 0^+} \frac{\|I^*(\hat{c} + \varepsilon b)\|_D - \|I^*(\hat{c})\|_D}{\varepsilon} \geq 0.$$

Since the dual norm is a maximum of linear functions over a compact set, we may apply Theorem 22 in the case that $\mathcal{X} = \{X : X \in M_{d,n}, \|X\| \leq 1\}$, $\mathcal{W} = M_{d,n}$, $f(W,X) = \langle W, X\rangle$, and evaluate Equation (16) for $W = I^*(\hat{c})$ and $\Delta = I^*(b)$ to obtain the inequality

$$\max\{\langle I^*(b), T\rangle \ : \ T \in \mathscr{D}\|I^*(\hat{c})\|\} \geq 0\,.$$

Using the fact that $b \in \mathcal{R}(I)$ and $\langle b, y\rangle = 0$, we can rephrase this inequality in the following fashion. For every $Z \in M_{d,n}$ such that $\langle Z, I^*(y)\rangle = 0$ we have that

$$\max\{\langle Z, I^*I(T)\rangle \ : \ T \in \mathscr{D}\|I^*(\hat{c})\|\} \geq 0.$$

To resolve this set of inequalities we use Lemma 21 in the appendix with $k = 1$, $J : M_{d,n} \to \mathbb{R}$ defined at $W \in M_{d,n}$ as $J(W) = \langle I^*(y), W\rangle$ and $\mathcal{W} := I^*I(\mathscr{D}\|I^*(\hat{c})\|)$. Since $J^* : \mathbb{R} \to M_{d,n}$ is given for $a \in \mathbb{R}$ as $J^*(a) = aI^*(y)$, we conclude that there exist $\lambda \in \mathbb{R}$ and $\tilde{W} \in \mathscr{D}\|I^*(\hat{c})\|$ such that

$$\lambda I^*(y) = I^*I(\tilde{W}).$$

This equation implies that $\lambda y - I(\tilde{W}) \in \mathcal{N}(I^*)$. However, recalling the fact that $y \in \mathcal{R}(I)$, we also have that $\lambda y - I(\tilde{W}) \in \mathcal{R}(I)$. Therefore, we have established that

$$\lambda y = I(\tilde{W}).$$

To identify the value of λ we use the fact that $\langle y, \hat{c} \rangle = 1$ and obtain that

$$\lambda = \langle \hat{c}, I(\tilde{W}) \rangle = \langle I^*(\hat{c}), \tilde{W} \rangle = \|I^*(\hat{c})\|_D = \theta.$$

Now, we define $\hat{W} = \frac{1}{\theta}\tilde{W}$ and note that $I(\hat{W}) = y$ and since $\|\tilde{W}\| = 1$ we obtain that

$$\phi \leq \|\hat{W}\| = \frac{1}{\theta}.$$

This inequality, combined with inequality (8) demonstrates that $\phi\theta = 1$ and that $\hat{W}$ is a solution to the primal problem (5).

To complete the proof, consider any $\hat{W}$ solving (5) and $\hat{c}$ solving (7) and it easily follows from inequality (9) and $\phi\theta = 1$ that $\hat{W} \in \theta^{-1}\mathscr{D}\|I^*(\hat{c})\|$. ■

Theorem 2 describes the relation between the set of solutions of the primal problem (5) and the dual problem (7). It also relates the set of solutions of the primal problem to the range of the adjoint operator I^*. This latter property, combined with Lemma 1, may be viewed as a general *representer theorem*, that is, the theorem implies that the solutions of the regularization problem (2) are matrices in the set $\mathscr{D}\|I^*(\tilde{c})\|$, for some $\tilde{c} \in \mathbb{R}^m$. However, additional effort is required to obtain a concrete representation of such solution. For example, for the Frobenius norm, $\mathscr{D}\|X\| = \{X/\|X\|\}$ and, so, the optimality condition becomes $\hat{W} = I^*(\tilde{c})$. We refer to this condition throughout the paper as the *standard representer theorem*, see Argyriou et al. (2009) and references therein. In other words, the standard representer theorem for $\hat{W}$ means that $\hat{W} \in \mathcal{R}(I^*)$.

We make no claim of originality for Theorem 2 as its proof uses well established tools of convex analysis. On the contrary, we emphasize the utility of this result for machine learning. Alternatively, we can approach the minimal norm interpolation problem by use of the Lagrangian, defined, for $W \in M_{d,n}$ and $\lambda \in \mathbb{R}^m$, as

$$\mathcal{L}(W, \lambda) = \|W\| + \langle W, I^*(\lambda) \rangle - \langle y, \lambda \rangle.$$

4. Representer Theorems for Orthogonally Invariant Norms

In this section, we focus our attention on matrix norms which are invariant under left and right multiplication by orthogonal matrices. As we shall see, for such norms, the representer theorem can be written in terms of the singular value decomposition. In addition, in Section 4.3, we shall describe a subclass of OI-norms for which representer theorems can be phrased in terms of matrix multiples of the adjoint operator value $I^*(\hat{c})$. This type of representer theorem arises in *multi-task learning* as described in Argyriou et al. (2009). That is, each of the columns of the optimal matrix lies in the span of the corresponding columns of the input matrices X_i.

4.1 Notation

Let $W \in M_{d,n}$ be a matrix and set $r = \min\{d,n\}$. We express the singular value decomposition of the matrix W in the form

$$W = U\Sigma V^\top,$$

where $U \in O_d, V \in O_n$ and $\Sigma \in M_{d,n}$ is a diagonal matrix with nonnegative elements, that is $\Sigma = \mathrm{diag}(\sigma(W))$, where $\sigma(W) = (\sigma_i(W) : i \in \mathbb{N}_r) \in \mathbb{R}^r_+$. We assume that the singular values are *ordered* in a non-increasing sense, that is,

$$\sigma_1(W) \geq \cdots \geq \sigma_r(W) \geq 0\,.$$

Note that $\sigma(W)$ is uniquely defined in this way. Sometimes we also use $\Sigma(W)$ to denote the diagonal matrix Σ. The components of $\sigma(W)$ are the *singular values* of W. They are equal to the square root of the largest r eigenvalues of $W^\top W$, which are the same as those of $WW^\top$. We shall call functions of the singular values of a matrix *spectral functions*.

In the case of a symmetric matrix $A \in \mathbf{S}^n$, we similarly write

$$A = U\Lambda U^\top$$

for a spectral decomposition of A, where $U \in O_n$, $\Lambda = \mathrm{Diag}(\lambda(A))$ and $\lambda(A) = (\lambda_j : j \in \mathbb{N}_n)$ has components ordered in non-decreasing sense

$$\lambda_1(A) \geq \cdots \geq \lambda_n(A)\,.$$

In addition, for $x \in \mathbb{R}^r$, we shall use $|x|$ to denote the vector of absolute values $(|x_i| : i \in \mathbb{N}_r)$. Finally, for two vectors $x, y \in \mathbb{R}^r$ we write $x \leq y$ whenever, for all $i \in \mathbb{N}_r$, $x_i \leq y_i$.

4.2 Orthogonally Invariant Norms

A norm $\|\cdot\|$ on $M_{d,n}$ is called *orthogonally invariant* whenever, for every $U \in O_d$, $V \in O_n$ and $W \in M_{d,n}$, we have that

$$\|UWV^\top\| = \|W\|\,.$$

It is clear from the definition that an OI-norm that $\|\cdot\|$ is a spectral function. That is, for some function f, we have that $\|W\| = f(\sigma(W))$.

The remaining conditions on f which characterize OI-norms were given by von Neumann (1962) (see also Horn and Johnson, 1991, Section 3.5). He established that OI-norms are exactly *symmetric gauge functions* (SG-functions) of the singular values. To this end, we let $\mathcal{P}_r$ be the subset of $r \times r$ permutation matrices.

Definition 3 *A function $f : \mathbb{R}^r \to \mathbb{R}_+$ is called an SG-function whenever the following properties hold:*

1. *f is a norm on $\mathbb{R}^r$;*
2. *$f(x) = f(|x|)$ for all $x \in \mathbb{R}^r$;*
3. *$f(Px) = f(x)$ for all $x \in \mathbb{R}^r$ and all permutation matrices $P \in \mathcal{P}_r$.*

Property 2 states that f is *absolutely* or *gauge invariant*. Property 3 states that f is *symmetric* or *permutation invariant*. Hence, an SG-function is an absolutely symmetric norm.

Von Neumann's result is stated in the following theorem.

Theorem 4 *If $\|\cdot\|$ is an OI-norm on $M_{d,n}$ then there exists an SG-function $f:\mathbb{R}^r\to\mathbb{R}_+$ such that $\|W\|=f(\sigma(W))$, for all $W\in M_{d,n}$. Conversely, if $f:\mathbb{R}^r\to\mathbb{R}$ is an SG-function then the norm defined at $W\in M_{d,n}$, as $\|W\|=f(\sigma(W))$ is orthogonally invariant.*

The best known example of OI-norms are the *Schatten p-norms*, where $p\geq 1$, They are defined, for every $W\in M_{d,n}$, as

$$\|W\|_p=\left(\sum_{i\in\mathbb{N}_r}(\sigma_i(W))^p\right)^{\frac{1}{p}}$$

and, for $p=\infty$, as

$$\|W\|_\infty=\sigma_1(W).$$

The Schatten $1-$norm is sometimes called the *trace norm* or *nuclear norm*. Other common values of p give rise to the *Frobenius norm* ($p=2$) and the *spectral norm* ($p=\infty$). The Frobenius norm can also be written as $\sqrt{\mathrm{tr}W^\top W}$ and the spectral norm is alternatively expressed as $\max\{\|Wx\|_2:\|x\|_2=1\}$, where the subscript on the vector norm indicates the Euclidean norm of that vector. Another well-known family of OI-norms are the *Ky Fan norms* defined, for every $W\in M_{d,n}$ as

$$\|W\|_{(k)}=\sum_{i\in\mathbb{N}_k}\sigma_i(W)$$

where $1\leq k\leq r$ (the cases $k=1$ and $k=r$ are the spectral and trace norms, respectively). For more examples and for many interesting results involving OI-norms, we refer the reader to (Bhatia, 1997, Sec. IV.2) and (Horn and Johnson, 1991, Sec. 3.5).

We also mention, in passing, a formula from Argyriou et al. (2007b) which is useful for algorithmic developments. Specifically, we recall, for $p\in(0,2]$, that

$$\|W\|_p=\inf\left\{\langle WW^\top,D^{-\frac{2-p}{p}}\rangle:D\in\mathbf{S}^d_{++},\mathrm{tr}D\leq 1\right\}. \tag{10}$$

When $p\in[1,2]$, the function

$$(W,D)\mapsto\langle WW^\top,D^{-\frac{2-p}{p}}\rangle$$

is jointly convex in W and D and, so, the infimum in (10) is convex in W, in agreement with the convexity of the norm of $\|W\|_p$. Furthermore, if $WW^\top$ is invertible and $p\in(0,2]$, then the infimum is uniquely attained by the matrix

$$D=\frac{(WW^\top)^{\frac{p}{2}}}{\mathrm{tr}(WW^\top)^{\frac{p}{2}}}.$$

In machine learning practice, regularization with the trace norm has been proposed for collaborative filtering and multi-task learning (Abernethy et al., 2009; Argyriou et al., 2007a,b; Maurer, 2006a; Srebro et al., 2005, and references therein) and related problems (Yuan et al., 2007). If $\Omega(W)=\mathrm{rank}(W)$ the regularization problem (1) is non-convex. However, a common technique that overcomes this issue is to replace the rank by the trace norm (Fazel et al., 2001). The trace norm is the ℓ_1 norm on the singular values and hence there is an analogy to regularization of a vector

variable with the ℓ_1 norm, which is often used to obtain sparse solutions, see Candès and Recht (2008) and reference therein. In analogy to ℓ_1 regularization, it has recently been shown that for certain configurations of the input data the low rank solution can be recovered using the trace norm approach (Candès and Recht, 2008; Recht et al., 2008). More generally, regardless of the rank of the solution, it has been demonstrated that this approach allows for accurate joint estimation of multiple related tasks even when there are only *few* data points available for each task (Srebro et al., 2005; Argyriou et al., 2007a). One motivation is to approximate a matrix with a (possibly low-rank) factorization (Srebro et al., 2005). Another is that fitting multiple learning tasks simultaneously, so that they share a small set of orthogonal features, leads to a trace norm problem (Argyriou et al., 2007a).

The spectral norm, $\|\cdot\|_\infty$, is also of interest in the context of filter design (Zames, 1981) in control theory. Moreover, Schatten p-norms in the range $p \in [1,2]$ can be used for trading off sparsity of the model against task independence in multi-task learning (Argyriou et al., 2007b). In general, OI-norms are a natural class of regularizers to consider, since many matrix optimization problems can be posed in terms of the spectrum of the matrix.

We now proceed by reviewing some facts on duals of OI-norms. To this end, we first state a useful inequality, which can be found, for example, in (Horn and Johnson, 1991, ex. 3.3.10). This inequality also originates from von Neumann (1962) and is sometimes called *von Neumann's trace theorem* or *Ky Fan inequality*.

Lemma 5 *For any $X, Y \in M_{d,n}$, we have that*

$$\langle X, Y \rangle \leq \langle \sigma(X), \sigma(Y) \rangle \tag{11}$$

and equality holds if and only if there are $U \in O_d$ and $V \in O_n$ such that $X = U\Sigma(X)V^\top$ and $Y = U\Sigma(Y)V^\top$.

We emphasize that equality in (11) implies that the matrices X and Y admit the same ordered system of singular vectors, where the ordering is given by *ordering of the singular values*. It is also important to note that this inequality is stronger than the Cauchy-Schwarz inequality for the Frobenius norm, $\langle X, Y \rangle \leq \|X\|_2 \|Y\|_2$. Moreover, in the case of diagonal matrices one obtains a vector inequality due to Hardy et al. (1988)

$$\langle x, y \rangle \leq \langle [x], [y] \rangle ,$$

where $x, y \in \mathbb{R}^d$ and $[x]$ denotes the vector consisting of the components of x in non-increasing order. Let us also mention that apart from norm duality, Lemma 5 underlies many analytical properties of spectral functions, such as convexity, Fenchel conjugacy, subgradients and differentiability (see, for example, Lewis, 1995 for a review).

For our purposes, inequality (11) can be used to compute the dual of an OI-norm in terms of the dual of the corresponding SG-function. This is expressed in the following lemmas, which follow easily from (11) (see also Bhatia, 1997, Secs. IV.1, IV.2).

Lemma 6 *If the norm $\|\cdot\|$ on $M_{d,n}$ is orthogonally invariant and f is the corresponding SG-function, then the dual norm is given, for $W \in M_{d,n}$, by*

$$\|W\|_D = f_D(\sigma(W))$$

where $f_D : \mathbb{R}^r \to \mathbb{R}_+$ is the dual norm of f.

Lemma 7 *$\|\cdot\|$ is an OI-norm on $M_{d,n}$ if and only if $\|\cdot\|_D$ is orthogonally invariant. Also, $f:\mathbb{R}^r\to\mathbb{R}$ is an SG-function if and only if f_D is an SG-function.*

The next useful formula describes the peak set for any OI-norm.

Lemma 8 *Let $W\in M_{d,n}\setminus\{0\}$ and $W=U\Sigma(W)V^\top$ its singular value decomposition. If the norm $\|\cdot\|$ is orthogonally invariant and f is the corresponding SG-function, then*

$$\mathscr{D}\|W\| = \{Z : Z = U\Sigma(Z)V^\top, \sigma(Z)\in\mathscr{D}f(\sigma(W))\}.$$

Lemma 9 *Let $W\in M_{d,n}\setminus\{0\}$ and $W=U\Sigma(X)V^\top$ its singular value decomposition. If the norm $\|\cdot\|$ is orthogonally invariant and the corresponding SG-function f is* differentiable *at $\sigma(W)$, then $\|\cdot\|$ is differentiable at W and*

$$\nabla\|W\| = U\,\nabla f(\sigma(W))\,V^\top.$$

Lemma 10 *If f is an SG-function, $x\in\mathbb{R}^r\setminus\{0\}$ and $w\in\mathscr{D}f(x)$, then $\overline{w}\in\mathscr{D}f(\bar{x})$, where $\overline{w},\bar{x}\in\mathbb{R}^r$ are the vectors with elements the absolute values of w,x, respectively, in decreasing order. Moreover, $|w|,|x|$ can yield $\overline{w},\bar{x}$ with a simultaneous permutation of their elements.*

In other words, duality of OI-norms translates to duality of SG-functions. Norm duality preserves orthogonal invariance as well as the symmetric gauge properties. And dual pairs of matrices with respect to OI-norms directly relate to dual pairs of vectors with respect to SG-functions. Similarly, (sub)gradients of OI-norms correspond to (sub)gradients of SG-functions. In fact, Lemmas 8 and 9 hold, more generally, for all symmetric functions of the singular values (Lewis, 1995).

As an example, the dual of a Schatten p-norm $\|\cdot\|_p$ is the norm $\|\cdot\|_q$, where $\frac{1}{p}+\frac{1}{q}=1$. For $p>1$ and every $W=U\Sigma(W)V^\top\in M_{d,n}\setminus\{0\}$, one can readily obtain the set of duals from the equality conditions in Hölder's inequality. These give that

$$\mathscr{D}\|W\|_p = \left\{Z : Z = U\Sigma(Z)V^\top, \sigma_i(Z) = \frac{(\sigma_i(W))^{q-1}}{\|\sigma(W)\|_q^{q-1}}, i\in\mathbb{N}_r\right\}.$$

Moreover, this norm is differentiable for $p>1$ and the gradient is given by

$$\nabla\|W\|_p = U\,\mathrm{Diag}(\lambda)V^\top\frac{1}{\|\sigma(X)\|_p^{p-1}},$$

where $\lambda_i=(\sigma_i(W))^{p-1}, i\in\mathbb{N}_r$.

Before continuing to the main result about OI-norms, we briefly review the relation between *regularization* and *interpolation* problems, mentioned at the end of Section 2. We are interested in obtaining representer theorems and optimality conditions, in general, for regularization problems of the form (2). We shall focus, however, on representer theorems for interpolation problems of the form (3).

Let $\Omega: M_{d,n}\to\mathbb{R}$ be a given regularizer and assume that, for every $y\in\mathbb{R}^m$ and linear operator $I: M_{d,n}\to\mathbb{R}^m$, there exists some solution of (3) satisfying a prescribed representer theorem. Then, by Lemma 1, for every $y\in\mathbb{R}^m$, $I: M_{d,n}\to\mathbb{R}^m$ and $E:\mathbb{R}^m\times\mathbb{R}^m\to\mathbb{R}$, the same representer theorem holds for some solution of problem (2). In the remainder of the paper we shall prove optimality conditions for interpolation problems, which thus equally apply to regularization problems.

Conversely, the representer theorem for the regularization problem (2) associated with certain choices of the function Ω and E , will also hold for the corresponding interpolation problems (3). To illustrate this idea, we adopt a result from Argyriou et al. (2009), which concerns the standard representer theorem,

$$\hat{W} \in \mathcal{R}(I^*)\,.$$

Theorem 11 *Let $E : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ and $\Omega : M_{d,n} \to \mathbb{R}$ be a function with the following properties:*

(i) *E is lower semicontinuous and bounded from below;*

(ii) *Ω is lower semicontinuous and has bounded sublevel sets, that is, for every $\lambda \in \mathbb{R}$, the set $\{W : W \in M_{d,n}, \Omega(W) \leq \lambda\}$ is bounded;*

(iii) *for some $v \in \mathbb{R}^m \setminus \{0\}, y \in \mathbb{R}^m$, there exists a* unique *minimizer of $\min\{E(av, y) : a \in \mathbb{R}\}$ and this minimizer does not equal zero.*

If, for all choices of I and y, there exists a solution $\hat{W} \in \mathcal{R}(I^*)$ of (2)*, then, for all choices of I and y such that $y \in \mathcal{R}(I)$, there exists a solution $\hat{W} \in \mathcal{R}(I^*)$ of* (3)*.*

As noted in Argyriou et al. (2009), the square loss, hinge loss or logistic loss are all valid error functions in this theorem. The above results allow us to focus on the interpolation problems, as a devise to study the regularization problem.

We are now ready to describe the main result of this section, which concerns the form of the solution of interpolation problems (5) for the class of OI-norms.

Theorem 12 *Assume that $\|\cdot\|$ is an OI-norm and let f be the corresponding SG-function. If the matrix $\hat{W} \in M_{d,n} \setminus \{0\}$ is a solution of* (5) *and the vector $\hat{c} \in \mathbb{R}^m$ is a solution of* (7)*, then*

$$\hat{W} = U\,\Sigma(\hat{W})\,V^\top\,,\; I^*(\hat{c}) = U\,\Sigma(I^*(\hat{c}))\,V^\top$$

for some $U \in O^d, V \in O^n$, and

$$\sigma(\hat{W}) \in \frac{1}{\|I^*(\hat{c})\|_D}\,\partial f(\sigma(I^*(\hat{c})))\,.$$

Proof By Theorem 2 we obtain that $\|I^*(\hat{c})\|_D \hat{W} \in \partial\|I^*(\hat{c})\|$. We can write $I^*(\hat{c})$ in terms of its singular value decomposition, $I^*(\hat{c}) = U\,\Sigma(I^*(\hat{c}))\,V^\top$ with $U \in O^d, V \in O^n$. Using Lemma 8 we conclude that

$$\|I^*(\hat{c})\|_D \hat{W} = U\,(\|I^*(\hat{c})\|_D \Sigma(\hat{W}))\,V^\top,$$

where $\|I^*(\hat{c})\|_D\,\sigma(\hat{W}) \in \partial f(\sigma(I^*(\hat{c})))$. ■

This theorem implies that, in order to solve the minimal norm interpolation problem (5), we may first solve the dual problem (7) and then find a matrix in the peak set of $I^*(\hat{c})$ scaled by $1/\|I^*(\hat{c})\|_D$, which interpolates the data. The latter step in turn requires computing a singular value decomposition of $I^*(\hat{c})$ and then solving a non-linear system of equations. However, when the SG-function is smooth, there is a unique elements in the peak set and, so, there is no need to solve the non-linear equations. For example, if $\|\cdot\|$ is the Frobenius norm, Theorem 12 readily yields the standard representer theorem.

4.3 Admissible Orthogonally Invariant Norms

In this section, we define a subclass of OI-norms, which obey an improved version of the representer theorem presented above.

We begin with a definition.

Definition 13 *A norm $\|\cdot\|$ on $\mathbb{R}^r$ is said to be* admissible *if for any $x \in \mathbb{R}^r$, any $k \in \mathbb{N}_r$ such that $x_k \neq 0$ we have that*

$$\|x^k\| < \|x\|$$

where x^k is the vector all of whose components agree with x, except the k-th component which is zero.

The simplest example of admissible norms are the ℓ_p norm on $\mathbb{R}^d$, $\|\cdot\|_p$, for $p \in [1,\infty)$. From this norm we can form other admissible norms in various ways. Specifically, for any $p_1, p_2 \in [1,\infty)$, we see that the norm $\|\cdot\|_{p_1} + \|\cdot\|_{p_2}$ or the norm $\max\{\|\cdot\|_{p_1}, \|\cdot\|_{p_2}\}$ are both admissible. Note that some of these norms are not strictly convex. Also compare this definition to that of weakly monotone norms (Horn and Johnson, 1985, Def. 5.5.13).

Lemma 14 *If $\|\cdot\|$ is an admissible norm on $\mathbb{R}^r$, $x \in \mathbb{R}^r \backslash \{0\}$ and $w \in \mathscr{D}\|x\|$, then for any $k \in \mathbb{N}_r$ with $x_k = 0$ it holds that $w_k = 0$.*

Conversely, assume that, for every $x \in \mathbb{R}^r \backslash \{0\}$, $w \in \mathscr{D}\|x\|$ and $k \in \mathbb{N}_r$, if $x_k = 0$ it holds that $w_k = 0$. Then $\|\cdot\|$ is admissible.

Proof Let $w \in \mathscr{D}\|x\|$, where $x \in \mathbb{R}^r \backslash \{0\}$, with $x_k = 0$. Suppose to the contrary that $w_k \neq 0$. Since $\|\cdot\|$ is admissible it follows that $\|w^k\| < \|w\|$, and so, we get that $\|w^k\| < 1$, because $\|w\| = 1$. However, we also have that

$$\|x\|_D = \langle w, x\rangle = \langle w^k, x\rangle \leq \|w^k\|\|x\|_D$$

from which it follows that $\|w^k\| \geq 1$. This proves the first part of the claim.

For the converse, we consider a $w \in \mathbb{R}^r \backslash \{0\}$ with $w_k \neq 0$. We shall show that $\|w^k\| < \|w\|$. To this end, we choose $x \in \mathscr{D}\|w^k\|_D$ and then we choose $y \in \mathscr{D}\|x^k\|$. By our hypothesis, we conclude that $y_k = 0$ and by our choice we have, in particular that $1 = \|y\| = \|x\|_D$. Consequently, it follows that

$$\|x^k\|_D = \langle y, x^k\rangle = \langle y, x\rangle \leq \|y\|\|x\|_D = 1$$

from which conclude that

$$\|w^k\| = \langle w^k, x\rangle = \langle w, x^k\rangle \leq \|w\|. \tag{12}$$

Moreover, if equality holds in this inequality it would follow that $\frac{w}{\|w\|} \in \mathscr{D}\|x^k\|$. But then, we can invoke our hypothesis once again and obtain a contradiction. That is, inequality (12) is strict and therefore $\|\cdot\|$ is an admissible norm, as asserted. ■

The above observation leads us to consider the following subclass of OI-norms.

Definition 15 *A norm $\|\cdot\|$ on $M_{d,n}$ is said to be* admissible orthogonally invariant *if there exists an admissible vector norm $\|\cdot\|$ on $\mathbb{R}^r$ such that, for every $W \in M_{d,n}$, we have that $\|W\| = \|\sigma(W)\|$.*

Examples of non-admissible OI-norms are the spectral norm, the Ky Fan norms $\|\cdot\|_{(k)}$ for $1 \leq k < r$ and the norm $\max\{\|\cdot\|_1, \alpha\|\cdot\|_\infty\}$ for $\alpha \in (1,\infty)$.

We have now accumulated sufficient information on admissible OI-norms to present an improved representer theorem for problem (5). We shall prove below, for any admissible OI-norm, $\hat{W}$ can be expressed as

$$\hat{W} = \sum_{i \in \mathbb{N}_m} \hat{c}_i X_i R.$$

In other words, $\hat{W}$ is obtained by first applying the standard representer theorem and then multiplying it from the right by the matrix R. In the case of the Frobenius norm $R = I_n$.

Theorem 16 *If $\|\cdot\|$ is admissible orthogonally invariant, the matrix $\hat{W} \in M_{d,n} \setminus \{0\}$ is a solution of* (5) *and the vector $\hat{c} \in \mathbb{R}^m$ is a solution of* (7)*, then there exists a matrix $R \in \mathbf{S}^n_+$ such that*

$$\hat{W} = I^*(\hat{c})R \tag{13}$$

and the eigenvectors of R are right singular vectors of $I^*(\hat{c})$.

Proof By Theorem 12, there exists $I^*(\hat{c}) = U\Sigma(I^*(\hat{c}))V^\top$, obtained from a dual solution $\hat{c}$ of (7), such that $\|I^*(\hat{c})\|_D \hat{W} = U\,\mathrm{Diag}(\lambda)V^\top$, where $\lambda \in \partial f(\sigma(I^*(\hat{c})))$ and f is the SG-function associated with $\|\cdot\|$. Since f is admissible, Lemma 14 implies that $\lambda_i = 0$ whenever $\sigma_i(I^*(\hat{c})) = 0$. Hence there exists $\mu \in \mathbb{R}^r_+$ such that $\lambda_i = \sigma_i(I^*(\hat{c}))\mu_i$, $i \in \mathbb{N}_r$, and $\mu_i = 0$ whenever $\sigma_i(I^*(\hat{c})) = 0$. Thus, $\|I^*(\hat{c})\|_D \hat{W} = U\Sigma(I^*(\hat{c}))V^\top V\,\mathrm{Diag}(\mu)V^\top$ and the corollary follows by selecting $R = \dfrac{1}{\|I^*(\hat{c})\|_D} V\,\mathrm{Diag}(\mu)V^\top$. ■

Note that, in the above theorem, the eigenvectors of R need not correspond to right singular vectors of $I^*(\hat{c})$ according to a simultaneous ordering of the eigenvalues / singular values.

We may also state a converse of Theorem 16, that is, the only OI-norms which satisfy property (13) are admissible.

Theorem 17 *If $\|\cdot\|$ is orthogonally invariant and condition* (13) *holds (without any conditions on $R \in M_{n,n}$), for every linear operator $I : M_{d,n} \to \mathbb{R}^m$, $y \in \mathcal{R}(I) \setminus \{0\}$, every solution $\hat{W}$ of* (5) *and every solution $\hat{c}$ of* (7)*, then the norm $\|\cdot\|$ is admissible orthogonally invariant.*

Proof Let f be the SG-function corresponding to $\|\cdot\|$. Let arbitrary $x \in \mathbb{R}^r \setminus \{0\}$ and $w \in \partial f(x)$. Define $\overline{x}, \overline{w} \in \mathbb{R}^r_+$ to be the vectors with elements the absolute values of x, w, respectively, in descending order. By Lemma 10, we obtain that $\overline{w} \in \partial f(\overline{x})$. Define also $X = \mathrm{Diag}(\overline{x}) \in M_{d,n}$ and $W = \mathrm{Diag}(\overline{w}) \in M_{d,n}$. By Lemma 8, we obtain that $W \in \partial\|X\|$. Now, consider the problem $\min\{\|Z\| : Z \in M_{d,n}, \langle Z, X\rangle = \|X\|_D\}$, whose set of solutions is $\partial\|X\|$. By hypothesis, $W = cXR$ for some $R \in M_{n,n}$ and for $c = \frac{1}{\|X\|_D}$ (the only solution of the dual problem). Therefore, $\mathrm{Diag}(\overline{w}) = c\,\mathrm{Diag}(\overline{x})R$ and hence $\overline{x}_k = 0$ implies $\overline{w}_k = 0$, for all $k \in \mathbb{N}_r$. By Lemma 10, this implies in turn that $w_k = 0$ if $x_k = 0$, for all $k \in \mathbb{N}_r$. Combining with Lemma 14, we deduce that f is admissible, as required. ■

We remark that there exist norms on $M_{d,n}$ which are not orthogonally invariant and satisfy condition (13). In fact, given two non-singular matrices $Q \in M_{d,d}$ and $M \in M_{n,n}$, the norm $W \mapsto \|QWM\|_2$ is not orthogonally invariant, but the representer theorem is easily seen to be

$$\hat{W} = (Q^\top Q)^{-1} I^*(\hat{c})(MM^\top)^{-1}.$$

Furthermore, concerning the converse of Theorem 16, it can be shown that if we restrict the eigenvectors of R to be right singular vectors of $I^*(\hat{c})$, then $\|\cdot\|$ has to be orthogonally invariant.

Moreover, if the norm $\|\cdot\|$ is not admissible, then it can be shown that for every solution $\hat{c}$ of the dual problem there exists a solution of the primal satisfying (13). As an example, see Corollary 20 below (spectral norm). For a characterization of functions yielding such representer theorems, see Argyriou et al. (2009).

Returning to Theorem 12, for the Schatten p-norms we have the following corollary. To state it, we use the notation A^{q-1} as a shorthand for the matrix $U \operatorname{Diag}(\sigma_i(A)^{q-1})_{i\in\mathbb{N}_r} V^\top$ when $A = U\Sigma(A)V^\top$.

Corollary 18 *If the matrix $\hat{W} \in M_{d,n} \setminus \{0\}$ is a solution of* (5) *for the Schatten $p-$norm, with $p \in (1,\infty)$, then there exists a vector $\hat{c} \in \mathbb{R}^m$ such that*

$$\hat{W} = \frac{I^*(\hat{c})^{q-1}}{\|I^*(\hat{c})\|_q^q},$$

where $\frac{1}{p} + \frac{1}{q} = 1$.

Proof The corollary follows directly from Theorem 12 and the description of $\partial\|\cdot\|_p$ in Section 4.2. ■

The above corollary does not cover the cases that $p = 1$ or $p = \infty$. We state them separately.

Corollary 19 *If $\hat{W} \in M_{d,n} \setminus \{0\}$ is a solution of* (5) *for the trace norm, $\hat{c} \in \mathbb{R}^m$ a solution of* (7) *and $I^*(\hat{c}) = \sum_{i\in\mathbb{N}_r} \sigma_i(I^*(\hat{c}))u_i v_i^\top$ is a singular value decomposition, then*

$$\hat{W} = \frac{1}{\sigma_1(I^*(\hat{c}))} \sum_{i\in\mathbb{N}_{r_{\max}}} \lambda_i u_i v_i^\top,$$

for some $\lambda_i \geq 0, i \in \mathbb{N}_{r_{\max}}$ such that $\sum_{i\in\mathbb{N}_{r_{\max}}} \lambda_i = 1$, where $r_{\max}$ is the multiplicity of the largest singular value $\sigma_1(I^*(\hat{c}))$. Moreover, $\hat{W} = I^*(\hat{c})R$, where

$$R = \frac{1}{\sigma_1^2(I^*(\hat{c}))} \sum_{i\in\mathbb{N}_{r_{\max}}} v_i v_i^\top.$$

Proof The corollary follows from Theorem 12 and the description of $\partial\|\cdot\|_1$. From the definition, it is easy to obtain that, for every $x \in \mathbb{R}^r_+$, $\partial\|x\|_1 = \{y \in \mathbb{R}^r_+ : y_i = 0, \text{ if } x_i < \|x\|_\infty, \sum_{i\in\mathbb{N}_r} y_i = 1\}$. Thus, $\sigma_1(I^*(\hat{c}))\hat{W} = \|I^*(\hat{c})\|_\infty \hat{W} = U\Lambda V^\top$, for $\Lambda = \operatorname{Diag}(\lambda)$ and $\lambda_i = 0$ for $i > r_{\max}$, $\sum_{i\in\mathbb{N}_{r_{\max}}} \lambda_i = 1$. Since $\Lambda = \frac{1}{\sigma_1(I^*(\hat{c}))}\Sigma(I^*(\hat{c}))\Lambda$, R can be selected as $\frac{1}{\sigma_1^2(I^*(\hat{c}))} V\Lambda V^\top$. ■

Corollary 20 *If the matrix $\hat{W} \in M_{d,n} \setminus \{0\}$ is a solution of* (5) *for the spectral norm, $\hat{c} \in \mathbb{R}^m$ a solution of* (7) *and $I^*(\hat{c}) = \sum_{i \in \mathbb{N}_r} \sigma_i(I^*(\hat{c}))u_i v_i^\top$ is a singular value decomposition, then*

$$\hat{W} = \frac{1}{\|I^*(\hat{c})\|_1} \sum_{i=1}^{\operatorname{rank}(I^*(\hat{c}))} u_i v_i^\top + \sum_{i=\operatorname{rank}(I^*(\hat{c}))+1}^{r} \alpha_i u_i v_i^\top , \tag{14}$$

for some $\alpha_i \in [0,1]$, $i = \operatorname{rank}(I^*(\hat{c})) + 1, \ldots, r$.

Proof The corollary follows from Theorem 12 and the fact that, for every $x \in \mathbb{R}^r_+$, $\mathscr{D}\|x\|_\infty = \{y \in [-1,1]^r : y_i = 1, \text{ if } x_i > 0\}$. ■

The above corollary also confirms that representation (13) does not apply to the spectral norm (which is not admissible orthogonally invariant). Indeed, from (14) it is clear that the range of $\hat{W}$ can be a superset of the range of $I^*(\hat{c})$.

To recapitulate the results presented in this section, Theorem 12 allows one to obtain the solutions of the primal minimum norm interpolation problem (5) from those of its dual problem (7), which involves m variables. This is true for *all OI-norms*, even though the representer theorem in the form (13) applies only to admissible OI-norms. Part of the appeal of OI-norms is that computing primal solutions from dual ones reduces to a vector norm optimization problem. Indeed, given a solution of the dual problem, one just needs to compute the *singular value decomposition* of the matrix $I^*(\hat{c})$ and *the peak set of the SG-function f at the singular values*. The associated primal solutions are then easily obtained by keeping the same row and column spaces and using elements of the peak set in place of the singular values. In fact, in many cases, the latter problem of computing the peak set of f may be straightforward. For example, if f_D is differentiable (except at zero), each dual solution is associated with a single primal one, which equals a multiple of the gradient of f_D at the dual solution.

4.4 Related Work

The results of Section 4 are related to other prior work, besides the already mentioned literature on representer theorems for the case of the vector L_2 norm (that is, for $n = 1$). In particular, the representer theorem for the trace norm (Corollary 19) has been stated in Srebro et al. (2005). Also, the representation (13) in Theorem 16 relates to the representer theorems proven in Argyriou et al. (2009); Abernethy et al. (2009). The results in Abernethy et al. (2009) apply to the case of the trace norm and when the X_i are rank one matrices. The results in Argyriou et al. (2009) give representer theorems for a broad class of functions, of which differentiable OI-norms are members. However, as mentioned before, Theorem 16 requires additional conditions on matrix R. In particular, the requirement on the eigenvectors of this matrix holds only for admissible OI-norms.

5. Conclusion and Future Work

We have characterized the form of the solution of regularization with an orthogonally invariant penalty term. Our result depends upon a detailed analysis of the corresponding minimal norm interpolation problem. In particular, we have derived a dual problem of the minimal norm interpolation problem and established the relationship between these two problems. The dual problem involves optimization over a vector of parameters whose size equals the number of data points. In practical circumstances, this number may be smaller than the dimension of the matrix we seek, thus our result

should prove useful in the development of optimization algorithms for orthogonally invariant norm regularization. For example, one could combine our result with Lemma 9 in order to implement gradient methods for solving the dual problem. Note however that the dual problem involves a singular value decomposition, and more effort is needed in elucidating the algorithmic implications of the results presented here.

Acknowledgments

The work of the first author and third author was partially supported by EPSRC Grant EP/D071542/1. The work of the second author was supported by NSF Grant ITR-0312113 and Air Force Grant AFOSR-FA9550.

Appendix A.

Here, we describe two results which we have used in the paper. Recall that for every linear operator $J : M_{d,n} \to \mathbb{R}^k$, the linear spaces $\mathcal{R}(J)$ and $\mathcal{N}(J)$ denote the range and the kernel of J, respectively.

Lemma 21 *Let $\mathcal{W}$ be a nonempty, convex and compact subset of $M_{d,n}$ and let $J : M_{d,n} \to \mathbb{R}^k$ be a linear operator. The set $\mathcal{R}(J^*)$ intersects $\mathcal{W}$ if and only if, for every $X \in \mathcal{N}(J)$ the following inequality holds*

$$\max\{\langle X, W\rangle : W \in \mathcal{W}\} \geq 0. \tag{15}$$

Proof Suppose that there exist $z \in \mathbb{R}^k$ and $T \in \mathcal{W}$ such that $J^*(z) = T$. Then for any $X \in M_{d,n}$ with $J(X) = 0$ we have that $\langle X, T\rangle = 0$ and, so, inequality (15) holds true.

Now, suppose that $\mathcal{R}(J^*) \cap \mathcal{W} = \emptyset$. Then, there is a hyperplane which strictly separates $\mathcal{R}(J^*)$ from $\mathcal{W}$ (see, for example, Rockafellar, 1970, Cor. 11.4.2). That is, there exist $W_0 \in M_{d,n}$ and $\mu \in \mathbb{R}$ such that, for all $z \in \mathbb{R}^k$,

$$\langle W_0, J^*(z)\rangle + \mu \geq 0,$$

while, for all $W \in \mathcal{W}$,

$$\langle W_0, W\rangle + \mu < 0.$$

The first inequality implies that $J(W_0) = 0$. To see this, we choose any $z_0 \in \mathbb{R}^k$ and $\lambda \in \mathbb{R}$ and let $z = \lambda z_0$ in the first inequality. Now, we allow $\lambda \to \pm\infty$, to obtain that $\langle W_0, J^*(z_0)\rangle = 0$. Therefore, the first inequality simplifies to the statement that $\mu \geq 0$.

The second inequality implies that

$$\max\{\langle W_0, W\rangle : W \in \mathcal{W}\} < -\mu \leq 0,$$

which contradicts (15) and proves the result. ■

Next, we state an important rule for taking directional derivatives of a convex function expressed as a maximum of a family of convex functions. For this purpose, recall that the right directional derivative of a function $g : \mathcal{W} \to \mathbb{R}$ in the direction Δ at $W \in \mathcal{W}$ is defined as

$$g'_+(W;\Delta) = \lim_{\lambda \to 0^+} \frac{g(W + \lambda\Delta) - g(W)}{\lambda}.$$

Theorem 22 *Let $\mathcal{W}$ be a convex subset and $\mathcal{X}$ a compact subset of $M_{d,n}$ and $f : \mathcal{W} \times \mathcal{X} \to \mathbb{R}$. If, for every $W \in \mathcal{W}$, the function $X \mapsto f(W,X)$ is continuous on $\mathcal{X}$ and, for every $X \in \mathcal{X}$ the function $W \mapsto f(W,X)$ is convex on $\mathcal{W}$, then the convex function $g : \mathcal{W} \to \mathbb{R}$ defined at $W \in \mathcal{W}$ as*

$$g(W) := \max\{f(W,X) : X \in \mathcal{X}\}$$

has a right directional derivative at W in the direction $\Delta \in M_{d,n}$, given as

$$g'_+(W;\Delta) = \max\{f'_+(W;\Delta,X) : X \in M(W)\}, \tag{16}$$

where $M(W) = \{X : X \in \mathcal{X}, f(W,X) = g(W)\}$.

References

J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009.

Y. Amit and M. Fink and N. Srebro and S. Ullman. Uncovering shared structures in multiclass classification, In *Proceedings of the Twenty-Fourth International Conference on Machine Learning,* 2007.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, 2007b.

A. Argyriou, C.A. Micchelli, and M. Pontil. When is there a representer theorem? Vector versus matrix regularizers. *Journal of Machine Learning Research*, 10:2507-2529, 2009.

R. Bhatia. *Matrix Analysis*. Graduate Texts in Mathematics. Springer, 1997.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, 2005.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9:717-772, 2008.

G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, 2008.

T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proceedings, American Control Conference*, volume 6, pages 4734–4739, 2001.

G. H. Hardy, J. E. Littlewood, and G. Pólya. *Inequalities*. Cambridge University Press, 1988.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

A. J. Izenman. Reduced-rank regression for the multivariate linear model. *Journal of Multivariate Analysis*, 5:248–264, 1975.

A. S. Lewis. The convex analysis of unitarily invariant matrix functions. *Journal of Convex Analysis*, 2(1/2):173–183, 1995.

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, 2006a.

A. Maurer. Learning similarity with operator-valued large-margin classifier. *Journal of Machine Learning Research*, 9:1049–1082, 2008a.

C. A. Micchelli and A. Pinkus. Variational problems arising from balancing several error criteria. *Rendiconti di Matematica, Serie VII*, 14:37–86, 1994.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. Preprint, 2008.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

B. Schölkopf, R. Herbrich, and A.J. Smola. A generalized representer theorem. In *Proceedings of the Fourteenth Annual Conference on Computational Learning Theory*, 2001.

N. Srebro, J. D. M. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, pages 1329–1336. MIT Press, 2005.

J. von Neumann. Some matrix inequalities and metrization of matric-space. In *Tomsk University Review, 1:286-300, 1937*, volume IV of *Collected Works*, pages 205–218. Pergamon, Oxford, 1962.

G. Wahba. *Spline Models for Observational Data*, volume 59 of *Series in Applied Mathematics*. SIAM, Philadelphia, 1990.

M. Yuan, A. Ekici, Z. Lu, and R. Monteiro. Dimension reduction and coefficient estimation in multivariate linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):329–346, 2007.

G. Zames. Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses. *IEEE Transactions on Automatic Control*, 26(2):301–320, 1981.

Journal of Machine Learning Research 11 (2010) 955-984 Submitted 7/08; Revised 10/09; Published 2/10

Generalized Expectation Criteria for Semi-Supervised Learning with Weakly Labeled Data

Gideon S. Mann* GIDEON.MANN@GMAIL.COM
Google Inc.
76 9th Avenue
New York, NY 10011

Andrew McCallum MCCALLUM@CS.UMASS.EDU
Department of Computer Science
140 Governors Drive
University of Massachusetts, Amherst, 01003

Editor: Michael Collins

Abstract

In this paper, we present an overview of *generalized expectation criteria (GE)*, a simple, robust, scalable method for semi-supervised training using weakly-labeled data. GE fits model parameters by favoring models that match certain expectation constraints, such as marginal label distributions, on the unlabeled data. This paper shows how to apply generalized expectation criteria to two classes of parametric models: maximum entropy models and conditional random fields. Experimental results demonstrate accuracy improvements over supervised training and a number of other state-of-the-art semi-supervised learning methods for these models.

Keywords: generalized expectation criteria, semi-supervised learning, logistic regression, conditional random fields

1. Introduction

Semi-supervised learning, where a small amount of human annotation is combined with a large amount of unlabeled data to yield an accurate classifier, has received a significant amount of attention from the research community. However, there are surprisingly few cases of its use in applications, where the emphasis is on solving a task, not on advancing theoretical understanding. This may be partially due to the natural time it takes for new machine learning ideas to propagate to practitioners, but we believe it is also due in large part to the inherent difficulty of the task and the unreliability of existing methods.

Instead of addressing the difficulties of semi-supervised learning head-on, we propose to use weakly labeled data ("side-information") in semi-supervised learning. To use this data, we present *generalized expectation critera (GE)*, a method initially described as *expectation regularization* in Mann and McCallum (2007). GE represents a new family of semi-supervised learning, where models are fit by minimizing model divergence from an input distribution. To calculate the divergence from the input distribution there is no need for additional training data, as the expected distribution on the unlabeled data can be used. These terms can be easily integrated with other terms, such as traditional log-likehood.

*. Gideon S. Mann is the corresponding author.

The experiments in this paper explore an illustrative special case of GE, *label regularization*, where a marginal distribution over output labels is applied as an expectation constraint. We investigate two parametric models: maximum entropy models and their structured output analog, conditional random fields. We demonstrate that for both of these models, label regularization is able to provide performance gains over other supervised and semi-supervised learning methods.

Generalized expectation criteria have a number of advantages over alternative semi-supervised learning techniques that make it suitable for use in practice. It is simple, making it easy to implement and use. It requires no additional processing such as constructing an inverted index for graph construction or pre-clustering unlabeled data. Since it can handle a wide spectrum of side-information, human intuitions about the problem can be explicitly communicated to the learning process, in contrast to other methods which can require opaque supervision, such as carefully tuned initialization, or specification of "contrastive examples" and other "auxiliary functions". We apply GE in this paper to discriminative models, and thus it is able to robustly handle overlapping, non-independent feature sets and yields transparent confidence estimates (in the form of probabilities). Additionally, GE has all of the advantages of parametric models, in particular scalability and a small memory footprint at test time.

2. Related Work

Traditional *supervised learning* takes as input fully labeled data, a set of tuples $\mathcal{D} = \{(x, y)\}$, where x is the input and y the desired output. The learner is a function which maps the input to a predictive function: $J(\mathcal{D}) = f$, where $f : x \rightarrow y$. Supervised learning is powerful, but the amount of labeled data needed can require significant human time and effort to create. In an effort to reduce the need for human effort, the machine learning community has explored semi-supervised learning. In *semi-supervised learning* approaches, a small amount of labeled data is augmented by unlabeled data, a set of elements $\mathcal{U} = \{x\}$, which it exploits to chose a similar function: $J(\mathcal{D} \cup \mathcal{U}) = f$. This section presents the main methods for semi-supervised learning from labeled and unlabeled data: 1) bootstrapping, 2) expectation maximization, 3) feature discovery, 4) decision boundaries in sparse region methods, and 5) graph-based methods.

In contrast to these methods, GE criteria exploit semi-supervised learning from weakly labeled data. With this scenario, there be no labeled data. Instead, in addition to unlabeled data there is side information that has been provided to the learner, for example, expectation constraints like marginal label distributions $\tilde{g}_y = p(y)$.[1] Section 2.2 reviews prior work in this area.

2.1 Semi-supervised Learning with Labeled and Unlabeled Instances

There are five main prior categories of semi-supervised learning approaches: bootstrapping, expectation maximization, feature discovery, decision boundaries in sparse regions, and graph-based methods.

2.1.1 BOOTSTRAPPING

In bootstrapping, or self-training approaches, a classifier is first trained on the fully labeled instances and then is applied to unlabeled instances. Some subset of those newly labeled instances are then used (in conjunction with the original labeled instances) to retrain the model.

1. Liang et al. (2009) presents a taxonomy of side-information.

Algorithm 1 Bootstrapping for semi-supervised learning

$f^{(0)} \leftarrow J(\mathcal{D})$
repeat
  $\mathcal{U}_B \leftarrow \cup_{x_i \in \mathcal{U}}(x_i, f^{(t-1)}(x_i))$
  $f^{(t)} \leftarrow J(\mathcal{D} \cup \mathcal{U}_B)$
until done

One of the most successful examples of this work is Yarowsky (1995), where a small 'seed set' of labeled instances is incrementally augmented. Co-training (Blum and Mitchell, 1998) looks at the case where two complementary classifiers can both be applied to a particular problem. Abney (2004) provides a deeper understanding of these methods by demonstrating that they optimize a natural objective function. However, these methods typically require continual human intervention in order to avoid performance loss during the bootstrapping process, such as in Riloff and Shepherd (2000).

2.1.2 EXPECTATION MAXIMIZATION

Generative models trained by expectation maximization (Dempster et al., 1977) have been widely studied for semi-supervised learning. EM consists of two steps: an expectation step $Q(\theta|\theta^{(t)}) = E_{p(y|x,\theta^{(t)}}[\log L(\theta;x,y)]$, and a maximization step, $\theta^{(t+1)} = \texttt{argmax}_\theta\, Q(\theta|\theta^{(t)})$. To apply EM to semi-supervised data, the log-likelihood function, $\log L(\theta;x,y)$, is set to be a composite of labeled and unlabeled data (possibly with a weighting factor to down-weight the contribution to the likelihood from the unlabeled data). One popular example of the use of EM for generative models is Nigam et al. (1998) which presents a naïve Bayes model for text classification trained using EM and semi-supervised data.

EM has also been applied to structured classification problems such as part-of-speech tagging (Klein and Manning, 2004), where EM can succeed after very careful and clever initialization. While these models can often be very effective, especially when used with "prototypes" (Haghighi and Klein, 2006b), they cannot efficiently accommodate non-independent features, for example, those that span multiple inputs. In these cases, the dynamic program required to compute the feature expectations over all input positions quickly becomes intractable, as building the lattice requires exponential space in the length of input features.

EM for discriminative models has also been explored. Wang et al. (2002) proposes a EM based model which instead of the likelihood, maximizes the entropy given the latent variables. Alternatively, Salakhutdinov et al. (2003) present an expected gradient method, which can be used for discriminative models which have some partially observed labels, and McCallum et al. (2005) uses this method for conditionally trained CRFs. While this method is appealing in the case of input data that consists of sequences with partially hidden variables, it cannot be applied to scenarios with fully unlabeled instances.

Another twist on this technique is to blend generative and discriminative models, by combining ML estimates over the labeled data with EM parameter estimates over the unlabeled data, for a joint model which combines a CRF and a HMM (Suzuki et al., 2007; Suzuki and Isozaki, 2008). In this formulation, the log-likelihood can be viewed as two separate log-likelihood functions $L_1(\theta)$ and $L_2(\theta)$ which respectively correspond to the CRF and HMM log-likelihood. When optimizing

$L_1(\theta)$ (using maximum likelihood), the HMM parameters are held constant, and the reverse when optimizing $L_2(\theta)$.

While EM can sometimes works well, it is often fragile and finds solutions that are worse than the equivalent supervised model. Merialdo (1994) gives a classic example where EM fails to help part-of-speech tagging. Cozman and Cohen (2006) discuss the risks of using EM and describe situations where it can fail. Additionally, the generative models on which EM depends often perform worse than discriminative models.

2.1.3 FEATURE DISCOVERY

As alternative to estimating a classifier directly with unlabeled data, a number of groups have explored using the unlabeled data for feature induction or feature discovery, and those features are then embedded into a traditional supervised learning problem. A latent clustering is applied over the unlabeled data, to learn a function f_l, which is used to provide additional features $f_l(x) = z$, these features are then used to augment the original labeled data: $\bar{\mathcal{D}} = \cup_{(x_d,y_d)\in\mathcal{D}}(x_d \cup z_d, y_d)$, and then supervised learning proceeds as usual. For example, Miller et al. (2004) and Ganchev et al. (2007) apply the method described in Brown et al. (1992) to cluster all of the word tokens in a large unsupervised corpus. Then for a given sentence, in addition to standard features, additional features corresponding to the latent clusters of the tokens in the sentence, are added. This technique, along with similar approachs (Freitag, 2004; Li and McCallum, 2005), have yielded small but consistent success. This method can be applied independently of the particular training method and in Section 6.3, we explore combining our method with those described by Miller et al. (2004).

Ando and Zhang (2005) use a similar method, but the clustering they explore is composed of auxiliary problems (e.g., predict a given token given the token context). In their method, they estimate the parameters for a linear classifier for each auxiliary problem, and then these parameters are embedded as a transformation of the parameters for a linear classifier for the original problem. Although this method has produced impressive gains, it is quite sensitive to the selection of auxiliary information, and making good selections requires significant insight. F. Pereira and J. Blitzer.[2] note that the list of tricks necessary to get the method of Ando and Zhang (2005) to work includes: oversampling positive instances, selecing the unlabeled data carefully, scaling real-valued features, and choosing the appropriate feature types.

Additionally, feature discovery as a semi-supervised learning technique relies on having a substantial amount of labeled data for training. It cannot be used in cases where only a limited amount of labeled data is available.

2.1.4 DECISION BOUNDARIES IN SPARSE REGIONS

Another family of methods uses the intuition that decision boundaries ought to fall in low-density regions (corresponding to an assumption of class separability) and thus fit discriminative models with this objective in mind. Clearly, if the cluster assumption is violated (i.e., the classes are not widely separable), assigning decision boundaries to low density regions is a poor choice. One illustrative example is entropy regularization (Grandvalet and Bengio, 2004), where a traditional conditional label log-likelihood objective function is augmented with an additional term that minimizes the

2. Conveyed in personal communication.

entropy of the label distribution on the unlabeled data:

$$O(\theta;\mathcal{D},\mathcal{U}) = L(\theta;\mathcal{D}) + H(\theta;\mathcal{U}) \\ = \sum_{d\in\mathcal{D}} \log p(y_d|x_d) - \lambda \sum_{u\in\mathcal{U}} \sum_{y} p(y|x_u) \log p(y|x_u) \,.$$

This objective function favors parameter settings where the model is certain of the labels on given unlabeled data. In entropy regularization, the hyper-parameter λ has a dramatic effect on the performance of the learner, since it must be tuned with regards to the amount of labeled and unlabeled data. Entropy regularization is particularly difficult to apply in cases of very small amounts of labeled data, since in one degenerate case, the model could select one output label for all possible inputs. Studies on structured output models in Jiao et al. (2006) experimentally demonstrate that careful tuning of λ is mandatory.

Transductive support vector machines (TSVMs) (Joachims, 1999) add a constraint to the SVM optimization function in order to preserve the margin over unknown test labels:

$$(\{y_u\},\theta) = \underset{\{y_u\},\theta}{\texttt{argmin}} \frac{1}{2}||\theta||^2 \quad \textit{subject to} \begin{cases} \forall_{x_i\in\mathcal{D}} & y_i[\theta\cdot x_i + b] \geq 1 \\ \forall_{x_u\in\mathcal{D}} & y_u[\theta\cdot x_u + b] \geq 1 \end{cases} .$$

It is combinatorially intractable to do a brute-force search over all possible labelings $\{y_u\}$, so an approximation search must be undertaken. Even with these approximations, the algorithm as originally proposed has running time $O(n^3)$. Sindhwani and Keerthi (2006) propose a method for speeding up training in some cases. In our experience, like entropy regularization, TSVMs also require extensive and delicate tuning of meta-parameters. We note that Sindhwani and Keerthi (2006) report results with meta-parameters tuned on test data. Benchmark tests have shown that entropy regularization performs as well as TSVMs (when the SVM is given a linear kernel) (Chapelle et al., 2006). Another related method is information regularization (Corduneanu and Jaakkola, 2003), which measures distance via the mutual information between a classifier and the marginal distribution $p(x)$.

2.1.5 GRAPH-BASED METHODS

Graph-based (manifold) methods can be very accurate when applied to semi-supervised learning. In these methods, a graph, typically with weighted edges, is constructed spanning the labeled and unlabeled instances. Thereafter, unlabeled instances are assigned labels according to their neighbors. Zhu and Ghahramani (2002) propose label propagation, where labels propagate from labeled instances to unlabeled instances (see Algorithm 2). In this formulation, there are two significant

Algorithm 2 Label propagation

repeat

$\forall_{x_u\in\mathcal{U}}:\ p(y_u^{(t)}|x_u^{(t)}) = \frac{1}{Z}\sum_{j\in\mathcal{N}(x_u)} q(j\rightarrow u) p(y_j^{(t-1)}|x_j)$

until $p(y_u^{(t)}|x_u^{(t)})$ converges

choices that must be made: the graph structure (the neighborhood $\mathcal{N}(x)$ for each instance) and the transition function $q(j\rightarrow i)$. Szummer and Jaakkola (2002) present a closely related approach which uses random walks through the graph to assign labels. More distantly related, Li and McCallum (2004) examine a method which performs an implicit clustering over points, as it simultaneously estimates pair-wise distance and classification boundaries.

As originally proposed, graph-based methods are slow, requiring time $O(n^3)$ or on average $O(kn^2)$ where k is the number of neighbors (similar to TSVMs). By sub-sampling unlabeled data, one can reduce run-time from $O(n^3)$ to $O(m^2n)$, where m is the subsampled number of unlabeled data points (Delalleau et al., 2006), but subsampling does not take full advantage of available unlabeled data. Zhu and Lafferty (2005) propose alternative methods for reducing the time compleity to $O(m^3), m < n$, but these may also impact performance. For structured output spaces, Lafferty et al. (2004) and Altun et al. (2005) have looked at approaches using these methods. However, the high running time of these methods has prevented wide-scale adoption, and they have been tested predominantly on synthetic or toy examples (e.g., with 5 labeled examples). Recently, Baluja et al. (2008) proposed a method for performing graph-based semi-supervised learning in parallel.

Since these are non-parametric models, they do not build a compact encoding of the model, and so it is not always clear how to apply them inductively (on new unlabeled data). At the very least, labeled and unlabeled data must be stored in order to classify new examples. In this paper we compare against a representative graph-based label propagation method called Quadratic Cost Criterion (QC) (Bengio et al., 2006) whose results are reported in Chapelle et al. (2006).

2.1.6 DIFFICULT APPLICATIONS

There are cases of semi-supervised learning being used in application settings, however, not without difficulty. In fact, a broad survey of semi-supervised learning methods (Chapelle et al., 2006) found that they do not uniformly beat supervised methods and that there is no clear winner from among the methods. This conclusion reflects the experimental evidence and theoretical support from a large span of work.

Expectation-maximization is notoriously fickle for semi-supervised learning. In a classic result (Merialdo, 1994) attempts semi-supervised learning to improve HMM part-of-speech tagging and finds that EM with unlabeled data reduces accuracy. Ng and Cardie (2003) also apply EM but finds that it fails to improve performance, as do Grenager et al. (2005) (without their tricky initialization). Cozman and Cohen (2006) discuss use cases where EM might fail to work.

Krogel and Scheffer (2004) use transductive SVMs for the functional genomics KDD Cup challenge and find that not only does it fails to improve performance but it even deteriorates performance. Ifrim and Weikum (2006) also find that TSVMs deteriorate performance. Kockelkorn et al. (2003) use transductive SVMs for text classification, but complain that it is computationally costly. Zhang and Oles (2000) discuss theoretical reasons why TSVMs might fail to work in various scenarios.

Macskassy and Provost (2006) apply harmonic mixing to classification of relational data, however the running time of harmonic mixing proves to be a barrier to its use. In the case of word sense disambiguation, Niu et al. (2005) has looked at label propagation, and found that the metric for graph construction has a dramatic effect on performance. Chen et al. (2005) look at combining manifold methods (e.g., ISOMAP) with semi-supervised learning, but finds that the methods are too fragile in their tuning parameters to be effective. Blum and Chawla (2001) also cite fragility in the tuning parameters as a problem for their graph-based method.

2.2 Semi-supervised Learning with Weakly Labeled Data

As an alternative to semi-supervised learning with labeled and unlabeled data, a number of methods have investigated semi-supervised learning with weakly labeled data or side information, though none with the expressiveness in labeling allowed by GE criteria.

Graph-based methods have used class proportions for post-processing to set thresholds on label propagation (Zhu et al., 2003). Schuurmans (1997) uses predicted label distributions on unlabeled data for model structure selection (as opposed to parameter estimation). More distantly, conditional harmonic mixing (Burges and Platt, 2006) minimizes over each point the KL-divergence between the currently predicted label distribution and the distribution predicted by its neighbors. Wang et al. (2004) also look at methods for incorporating class proportions into classification. In their model, they pseudolabel instances and provide them as constraints for the model to handle.

The use of side information to train a parametric classifier has been explored before by Schapire et al. (2002) who uses a boosted ensemble of weak learners set from human-generated expected distributions. There are significant differences between GE and this work, in particular, Schapire et al. (2002) match distributions on a per-instance basis, while generalized expectation criteria match a global distribution. Thus in the model proposed by Schapire et al. (2002), every example has to match the distribution given as input.[3] Graca et al. (2008) integrates similar types of instance-based constraints into EM learning, where the constraints restrict the space over which the model calculates the expectations of the hidden variables.

Like Schapire et al. (2002), Jin and Liu (2005) present a model for incorporating class proportions into discriminative models which places a expected class distribution over each instance. Unlike Schapire et al. (2002), these distributions start from a fixed point and then are allowed to change during training.

In contrastive estimation (Smith and Eisner, 2005), EM is performed over a restricted log-likelihood function, where instead of $L(\theta) = \sum_i \log p(x_i; \theta)$, the contrastive estimation log-likelihood function is $L_{CE}(\theta) = \sum_i \log p(x_i | \mathcal{N}(x_i); \theta)$. The neighborhood function $\mathcal{N}(x_i)$ must be highly tuned, and even slight variations in it can have significant impact on error. More crucially, the bias introduced by choosing $\mathcal{N}(x_i)$ is difficult to predict and unintuitive.

Haghighi and Klein (2006b) take prototypes as input to their method, and then uses SVD to link up words to prototypes with similar co-occurrence patterns (e.g., "Inner Richmond" has the label NEIGHBORHOOD). Haghighi and Klein (2006a) extends this framework to context-free grammar induction. Another group that has investigated integrating constraints into structure output learning is Chang et al. (2007) which integrates constraints into unsupervised learning with HMM. In their method, they reranking candidate labelings by constraint violations and then use a threshold set of these candidates for re-training in a Viterbi-like approximation to expectation maximization.

Generalized expectation criteria are unique in that it uses the expected distribution as the sole criterion for optimizing the model parameters on one set of unlabeled data (though it may also use labeled data). Most other methods do not try to directly fit these expectations, but use them instead as heuristics within a more complicated semi-supervised learning model. When we compare with techniques that use the label distribution (e.g., naïve Bayes with a fixed label prior), we find they do worse than GE, which demonstrates that GE is able to use these class distributions more effectively than other methods.

3. Label regularization is impossible under the Schapire et al. (2002) model, since if the model exactly matched the label expectation on a per-instance basis, in application it would assign all instances to the majority class.

3. Generalized Expectation Criteria

When a person is designing a classifier, they frequently have intuitions about the data they are trying to classify. For example, someone designing a part-of-speech tagger might know that 'nouns' are quite frequent, whereas 'conjunctions' are much less common. Manually labeling data can be a round-about way of providing this information to the machine learning model, and weak labeling provides another route for biasing the model with this information. It would be difficult to hypothesize priors over *model parameters* to capture this intuition. With GE the designer sets priors over *model expectations*, and since these expectations have a relatively transparent interpretation to the human designer, they provide an appealing route for injecting bias into the classifier. GE can effectively learn from a wide variety of side information, including expectation constraints which hold over global properties of the classifier (e.g., label marginals in label regularization), constraints over individual instances, and expectation constraints which are more expressive than the base model can model directly (e.g., a three label sequence in a markov order 1 CRF).

A **generalized expectation (GE) criterion** is a term in a parameter estimation objective function that assigns scores to values of a model expectation. First some standard notation: x is the input, y the output, and θ the parameters for a given model. Given a set of unlabeled data $\mathcal{U} = \{x\}$ and a conditional model $p(y|x;\theta)$, a GE criterion $G(\theta;\mathcal{U})$ is defined by a score function S and a constraint function $G(x,y)$:

$$G(\theta;\mathcal{U}) = S(E_{\mathcal{U}}[E_{p(y|x;\theta)}[G(x,y)]])\,.$$

In this light, GE criteria can be viewed as an replacement for a maximum likelihood estimator and can be maximized alone to yield a parameter estimate θ. For a particular choice of model family and parameterization, many different choices for score functions and constraint functions may be explored. In this paper we consider a subset of GE criteria which express a preference for a particular value of a constraint $\tilde{g}_{x,y}$, and apply the KL-divergence to compute model divergence from this constraint:

$$G(\theta;\mathcal{U}) = D(\tilde{g}_{x,y}||E_{\mathcal{U}}[p(y|x;\theta)G(x,y)])\,.$$

Other work has considered squared loss and constraint functions which are more and less expressive than the model parameterization (Druck et al., 2009a).

GE criteria can be used as a sole criterion for an objective function (e.g., Mann and McCallum 2008). In this work, we combine it the log-likelihood, $L(\theta)$ to form a composite objective function:

$$O(\theta;\mathcal{U},\mathcal{D}) = L(\theta;\mathcal{D}) + G(\theta;\mathcal{U})\,.$$

Alternatively, an entropy regularization term (Grandvalet and Bengio, 2004)[4] can be combined into the above objective function in the same manner:

$$O(\theta;\mathcal{U},\mathcal{D}) = L(\theta;\mathcal{D}) + G(\theta;\mathcal{U}) + H(\theta;\mathcal{U})\,.$$

GE criteria can be interpreted as a generalization of traditional maximum likelihood. First, GE allows a variety of scoring functions (e.g., KL-divergence or mean-squared error from a reference

4. Entropy regularization cannot be framed as a instance of GE, but a generalization could encompass both: $G(\theta;\mathcal{U}) = S(E_{\mathcal{U}}[E_{p(y|x;\theta)}[F(x,y)]])$, where F is an arbitrary function over a particular (x,y) tuple (e.g., for entropy regularization $F = \log p(y|x;\theta)$.

distribution) and thus can incorporate information from a source other than empirical feature counts derived from the training data (e.g., human intuition, empirical counts derived from alternative data sources). Second, there need not be a one-to-one relationship between GE terms and features. For example, we can express preferences on a subset of model features (and leave others unconstrained), or on marginal distributions larger than model factors. In this paper, we apply a constraint for only one feature obtained from human intuition about the problem.

We explore **label regularization**, where the constraints $\tilde{g}$ are expectations of model marginal distributions on the expected output labels. We look at functions $G(x,y) = \mathbf{1}(y)$, and use various estimated label marginal distributions:

$$\tilde{g}_{x,y} = \tilde{p}(y)\,.$$

The effect of adding this term is to ensure that the model applied to the unlabeled data matches the label proportions.[5] Note that this does not force the conditional label distribution for each instance to conform to this constraint, but rather it encourages the model to meet this constraint in aggregate over all instances.

Similar to label regularization, Quadrianto et al. (2008) uses label proportions to learn classifiers, though there are some interesting differences from our work. Their method relies on having multiple training subsets with very different class distributions, whereas we only use one data set with a single set of label proportions. Their work concentrates on non-structured classification, whereas we extend our method to the structured output case.

3.1 Recent Work on GE Criteria and Related Methods

Since the initial proposal of GE criteria (under the name *expectation regularization*) in Mann and McCallum (2007), there has been a flurry of recent work on generalized expectation criteria and related methods which apply expectation constraints for weak learning.

In a set of user experiments, Druck et al. (2008) compares traditional labeled data and *labeled features* (which can be used to build feature marginal distributions). That study finds that given the same amount of time, human annotation in the form of labeled features and classifiers trained using GE criteria outperform human annotation of traditional labeled instances and maximum likelihood training. For the structured output case, Mann and McCallum (2008) has shown that expectations over features for CRF learning, similar to the prototypes proposed in Haghighi and Klein (2006b), are more effective when used with GE than similar numbers of labeled tokens used to train a CRF. Druck et al. (2009a) extends these methods to conditional random field dependency parsing models, and shows that in that case as well feature marginal distributions can be effectively used to guide training.

A few groups have investigated related methods for incorporating expectation constraints. Ganchev et al. (2009) use expectation constraints over aligned sentences and a source-language parser induce dependency grammar on a target language, using a generative method related to expectation-maximization and a discriminative model closely related to GE criteria. Bellare et al. (2009) presents an alternative objective function to learn using expectation constraints over unlabeled data.

Since the emphasis is on reducing human annotation time, it is a clear question as to whether active learning can be applied to help choose labeled features or expectation constraints. Druck

5. The mathematics is unchanged for expectation constraints over any single features, but the experiments concern only this simple scenario.

et al. (2009b) pursues this question and uses active learning to choose which features to use with GE criteria. Along those lines, Liang et al. (2009) proposes a notion of 'measurements' to encapsulate the variety of weakly labeled data, and uses active learning to guide which measurements are provided from the human annotator to guide learning.

4. GE Criteria for Log-linear Models

In this section, we describe how to apply the GE criteria proposed above to conditionally trained log-linear models, starting with conditional maximum-entropy models, aka multinomial logistic regression models (Berger et al., 1996). In these models, there are k scalar feature functions $\psi_k(z,y)$, and the probability of the label y for input x is calculated by

$$p(y|x;\theta) = \frac{1}{Z(x)} \exp\left(\sum_k \theta_k \psi_k(x,y)\right),$$

where $Z(x) = \sum_{y'} \exp(\sum_k \theta_k \psi_k(x,y'))$ is the partition function. Given training data $\mathcal{D}$, the model is trained by maximizing the log-likelihood of the labels (with a Gaussian prior for regularization):

$$\begin{aligned} O(\theta;\mathcal{D}) &= \log L(\theta;\mathcal{D}) \\ &= \sum_{d\in\mathcal{D}} \log p(y_d|x_d;\theta) - \frac{\sum_k \theta_k^2}{2\sigma^2}. \end{aligned}$$

This can be done by gradient methods (Malouf, 2002), where the gradient of the likelihood is

$$\frac{\partial}{\partial\theta_k} O(\theta;\mathcal{D}) = \sum_d \left(\psi_k(x_d,y_d) - \sum_y p(y|x_d;\theta)\psi_k(x_d,y)\right) + \frac{\theta_k}{\sigma^2}.$$

For semi-supervised discriminative training, we augment the objective function by adding the generalized expectation criteria objective function terms.

$$\begin{aligned} O(\theta;D,U) &= L(\theta;\mathcal{D}) + G(\theta;\mathcal{U}) \\ &= \sum_d \log p(y_d|x_d);\theta) - \frac{\sum_k \theta_k^2}{2\sigma^2} - \lambda D(\tilde{g}_{x,y}||E_{\mathcal{U}}[E_{p(y|x;\theta)}[G(x,y)]]). \end{aligned}$$

Note that here the side information comes in the expectation constraints $\tilde{g}_{x,y}$ which specify particular priors for model marginal. In practice, we find that the hyper-parameters do not need to be extensively tuned. In particular, λ does not need tuning for each data set, and can be set simply to $\lambda = 10 \times$ # labeled examples.[6]

The form of the GE criteria lends itself to optimization by gradient based methods. After dropping terms which are constant with respect to the partial derivative, we are left with:

$$\begin{aligned} \frac{\partial}{\partial\theta_k} G(\theta;\mathcal{U}) &\propto \frac{\partial}{\partial\theta_k} \sum_y \tilde{g}_{x,y} \log \sum_{x\in\mathcal{U}} p(y|x;\theta) G(x,y) \\ &= \sum_y \left(\frac{\tilde{g}_{x,y}}{\sum_{x\in\mathcal{U}} p(y|x;\theta) G(x,y)}\right) \sum_{x\in\mathcal{U}} \frac{\partial}{\partial\theta_k} p(y|x;\theta) G(x,y) \\ &= \sum_y \left(\frac{\tilde{g}_{x,y}}{\sum_{x\in\mathcal{U}} p(y|x;\theta) G(x,y)}\right) \sum_{x\in U} p(y|x;\theta) G(x,y) \left(\psi_k(x,y) - \sum_{y'} p(y'|x;\theta)\psi_k(x)\right). \end{aligned}$$

6. As support for this value of λ, notice that the KL-divergence is significantly smaller than the likelihood as the likelihood is proportional to the number of examples, while the KL-divergence is not.

GE criteria aren't convex, and this can be shown by a contradictory example. Take a simple version of the GE criteria: $G(\theta;\mathcal{U}) = \sum_y \log \sum_x p(y|x;\theta)$. In this setting, for an arbitrary label y_i you can find parameter settings where $\forall x, p(y_i|x) = 1$, by having a parameter $\theta = \{\theta_k = \infty, \forall j \neq k, \theta_j = 0\}$ where $\psi_k(x,y) = \mathbf{1}\{y = y_i\}$. In this setting, it's pretty clear that multiple optima exist, and that other settings of θ yield smaller values of $G(\theta;\mathcal{U})$.

Label regularization can occasionally find a degenerate solution where, rather than the expectation of all instances matching the input distribution, instead, the distribution over labels for *each instance* will match the given distribution on every example. For example, given a three class classification task, if the labeled class distribution $\hat{p}_j(y) = \{.5,.35,.15\}$, it will find a solution such that $\tilde{p}(y;\theta) = \{.5,.35,.15\}$ for every instance. As a result, all the test instances will be assigned the same label.

One solution, appealing to 0/1 loss, would be to simply measure and match the expectation over winning class counts, but this is not differentiable. So instead, we make $p(y|x;\theta)$ more peaked using a less than 1.

$$p(y|x;\theta) \propto \exp\left(\frac{1}{T}\sum_k \theta_k \psi_k(x)\right).$$

This is differentiable and thus amenable to many gradient ascent methods. In practice we find that this meta-parameter does not require fine-tuning. Across all data sets we simply use $T = 0.1$ for multi-class problems and $T = 1$ for binary classification problems, and we find this to work well.

4.1 CRF Training

The previous section has shown the application of generalized expectation criteria to classification models. However, GE can additionally be applied to structured models. In this section, we examine the case of linear chain structured conditional random fields (Lafferty et al., 2001), and derive the GE gradient for this model.

Linear-chain CRFs are a discriminative probabilistic model over sequences $\mathbf{x} = \langle x_1..x_n\rangle$ of feature vectors and label sequences $\mathbf{y} = \langle y_1..y_n\rangle$, where $|\mathbf{x}| = |\mathbf{y}| = n$, and each label $y_i \in s$. This model is analogous to maximum entropy models for structured outputs, where expectations can be efficiently calculated by dynamic programming. For a linear-chain CRF of Markov order one

$$p(\mathbf{y}|\mathbf{x};\theta) = \frac{1}{Z(\mathbf{x})}\exp\left(\sum_k \theta_k \Psi_k(\mathbf{x},\mathbf{y})\right),$$

where $\Psi_k(\mathbf{x},\mathbf{y}) = \sum_i \psi_k(\mathbf{x}, y_i, y_{i+1}, i)$, and the partition function $Z(\mathbf{x}) = \sum_{\mathbf{y}'} \exp(\sum_k \theta_k \Psi_k(\mathbf{x},\mathbf{y}'))$. Given training data $\mathcal{D}$, the model is trained by maximizing the log-likelihood,

$$O(\theta;\mathcal{D}) = \sum_d \log p(\mathbf{y}_d|\mathbf{x}_d;\theta) - \frac{\sum_k \theta_k^2}{2\sigma^2},$$

by gradient-based methods where the gradient of the likelihood is (similar to the non-structured case):

$$\frac{\partial}{\partial\theta_k} O(\theta;D) = \sum_d \left(\Psi_k(\mathbf{x}_d,\mathbf{y}_d) - \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}_d;\theta)\Psi_k(\mathbf{x}_d,\mathbf{y})\right) + \frac{\theta_k}{\sigma^2}.$$

The second term (the expected counts of the features given the model) can be computed in a tractable amount of time, since according to the Markov assumption, the feature expectations can be rewritten:

$$\sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x};\theta)\Psi_k(\mathbf{x},\mathbf{y}) = \sum_i \sum_{y_i,y_{i+1}} p(y_i,y_{i+1}|\mathbf{x};\theta)\psi_k(\mathbf{x},y_i,y_{i+1},i)\,.$$

A dynamic program (the forward/backward algorithm) then computes in time $O(n|s|^2)$ all the needed probabilities $p_\theta(y_i,y_{i+1})$, where n is the sequence length, and $|s|$ is the number of labels.

4.2 Semi-supervised training with Generalized Expectation Criteria

To add unlabeled data regularization to the CRF training, just as with the maximum entropy model, we augment the objective function with the regularization term:

$$\begin{aligned} O(\theta;\mathcal{D},\mathcal{U}) =& L(\theta;\mathcal{D}) + G(\theta;\mathcal{U}) \\ =& \sum_d \log p(\mathbf{y}_d|\mathbf{x}_d;\theta) - \frac{\sum_k \theta_k^2}{2\sigma^2} - \lambda D(\tilde{g}_{x,y}||E_{\mathcal{U}}[E_{p(\mathbf{y}|\mathbf{x};\theta)}[\sum_i G(\mathbf{x},y_i)]])\,. \end{aligned}$$

Note that we restrict the constraint function to functions over one output label, $G(\mathbf{x},y_i)$. Druck et al. (2009a) has looked at extending the method to arbitrary constraint functions $G(\mathbf{x},\mathbf{y})$, but here we only consider constraint functions over functions of one label.

The derivation of the gradient for $G(\theta;\mathcal{U})$ is somewhat more complicated than in the unstructured case, but follows roughly the same line. s is the set of permissible output labels. $\mathbf{y}_{(m=s)} = \{\mathbf{y} : y_m = s\}$. The gradient is then:

$$\begin{aligned} \frac{\partial}{\partial\theta_k} G(\theta;\mathcal{U}) \propto& \frac{\partial}{\partial\theta_k} \sum_s \tilde{g}_{\mathbf{x},s} \log \sum_{\mathbf{x}\in\mathcal{U}} \sum_m \sum_{\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s) \\ =& \sum_s \left(\frac{\tilde{g}_{\mathbf{x},s}}{\sum_{\mathbf{x}\in\mathcal{U},m,\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s)} \right) \sum_{\mathbf{x}\in\mathcal{U}} \sum_m \sum_{\mathbf{y}_{(m=s)}} \frac{\partial}{\partial\theta_k} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s) \\ \texttt{Now define:} \frac{\tilde{g}}{G} =& \left(\frac{\tilde{g}_{\mathbf{x},s}}{\sum_{\mathbf{x}\in\mathcal{U},m,\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s)} \right) \\ =& \sum_s \frac{\tilde{g}}{G} \sum_{\mathbf{x}\in\mathcal{U}} \sum_m \sum_{\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s) \left(\Psi_k(\mathbf{x},\mathbf{y}_{(m=s)}) - \sum_{\mathbf{y}'} p(\mathbf{y}'|\mathbf{x}) \Psi_k(\mathbf{x},\mathbf{y}) \right) \\ =& \sum_s \frac{\tilde{g}}{G} \sum_{\mathbf{x}\in\mathcal{U}} \sum_m \sum_{\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(x,s) \Psi_k(\mathbf{x},\mathbf{y}_{(m=s)}) \\ & - \sum_s \frac{\tilde{g}}{G} \sum_{\mathbf{x}\in\mathcal{U}} \sum_m \sum_{\mathbf{y}_{(m=s)}} p(\mathbf{y}_{(m=s)}|\mathbf{x}) G(\mathbf{x},s) \sum_{\mathbf{y}'} p(\mathbf{y}'|\mathbf{x}) \Psi_k(\mathbf{x},\mathbf{y}) \Bigg) \\ =& \sum_s \frac{\tilde{g}}{G} \sum_{\mathbf{x}\in\mathcal{U}} \left(\sum_i \sum_{y_i,y_{i+1}} \psi_k(\mathbf{x},y_i,y_{i+1},i) \sum_m p(y_i,y_{i+l},y_m=s|\mathbf{x}) G(\mathbf{x},s) \right) \\ & - \sum_s \frac{\tilde{g}}{G} \sum_{\mathbf{x}\in\mathcal{U}} \left(\sum_i \sum_{y_i,y_{i+1}} \psi_k(\mathbf{x},y_i,y_{i+1},i) \right) \left(\sum_m p(y_m=s|\mathbf{x}) G(\mathbf{x},s) \right)\,. \end{aligned}$$

After combining terms and rearranging we arrive at the final form of the gradient:

$$= \sum_{\mathbf{x}\in\mathbf{U}} \sum_{i} \sum_{y_i, y_{i+1}} \psi_k(\mathbf{x}, y_i, y_{i+1}, i) \times \sum_{s} \frac{\tilde{g}}{G} \left(\sum_{m} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta) G(\mathbf{x}, s) - p(y_i, y_{i+1}|\mathbf{x};\theta) \sum_{m} p(y_m = s|\mathbf{x};\theta) G(\mathbf{x}, s) \right).$$

Here, the second term is easily obtainable from forward/backward, but the first term is a little more complicated to compute. Computing this term naively would require multiple runs of constrained forward/backward. Here we propose a more efficient method that requires only one run of forward/backward.[7] For the sake of simplicity, we omit the constraint function $G(\mathbf{x}, s)$; its addition is trivial. First we decompose the probability into two parts:

$$\sum_{m} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta) = \sum_{m=1}^{i} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta) + \sum_{m=i+1}^{n} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta).$$

Similar to forward/backward we build a lattice of intermediate results that then can be used to calculate the quantity of interest:

$$\begin{aligned}
&\sum_{m=1}^{i} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta) \\
&= p(y_i, y_{i+1}|\mathbf{x};\theta)\delta(y_i, s) + \sum_{m=1}^{i-1} p(y_i, y_{i+1}, y_m = s|\mathbf{x};\theta) \\
&= p(y_i, y_{i+1}|\mathbf{x};\theta)\delta(y_i, s) + \left(\sum_{y_{i-1}} \sum_{m=1}^{i-1} p(y_{i-1}, y_i, y_m = s|\mathbf{x};\theta) \right) p(y_{i+1}|y_i, \mathbf{x};\theta).
\end{aligned}$$

For each label s, it requires one pass to create a lattice with $\sum_{m=1}^{i-1} p(y_{i-1}, y_i, y_m = s|\mathbf{x};\theta)$ for all pairs (y_i, y_{i+1}). $\sum_{m=i+1}^{n} p(y_{i-1}, y_i, y_m = s|\mathbf{x};\theta)$ can be computed in the same fashion. To compute the lattices it takes time $O(n|s|^2)$, and one lattice must be computed for each label so the total time is $O(n|s|^3)$.

5. Experimental Results for Classifiers

We present two sets of experiments: experiments on maximum entropy models and conditional random fields for the special case of generalized expectation criteria, *label regularization*. For this set of experiments, we evaluate on five different data sets, and compare against seven different semi-supervised and supervised-only methods. We present learning curves, where the amount of labeled training data is gradually increased from one instance per class up to thousands of instances and demonstrate that generalized expectation criteria are able to show improvements for both types of scenarios. We present experiments with noisy expected distributions, and show that the method is robust with respect to a variety of settings for λ and temperature. We do not vary the gaussian regularizer, but leave a default value. Unpublished experiments by G. Druck[8] have suggested that

7. Kakade et al. (2002) present a related method that computes $p(y_{1..i} = s_{1..i}|y_{i+1} = s)$.
8. Conveyed in personal communication.

Name	# Test Examples	# Unlabeled Examples	# features	# classes
SRAA	20k	20k	77,494	4
POS	20k	20k	11,520	44
SecStr	1000	83k	314 (45,436)	2
BIOII	100k	100k	54,958	3
CoNLL03	100k	100k	114,264	9

Table 1: The data sets are complex: they have dramatic class skews, highly inter-dependent features, and large amounts of data. The SecStr data set has 315 atomic features, and 45k features when pairwise feature conjunctions are used.

while gaussian regularization can have an effect for label regularization, for more complicated GE variants it doesn't dramatically affect performance. We begin training with parameters set at 0 (even though the objective function may not be convex).

5.1 Experimental Set-up

First, we examine a protein secondary structure prediction task (**SecStr**), as extensively evaluated in Chapelle et al. (2006), compare with the published results and show that label regularization is able to outperform previous methods. Next, we examine three especially difficult natural language processing tasks: the CoNLL03 named-entity recognition task (**CoNLL03**), Part of speech tagging of the Wall Street Journal (**POS**), and the 2007 BiocreativeII evaluation (**BIOII**), using a sliding window classifier.[9] Finally, one of the main targets for semi-supervised learning is text classification (Nigam et al., 2006), and we evaluate on the simulated/real auto/aviation (**SRAA**) task. The tasks are large in scale, with up to hundreds of thousands of instances and features (see Table 1). They have complex characteristics such as heavily inter-dependent features and highly skewed class distributions.

Across all of the experiments, for supervised comparisons, we compare with naïve Bayes and maximum entropy models, for semi-supervised comparisons we compare with naïve Bayes trained with EM and maximum entropy models trained with entropy regularization. On some tasks, in particular the sliding window NLP tasks, the number of features per instance varied dramatically, and so we used document length normalization for the naïve Bayes approaches as we found it to significantly improve accuracy. On the secondary structure prediction (**SecStr**), we had access to published results for a supervised SVM using a radial-basis function (RBF) kernel, a Cluster Kernel (Weston et al., 2006) and a graph based-method, the Quadratic Cost Criterion with Class Mean Normalization (Bengio et al., 2006) trained using various data sub-sampling schemes (Delalleau et al., 2006): a random sampler and two smarter variations. Presumably, training a graph-based method on the entire unlabeled training set would have been technically infeasible.

For **CoNLL03**, **POS**, **BIOII**, and **SRAA**, we performed inductive learning, splitting the data randomly into two sections, training and test. From the training set, we randomly chose some instances to be labeled and set the remainder to be hidden. Out of those hidden, we then select a

9. The sliding window classifier makes independent decisions for each element in the sequence. While finite-state methods could also be applied in these cases, the cost of training label regularization would be prohibitive, and we found that these methods work well.

	# Labeled Instances		
	2	100	1000
SVM (supervised)		55.41	**66.29**
Cluster Kernel		57.05	65.97
QC randsub (CMN)		57.68	59.16
QC smartonly (CMN)		57.86	59.29
QC smartsub (CMN)		57.74	59.16
Naive Bayes (supervised)	52.42% (± 0.4)	57.12% (± 0.7)	64.47% (± 0.2)
Naive Bayes EM	50.79% (± 1.5)	57.34% (± 1.1)	57.60% (± .009)
MaxEnt (supervised)	52.42% (± 0.5)	56.74% (± 1.1)	65.43% (± 0.3)
MaxEnt + Ent. Min.	49.40% (± 2.1)	54.45% (± 1.8)	58.28% (± 0.1)
MaxEnt + GE	**57.08%** (± .03)	**58.51%** (± 0.4)	65.44% (± 0.3)

Table 2: Label regularization outperforms other semi-supervised learning methods at 100 labeled data points. At one instance per class, its performance is better than the *supervised* SVM and maximum entropy model at 100. Standard error is reported for experiments that were run locally. Other experimental performance is taken from the literature.

fixed number to use for unsupervised learning. We then evaluate the model on the hidden test data. We repeat this evaluation five times for each of the models.

The **SecStr** task was set up in what is commonly called transductive learning, where the model is evaluated on hidden training data. For this task, the labeled/unlabeled splits were provided with the data from Chapelle et al. (2006) and evaluation is on hidden training data. In order to provide a somewhat more fair comparison with the RBF kernels used by the other methods on this task, the feature set used by the maximum entropy model and naïve Bayes models is augmented by pairwise feature conjunctions.[10]

For the maximum entropy model trained with entropy regularization, after some experimentation, we weighted its contribution to the objective function with

$$\lambda = \text{\# labeled data points / \# unlabeled data points}.$$

This was shown to yield relatively good performance. For the first set of experiments, we use label proportions estimated from all of the data, corresponding to a use-case where a user gives this knowledge to the system during training. Section 5.3 presents experiments showing robustness to noisy label proportions both when smoothed towards a uniform distribution and when sampled from a limited number of training examples. Across the experiments, we observed that label regularization trains in time linear in the amount of unlabeled data, and since the resulting model is parametric, it is linear in the number of features for evaluation.

5.2 Learning Curves

For the first set of experiments, we experimented with varying the number of supervised training example, while keeping the unlabeled data set size the same (with sizes of the unlabeled data shown

10. Though, as one anonymous reviewer noted, this makes them strictly more expressive then kernel methods with quadratic features.

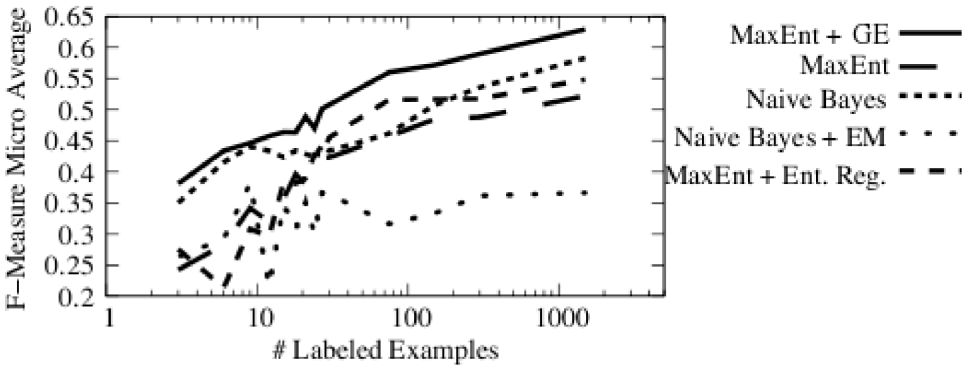


Figure 1: BIOII: Label regularization (GE) outperforms all other methods. The x-axis represents increasing numbers of labeled data instances. The y-axis is the F-measure micro average across all classes.

in Table 1). We added examples in a balanced way, going from 1 example per class up to 500 examples per class (when possible). For each data point we ran 5 trials, where the data was partitioned into training/test/unlabeled uniformly at random. The sole exception was for the **SecStr** data where the data splits and tests were pre-specified.

As Table 2 shows, for **SecStr**, label regularization outperforms the other methods at 100 labeled points, and approaches the cluster kernel method on 1000 points. While performance results were not available for the other methods for two instances per class, we ran label regularization for this case and found that it outperforms the *supervised* SVM and maximum entropy model when they are trained with 100 labeled points. In these experiments QC is not run over the complete data (presumably because of scalability problems), but operates on a subset, either selected randomly (randsub) or in a smarter fashion (smartonly and smartsub), while the label regularization method uses the complete data.

Figures 1, 2, 3, and 4 show classifier performance using a fixed amount of unlabeled data as greater amounts of labeled data is added. Label regularization yields significant benefits over the other methods for **POS**, **BIOII**, and **CoNLL03** for all amounts of labeled data. Label regularization on **SRAA** shows a benefit over the fully supervised maximum entropy model but its accuracy is not as high as that obtained by the EM-trained naïve Bayes learner. This may be partly explained by the fact that the baseline performance of the discriminative maximum entropy model is much lower than the generative naïve Bayes model, so that label regularization starts off at a considerable deficit.

While alternative methods often result in degredations of performance over their supervised counter parts (EM, entropy regularization, cluster kernels), in these experiments label regularization consistently yielded improved accuracy. Additionally, the benefit of label regularization is more apparent as the feature sets and numbers of unlabeled instances increase, with the least improvements on one of the simplest tasks, the **SRAA** text classification task.

These experiments demonstrate that label regularization can at least match, and in many cases beat, alternative methods of semi-supervised learning, given minimal additional information and access to large samples of unlabeled data. The successes of GE suggest more investigation of addi-

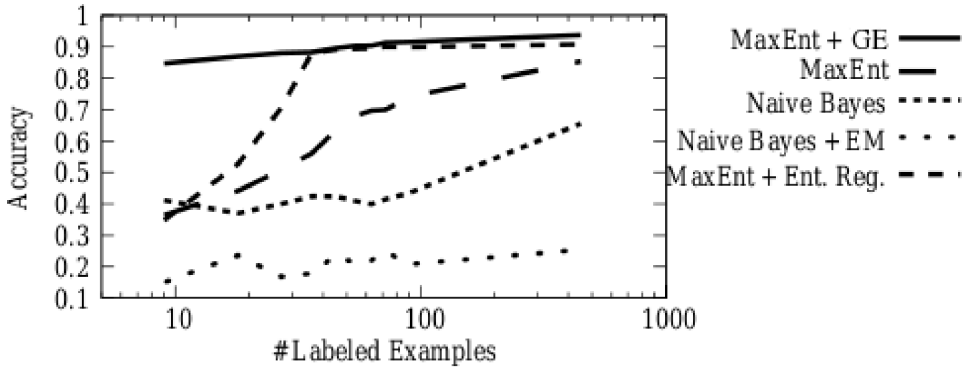


Figure 2: CoNLL03: Label regularization (GE) outperforms all other methods. The x-axis represents increasing numbers of labeled instances per class, and the y-axis is accuracy.

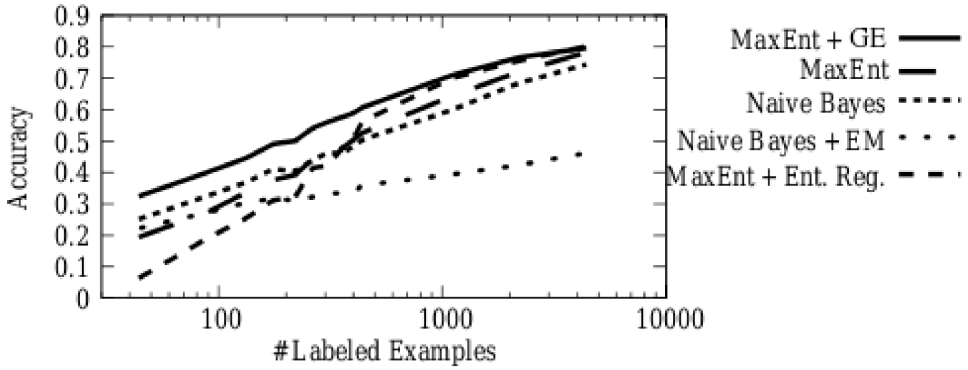


Figure 3: POS: Label regularization (GE) outperforms all other methods, though performance improvements over supervised maximum entropy methods appear to level off at 1300 labeled instances.

tional, alternative modalities of supervision which will generate data that can be added to supervised classifiers and combined with unlabeled data in order to improve performance.

5.3 Noisy Priors

The previous section assumes that the system has accurate knowledge of the distributions over the labels. In this section, we perform a sensitively analysis by gradually smoothing the class distribution until it reaches a uniform distribution. We add noisy counts ν to the true counts $c(y)$:

$$\hat{p}_j(\mathbf{y})(y) = \frac{c(y)+\nu}{\sum_{y'}(c(y')+\nu)}.$$

As more noise is added, the input distribution converges to uniform.

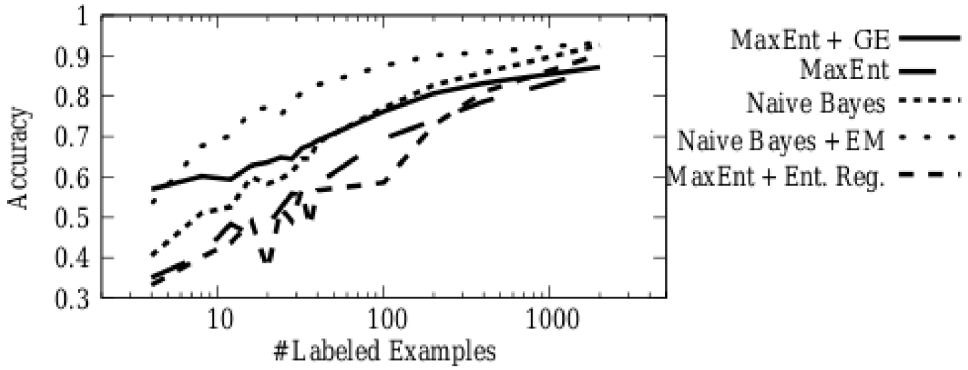


Figure 4: SRAA: Label regularization (GE) outperforms its supervised maximum entropy counterpart and entropy regularization and is the winner at one labeled instance per lass. After that, naïve Bayes EM is the clear winner.

Figure 5 demonstrates the effect of increasing noise in the system. At $\nu = 1{,}000$, the majority class probability drops from 84% to 80% and there is almost no loss of performance. At $\nu = 10{,}000$ are added, the majority class probability drops to 61% and there is only a slight loss of performance. At $\nu = 1e07$ the majority class probability has dropped to 11%, a virtually uniform distribution, and performance has leveled off. These results are encouraging as they suggest that relatively large changes (of 20% absolute, 27% relative) can be tolerated without major losses in accuracy. Even when the human has no domain knowledge to contribute, label distribution estimates of sufficient accuracy should be obtainable from a reasonably small number of labeled examples.

To test this assumption, we performed another set of experiments, where instead of smoothing the input distributions towards a uniform distribution, we sampled them from the data, varying the number of instances used in the sample. These points were sampled from the data, and then the data was partitioned into test/train/unlabel splits. Figure 6 and 7 demonstrate the effect of sampled distributions, as opposed to distributions smoothed towards a uniform distribution. For the **CoNLL3** data set, as can be seen in Figure 6, after sampling from 1000 points, the performance of the classifier doesn't get worse, suggesting that only a small amount of prior knowledge or labeling is necessary for determining accurate input distributions. With the **POS** data, it appears that as you increase the number of points used to compute the sample performance improves, though it appears to begin to level out at 1000 points. Because this data set is significantly smaller, we were unable to continue running experiments with larger numbers of sampled points to evaluate when the performance begins to level out.

5.4 Robustness

Along with robustness in the face of noise from the estimated label proportions, the model is robust to changes in λ and temperature. As can be seen in Figure 8, λ and temperature have a wide plateau over which their performance is stable. At some extreme values of λ and temperature, the performance degrades, and can drop below supervised performance. This trend was observed for 500 labeled examples (shown in the figure), as well as in cases when there as little as one

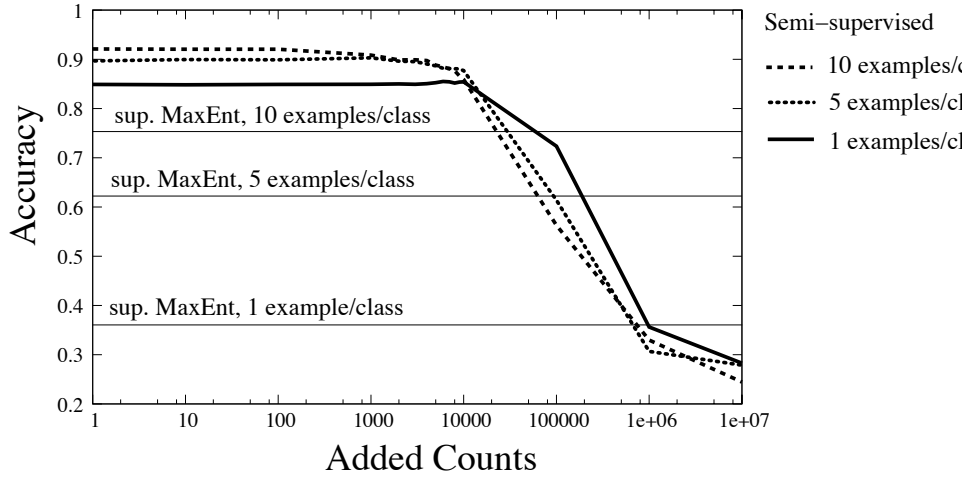


Figure 5: CoNLL03: The x-axis represents increasing amount of noise towards a uniform distribution. On this data set, the majority class is 84% of the instances, and so the uniform distribution is an extremely poor approximation. Performance suffers little when the majority class proportion is erroneously given as 61%($\nu = 10,000$)

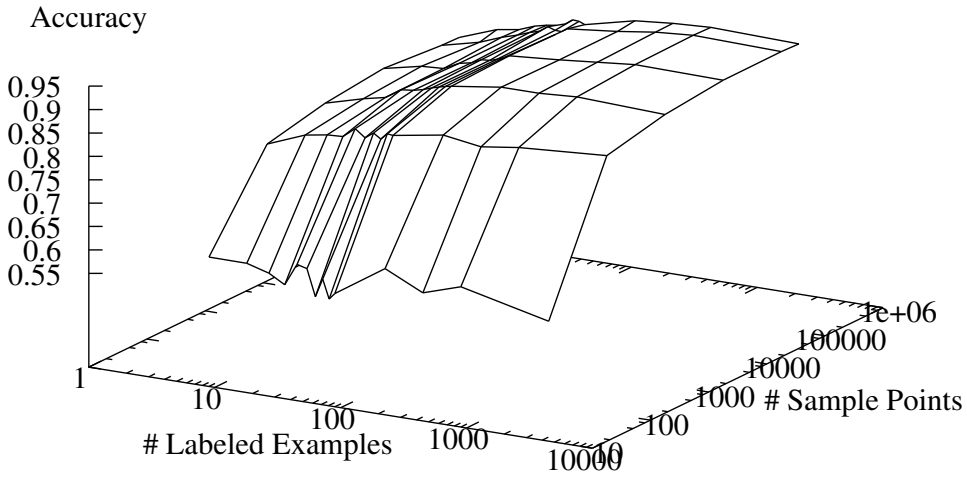


Figure 6: CoNLL03: An input label distribution with sampling noise. After 1000 points, sampling from more points doesn't appear to lead to performance improvements.

labeled example for a number of the data sets. For other semi-supervised techniques such as entropy regularization, extensive tuning is required across for each individual data set and labeled/unlabeled data set sizes in order to improve upon supervised-only performance (Jiao et al., 2006).

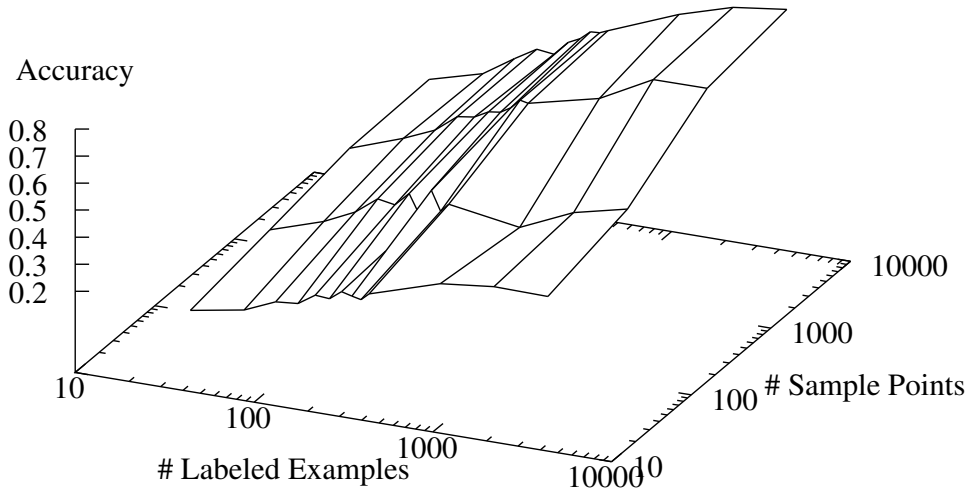


Figure 7: POS: An input label distribution with sampling noise. Since POS has many classes (44), access to an accurate sampling distribution has a larger impact on performance, and the graph suggests that even higher precision in sampling would lead to higher accuracy. Note that we were unable to sample as many points as in the previous example beause of limited amounts of available data.

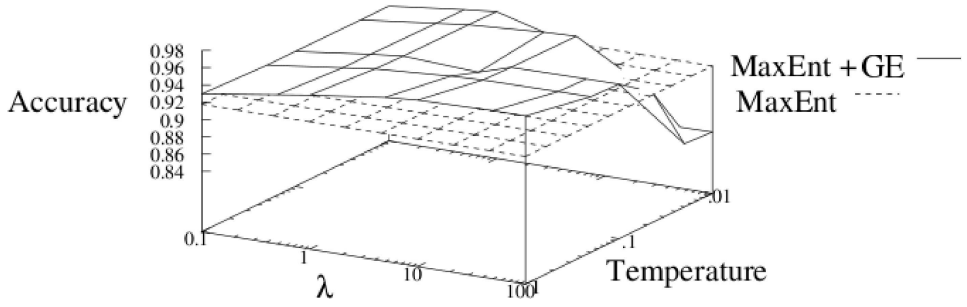


Figure 8: CoNLL03: For a wide range of λ and temperature the performance is similar and surpasses the purely supervised performance.

5.5 Running Time

Figure 9 shows the running time per optimization iteration for the two largest tasks, **CoNLL** and **BIOII**. The slope variation between the running times can be accounted for by the number of features in each of the data sets.

5.6 Mechanism of Effect

One question that needs to be addressed is whether label regularization is improving performance solely by adjusting the label proportions or operating in some other fashion. Though certainly correcting label proportions is one pathway for improved performance we have two pieces of evidence that additional learning is happening in the model. First, when we allow other classifiers access to

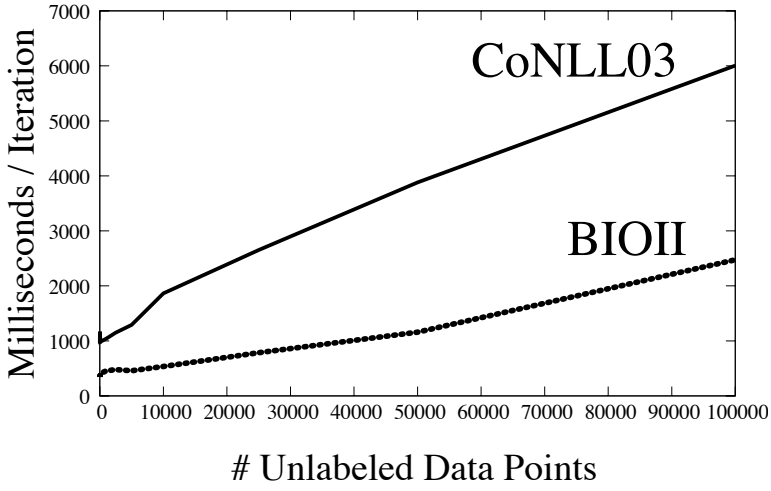


Figure 9: Label regularization is linear in the number of unlabeled examples, without requiring sub-sampling.

the label proportions, they are unable to reach the same gains as achieved with GE. For example, in all of the experiments the naïve Bayes classifier has its label prior fixed to the input distribution as it is subsequently trained with EM, yet it typically fails to reach the same level of performance as achieved with the GE methods. Second, Druck et al. (2008) reports results of experiments using GE criteria where the input distributions were only allowed to affect the features they were conditioned on (corresponding to only being able to adjust the label proportions). In these experiments, when GE could only adjust the features specified by the input distributions it achieved significantly worse results, thus indicating that learning parameter values for features not directly conditioned by the input distributions has a dramatic effect on classifier accuracy.

5.7 Combining Label Regularization and Entropy Regularization

Generalized expectation criteria can often be easily combined with other models. Any semi-supervised model in the parametric model family, such as expected gradient methods (Salakhutdinov et al., 2003), can be easily combined with GE, and certain generative models such as naïve MRFs (Druck et al., 2007) can be simply combined as well. More distantly, just as various models can be augmented with regularization terms (as in ridge regression for linear regression models), GE may be augmented in the same way. In this paper we used a Gaussian prior and minimized KL-divergence from input distributions with gradient methods here, in other cases it might require an alternative penalty term from the input distributions and a different minimization technique.

Here we examine combining label regularization with entropy regularization where the objective function is augmented with more than one regularization criterion. For many of the experiments, combining label regularization and entropy regularization does not lead to improvements. Two exceptions were experiments on **SRAA** and the **SecStr** data sets. Notably, on **SecStr**, combined entropy regularization and label regularization yields a performance of 66.30, a level which matches the performance of the supervised radial-basis SVM and beats all other unsupervised methods. For **SRAA**, Figure 10 shows that when entropy regularization is added to label regularization, there

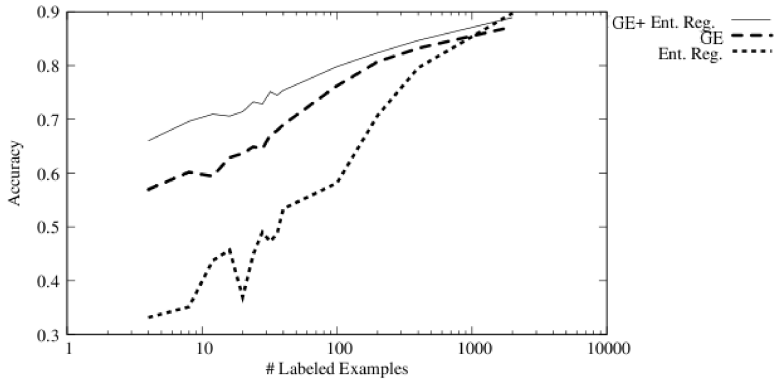


Figure 10: SRAA: Combining label regularization and entropy regularization can be easily accomplished, and yields improvements over label regularization alone for this data set.

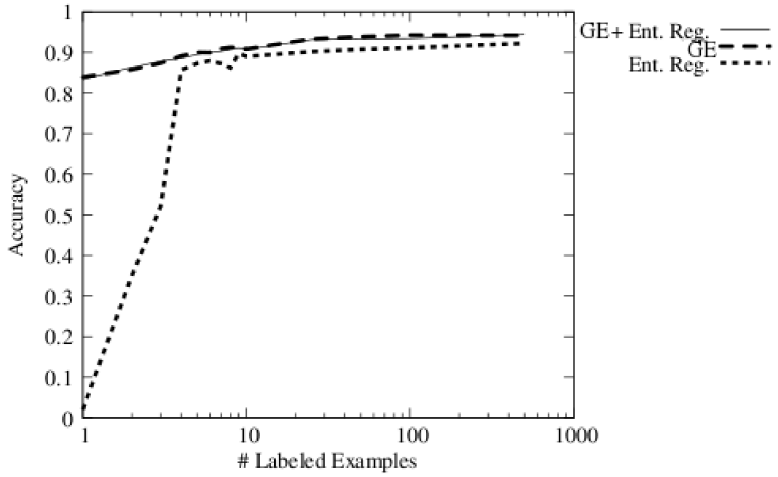


Figure 11: CoNLL03: On this data set, combining label regularization and entropy regularization does not lead to any benefit. The GE+Entropy regularization curve exactly overlaps the GE curve.

can sometimes be a benefit over the use of label regularization or entropy regularization alone. In comparison, Figure 11 shows that there is little or no difference in performance when entropy regularization is combined with label regularization in the case of **CoNLL03**.

6. Experimental Results for Conditional Random Fields

In this section, we examine the performance of label regularization for conditional random fields. We look at two data sets, Craigslist Apartment listings, and citation data. We compare against two previous methods for semi-supervised learning of conditional random fields, entropy regularization and the clustering method proposed by Miller et al. (2004) in Section 6.3. We demonstrate that label regularization can achieve higher accuracy than purely supervised training, and can beat or match

MRF (prototypes)	53.7%
MRF (prototypes) + cluster	71.5%
HMM supervised only (100)	74.4%
CRF supervised only (100)	75.8% ($\pm$ 0.3)
CRF supervised (100) + Ent. Reg. (2k)	76.7% ($\pm$ 0.4)
CRF supervised (100) + GE (2k)	77.1% ($\pm$ 0.3)

Table 3: APT: A CRF trained semi-supervised with 100 labeled examples and 2k unlabeled exampled has the highest performance, beating the strictly supervised CRF and the CRF trained semi-supervised with entropy regularization. Standard error is shown in parentheses for experiments run locally.

the performance of entropy regularization or clustering. We do not vary the gaussian regularizer, but leave a default value. We begin training with parameters set at 0 (even though the objective function may not be convex). Later experiments start from a model initialized by a non-structured classifier trained with GE, which we observed to yield higher accuracy (Mann and McCallum, 2008).

6.1 Apartment Listings

First we examine performance on the apartment data set (**APT**) initially presented by Grenager et al. (2005) and later examined by Haghighi and Klein (2006b), with label regularization. This data was collected in June 2004 from craigslist.com, and consists of 302 hand-labeled ads where each ad is labeled with 12 fields (e.g., SIZE, RENT, NEIGHBORHOOD, FEATURES). The average ad has 119 tokens in 8.7 fields.

For this task, sliding window models perform poorly as the fields are "sticky," (i.e., the best way to predict the next label is from the previous label). We set label regularization λ as before, but set the entropy regularization λ to 0.01 times the number of labeled examples divided by the number of unlabeled examples. For these experiments we used 2,000 unlabeled apartments listings.

We performed minimal feature engineering, using only standard capitalization and word class features (e.g., "digits"). We additionally used a feature that is the exact token string of the previous word. The use of flexible, non-independent features demonstrates the benefit of the greater expressive power of discriminatively trained CRFs; with these features alone, the CRF out-performs the supervised HMM.

Table 3 compares the performance of our system relative to previous supervised and semi-supervised systems,[11] where at the maximum amount of training data, label regularization achieves a 1.3% accuracy improvement over the purely supervised CRF, and a 0.4% accuracy improvement over semi-supervised learning via entropy regularization.

Table 4 shows learning curve performance for the CRF with a variety of different settings. With accurate input distributions, the CRF trained with label regularization achieves the highest performance, for all but with one example, where entropy regularization is the highest performer. Label regularization gives a 8% absolute accuracy improvement at the lower levels of training data and a 1.3% boost (a 5% relative error reduction) for a highly trained sequence model. Unlike entropy

11. We did not implement the ad-hoc boundary pre-processing and post-processing as is performed by the unsupervised MRF. These boundary features gave a 3% improvement for Haghighi and Klein (2006b).

# Supervised	Supervised	Entropy Regularization	GE: Estimated Priors	GE: Accurate Priors
1	41.2% (± 1.7)	44.3% (± 0.3)	41.3% (± 1.7)	**45.7%** (± 1.7)
5	48.4% (± 1.8)	45.8% (± 1.1)	51.9% (± 1.6)	**56.2%** (± 0.6)
10	55.9% (± 1.3)	58.0% (± 2.5)	57.5% (± 1.2)	**63.0 %** (± 0.2)
50	71.7% (± 0.5)	**74.1 %** (± 0.2)	73.7 % (± 0.4)	74.0 % (± 0.6)
100	75.8% (± 0.3)	76.7 % (± 0.4)	**77.4 %** (± 0.2)	77.1 % (± 0.3)

Table 4: APT: Use of input distributions estimated from limited labeled data yields smaller improvements over the true distributions, but still improves over the strictly supervised accuracy, and at large levels of training data, nearly matches the accuracy achieved by the true distributions. Standard error is shown in parentheses for the experiments.

# Supervised	Supervised	Entropy Regularization	GE: Estimated Priors	GE: Accurate Priors
1	24.9% (± 3.6)	17.5% (± 3.2)	25.5% (± 3.8)	**37.3%** (± 2.1)
5	52.4% (± 0.5)	27.0% (± 2.7)	52.4% (± 5.6)	**54.7%** (± 0.5)
10	56.1% (± 0.6)	35.2% (± 3.3)	55.5% (± 0.7)	**57.9%** (± 0.4)
50	67.8% (± 0.6)	66.5% (± 0.8)	67.5% (± 0.3)	**68.0%** (± 0.6)
100	72.7% (± 0.5)	**73.5%** (± 0.5)	72.4% (± 0.3)	72.0% (± 0.6)

Table 5: CITE: Label regularization (GE) is able to significantly improve on the purely supervised cases at very low levels of training data. At higher levels of training data, it makes less of an impact. Standard error is shown in parentheses for the experiments.

regularization, label regularization improves over supervised learning across all amounts of training data.

In addition to testing with accurate proportions, we also examined performance when input distributions were read directly off of the minimal training label sequence. When these noisy input distributions were used, the performance was less than with entropy regularization at lower levels of training data, but it still provided consistent gains in performance across all training settings.

6.2 Citation Data

In addition to experiments on apartment data, we also ran experiments on citation data (**CITE**) as given by Grenager et al. (2005). This data set consists of 500 hand-annotated citations taken from the reference sections of different computer science papers. Each citation is annotated with 13 fields (e.g., AUTHOR, TITLE, DATE, JOURNAL), and on average the citation has 35 tokens annotated with 5.5 fields. For this experiment we used 1,000 unlabeled examples, with $\lambda = 1$ times the number of unlabeled data points, and the entropy regularizer set as before. On this set of data, as shown in Table 5, label regularization gives a slight win at low levels of training data, but at higher levels, it does not provide any improvement. Entropy regularization unfortunately consistently decreases performance. Here, estimating the input distributions from the minimal training data leads to performance decreases beyond supervised training. The difference between the two data sets suggest that more work is needed to understand in what situations label regularization can be expected to work well.

# Supervised	Supervised	Supervised + Clustering	GE	GE + Clustering
1	41.2% (± 1.7)	44.5% (± 0.9)	45.7% (± 1.7)	**50.0%** (± 0.3)
5	48.4% (± 1.8)	54.3% (± 1.8)	56.2% (± 0.6)	**58.9%** (± 1.1)
10	55.9% (± 1.3)	61.2% (± 1.5)	63.0 % (± 0.2)	**64.4%** (± 0.9)
50	71.7% (± 0.5)	**74.1%** (± 0.5)	74.0 % (± 0.6)	**74.1%** (± 0.3)
100	75.8% (± 0.3)	**77.6%** (± 0.1)	77.1 % (± 0.3)	**77.6%** (± 0.2)

Table 6: APT: Using clustering features gives an additional gain to label regularization for low levels of training data, but it doesn't provide any additional benefit at higher levels of training data. Standard error is shown in parentheses for the experiments.

One thing to note from the experiments, is that as training data increases, the performance gap between the true distributions and estimated input distributions diminishes, until at 100 supervised examples, it almost matches the accuracy achieved by the true distributions. These results are at once encouraging and surprising, and suggest two possible reasons for improvement. First, perhaps the CRF isn't matching the proportions implicit in the labeled data, even though it has access to this information. Second, it suggests that the unlabeled data does have an effect beyond that of helping to readjust the classifier towards the input distributions. In particular, perhaps new features are being brought in by inclusion of the unlabeled examples.

The strengths of label regularization over entropy regularization are two-fold. First, GE gives an overall win in performance across many different levels of training data. Second, GE gives consistent gains. GE rarely performs worse than purely supervised training (when the input distributions are accurate), whereas the performance of entropy regularization is erratic, sometimes yielding a gain in performance, in other cases leading to a severe decrease in performance.

6.3 Clustering Features

Miller et al. (2004) proposes a method of using unsupervised clusters to improve performance. In his method, the unlabeled data undergoes word clustering, and then features corresponding to clusters are added during supervised training. A similar method can be applied here, in which features corresponding to unsupervised word clusters are added during semi-supervised training. We applied this method to the apartment data, and for very low level of training, found encouraging performance gains. Table 6 shows that for one labeled example, 2,000 unlabeled examples, and unsupervised clustering, the system achieves almost a 5% improvement by using these unsupervised cluster features (45.7% to 50.0%). At higher levels of training data, label regularization and clustering features interact poorly, and using them together lead to no improvement. While at lower levels of training data, label regularization is able to achieve more gains than the clustering method, at higher levels of performance, it only matches performance. We have also attempted to use sim features as proposed in Haghighi and Klein (2006b), but found these methods difficult to tune (e.g., what SVD rank to retain).

7. Conclusion

This paper has presented *generalized expectation criteria*, a simple, robust, scalable method for semi-supervised learning. This method penalizes models by divergence between the model's expectations over the unlabeled data and input conditional probabilities, which can be estimated from labeled data or given as a priori knowledge by a human annotator. An important special case, *label regularization* is empirically explored for the case of maximum entropy models where we find it to provide accuracy improvements over entropy regularization, naïve Bayes EM, Quadratic Cost Criterion (a representative graph-based method) and a cluster kernel SVM. We show that the method is robust to noise in the estimates of the input conditional probabilities, the meta-parameters need little or no tuning, and that it runs in linear time with increasing numbers of unlabeled examples.

We additionally present extensions of the method to conditional random fields. In this case, the gradient computation is complicated and requires the use of a specialized dynamic program which computes $\sum_j p(y_i, y_{i+1}, y_j = l|\mathbf{x};\theta)$. Conditional random fields trained with label regularization outperform alternative methods for semi-supervised training such as entropy regularization, and can achieve 1% to 8% improvement over supervised only approaches.

The simplicity of these methods, their robustness, and their high performance give promise that they may have a wide application and impact in use for semi-supervised machine learning.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant # IIS-0326249.

References

S. Abney. Understanding the yarowsky algorithm. *Computational Linguistics*, 30:3, 2004.

Y. Altun, D. McAllester, and M. Belkin. Maximum margin semi-supervised learning for structured variables. In *NIPS*, 2005.

R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6, 2005.

S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. Video suggestion and discovery for youtube: Taking random walks through the view graph. In *WWW*, 2008.

K. Bellare, G. Druck, and A. McCallum. Alternating projections for learning with expectation constraints. In *UAI*, 2009.

Y. Bengio, O. Dellalleau, and N. Le Roux. Label propagation and quadratic criterion. In O. Chapelle, B. Schlkopf, and A. Zien, editors, *Semi-Supervised Learning*. MIT Press, 2006.

A. L. Berger, V. J. Della Pietra, and S. A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1), 1996.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincut. In *ICML*, 2001.

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998.

P. F. Brown, V. J. D. Pietra, P. V. DeSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, pages 467–479, 1992.

C.J.C Burges and J.C. Platt. Semi-supervised learning with conditional harmonic mixing. In O. Chapelle, B. Schlkopf, and A. Zien, editors, *Semi-Supervised Learning*. MIT Press, 2006.

M.-W. Chang, L. Ratinov, and D. Roth. Guiding semi-supervision with constraint-driven learning. In *ACL*, 2007.

O. Chapelle, B. Scholkopf, and A. Zien. Analysis of Benchmarks. In O. Chapelle, A. Zien, and B. Scholkopf, editors, *Semi-Supervised Learning*. MIT Press, 2006.

M. Chen, I-H. Lee, G. Wu, Y. Wu, and E. Chang. Manifold learning, a promised land or work in progress? In *IEEE/International Conference on Multimedia and Expo*, 2005.

A. Corduneanu and T. Jaakkola. On information regularization. In *UAI*, 2003.

F. Cozman and I. Cohen. Risks of Semi-Supervised Learning. In O. Chapelle, A. Zien, and B. Scholkopf, editors, *Semi-Supervised Learning*. MIT Press, 2006.

O. Delalleau, Y. Bengio, and N. Le Roux. Large-scale algorithms. In Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors, *Semi-Supervised Learning*, 2006.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Stat. Soc.*, 39:1–38, 1977.

G. Druck, C. Pal, X. Zhu, and A. McCallum. Semi-supervised classification with hybrid generative discriminative methods. In *KDD*, 2007.

G. Druck, G. S. Mann, and A. McCallum. Learning from labeled features using generalized expectation criteria. In *SIGIR*, 2008.

G. Druck, G. Mann, and A. McCallum. Semi-supervised learning of dependency parsers using generalized expectation criteria. In *ACL/IJCNLP*, 2009a.

G. Druck, B. Settles, and A. McCallum. Active learning by labeling features. In *EMNLP*, 2009b.

D. Freitag. Trained named entity recognition using distributional clusters. In *EMNLP*, 2004.

K. Ganchev, K. Crammer, F. Pereira, G. Mann, A. McCallum, S. Carroll, Y. Jin, and P. White. Penn/umass/chop biocreativeii systems. In *BioCreativeII*, 2007.

K. Ganchev, J. Gillenwater, and B. Taskar. Dependency grammar induction via bitext projection constraints. In *ACL/IJCNLP*, 2009.

J. Graca, K. Ganchev, and B. Taskar. Expectation maximization and posterior constraints. In *NIPS*, 2008.

Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2004.

T. Grenager, D. Klein, and C. Manning. Unsupervised learning of field segmentation models for information extraction. In *ACL*, 2005.

A. Haghighi and D. Klein. Prototype-driven grammar induction. In *COLING-ACL*, 2006a.

A. Haghighi and D. Klein. Prototype-driven learning for sequence models. In *NAACL*, 2006b.

G. Ifrim and G. Weikum. Transductive learning for text classification using explicit knowledge models. In *PKDD*, 2006.

F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *COLING/ACL*, 2006.

R. Jin and Y. Liu. A framework for incorporating class priors into discriminative classification. In *PAKDD*, 2005.

T. Joachims. Transductive inference for text classification using support vector machines. In *ICML*, 1999. URL `citeseer.ist.psu.edu/joachims99transductive.html`.

S. Kakade, Y-W. Teg, and S.Roweis. An alternate objective function for markovian fields. In *ICML*, 2002.

D. Klein and C. Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *ACL*, 2004.

M. Kockelkorn, A. Luneburg, and T. Scheffer. Using transduction and multi-view learning to answer emails. In *PKDD*, 2003.

M. Krogel and T. Scheffer. Multi-relational learning, text mining, and semi-supervised learing for functional genomics. *Machine Learning*, 57, 2004.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

J. Lafferty, Y. Liu, and X. Zhu. Kernel conditional random fields:repsentation, clique selection, and semi-supervised learning. In *ICML*, 2004.

W. Li and A. McCallum. A note on semi-supervised learning using markov random fields. Computer science technical note, University of Massachusetts, Amherst, MA, 2004.

W. Li and A. McCallum. Semi-supervised sequence modeling with syntactic topic models. In *AAAI*, 2005.

P. Liang, M. Jordan, and D. Klein. Lerning from measurements in exponential families. In *ICML*, 2009.

S. Macskassy and F. Provost. Classification in networked data. Technical Report CeDER-04-08, New York University, 2006.

R. Malouf. A comparison of algorithms for maximum enotrpy parameter estimation. In *COLING*, 2002.

G. Mann and A. McCallum. Simple, robust, scalable semi-supervised learning via expectation regularization. In *ICML*, 2007.

G. Mann and A. McCallum. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *ACL*, 2008.

A. McCallum, K. Bellare, and F. Pereira. A conditional random field for discriminatively-trained finite-state string edit distance. In *UAI*, 2005.

P. Merialdo. Tagging english text with a probabilistic model. *Computational Linguistics*, 1994.

S. Miller, J. Guinness, and A. Zamanian. Name tagging with word clusters and discriminative training. In *ACL*, 2004.

V. Ng and C. Cardie. Weakly supervised natural language learning without redundant views. In *HLT-NAACL*, 2003.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *AAAI*, 1998.

K. Nigam, A. McCallum, and T. Mitchell. Semi-supervised Text Classification Using EM. In O. Chapelle, A. Zien, and B. Scholkopf, editors, *Semi-Supervised Learning*. MIT Press, 2006.

Z.-Y. Niu, D.-H. Ji, and C. L. Tam. Word sense disambiguation using label propagation based semi-supervised learning. In *ACL*, 2005.

N. Quadrianto, A. J. Smola, T.S. Caetano, and Q.V. Le. Estimating labels from label proportions. In *ICML*, 2008.

E. Riloff and J. Shepherd. A Corpus-based Bootstrapping Algorithm for Semi-Automated Semantic Lexicon Construction. *Journal for Natural Language Engineering*, 2000. forthcoming.

R. Salakhutdinov, S. Roweis, and Z. Ghahramani. Optimization with em and expectation-conjugate-gradient. In *ICML*, 2003.

R. Schapire, M. Rochery, M. Rahim, and N. Gupta. Incorporating prior knowledge into boosting. In *ICML*, 2002.

D. Schuurmans. A new metric-based approach to model selection. In *AAAI*, 1997.

V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear svms. In *SIGIR*, 2006.

N. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *ACL*, 2005.

J. Suzuki and H. Isozaki. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *ACL*, 2008.

J. Suzuki, A. Fujino, and H. Isozaki. Semi-supervised structured output learning based on a hybrid generative and discriminative approach. In *EMNLP-CoNLL*, 2007.

Martin Szummer and Tommi Jaakkola. Partially labeled classification with markov random walks. In *NIPS*, volume 14, 2002.

L. Wang, P. Xue, and K. Chan. Incorporating prior knowledge into svm for image retrieval. In *ICPR*, 2004.

S. Wang, R. Rosenfeld, Y. Zhao, and D. Schuurmans. The latent maximum entropy principle. In *IEEE ISIT*, 2002.

J. Weston, C. Leslie, E. Ie, and W. S. Noble. Semi-supervised protein classification using cluster kernels. In Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors, *Semi-Supervised Learning*, 2006.

D. Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of ACL*, 1995.

T. Zhang and F.J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *ICML*, 2000.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, CMU, 2002.

X. Zhu and J. Lafferty. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *ICML*, 2005.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic mixtures. In *ICML*, 2003.

Journal of Machine Learning Research 11 (2010) 985-1042 Submitted 12/08; Revised 8/09; Published 2/10

Kronecker Graphs: An Approach to Modeling Networks

Jure Leskovec JURE@CS.STANFORD.EDU
Computer Science Department
Stanford University
Stanford, CA 94305

Deepayan Chakrabarti DEEPAY@YAHOO-INC.COM
Yahoo! Research
2821 Mission College Blvd
Santa Clara, CA 95054

Jon Kleinberg KLEINBER@CS.CORNELL.EDU
Computer Science Department
Cornell University
Ithaca, NY 14853

Christos Faloutsos CHRISTOS@CS.CMU.EDU
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Zoubin Ghahramani ZOUBIN@ENG.CAM.AC.UK
Department of Engineering
University of Cambridge
Cambridge CB2 1PZ, UK

Editor: Fan Chung

Abstract

How can we generate realistic networks? In addition, how can we do so with a mathematically tractable model that allows for rigorous analysis of network properties? Real networks exhibit a long list of surprising properties: Heavy tails for the in- and out-degree distribution, heavy tails for the eigenvalues and eigenvectors, small diameters, and densification and shrinking diameters over time. Current network models and generators either fail to match several of the above properties, are complicated to analyze mathematically, or both. Here we propose a generative model for networks that is both mathematically tractable and can generate networks that have all the above mentioned structural properties. Our main idea here is to use a non-standard matrix operation, the *Kronecker product*, to generate graphs which we refer to as "Kronecker graphs".

First, we show that Kronecker graphs naturally obey common network properties. In fact, we rigorously *prove* that they do so. We also provide empirical evidence showing that Kronecker graphs can effectively model the structure of real networks.

We then present KRONFIT, a fast and scalable algorithm for fitting the Kronecker graph generation model to large real networks. A naive approach to fitting would take super-exponential time. In contrast, KRONFIT takes *linear* time, by exploiting the structure of Kronecker matrix multiplication and by using statistical simulation techniques.

Experiments on a wide range of large real and synthetic networks show that KRONFIT finds accurate parameters that very well mimic the properties of target networks. In fact, using just

four parameters we can accurately model several aspects of global network structure. Once fitted, the model parameters can be used to gain insights about the network structure, and the resulting synthetic graphs can be used for null-models, anonymization, extrapolations, and graph summarization.

Keywords: Kronecker graphs, network analysis, network models, social networks, graph generators, graph mining, network evolution

1. Introduction

What do real graphs look like? How do they evolve over time? How can we generate synthetic, but realistic looking, time-evolving graphs? Recently, network analysis has been attracting much interest, with an emphasis on finding patterns and abnormalities in social networks, computer networks, e-mail interactions, gene regulatory networks, and many more. Most of the work focuses on static snapshots of graphs, where fascinating "laws" have been discovered, including small diameters and heavy-tailed degree distributions.

In parallel with discoveries of such structural "laws" there has been effort to find mechanisms and models of network formation that generate networks with such structures. So, a good realistic network generation model is important for at least two reasons. The first is that it can generate graphs for extrapolations, hypothesis testing, "what-if" scenarios, and simulations, when real graphs are difficult or impossible to collect. For example, how well will a given protocol run on the Internet five years from now? Accurate network models can produce more realistic models for the future Internet, on which simulations can be run. The second reason is more subtle. It forces us to think about network properties that generative models should obey to be realistic.

In this paper we introduce Kronecker graphs, a generative network model which obeys all the main static network patterns that have appeared in the literature (Faloutsos et al., 1999; Albert et al., 1999; Chakrabarti et al., 2004; Farkas et al., 2001; Mihail and Papadimitriou, 2002; Watts and Strogatz, 1998). Our model also obeys recently discovered temporal evolution patterns (Leskovec et al., 2005b, 2007a). And, contrary to other models that match this combination of network properties (as for example, Bu and Towsley, 2002; Klemm and Eguíluz, 2002; Vázquez, 2003; Leskovec et al., 2005b; Zheleva et al., 2009), Kronecker graphs also lead to tractable analysis and rigorous proofs. Furthermore, the Kronecker graphs generative process also has a nice natural interpretation and justification.

Our model is based on a matrix operation, the *Kronecker product*. There are several known theorems on Kronecker products. They correspond exactly to a significant portion of what we want to prove: heavy-tailed distributions for in-degree, out-degree, eigenvalues, and eigenvectors. We also demonstrate how a Kronecker graphs can match the behavior of several real networks (social networks, citations, web, internet, and others). While Kronecker products have been studied by the algebraic combinatorics community (see, e.g., Chow, 1997; Imrich, 1998; Imrich and Klavžar, 2000; Hammack, 2009), the present work is the first to employ this operation in the design of network models to match real data.

Then we also make a step further and tackle the following problem: Given a large real network, we want to generate a synthetic graph, so that the resulting synthetic graph matches the properties of the real network as well as possible.

Ideally we would like: (a) A graph generation model that *naturally* produces networks where many properties that are also found in real networks naturally emerge. (b) The model parameter estimation should be fast and scalable, so that we can handle networks with millions of nodes. (c)

The resulting set of parameters should generate realistic-looking networks that match the statistical properties of the target, real networks.

In general the problem of modeling network structure presents several conceptual and engineering challenges: Which generative model should we choose, among the many in the literature? How do we measure the goodness of the fit? (Least squares don't work well for power laws, for subtle reasons!) If we use likelihood, how do we estimate it faster than in time quadratic on the number of nodes? How do we solve the node correspondence problem, that is, which node of the real network corresponds to what node of the synthetic one?

To answer the above questions we present KRONFIT, a fast and scalable algorithm for fitting Kronecker graphs by using the maximum likelihood principle. When calculating the likelihood there are two challenges: First, one needs to solve the node correspondence problem by matching the nodes of the real and the synthetic network. Essentially, one has to consider all mappings of nodes of the network to the rows and columns of the graph adjacency matrix. This becomes intractable for graphs with more than tens of nodes. Even when given the "true" node correspondences, just evaluating the likelihood is still prohibitively expensive for large graphs that we consider, as one needs to evaluate the probability of each possible edge. We present solutions to both of these problems: We develop a Metropolis sampling algorithm for sampling node correspondences, and approximate the likelihood to obtain a *linear* time algorithm for Kronecker graph model parameter estimation that scales to large networks with millions of nodes and edges. KRONFIT gives orders of magnitude speed-ups against older methods (20 minutes on a commodity PC, versus 2 days on a 50-machine cluster).

Our extensive experiments on synthetic and real networks show that Kronecker graphs can efficiently model statistical properties of networks, like degree distribution and diameter, while using only four parameters.

Once the model is fitted to the real network, there are several benefits and applications:

(a) *Network structure:* the parameters give us insight into the global structure of the network itself.

(b) *Null-model:* when working with network data we would often like to assess the significance or the extent to which a certain network property is expressed. We can use Kronecker graph as an accurate null-model.

(c) *Simulations:* given an algorithm working on a graph we would like to evaluate how its performance depends on various properties of the network. Using our model one can generate graphs that exhibit various combinations of such properties, and then evaluate the algorithm.

(d) *Extrapolations:* we can use the model to generate a larger graph, to help us understand how the network will look like in the future.

(e) *Sampling:* conversely, we can also generate a smaller graph, which may be useful for running simulation experiments (e.g., simulating routing algorithms in computer networks, or virus/worm propagation algorithms), when these algorithms may be too slow to run on large graphs.

(f) *Graph similarity:* to compare the similarity of the structure of different networks (even of different sizes) one can use the differences in estimated parameters as a similarity measure.

(g) *Graph visualization and compression:* we can compress the graph, by storing just the model parameters, and the deviations between the real and the synthetic graph. Similarly, for visualization purposes one can use the structure of the parameter matrix to visualize the backbone of the network, and then display the edges that deviate from the backbone structure.

(h) *Anonymization:* suppose that the real graph cannot be publicized, like, for example, corporate e-mail network or customer-product sales in a recommendation system. Yet, we would like to share our network. Our work gives ways to such a realistic, 'similar' network.

The current paper builds on our previous work on Kronecker graphs (Leskovec et al., 2005a; Leskovec and Faloutsos, 2007) and is organized as follows: Section 2 briefly surveys the related literature. In section 3 we introduce the Kronecker graph model, and give formal statements about the properties of networks it generates. We investigate the model using simulation in Section 4 and continue by introducing KRONFIT, the Kronecker graphs parameter estimation algorithm, in Section 5. We present experimental results on a wide range of real and synthetic networks in Section 6. We close with discussion and conclusions in Sections 7 and 8.

2. Relation to Previous Work on Network Modeling

Networks across a wide range of domains present surprising regularities, such as power laws, small diameters, communities, and so on. We use these patterns as sanity checks, that is, our synthetic graphs should match those properties of the real target graph.

Most of the related work in this field has concentrated on two aspects: properties and patterns found in real-world networks, and then ways to find models to build understanding about the emergence of these properties. First, we will discuss the commonly found patterns in (static and temporally evolving) graphs, and finally, the state of the art in graph generation methods.

2.1 Graph Patterns

Here we briefly introduce the network patterns (also referred to as properties or statistics) that we will later use to compare the similarity between the real networks and their synthetic counterparts produced by the Kronecker graphs model. While many patterns have been discovered, two of the principal ones are heavy-tailed degree distributions and small diameters.

Degree distribution: The degree-distribution of a graph is a power law if the number of nodes N_d with degree d is given by $N_d \propto d^{-\gamma} \quad (\gamma > 0)$ where γ is called the power law exponent. Power laws have been found in the Internet (Faloutsos et al., 1999), the Web (Kleinberg et al., 1999; Broder et al., 2000), citation graphs (Redner, 1998), online social networks (Chakrabarti et al., 2004) and many others.

Small diameter: Most real-world graphs exhibit relatively small diameter (the "small- world" phenomenon, or "six degrees of separation" Milgram, 1967): A graph has diameter D if every pair of nodes can be connected by a path of length at most D edges. The diameter D is susceptible to outliers. Thus, a more robust measure of the pair wise distances between nodes in a graph is the *integer effective diameter* (Tauro et al., 2001), which is the minimum number of links (steps/hops) in which some fraction (or quantile q, say $q = 0.9$) of all connected pairs of nodes can reach each other. Here we make use of *effective diameter* which we define as follows (Leskovec et al., 2005b). For each natural number h, let $g(h)$ denote the fraction of connected node pairs whose shortest

connecting path has length at most h, that is, at most h hops away. We then consider a function defined over all positive real numbers x by linearly interpolating between the points $(h, g(h))$ and $(h+1, g(h+1))$ for each x, where $h = \lfloor x \rfloor$, and we define the *effective diameter* of the network to be the value x at which the function $g(x)$ achieves the value 0.9. The effective diameter has been found to be small for large real-world graphs, like Internet, Web, and online social networks (Albert and Barabási, 2002; Milgram, 1967; Leskovec et al., 2005b).

Hop-plot: It extends the notion of diameter by plotting the number of reachable pairs $g(h)$ within h hops, as a function of the number of hops h (Palmer et al., 2002). It gives us a sense of how quickly nodes' neighborhoods expand with the number of hops.

Scree plot: This is a plot of the eigenvalues (or singular values) of the graph adjacency matrix, versus their rank, using the logarithmic scale. The scree plot is also often found to approximately obey a power law (Chakrabarti et al., 2004; Farkas et al., 2001). Moreover, this pattern was also found analytically for random power law graphs (Mihail and Papadimitriou, 2002; Chung et al., 2003).

Network values: The distribution of eigenvector components (indicators of "network value") associated to the largest eigenvalue of the graph adjacency matrix has also been found to be skewed (Chakrabarti et al., 2004).

Node triangle participation: Edges in real-world networks and especially in social networks tend to cluster (Watts and Strogatz, 1998) and form triads of connected nodes. Node triangle participation is a measure of transitivity in networks. It counts the number of triangles a node participates in, that is, the number of connections between the neighbors of a node. The plot of the number of triangles Δ versus the number of nodes that participate in Δ triangles has also been found to be skewed (Tsourakakis, 2008).

Densification power law: The relation between the number of edges $E(t)$ and the number of nodes $N(t)$ in evolving network at time t obeys the *densification power law* (DPL), which states that $E(t) \propto N(t)^a$. The *densification exponent* a is typically greater than 1, implying that the average degree of a node in the network is *increasing* over time (as the network gains more nodes and edges). This means that real networks tend to sprout many more edges than nodes, and thus densify as they grow (Leskovec et al., 2005b, 2007a).

Shrinking diameter: The effective diameter of graphs tends to shrink or stabilize as the number of nodes in a network grows over time (Leskovec et al., 2005b, 2007a). This is somewhat counterintuitive since from common experience as one would expect that as the volume of the object (a graph) grows, the size (i.e., the diameter) would also grow. But for real networks this does not hold as the diameter shrinks and then seems to stabilize as the network grows.

2.2 Generative Models of Network Structure

The earliest probabilistic generative model for graphs was the Erdős-Rényi (Erdős and Rényi, 1960) random graph model, where each pair of nodes has an identical, independent probability of being joined by an edge. The study of this model has led to a rich mathematical theory. However, as the model was not developed to model real-world networks it produces graphs that fail to match real networks in a number of respects (for example, it does not produce heavy-tailed degree distributions).

The vast majority of recent network models involve some form of *preferential attachment* (Barabási and Albert, 1999; Albert and Barabási, 2002; Winick and Jamin, 2002; Kleinberg et al.,

1999; Kumar et al., 1999; Flaxman et al., 2007) that employs a simple rule: new node joins the graph at each time step, and then creates a connection to an existing node u with the probability proportional to the degree of the node u. This leads to the "rich get richer" phenomena and to power law tails in degree distribution. However, the diameter in this model grows slowly with the number of nodes N, which violates the "shrinking diameter" property mentioned above.

There are also many variations of preferential attachment model, all somehow employing the "rich get richer" type mechanism, for example, the "copying model" (Kumar et al., 2000), the "winner does not take all" model (Pennock et al., 2002), the "forest fire" model (Leskovec et al., 2005b), the "random surfer model" (Blum et al., 2006), etc.

A different family of network methods strives for small diameter and local clustering in networks. Examples of such models include the *small-world* model (Watts and Strogatz, 1998) and the Waxman generator (Waxman, 1988). Another family of models shows that heavy tails emerge if nodes try to optimize their connectivity under resource constraints (Carlson and Doyle, 1999; Fabrikant et al., 2002).

In summary, most current models focus on modeling only one (static) network property, and neglect the others. In addition, it is usually hard to analytically analyze properties of the network model. On the other hand, the Kronecker graph model we describe in the next section addresses these issues as it matches multiple properties of real networks at the same time, while being analytically tractable and lending itself to rigorous analysis.

2.3 Parameter Estimation of Network Models

Until recently relatively little effort was made to fit the above network models to real data. One of the difficulties is that most of the above models usually define a mechanism or a principle by which a network is constructed, and thus parameter estimation is either trivial or almost impossible.

Most work in estimating network models comes from the area of social sciences, statistics and social network analysis where the *exponential random graphs*, also known as $p*$ model, were introduced (Wasserman and Pattison, 1996). The model essentially defines a log linear model over all possible graphs G, $p(G|\theta) \propto \exp(\theta^T s(G))$, where G is a graph, and s is a set of functions, that can be viewed as summary statistics for the structural features of the network. The $p*$ model usually focuses on "local" structural features of networks (like, for example, characteristics of nodes that determine a presence of an edge, link reciprocity, etc.). As exponential random graphs have been very useful for modeling small networks, and individual nodes and edges, our goal here is different in a sense that we aim to accurately model the structure of the network as a whole. Moreover, we aim to model and estimate parameters of networks with millions of nodes, while even for graphs of small size (> 100 nodes) the number of model parameters in exponential random graphs usually becomes too large, and estimation prohibitively expensive, both in terms of computational time and memory.

Regardless of a particular choice of a network model, a common theme when estimating the likelihood $P(G)$ of a graph G under some model is the challenge of finding the correspondence between the nodes of the true network and its synthetic counterpart. The node correspondence problem results in the factorially many possible matchings of nodes. One can think of the correspondence problem as a test of graph isomorphism. Two isomorphic graphs G and G' with differently assigned node IDs should have same likelihood $P(G) = P(G')$ so we aim to find an accurate mapping between the nodes of the two graphs.

SYMBOL	DESCRIPTION
G	Real network
N	Number of nodes in G
E	Number of edges in G
K	Kronecker graph (synthetic estimate of G)
K_1	Initiator of a Kronecker graphs
N_1	Number of nodes in initiator K_1
E_1	Number of edges in K_1 (the expected number of edges in $\mathcal{P}_1$, $E_1 = \sum \theta_{ij}$)
$G \otimes H$	Kronecker product of adjacency matrices of graphs G and H
$K_1^{[k]} = K_k = K$	k^{th} Kronecker power of K_1
$K_1[i,j]$	Entry at row i and column j of K_1
$\Theta = \mathcal{P}_1$	Stochastic Kronecker initiator
$\mathcal{P}_1^{[k]} = \mathcal{P}_k = \mathcal{P}$	k^{th} Kronecker power of $\mathcal{P}_1$
$\theta_{ij} = \mathcal{P}_1[i,j]$	Entry at row i and column j of $\mathcal{P}_1$
$p_{ij} = \mathcal{P}_k[i,j]$	Probability of an edge (i,j) in $\mathcal{P}_k$, that is, entry at row i and column j of $\mathcal{P}_k$
$K = R(\mathcal{P})$	Realization of a Stochastic Kronecker graph $\mathcal{P}$
$l(\Theta)$	Log-likelihood. Log-prob. that Θ generated real graph G, $\log P(G\|\Theta)$
$\hat{\Theta}$	Parameters at maximum likelihood, $\hat{\Theta} = \text{argmax}_\Theta P(G\|\Theta)$
σ	Permutation that maps node IDs of G to those of $\mathcal{P}$
a	Densification power law exponent, $E(t) \propto N(t)^a$
D	Diameter of a graph
N_c	Number of nodes in the largest weakly connected component of a graph
ω	Fraction of times `SwapNodes` permutation proposal distribution is used

Table 1: Table of symbols.

An ordering or a permutation defines the mapping of nodes in one network to nodes in the other network. For example, Butts (2005) used permutation sampling to determine similarity between two graph adjacency matrices, while Bezáková et al. (2006) used permutations for graph model selection. Recently, an approach for estimating parameters of the "copying" model was introduced (Wiuf et al., 2006), however authors also note that the class of "copying" models may not be rich enough to accurately model real networks. As we show later, Kronecker graph model seems to have the necessary expressive power to mimic real networks well.

3. Kronecker Graph Model

The Kronecker graph model we propose here is based on a recursive construction. Defining the recursion properly is somewhat subtle, as a number of standard, related graph construction methods fail to produce graphs that densify according to the patterns observed in real networks, and they also produce graphs whose diameters increase. To produce densifying graphs with constant/shrinking diameter, and thereby match the qualitative behavior of a real network, we develop a procedure that is best described in terms of the *Kronecker product* of matrices.

3.1 Main Idea

The main intuition behind the model is to create self-similar graphs, recursively. We begin with an *initiator* graph K_1, with N_1 nodes and E_1 edges, and by recursion we produce successively larger graphs $K_2, K_3, \ldots$ such that the k^{th} graph K_k is on $N_k = N_1^k$ nodes. If we want these graphs to exhibit a version of the Densification power law (Leskovec et al., 2005b), then K_k should have $E_k = E_1^k$ edges. This is a property that requires some care in order to get right, as standard recursive constructions (for example, the traditional Cartesian product or the construction of Barabási et al., 2001) do not yield graphs satisfying the densification power law.

It turns out that the *Kronecker product* of two matrices is the right tool for this goal. The Kronecker product is defined as follows:

Definition 1 (Kronecker product of matrices) *Given two matrices $\mathbf{A} = [a_{i,j}]$ and $\mathbf{B}$ of sizes $n \times m$ and $n' \times m'$ respectively, the Kronecker product matrix $\mathbf{C}$ of dimensions $(n \cdot n') \times (m \cdot m')$ is given by*

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \doteq \begin{pmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \ldots & a_{1,m}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \ldots & a_{2,m}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}\mathbf{B} & a_{n,2}\mathbf{B} & \ldots & a_{n,m}\mathbf{B} \end{pmatrix}.$$

We then define the Kronecker product of two graphs simply as the Kronecker product of their corresponding adjacency matrices.

Definition 2 (Kronecker product of graphs, Weichsel, 1962) *If G and H are graphs with adjacency matrices $A(G)$ and $A(H)$ respectively, then the Kronecker product $G \otimes H$ is defined as the graph with adjacency matrix $A(G) \otimes A(H)$.*

Observation 1 (Edges in Kronecker-multiplied graphs)

$$\textit{Edge } (X_{ij}, X_{kl}) \in G \otimes H \textit{ iff } (X_i, X_k) \in G \textit{ and } (X_j, X_l) \in H.$$

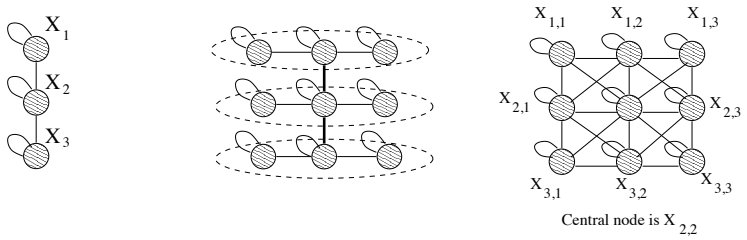
where X_{ij} and X_{kl} are nodes in $G \otimes H$, and X_i, X_j, X_k and X_l are the corresponding nodes in G and H, as in Figure 1.

The last observation is crucial, and deserves elaboration. Basically, each node in $G \otimes H$ can be represented as an ordered pair X_{ij}, with i a node of G and j a node of H, and with an edge joining X_{ij} and X_{kl} precisely when (X_i, X_k) is an edge of G and (X_j, X_l) is an edge of H. This is a direct consequence of the hierarchical nature of the Kronecker product. Figure 1(a–c) further illustrates this by showing the recursive construction of $G \otimes H$, when $G = H$ is a 3-node chain. Consider node $X_{1,2}$ in Figure 1(c): It belongs to the H graph that replaced node X_1 (see Figure 1(b)), and in fact is the X_2 node (i.e., the center) within this small H-graph.

We propose to produce a growing sequence of matrices by iterating the Kronecker product:

Definition 3 (Kronecker power) *The k^{th} power of K_1 is defined as the matrix $K_1^{[k]}$ (abbreviated to K_k), such that:*

$$K_1^{[k]} = K_k = \underbrace{K_1 \otimes K_1 \otimes \ldots K_1}_{k \text{ times}} = K_{k-1} \otimes K_1$$



(a) Graph K_1 (b) Intermediate stage (c) Graph $K_2 = K_1 \otimes K_1$

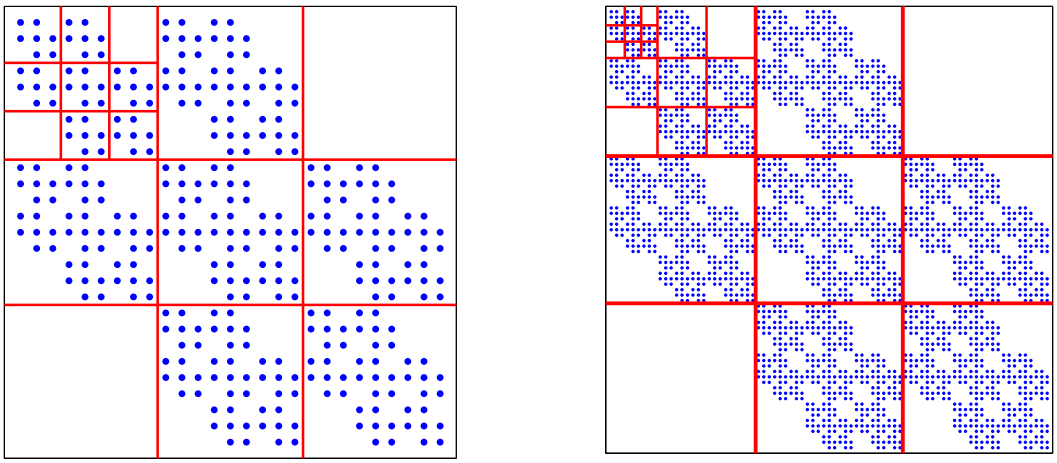
1	1	0
1	1	1
0	1	1

(d) Adjacency matrix of K_1

K_1	K_1	0
K_1	K_1	K_1
0	K_1	K_1

(e) Adjacency matrix of $K_2 = K_1 \otimes K_1$

Figure 1: *Example of Kronecker multiplication:* Top: a "3-chain" initiator graph and its Kronecker product with itself. Each of the X_i nodes gets expanded into 3 nodes, which are then linked using Observation 1. Bottom row: the corresponding adjacency matrices. See Figure 2 for adjacency matrices of K_3 and K_4.



(a) K_3 adjacency matrix (27×27) (b) K_4 adjacency matrix (81×81)

Figure 2: Adjacency matrices of K_3 and K_4, the 3^{rd} and 4^{th} Kronecker power of K_1 matrix as defined in Figure 1. Dots represent non-zero matrix entries, and white space represents zeros. Notice the recursive self-similar structure of the adjacency matrix.

Definition 4 (Kronecker graph) *Kronecker graph of order k is defined by the adjacency matrix* $K_1^{[k]}$*, where* K_1 *is the Kronecker initiator adjacency matrix.*

The self-similar nature of the Kronecker graph product is clear: To produce K_k from K_{k-1}, we "expand" (replace) each node of K_{k-1} by converting it into a copy of K_1, and we join these copies

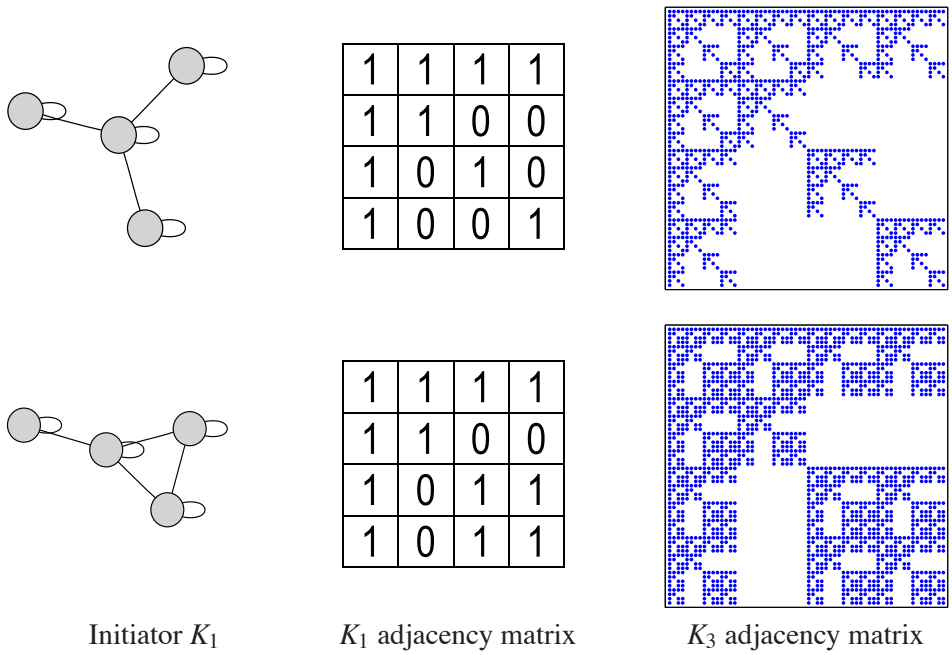


Figure 3: Two examples of Kronecker initiators on 4 nodes and the self-similar adjacency matrices they produce.

together according to the adjacencies in K_{k-1} (see Figure 1). This process is very natural: one can imagine it as positing that communities within the graph grow recursively, with nodes in the community recursively getting expanded into miniature copies of the community. Nodes in the sub-community then link among themselves and also to nodes from other communities.

Note that there are many different names to refer to Kronecker product of graphs. Other names for the Kronecker product are tensor product, categorical product, direct product, cardinal product, relational product, conjunction, weak direct product or just product, and even Cartesian product (Imrich and Klavžar, 2000).

3.2 Analysis of Kronecker Graphs

We shall now discuss the properties of Kronecker graphs, specifically, their degree distributions, diameters, eigenvalues, eigenvectors, and time-evolution. Our ability to prove analytical results about all of these properties is a major advantage of Kronecker graphs over other network models.

3.2.1 DEGREE DISTRIBUTION

The next few theorems prove that several distributions of interest are *multinomial* for our Kronecker graph model. This is important, because a careful choice of the initial graph K_1 makes the resulting multinomial distribution to behave like a power law or Discrete Gaussian Exponential (DGX) distribution (Bi et al., 2001; Clauset et al., 2007).

Theorem 5 (Multinomial degree distribution) *Kronecker graphs have multinomial degree distributions, for both in- and out-degrees.*

Proof Let the initiator K_1 have the degree sequence $d_1, d_2, \ldots, d_{N_1}$. Kronecker multiplication of a node with degree d expands it into N_1 nodes, with the corresponding degrees being $d \times d_1, d \times d_2, \ldots, d \times d_{N_1}$. After Kronecker powering, the degree of each node in graph K_k is of the form $d_{i_1} \times d_{i_2} \times \ldots d_{i_k}$, with $i_1, i_2, \ldots, i_k \in (1 \ldots N_1)$, and there is one node for each ordered combination. This gives us the multinomial distribution on the degrees of K_k. So, graph K_k will have multinomial degree distribution where the "events" (degrees) of the distribution will be combinations of degree products: $d_1^{i_1} d_2^{i_2} \ldots d_{N_1}^{i_{N_1}}$ (where $\sum_{j=1}^{N_1} i_j = k$) and event (degree) probabilities will be proportional to $\binom{k}{i_1 i_2 \ldots i_{N_1}}$. Note also that this is equivalent to noticing that the degrees of nodes in K_k can be expressed as the k^{th} Kronecker power of the vector $(d_1, d_2, \ldots, d_{N_1})$. ■

3.2.2 SPECTRAL PROPERTIES

Next we analyze the spectral properties of adjacency matrix of a Kronecker graph. We show that both the distribution of eigenvalues and the distribution of component values of eigenvectors of the graph adjacency matrix follow multinomial distributions.

Theorem 6 (Multinomial eigenvalue distribution) *The Kronecker graph K_k has a multinomial distribution for its eigenvalues.*

Proof Let K_1 have the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{N_1}$. By properties of the Kronecker multiplication (Loan, 2000; Langville and Stewart, 2004), the eigenvalues of K_k are the k^{th} Kronecker power of the vector of eigenvalues of the initiator matrix, $(\lambda_1, \lambda_2, \ldots, \lambda_{N_1})^{[k]}$. As in Theorem 5, the eigenvalue distribution is a multinomial. ■

A similar argument using properties of Kronecker matrix multiplication shows the following.

Theorem 7 (Multinomial eigenvector distribution) *The components of each eigenvector of the Kronecker graph K_k follow a multinomial distribution.*

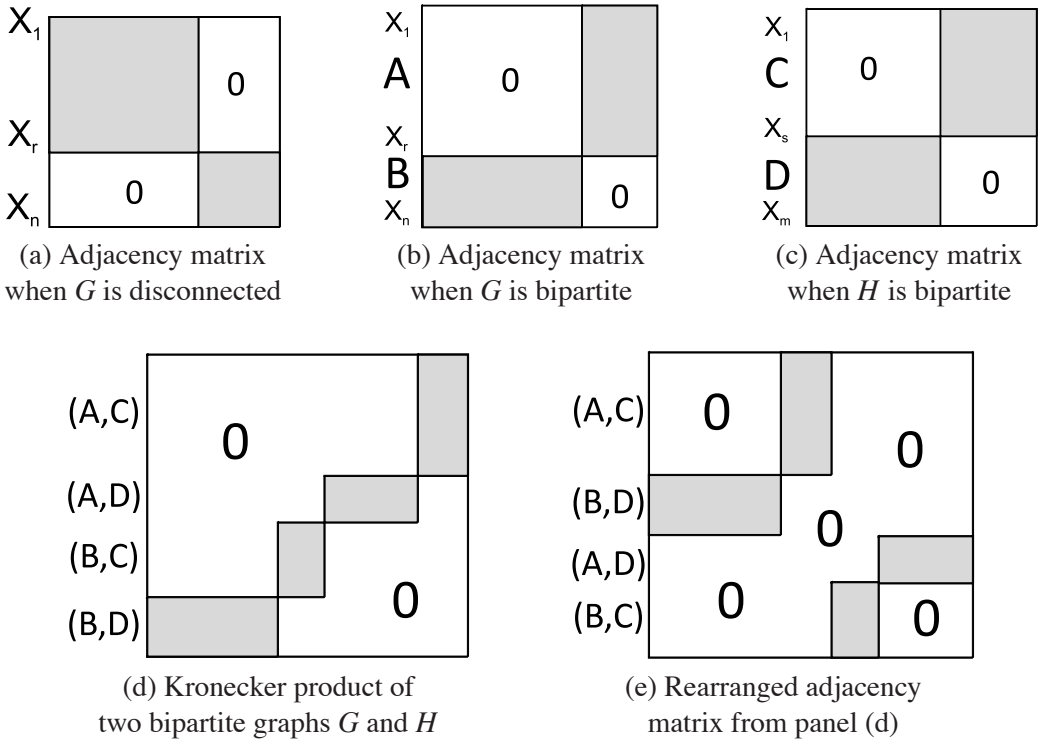
Proof Let K_1 have the eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_{N_1}$. By properties of the Kronecker multiplication (Loan, 2000; Langville and Stewart, 2004), the eigenvectors of K_k are given by the k^{th} Kronecker power of the vector: $(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_{N_1})$, which gives a multinomial distribution for the components of each eigenvector in K_k. ■

We have just covered several of the static graph patterns. Notice that the proofs were a direct consequences of the Kronecker multiplication properties.

3.2.3 CONNECTIVITY OF KRONECKER GRAPHS

We now present a series of results on the connectivity of Kronecker graphs. We show, maybe a bit surprisingly, that even if a Kronecker initiator graph is connected its Kronecker power can in fact be disconnected.

Lemma 8 *If at least one of G and H is a disconnected graph, then $G \otimes H$ is also disconnected.*



(a) Adjacency matrix when G is disconnected

(b) Adjacency matrix when G is bipartite

(c) Adjacency matrix when H is bipartite

(d) Kronecker product of two bipartite graphs G and H

(e) Rearranged adjacency matrix from panel (d)

Figure 4: Graph adjacency matrices. Dark parts represent connected (filled with ones) and white parts represent empty (filled with zeros) parts of the adjacency matrix. (a) When G is disconnected, Kronecker multiplication with any matrix H will result in $G \otimes H$ being disconnected. (b) Adjacency matrix of a connected bipartite graph G with node partitions A and B. (c) Adjacency matrix of a connected bipartite graph G with node partitions C and D. (e) Kronecker product of two bipartite graphs G and H. (d) After rearranging the adjacency matrix $G \otimes H$ we clearly see the resulting graph is disconnected.

Proof Without loss of generality we can assume that G has two connected components, while H is connected. Figure 4(a) illustrates the corresponding adjacency matrix for G. Using the notation from Observation 1 let graph let G have nodes $X_1, \ldots, X_n$, where nodes $\{X_1, \ldots X_r\}$ and $\{X_{r+1}, \ldots, X_n\}$ form the two connected components. Now, note that $(X_{ij}, X_{kl}) \notin G \otimes H$ for $i \in \{1, \ldots, r\}$, $k \in \{r+1, \ldots, n\}$, and all j, l. This follows directly from Observation 1 as (X_i, X_k) are not edges in G. Thus, $G \otimes H$ must at least two connected components. ■

Actually it turns out that both G and H can be connected while $G \otimes H$ is disconnected. The following theorem analyzes this case.

Theorem 9 *If both G and H are connected but bipartite, then $G \otimes H$ is disconnected, and each of the two connected components is again bipartite.*

Proof Again without loss of generality let G be bipartite with two partitions $A = \{X_1, \ldots X_r\}$ and $B = \{X_{r+1}, \ldots, X_n\}$, where edges exists only between the partitions, and no edges exist inside the

partition: $(X_i, X_k) \notin G$ for $i,k \in A$ or $i,k \in B$. Similarly, let H also be bipartite with two partitions $C = \{X_1, \ldots X_s\}$ and $D = \{X_{s+1}, \ldots, X_m\}$. Figures 4(b) and (c) illustrate the structure of the corresponding adjacency matrices.

Now, there will be two connected components in $G \otimes H$: 1^{st} component will be composed of nodes $\{X_{ij}\} \in G \otimes H$, where $(i \in A, j \in D)$ or $(i \in B, j \in C)$. And similarly, 2^{nd} component will be composed of nodes $\{X_{ij}\}$, where $(i \in A, j \in C)$ or $(i \in B, j \in D)$. Basically, there exist edges between node sets (A,D) and (B,C), and similarly between (A,C) and (B,D) but not across the sets. To see this we have to analyze the cases using Observation 1. For example, in $G \otimes H$ there exist edges between nodes (A,C) and (B,D) as there exist edges $(i,k) \in G$ for $i \in A, k \in B$, and $(j,l) \in H$ for $j \in C$ and $l \in D$. Similar is true for nodes (A,C) and (B,D). However, there are no edges cross the two sets, for example, nodes from (A,D) do not link to (A,C), as there are no edges between nodes in A (since G is bipartite). See Figures 4(d) and 4(e) for a visual proof. ■

Note that bipartite graphs are triangle free and have no self-loops. Stars, chains, trees and cycles of even length are all examples of bipartite graphs. In order to ensure that K_k is connected, for the remained of the paper we focus on initiator graphs K_1 with self loops on all of the vertices.

3.2.4 TEMPORAL PROPERTIES OF KRONECKER GRAPHS

We continue with the analysis of temporal patterns of evolution of Kronecker graphs: the densification power law, and shrinking/stabilizing diameter (Leskovec et al., 2005b, 2007a).

Theorem 10 (Densification power law) *Kronecker graphs follow the densification power law (DPL) with densification exponent* $a = \log(E_1)/\log(N_1)$.

Proof Since the k^{th} Kronecker power K_k has $N_k = N_1^k$ nodes and $E_k = E_1^k$ edges, it satisfies $E_k = N_k^a$, where $a = \log(E_1)/\log(N_1)$. The crucial point is that a is independent of k, and hence the sequence of Kronecker powers follows an exact version of the densification power law. ■

We now show how the Kronecker product also preserves the property of constant diameter, a crucial ingredient for matching the diameter properties of many real-world network data sets. In order to establish this, we will assume that the initiator graph K_1 has a self-loop on every node. Otherwise, its Kronecker powers may be disconnected.

Lemma 11 *If G and H each have diameter at most D and each has a self-loop on every node, then the Kronecker graph $G \otimes H$ also has diameter at most D.*

Proof Each node in $G \otimes H$ can be represented as an ordered pair (v,w), with v a node of G and w a node of H, and with an edge joining (v,w) and (x,y) precisely when (v,x) is an edge of G and (w,y) is an edge of H. (Note this exactly the Observation 1.) Now, for an arbitrary pair of nodes (v,w) and (v',w'), we must show that there is a path of length at most D connecting them. Since G has diameter at most D, there is a path $v = v_1, v_2, \ldots, v_r = v'$, where $r \leq D$. If $r < D$, we can convert this into a path $v = v_1, v_2, \ldots, v_D = v'$ of length exactly D, by simply repeating v' at the end for $D - r$ times. By an analogous argument, we have a path $w = w_1, w_2, \ldots, w_D = w'$. Now by the definition of the Kronecker product, there is an edge joining (v_i, w_i) and (v_{i+1}, w_{i+1}) for all $1 \leq i \leq D-1$, and so $(v,w) = (v_1,w_1), (v_2,w_2), \ldots, (v_D,w_D) = (v',w')$ is a path of length D connecting (v,w) to

(v', w'), as required. ■

Theorem 12 *If K_1 has diameter D and a self-loop on every node, then for every k, the graph K_k also has diameter D.*

Proof This follows directly from the previous lemma, combined with induction on k. ■

As defined in Section 2 we also consider the *effective diameter* D^*. We defined the q-effective diameter as the minimum D^* such that, for a q fraction of the reachable node pairs, the path length is at most D^*. The q-effective diameter is a more robust quantity than the diameter, the latter being prone to the effects of degenerate structures in the graph (e.g., very long chains). However, the q-effective diameter and diameter tend to exhibit qualitatively similar behavior. For reporting results in subsequent sections, we will generally consider the q-effective diameter with $q = 0.9$, and refer to this simply as the *effective diameter*.

Theorem 13 (Effective diameter) *If K_1 has diameter D and a self-loop on every node, then for every q, the q-effective diameter of K_k converges to D (from below) as k increases.*

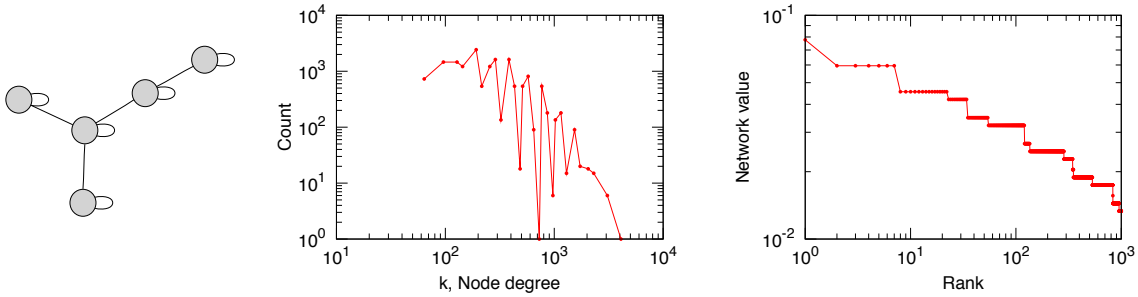
Proof To prove this, it is sufficient to show that for two randomly selected nodes of K_k, the probability that their distance is D converges to 1 as k goes to infinity.

We establish this as follows. Each node in K_k can be represented as an ordered sequence of k nodes from K_1, and we can view the random selection of a node in K_k as a sequence of k independent random node selections from K_1. Suppose that $v = (v_1, \ldots, v_k)$ and $w = (w_1, \ldots, w_k)$ are two such randomly selected nodes from K_k. Now, if x and y are two nodes in K_1 at distance D (such a pair (x, y) exists since K_1 has diameter D), then with probability $1 - (1 - \frac{1}{N_1^2})^k$, there is some index j for which $\{v_j, w_j\} = \{x, y\}$. If there is such an index, then the distance between v and w is D. As the expression $1 - (1 - \frac{1}{N_1^2})^k$ converges to 1 as k increases, it follows that the q-effective diameter is converging to D. ■

3.3 Stochastic Kronecker Graphs

While the Kronecker power construction discussed so far yields graphs with a range of desired properties, its discrete nature produces "staircase effects" in the degrees and spectral quantities, simply because individual values have large multiplicities. For example, degree distribution and distribution of eigenvalues of graph adjacency matrix and the distribution of the principal eigenvector components (i.e., the "network" value) are all impacted by this. These quantities are multinomially distributed which leads to individual values with large multiplicities. Figure 5 illustrates the staircase effect.

Here we propose a stochastic version of Kronecker graphs that eliminates this effect. There are many possible ways how one could introduce stochasticity into Kronecker graph model. Before introducing the proposed model, we introduce two simple ways of introducing randomness to Kronecker graphs and describe why they do not work.



(a) Kronecker initiator K_1 (b) Degree distribution of K_6 (6^{th} Kronecker power of K_1) (c) Network value of K_6 (6^{th} Kronecker power of K_1)

Figure 5: The "staircase" effect. Kronecker initiator and the degree distribution and network value plot for the 6^{th} Kronecker power of the initiator. Notice the non-smoothness of the curves.

Probably the simplest (but wrong) idea is to generate a large deterministic Kronecker graph K_k, and then uniformly at random flip some edges, that is, uniformly at random select entries of the graph adjacency matrix and flip them ($1 \rightarrow 0, 0 \rightarrow 1$). However, this will not work, as it will essentially superimpose an Erdős-Rényi random graph, which would, for example, corrupt the degree distribution—real networks usually have heavy tailed degree distributions, while random graphs have Binomial degree distributions. A second idea could be to allow a weighted initiator matrix, that is, values of entries of K_1 are not restricted to values $\{0,1\}$ but rather can be any non-negative real number. Using such K_1 one would generate K_k and then threshold the K_k matrix to obtain a binary adjacency matrix K, that is, for a chosen value of ε set $K[i,j] = 1$ if $K_k[i,j] > \varepsilon$ else $K[i,j] = 0$. This also would not work as the mechanism would selectively remove edges and thus the low degree nodes which would have low weight edges would get isolated first.

Now we define *Stochastic Kronecker graph model* which overcomes the above issues. A more natural way to introduce stochasticity to Kronecker graphs is to relax the assumption that entries of the initiator matrix take only binary values. Instead we allow entries of the initiator to take values on the interval $[0,1]$. This means now each entry of the initiator matrix encodes the probability of that particular edge appearing. We then Kronecker-power such initiator matrix to obtain a large stochastic adjacency matrix, where again each entry of the large matrix gives the probability of that particular edge appearing in a big graph. Such a stochastic adjacency matrix defines a probability distribution over all graphs. To obtain a graph we simply sample an instance from this distribution by sampling individual edges, where each edge appears independently with probability given by the entry of the large stochastic adjacency matrix. More formally, we define:

Definition 14 (Stochastic Kronecker graph) *Let $\mathcal{P}_1$ be a $N_1 \times N_1$* probability matrix*: the value $\theta_{ij} \in \mathcal{P}_1$ denotes the probability that edge (i,j) is present, $\theta_{ij} \in [0,1]$.*

Then k^{th} Kronecker power $\mathcal{P}_1^{[k]} = \mathcal{P}_k$, where each entry $p_{uv} \in \mathcal{P}_k$ encodes the probability of an edge (u,v).

To obtain a graph, an instance *(or* realization*), $K = R(\mathcal{P}_k)$ we include edge (u,v) in K with probability p_{uv}, $p_{uv} \in \mathcal{P}_k$.*

First, note that sum of the entries of $\mathcal{P}_1$, $\sum_{ij} \theta_{ij}$, can be greater than 1. Second, notice that in principle it takes $O(N_1^{2k})$ time to generate an instance K of a Stochastic Kronecker graph from the

probability matrix $\mathcal{P}_k$. This means the time to get a realization K is quadratic in the size of $\mathcal{P}_k$ as one has to flip a coin for each possible edge in the graph. Later we show how to generate Stochastic Kronecker graphs much faster, in the time *linear* in the expected number of edges in $\mathcal{P}_k$.

3.3.1 PROBABILITY OF AN EDGE

For the size of graphs we aim to model and generate here taking $\mathcal{P}_1$ (or K_1) and then explicitly performing the Kronecker product of the initiator matrix is infeasible. The reason for this is that $\mathcal{P}_1$ is usually dense, so $\mathcal{P}_k$ is also dense and one can not explicitly store it in memory to directly iterate the Kronecker product. However, due to the structure of Kronecker multiplication one can easily compute the probability of an edge in $\mathcal{P}_k$.

The probability p_{uv} of an edge (u,v) occurring in k-th Kronecker power $\mathcal{P} = \mathcal{P}_k$ can be calculated in $O(k)$ time as follows:

$$p_{uv} = \prod_{i=0}^{k-1} \mathcal{P}_1 \left[\left\lfloor \frac{u-1}{N_1^i} \right\rfloor (\mathrm{mod} N_1) + 1, \left\lfloor \frac{v-1}{N_1^i} \right\rfloor (\mathrm{mod} N_1) + 1 \right]. \quad (1)$$

The equation imitates recursive descent into the matrix $\mathcal{P}$, where at every level i the appropriate entry of $\mathcal{P}_1$ is chosen. Since $\mathcal{P}$ has N_1^k rows and columns it takes $O(k \log N_1)$ to evaluate the equation. Refer to Figure 6 for the illustration of the recursive structure of $\mathcal{P}$.

3.4 Additional Properties of Kronecker Graphs

Stochastic Kronecker graphs with initiator matrix of size $N_1 = 2$ were studied by Mahdian and Xu (2007). The authors showed a phase transition for the emergence of the giant component and another phase transition for connectivity, and proved that such graphs have constant diameters beyond the connectivity threshold, but are not searchable using a decentralized algorithm (Kleinberg, 1999).

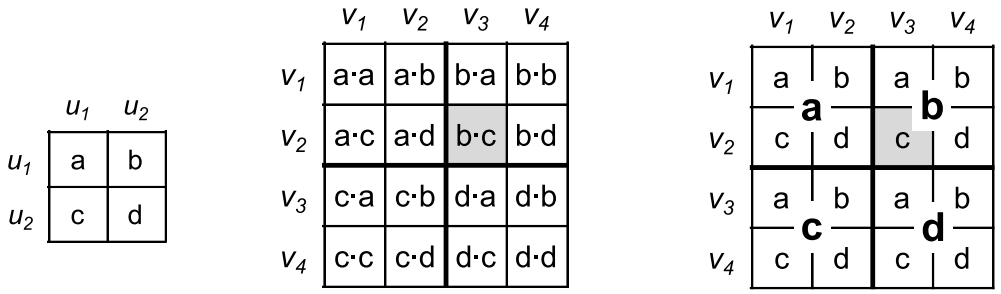
General overview of Kronecker product is given in Imrich and Klavžar (2000) and properties of Kronecker graphs related to graph minors, planarity, cut vertex and cut edge have been explored in Bottreau and Metivier (1998). Moreover, recently Tsourakakis (2008) gave a closed form expression for the number of triangles in a Kronecker graph that depends on the eigenvalues of the initiator graph K_1.

3.5 Two Interpretations of Kronecker Graphs

Next, we present two natural interpretations of the generative process behind the Kronecker graphs that go beyond the purely mathematical construction of Kronecker graphs as introduced so far.

We already mentioned the first interpretation when we first defined Kronecker graphs. One intuition is that networks are hierarchically organized into communities (clusters). Communities then grow recursively, creating miniature copies of themselves. Figure 1 depicts the process of the recursive community expansion. In fact, several researchers have argued that real networks are hierarchically organized (Ravasz et al., 2002; Ravasz and Barabási, 2003) and algorithms to extract the network hierarchical structure have also been developed (Sales-Pardo et al., 2007; Clauset et al., 2008). Moreover, especially web graphs (Dill et al., 2002; Dorogovtsev et al., 2002; Crovella and Bestavros, 1997) and biological networks (Ravasz and Barabási, 2003) were found to be self-similar and "fractal".

The second intuition comes from viewing every node of $\mathcal{P}_k$ as being described with an ordered sequence of k nodes from $\mathcal{P}_1$. (This is similar to the Observation 1 and the proof of Theorem 13.)



(a) 2×2 Stochastic Kronecker initiator $\mathcal{P}_1$

(b) Probability matrix $\mathcal{P}_2 = \mathcal{P}_1 \otimes \mathcal{P}_1$

(c) Alternative view of $\mathcal{P}_2 = \mathcal{P}_1 \otimes \mathcal{P}_1$

Figure 6: Stochastic Kronecker initiator $\mathcal{P}_1$ and the corresponding 2^{nd} Kronecker power $\mathcal{P}_2$. Notice the recursive nature of the Kronecker product, with edge probabilities in $\mathcal{P}_2$ simply being products of entries of $\mathcal{P}_1$.

Let's label nodes of the initiator matrix $\mathcal{P}_1$, $u_1, \ldots, u_{N_1}$, and nodes of $\mathcal{P}_k$ as $v_1, \ldots, v_{N_1^k}$. Then every node v_i of $\mathcal{P}_k$ is described with a sequence $(v_i(1), \ldots, v_i(k))$ of node labels of $\mathcal{P}_1$, where $v_i(l) \in \{u_1, \ldots, u_k\}$. Similarly, consider also a second node v_j with the label sequence $(v_j(1), \ldots, v_j(k))$. Then the probability p_e of an edge (v_i, v_j) in $\mathcal{P}_k$ is exactly:

$$p_e(v_i, v_j) = \mathcal{P}_k[v_i, v_j] = \prod_{l=1}^{k} \mathcal{P}_1[v_i(l), v_j(l)].$$

(Note this is exactly the Equation 1.)

Now one can look at the description sequence of node v_i as a k dimensional vector of attribute values $(v_i(1), \ldots, v_i(k))$. Then $p_e(v_i, v_j)$ is exactly the coordinate-wise product of appropriate entries of $\mathcal{P}_1$, where the node description sequence selects which entries of $\mathcal{P}_1$ to multiply. Thus, the $\mathcal{P}_1$ matrix can be thought of as the attribute similarity matrix, that is, it encodes the probability of linking given that two nodes agree/disagree on the attribute value. Then the probability of an edge is simply a product of individual attribute similarities over the k N_1-valued attributes that describe each of the two nodes.

This gives us a very natural interpretation of Stochastic Kronecker graphs: Each node is described by a sequence of categorical attribute values or features. And then the probability of two nodes linking depends on the product of individual attribute similarities. This way Kronecker graphs can effectively model homophily (nodes with similar attribute values are more likely to link) by $\mathcal{P}_1$ having high value entries on the diagonal. Or heterophily (nodes that differ are more likely to link) by $\mathcal{P}_1$ having high entries off the diagonal.

Figure 6 shows an example. Let's label nodes of $\mathcal{P}_1$ u_1, u_2 as in Figure 6(a). Then every node of $\mathcal{P}_k$ is described with an ordered sequence of k binary attributes. For example, Figure 6(b) shows an instance for $k = 2$ where node v_2 of $\mathcal{P}_2$ is described by (u_1, u_2), and similarly v_3 by (u_2, u_1). Then as shown in Figure 6(b), the probability of edge $p_e(v_2, v_3) = b \cdot c$, which is exactly $\mathcal{P}_1[u_2, u_1] \cdot \mathcal{P}_1[u_1, u_2] = b \cdot c$—the product of entries of $\mathcal{P}_1$, where the corresponding elements of the description of nodes v_2 and v_3 act as selectors of which entries of $\mathcal{P}_1$ to multiply.

Figure 6(c) further illustrates the recursive nature of Kronecker graphs. One can see Kronecker product as recursive descent into the big adjacency matrix where at each stage one of the entries

or blocks is chosen. For example, to get to entry (v_2, v_3) one first needs to dive into quadrant b following by the quadrant c. This intuition will help us in Section 3.6 to devise a fast algorithm for generating Kronecker graphs.

However, there are also two notes to make here. First, using a single initiator $\mathcal{P}_1$ we are implicitly assuming that there is one single and universal attribute similarity matrix that holds across all k N_1-ary attributes. One can easily relax this assumption by taking a different initiator matrix for each attribute (initiator matrices can even be of different sizes as attributes are of different arity), and then Kronecker multiplying them to obtain a large network. Here each initiator matrix plays the role of attribute similarity matrix for that particular attribute.

For simplicity and convenience we will work with a single initiator matrix but all our methods can be trivially extended to handle multiple initiator matrices. Moreover, as we will see later in Section 6 even a single 2×2 initiator matrix seems to be enough to capture large scale statistical properties of real-world networks.

The second assumption is harder to relax. When describing every node v_i with a sequence of attribute values we are implicitly assuming that the values of all attributes are uniformly distributed (have same proportions), and that every node has a unique combination of attribute values. So, all possible combinations of attribute values are taken. For example, node v_1 in a large matrix $\mathcal{P}_k$ has attribute sequence $(u_1, u_1, \ldots, u_1)$, and v_{N_1} has $(u_1, u_1, \ldots, u_1, u_{N_1})$, while the "last" node $v_{N_1^k}$ is has attribute values $(u_{N_1}, u_{N_1}, \ldots, u_{N_1})$. One can think of this as counting in N_1-ary number system, where node attribute descriptions range from 0 (i.e., "leftmost" node with attribute description $(u_1, u_1, \ldots, u_1)$) to N_1^k (i.e., "rightmost" node attribute description $(u_{N_1}, u_{N_1}, \ldots, u_{N_1})$).

A simple way to relax the above assumption is to take a larger initiator matrix with a smaller number of parameters than the number of entries. This means that multiple entries of $\mathcal{P}_1$ will share the same value (parameter). For example, if attribute u_1 takes one value 66% of the times, and the other value 33% of the times, then one can model this by taking a 3×3 initiator matrix with only four parameters. Adopting the naming convention of Figure 6 this means that parameter a now occupies a 2×2 block, which then also makes b and c occupy 2×1 and 1×2 blocks, and d a single cell. This way one gets a four parameter model with uneven feature value distribution.

We note that the view of Kronecker graphs where every node is described with a set of features and the initiator matrix encodes the probability of linking given the attribute values of two nodes somewhat resembles the Random dot product graph model (Young and Scheinerman, 2007; Nickel, 2008). The important difference here is that we multiply individual linking probabilities, while in Random dot product graphs one takes the sum of individual probabilities which seems somewhat less natural.

3.6 Fast Generation of Stochastic Kronecker Graphs

Generating a Stochastic Kronecker graph K on N nodes naively takes $O(N^2)$ time. Here we present a fast heuristic procedure that works well in practice and takes time *linear* in the number of edges to generate a graph. The intuition for fast generation of Stochastic Kronecker graphs comes from the recursive nature of the Kronecker product and is closely related to the R-MAT graph generator (Chakrabarti et al., 2004). In contrast to R-MAT Stochastic Kronecker graph initiator matrix encodes both the total number of edges in a graph and their structure. $\sum \theta_{ij}$ encodes the number of edges in the graph, while the proportions (ratios) of values θ_{ij} define how many edges each part of graph adjacency matrix will contain.

In order to illustrate the recursive nature of Kronecker graphs and to highlight the relation to R-MAT graph generator Figure 6 shows how probability matrix in panel (b) can be recursively decomposed as shown in panel (c) of Figure 6. To "arrive" to a particular edge (v_i, v_j) of $\mathcal{P}_k$ one has to make a sequence of k (in our case $k = 2$) decisions among the entries of $\mathcal{P}_1$, multiply the chosen entries of $\mathcal{P}_1$, and then placing the edge (v_i, v_j) with the obtained probability.

Thus, instead of flipping $O(N^2) = O(N_1^{2k})$ biased coins to determine the edges in a graph, we place E edges by directly simulating the recursion of the Kronecker product. Basically, we recursively choose sub-regions of matrix K with probability proportional to θ_{ij}, $\theta_{ij} \in \mathcal{P}_1$ until in k steps we descend to a single cell of the big adjacency matrix K and place an edge. For example, for (v_2, v_3) in Figure 6(c) we first have to choose b following by c.

More generally, the probability of each individual edge of $\mathcal{P}_k$ follows a Bernoulli distribution, as the edge occurrences are independent. By the Central Limit Theorem (Petrov, 1995) the number of edges in $\mathcal{P}_k$ tends to a normal distribution with mean $(\sum_{i,j=1}^{N_1} \theta_{ij})^k = E_1^k$, where $\theta_{ij} \in \mathcal{P}_1$. So, given a stochastic initiator matrix $\mathcal{P}_1$ we first sample the expected number of edges E in $\mathcal{P}_k$. Then we place E edges in a graph K, by applying the recursive descent for k steps where at each step we choose entry (i, j) with probability θ_{ij}/E_1 where $E_1 = \sum_{ij} \theta_{ij}$. Since we add $E = E_1^k$ edges, the probability that edge (v_i, v_j) appears in K is exactly $\mathcal{P}_k[v_i, v_j]$. However, in practice it can happen that more than one edge lands in the same (v_i, v_j) entry of big adjacency matrix K. If an edge lands in a already occupied cell we simply insert it again. Even though values of $\mathcal{P}_1$ are usually skewed, adjacency matrices of real networks are so sparse that collisions are not really a problem in practice as only around 1% of edges collide. It is due to these edge collisions the above procedure does not obtain exact samples from the graph distribution defined by the parameter matrix $\mathcal{P}$. However, in practice graphs generated by this fast linear time ($O(E)$) procedure are basically indistinguishable from graphs generated with the exact exponential time ($O(N^2)$) procedure.

Code for generating Kronecker graphs can be obtained at `http://snap.stanford.edu`.

3.7 Observations and Connections

Next, we describe several observations about the properties of Kronecker graphs and make connections to other network models.

- *Bipartite graphs:* Kronecker graphs can naturally model bipartite graphs. Instead of starting with a square $N_1 \times N_1$ initiator matrix, one can choose arbitrary $N_1 \times M_1$ initiator matrix, where rows define "left", and columns the "right" side of the bipartite graph. Kronecker multiplication will then generate bipartite graphs with partition sizes N_1^k and M_1^k.

- *Graph distributions:* $\mathcal{P}_k$ defines a distribution over all graphs, as it encodes the probability of all possible N_1^{2k} edges appearing in a graph by using an exponentially smaller number of parameters (just N_1^2). As we will later see, even a very small number of parameters, for example, 4 (2×2 initiator matrix) or 9 (3×3 initiator), is enough to accurately model the structure of large networks.

- *Extension of Erdős-Rényi random graph model:* Stochastic Kronecker graphs represent an extension of Erdős-Rényi (Erdős and Rényi, 1960) random graphs. If one takes $\mathcal{P}_1 = [\theta_{ij}]$, where every $\theta_{ij} = p$ then we obtain exactly the Erdős-Rényi model of random graphs $G_{n,p}$, where every edge appears independently with probability p.

- *Relation to the R-MAT model:* The recursive nature of Stochastic Kronecker graphs makes them related to the R-MAT generator (Chakrabarti et al., 2004). The difference between the two models is that in R-MAT one needs to separately specify the number of edges, while in Stochastic Kronecker graphs initiator matrix $\mathcal{P}_1$ also encodes the number of edges in the graph. Section 3.6 built on this similarity to devise a fast algorithm for generating Stochastic Kronecker graphs.

- *Densification:* Similarly as with deterministic Kronecker graphs the number of nodes in a Stochastic Kronecker graph grows as N_1^k, and the expected number of edges grows as $(\sum_{ij}\theta_{ij})^k$. This means one would want to choose values θ_{ij} of the initiator matrix $\mathcal{P}_1$ so that $\sum_{ij}\theta_{ij} > N_1$ in order for the resulting network to densify.

4. Simulations of Kronecker Graphs

Next we perform a set of simulation experiments to demonstrate the ability of Kronecker graphs to match the patterns of real-world networks. We will tackle the problem of estimating the Kronecker graph model from real data, that is, finding the most likely initiator $\mathcal{P}_1$, in the next section. Instead here we present simulation experiments using Kronecker graphs to explore the parameter space, and to compare properties of Kronecker graphs to those found in large real networks.

4.1 Comparison to Real Graphs

We observe two kinds of graph patterns—"static" and "temporal." As mentioned earlier, common static patterns include degree distribution, scree plot (eigenvalues of graph adjacency matrix vs. rank) and distribution of components of the principal eigenvector of graph adjacency matrix. Temporal patterns include the diameter over time, and the densification power law. For the diameter computation, we use the effective diameter as defined in Section 2.

For the purpose of this section consider the following setting. Given a real graph G we want to find Kronecker initiator that produces qualitatively similar graph. In principle one could try choosing each of the N_1^2 parameters for the matrix $\mathcal{P}_1$ separately. However, we reduce the number of parameters from N_1^2 to just two: α and β. Let K_1 be the initiator matrix (binary, deterministic). Then we create the corresponding stochastic initiator matrix $\mathcal{P}_1$ by replacing each "1" and "0" of K_1 with α and β respectively ($\beta \leq \alpha$). The resulting probability matrices maintain—with some random noise—the self-similar structure of the Kronecker graphs in the previous section (which, for clarity, we call *deterministic Kronecker graphs*). We defer the discussion of how to automatically estimate $\mathcal{P}_1$ from data G to the next section.

The data sets we use here are:

- CIT-HEP-TH: This is a citation graph for High-Energy Physics Theory research papers from pre-print archive ArXiv, with a total of N =29,555 papers and E =352,807 citations (Gehrke et al., 2003). We follow its evolution from January 1993 to April 2003, with one data-point per month.

- AS-ROUTEVIEWS: We also analyze a static data set consisting of a single snapshot of connectivity among Internet Autonomous Systems (RouteViews, 1997) from January 2000, with N =6,474 and E =26,467.

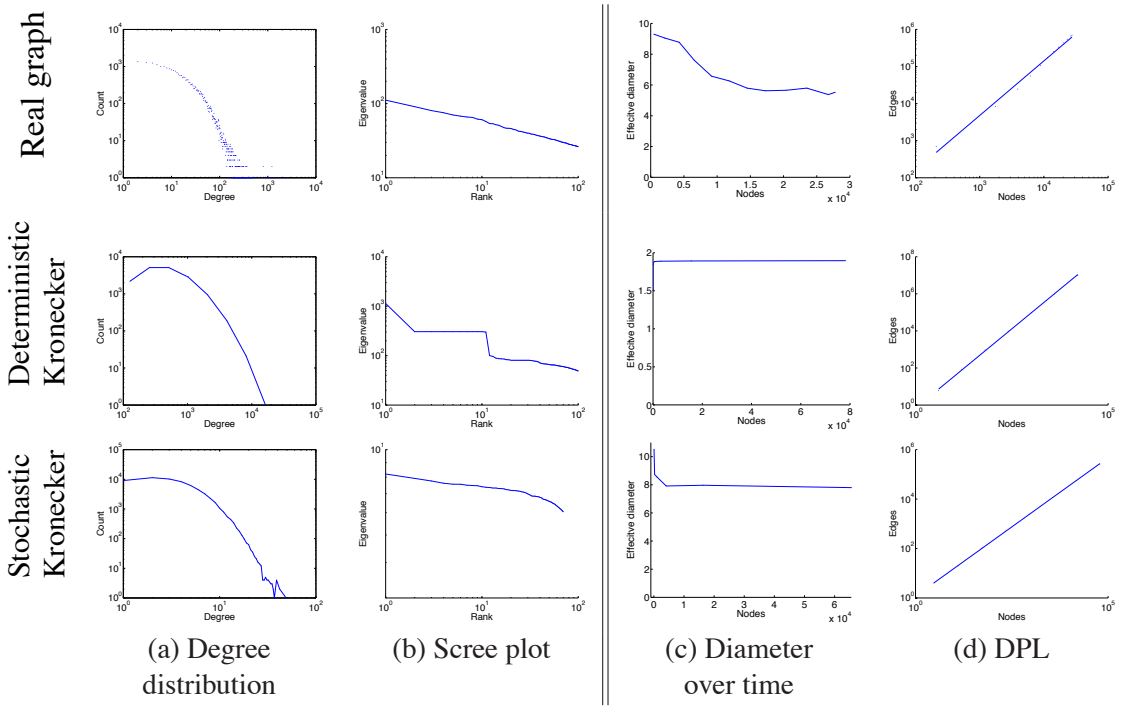
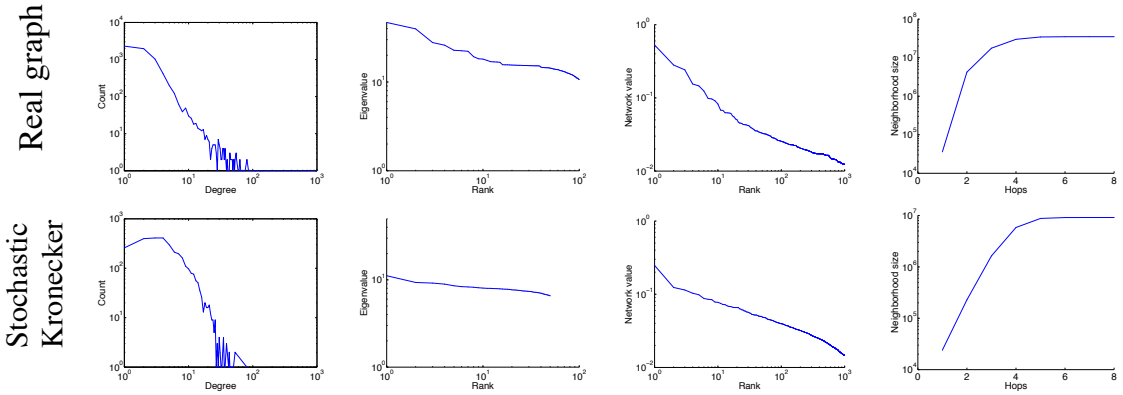


Figure 7: *Citation network (*CIT-HEP-TH*):* Patterns from the real graph (top row), the deterministic Kronecker graph with K_1 being a star graph on 4 nodes (center + 3 satellites) (middle row), and the Stochastic Kronecker graph ($\alpha = 0.41$, $\beta = 0.11$ – bottom row). *Static* patterns: (a) is the PDF of degrees in the graph (log-log scale), and (b) the distribution of eigenvalues (log-log scale). *Temporal* patterns: (c) gives the effective diameter over time (linear-linear scale), and (d) is the number of edges versus number of nodes over time (log-log scale). Notice that the Stochastic Kronecker graphs qualitatively matches all the patterns very well.

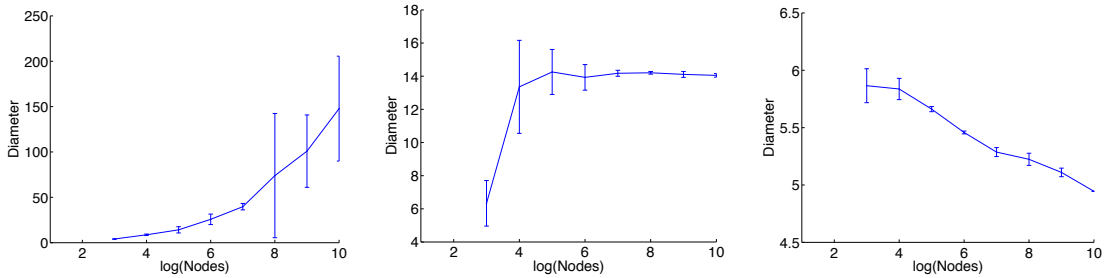
Results are shown in Figure 7 for the CIT-HEP-TH graph which evolves over time. We show the plots of two static and two temporal patterns. We see that the deterministic Kronecker model already to some degree captures the qualitative structure of the degree and eigenvalue distributions, as well as the temporal patterns represented by the densification power law and the stabilizing diameter. However, the deterministic nature of this model results in "staircase" behavior, as shown in scree plot for the deterministic Kronecker graph of Figure 7 (column (b), second row). We see that the Stochastic Kronecker graphs smooth out these distributions, further matching the qualitative structure of the real data, and they also match the shrinking-before-stabilization trend of the diameters of real graphs.

Similarly, Figure 8 shows plots for the static patterns in the *Autonomous systems* (AS-ROUTEVIEWS) graph. Recall that we analyze a single, static network snapshot in this case. In addition to the degree distribution and scree plot, we also show two typical plots (Chakrabarti et al., 2004): the distribution of *network values* (principal eigenvector components, sorted, versus rank) and the *hop-plot* (the number of reachable pairs $g(h)$ within h hops or less, as a function of the



(a) Degree distribution (b) Scree plot (c) "Network value" distribution (d) "Hop-plot"

Figure 8: *Autonomous systems (*AS-ROUTEVIEWS*):* Real (top) versus Kronecker (bottom). Columns (a) and (b) show the degree distribution and the scree plot, as before. Columns (c) and (d) show two more static patterns (see text). Notice that, again, the Stochastic Kronecker graph matches well the properties of the real graph.



(a) Increasing diameter $\alpha = 0.38, \beta = 0$ (b) Constant diameter $\alpha = 0.43, \beta = 0$ (c) Decreasing diameter $\alpha = 0.54, \beta = 0$

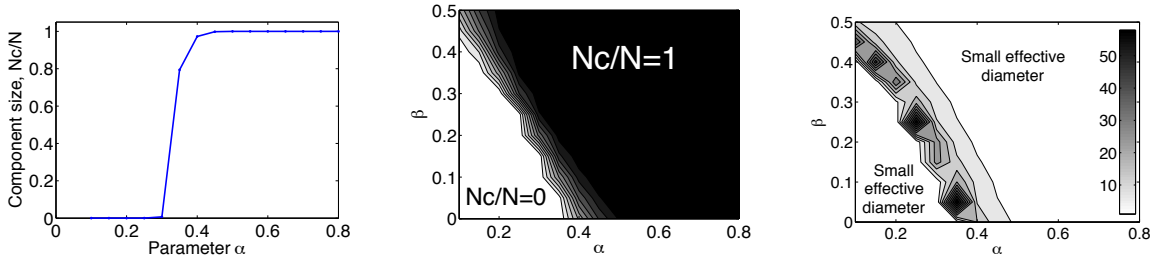
Figure 9: Effective diameter over time for a 4-node chain initiator graph. After each consecutive Kronecker power we measure the effective diameter. We use different settings of α parameter. $\alpha = 0.38, 0.43, 0.54$ and $\beta = 0$, respectively.

number of hops h). Notice that, again, the Stochastic Kronecker graph matches well the properties of the real graph.

4.2 Parameter Space of Kronecker Graphs

Last we present simulation experiments that investigate the parameter space of Stochastic Kronecker graphs.

First, in Figure 9 we show the ability of Kronecker Graphs to generate networks with increasing, constant and decreasing/stabilizing effective diameter. We start with a 4-node chain initiator graph (shown in top row of Figure 3), setting each "1" of K_1 to α and each "0" to $\beta = 0$ to obtain $\mathcal{P}_1$ that we then use to generate a growing sequence of graphs. We plot the effective diameter of each



(a) Largest component size (b) Largest component size (c) Effective diameter

Figure 10: Fraction of nodes in the largest weakly connected component (N_c/N) and the effective diameter for 4-star initiator graph. (a) We fix $\beta = 0.15$ and vary α. (b) We vary both α and β. (c) Effective diameter of the network, if network is disconnected or very dense path lengths are short, the diameter is large when the network is barely connected.

$R(\mathcal{P}_k)$ as we generate a sequence of growing graphs $R(\mathcal{P}_2), R(\mathcal{P}_3), \ldots, R(\mathcal{P}_{10})$. $R(\mathcal{P}_{10})$ has exactly 1,048,576 nodes. Notice Stochastic Kronecker graphs is a very flexible model. When the generated graph is very sparse (low value of α) we obtain graphs with slowly increasing effective diameter (Figure 9(a)). For intermediate values of α we get graphs with constant diameter (Figure 9(b)) and that in our case also slowly densify with densification exponent $a = 1.05$. Last, we see an example of a graph with shrinking/stabilizing effective diameter. Here we set the $\alpha = 0.54$ which results in a densification exponent of $a = 1.2$. Note that these observations are not contradicting Theorem 11. Actually, these simulations here agree well with the analysis of Mahdian and Xu (2007).

Next, we examine the parameter space of a Stochastic Kronecker graphs where we choose a star on 4 nodes as a initiator graph and parameterized with α and β as before. The initiator graph and the structure of the corresponding (deterministic) Kronecker graph adjacency matrix is shown in top row of Figure 3.

Figure 10(a) shows the sharp transition in the fraction of the number of nodes that belong to the largest weakly connected component as we fix $\beta = 0.15$ and slowly increase α. Such phase transitions on the size of the largest connected component also occur in Erdős-Rényi random graphs. Figure 10(b) further explores this by plotting the fraction of nodes in the largest connected component (N_c/N) over the full parameter space. Notice a sharp transition between disconnected (white area) and connected graphs (dark).

Last, Figure 10(c) shows the effective diameter over the parameter space (α, β) for the 4-node star initiator graph. Notice that when parameter values are small, the effective diameter is small, since the graph is disconnected and not many pairs of nodes can be reached. The shape of the transition between low-high diameter closely follows the shape of the emergence of the connected component. Similarly, when parameter values are large, the graph is very dense, and the diameter is small. There is a narrow band in parameter space where we get graphs with interesting diameters.

5. Kronecker Graph Model Estimation

In previous sections we investigated various properties of networks generated by the (Stochastic) Kronecker graphs model. Many of these properties were also observed in real networks. Moreover, we also gave closed form expressions (parametric forms) for values of these statistical network

properties, which allow us to calculate a property (e.g., diameter, eigenvalue spectrum) of a network directly from just the initiator matrix. So in principle, one could invert these equations and directly get from a property (e.g., shape of degree distribution) to the values of initiator matrix.

However, in previous sections we did not say anything about how various network properties of a Kronecker graph correlate and interdepend. For example, it could be the case that two network properties are mutually exclusive. For instance, perhaps only could only match the network diameter but not the degree distribution or vice versa. However, as we show later this is not the case.

Now we turn our attention to automatically estimating the Kronecker initiator graph. The setting is that we are given a real network G and would like to find a Stochastic Kronecker initiator $\mathcal{P}_1$ that produces a synthetic Kronecker graph K that is "similar" to G. One way to measure similarity is to compare statistical network properties, like diameter and degree distribution, of graphs G and K.

Comparing statistical properties already suggests a very direct approach to this problem: One could first identify the set of network properties (statistics) to match, then define a quality of fit metric and somehow optimize over it. For example, one could use the KL divergence (Kullback and Leibler, 1951), or the sum of squared differences between the degree distribution of the real network G and its synthetic counterpart K. Moreover, as we are interested in matching several such statistics between the networks one would have to meaningfully combine these individual error metrics into a global error metric. So, one would have to specify what kind of properties he or she cares about and then combine them accordingly. This would be a hard task as the patterns of interest have very different magnitudes and scales. Moreover, as new network patterns are discovered, the error functions would have to be changed and models re-estimated. And even then it is not clear how to define the optimization procedure to maximize the quality of fit and how to perform optimization over the parameter space.

Our approach here is different. Instead of committing to a set of network properties ahead of time, we try to directly match the adjacency matrices of the real network G and its synthetic counterpart K. The idea is that if the adjacency matrices are similar then the global statistical properties (statistics computed over K and G) will also match. Moreover, by directly working with the graph itself (and not summary statistics), we do not commit to any particular set of network statistics (network properties/patterns) and as new statistical properties of networks are discovered our models and estimated parameters will still hold.

5.1 Preliminaries

Stochastic graph models induce probability distributions over graphs. A generative model assigns a probability $P(G)$ to every graph G. $P(G)$ is the *likelihood* that a given model (with a given set of parameters) generates the graph G. We concentrate on the Stochastic Kronecker graphs model, and consider fitting it to a real graph G, our data. We use the maximum likelihood approach, that is, we aim to find parameter values, the initiator $\mathcal{P}_1$, that maximize $P(G)$ under the Stochastic Kronecker graph model.

This presents several challenges:

- **Model selection:** a graph is a single structure, and not a set of items drawn independently and identically-distributed (i.i.d.) from some distribution. So one cannot split it into independent training and test sets. The fitted parameters will thus be best to generate a *particular* instance of a graph. Also, overfitting could be an issue since a more complex model generally fits better.

- **Node correspondence:** The second challenge is the node correspondence or node labeling problem. The graph G has a set of N nodes, and each node has a unique label (index, ID). Labels do not carry any particular meaning, they just uniquely denote or identify the nodes. One can think of this as the graph is first generated and then the labels (node IDs) are randomly assigned. This means that two isomorphic graphs that have different node labels should have the same likelihood. A permutation σ is sufficient to describe the node correspondences as it maps labels (IDs) to nodes of the graph. To compute the likelihood $P(G)$ one has to consider all node correspondences $P(G) = \sum_\sigma P(G|\sigma)P(\sigma)$, where the sum is over all $N!$ permutations σ of N nodes. Calculating this *super-exponential* sum explicitly is infeasible for any graph with more than a handful of nodes. Intuitively, one can think of this summation as some kind of graph isomorphism test where we are searching for best correspondence (mapping) between nodes of G and $\mathcal{P}$.

- **Likelihood estimation:** Even if we assume one can efficiently solve the node correspondence problem, calculating $P(G|\sigma)$ naively takes $O(N^2)$ as one has to evaluate the probability of each of the N^2 possible edges in the graph adjacency matrix. Again, for graphs of size we want to model here, approaches with quadratic complexity are infeasible.

To develop our solution we use sampling to avoid the super-exponential sum over the node correspondences. By exploiting the structure of the Kronecker matrix multiplication we develop an algorithm to evaluate $P(G|\sigma)$ in *linear* time $O(E)$. Since real graphs are *sparse*, that is, the number of edges is roughly of the same order as the number of nodes, this makes fitting of Kronecker graphs to large networks feasible.

5.2 Problem Formulation

Suppose we are given a graph G on $N = N_1^k$ nodes (for some positive integer k), and an $N_1 \times N_1$ Stochastic Kronecker graphs initiator matrix $\mathcal{P}_1$. Here $\mathcal{P}_1$ is a parameter matrix, a set of parameters that we aim to estimate. For now also assume N_1, the size of the initiator matrix, is given. Later we will show how to automatically select it. Next, using $\mathcal{P}_1$ we create a Stochastic Kronecker graphs probability matrix $\mathcal{P}_k$, where every entry p_{uv} of $\mathcal{P}_k$ contains a probability that node u links to node v. We then evaluate the probability that G is a realization of $\mathcal{P}_k$. The task is to find such $\mathcal{P}_1$ that has the highest probability of realizing (generating) G.

Formally, we are solving:

$$\arg\max_{\mathcal{P}_1} P(G|\mathcal{P}_1). \tag{2}$$

To keep the notation simpler we use standard symbol Θ to denote the parameter matrix $\mathcal{P}_1$ that we are trying to estimate. We denote entries of $\Theta = \mathcal{P}_1 = [\theta_{ij}]$, and similarly we denote $\mathcal{P} = \mathcal{P}_k = [p_{ij}]$. Note that here we slightly simplified the notation: we use Θ to refer to $\mathcal{P}_1$, and θ_{ij} are elements of Θ. Similarly, p_{ij} are elements of $\mathcal{P}$ ($\equiv \mathcal{P}_k$). Moreover, we denote $K = R(\mathcal{P})$, that is, K is a realization of the Stochastic Kronecker graph sampled from probabilistic adjacency matrix $\mathcal{P}$.

As noted before, the node IDs are assigned arbitrarily and they carry no significant information, which means that we have to consider all the mappings of nodes from G to rows and columns of stochastic adjacency matrix $\mathcal{P}$. A priori all labelings are equally likely. A permutation σ of the set $\{1,\ldots,N\}$ defines this mapping of nodes from G to stochastic adjacency matrix $\mathcal{P}$. To evaluate the

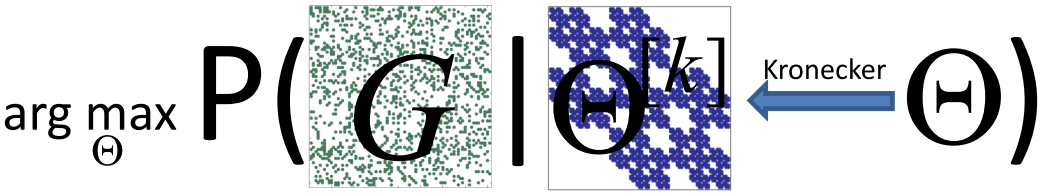


Figure 11: Kronecker parameter estimation as an optimization problem. We search over the initiator matrices Θ ($\equiv \mathcal{P}_1$). Using Kronecker multiplication we create probabilistic adjacency matrix $\Theta^{[k]}$ that is of same size as real network G. Now, we evaluate the likelihood by simultaneously traversing and multiplying entries of G and $\Theta^{[k]}$ (see Equation refeq:KronProbGPS). As shown by the figure permutation σ plays an important role, as permuting rows and columns of G could make it look more similar to $\Theta^{[k]}$ and thus increase the likelihood.

likelihood of G one needs to consider all possible mappings of N nodes of G to rows (columns) of $\mathcal{P}$. For convenience we work with *log-likelihood* $l(\Theta)$, and solve $\hat{\Theta} = \arg\max_\Theta l(\Theta)$, where $l(\Theta)$ is defined as:

$$\begin{aligned} l(\Theta) &= \log P(G|\Theta) = \log \sum_\sigma P(G|\Theta,\sigma)P(\sigma|\Theta) \\ &= \log \sum_\sigma P(G|\Theta,\sigma)P(\sigma). \end{aligned} \tag{3}$$

The likelihood that a given initiator matrix Θ and permutation σ gave rise to the real graph G, $P(G|\Theta,\sigma)$, is calculated naturally as follows. First, by using Θ we create the Stochastic Kronecker graph adjacency matrix $\mathcal{P} = \mathcal{P}_k = \Theta^{[k]}$. Permutation σ defines the mapping of nodes of G to the rows and columns of stochastic adjacency matrix $\mathcal{P}$. (See Figure 11 for the illustration.)

We then model edges as independent Bernoulli random variables parameterized by the parameter matrix Θ. So, each entry p_{uv} of $\mathcal{P}$ gives exactly the probability of edge (u,v) appearing.

We then define the likelihood:

$$P(G|\mathcal{P},\sigma) = \prod_{(u,v)\in G} \mathcal{P}[\sigma_u,\sigma_v] \prod_{(u,v)\notin G} (1-\mathcal{P}[\sigma_u,\sigma_v]), \tag{4}$$

where we denote σ_i as the i^{th} element of the permutation σ, and $\mathcal{P}[i,j]$ is the element at row i, and column j of matrix $\mathcal{P} = \Theta^{[k]}$.

The likelihood is defined very naturally. We traverse the entries of adjacency matrix G and then based on whether a particular edge appeared in G or not we take the probability of edge occurring (or not) as given by $\mathcal{P}$, and multiply these probabilities. As one has to touch all the entries of the stochastic adjacency matrix $\mathcal{P}$ evaluating Equation 4 takes $O(N^2)$ time.

We further illustrate the process of estimating Stochastic Kronecker initiator matrix Θ in Figure 11. We search over initiator matrices Θ to find the one that maximizes the likelihood $P(G|\Theta)$. To estimate $P(G|\Theta)$ we are given a concrete Θ and now we use Kronecker multiplication to create probabilistic adjacency matrix $\Theta^{[k]}$ that is of same size as real network G. Now, we evaluate the likelihood by traversing the corresponding entries of G and $\Theta^{[k]}$. Equation 4 basically traverses the

adjacency matrix of G, and maps every entry (u,v) of G to a corresponding entry (σ_u,σ_v) of $\mathcal{P}$. Then in case that edge (u,v) exists in G (i.e., $G[u,v]=1$) the likelihood that particular edge existing is $\mathcal{P}[\sigma_u,\sigma_v]$, and similarly, in case the edge (u,v) does not exist the likelihood is simply $1-\mathcal{P}[\sigma_u,\sigma_v]$. This also demonstrates the importance of permutation σ, as permuting rows and columns of G could make the adjacency matrix look more "similar" to $\Theta^{[k]}$, and would increase the likelihood.

So far we showed how to assess the quality (likelihood) of a particular Θ. So, naively one could perform some kind of grid search to find best Θ. However, this is very inefficient. A better way of doing it is to compute the gradient of the log-likelihood $\frac{\partial}{\partial\Theta}l(\Theta)$, and then use the gradient to update the current estimate of Θ and move towards a solution of higher likelihood. Algorithm 1 gives an outline of the optimization procedure.

However, there are several difficulties with this algorithm. First, we are assuming gradient descent type optimization will find a good solution, that is, the problem does not have (too many) local minima. Second, we are summing over exponentially many permutations in Equation 3. Third, the evaluation of Equation 4 as it is written now takes $O(N^2)$ time and needs to be evaluated $N!$ times. So, given a concrete Θ just naively calculating the likelihood takes $O(N!N^2)$ time, and then one also has to optimize over Θ.

Observation 2 *The complexity of calculating the likelihood $P(G|\Theta)$ of the graph G naively is $O(N!N^2)$, where N is the number of nodes in G.*

Next, we show that all this can be done in *linear time*.

5.3 Summing Over the Node Labelings

To maximize Equation 2 using algorithm 1 we need to obtain the gradient of the log-likelihood $\frac{\partial}{\partial\Theta}l(\Theta)$. We can write:

$$\begin{aligned}\frac{\partial}{\partial\Theta}l(\Theta) &= \frac{\sum_\sigma \frac{\partial}{\partial\Theta}P(G|\sigma,\Theta)P(\sigma)}{\sum_{\sigma'}P(G|\sigma',\Theta)P(\sigma')}\\ &= \frac{\sum_\sigma \frac{\partial \log P(G|\sigma,\Theta)}{\partial\Theta}P(G|\sigma,\Theta)P(\sigma)}{P(G|\Theta)}\\ &= \sum_\sigma \frac{\partial \log P(G|\sigma,\Theta)}{\partial\Theta}P(\sigma|G,\Theta). \qquad (5)\end{aligned}$$

Note we are still summing over all $N!$ permutations σ, so calculating Eq. 5 is computationally intractable for graphs with more than a handful of nodes. However, the equation has a nice form which allows for use of simulation techniques to avoid the summation over super-exponentially many node correspondences. Thus, we simulate draws from the permutation distribution $P(\sigma|G,\Theta)$, and then evaluate the quantities at the sampled permutations to obtain the expected values of log-likelihood and gradient. Algorithm 2 gives the details.

Note that we can also permute the rows and columns of the parameter matrix Θ to obtain equivalent estimates. Therefore Θ is not strictly identifiable exactly because of these permutations. Since the space of permutations on N nodes is very large (grows as $N!$) the permutation sampling algorithm will explore only a small fraction of the space of all permutations and may converge to one of the global maxima (but may not explore all $N_1!$ of them) of the parameter space. As we empirically

input : size of parameter matrix N_1, graph G on $N = N_1^k$ nodes, and learning rate λ
output: MLE parameters $\hat{\Theta}$ ($N_1 \times N_1$ probability matrix)

initialize $\hat{\Theta}_1$
while *not converged* **do**
    evaluate gradient: $\frac{\partial}{\partial \hat{\Theta}_t} l(\hat{\Theta}_t)$
    update parameter estimates: $\hat{\Theta}_{t+1} = \hat{\Theta}_t + \lambda \frac{\partial}{\partial \hat{\Theta}_t} l(\hat{\Theta}_t)$
end
return $\hat{\Theta} = \hat{\Theta}_t$

Algorithm 1: KRONFIT algorithm.

input : Parameter matrix Θ, and graph G
output: Log-likelihood $l(\Theta)$, and gradient $\frac{\partial}{\partial \Theta} l(\Theta)$

for $t := 1$ **to** T **do**
    $\sigma_t :=$ `SamplePermutation` (G, Θ)
    $l_t = \log P(G|\sigma^{(t)}, \Theta)$
    $grad_t := \frac{\partial}{\partial \Theta} \log P(G|\sigma^{(t)}, \Theta)$
end
return $l(\Theta) = \frac{1}{T} \sum_t l_t$, and $\frac{\partial}{\partial \Theta} l(\Theta) = \frac{1}{T} \sum_t grad_t$

Algorithm 2: Calculating log-likelihood and gradient

show later our results are not sensitive to this and multiple restarts result in equivalent (but often permuted) parameter estimates.

5.3.1 SAMPLING PERMUTATIONS

Next, we describe the Metropolis algorithm to simulate draws from the permutation distribution $P(\sigma|G,\Theta)$, which is given by

$$P(\sigma|G,\Theta) = \frac{P(\sigma,G,\Theta)}{\sum_\tau P(\tau,G,\Theta)} = \frac{P(\sigma,G,\Theta)}{Z}$$

where Z is the normalizing constant that is hard to compute since it involves the sum over $N!$ elements. However, if we compute the likelihood ratio between permutations σ and σ' (Equation 6) the normalizing constants nicely cancel out:

$$\frac{P(\sigma'|G,\Theta)}{P(\sigma|G,\Theta)} = \prod_{(u,v)\in G} \frac{\mathcal{P}[\sigma'_u,\sigma'_v]}{\mathcal{P}[\sigma_u,\sigma_v]} \prod_{(u,v)\notin G} \frac{(1-\mathcal{P}[\sigma'_u,\sigma'_v])}{(1-\mathcal{P}[\sigma_u,\sigma_v])} \tag{6}$$

$$= \prod_{\substack{(u,v)\in G \\ (\sigma_u,\sigma_v)\neq(\sigma'_u,\sigma'_v)}} \frac{\mathcal{P}[\sigma'_u,\sigma'_v]}{\mathcal{P}[\sigma_u,\sigma_v]} \prod_{\substack{(u,v)\notin G \\ (\sigma_u,\sigma_v)\neq(\sigma'_u,\sigma'_v)}} \frac{(1-\mathcal{P}[\sigma'_u,\sigma'_v])}{(1-\mathcal{P}[\sigma_u,\sigma_v])}. \tag{7}$$

This immediately suggests the use of a Metropolis sampling algorithm (Gamerman, 1997) to simulate draws from the permutation distribution since Metropolis is solely based on such ratios

input : Kronecker initiator matrix Θ and a graph G on N nodes
output: Permutation $\sigma^{(i)} \sim P(\sigma|G,\Theta)$

$\sigma^{(0)} := (1,\ldots,N)$
$i = 1$
repeat
  Draw j and k uniformly from $(1,\ldots,N)$
  $\sigma^{(i)} :=$ `SwapNodes`$(\sigma^{(i-1)}, j, k)$
  Draw u from $U(0,1)$
  if $u > \frac{P(\sigma^{(i)}|G,\Theta)}{P(\sigma^{(i-1)}|G,\Theta)}$ **then**
    $\sigma^{(i)} := \sigma^{(i-1)}$
  end
  i = i + 1
until $\sigma^{(i)} \sim P(\sigma|G,\Theta)$
return $\sigma^{(i)}$
Where $U(0,1)$ is a uniform distribution on $[0,1]$, and $\sigma' :=$ `SwapNodes(`σ, j, k`)` is the permutation σ' obtained from σ by swapping elements at positions j and k.

Algorithm 3: `SamplePermutation(`G,Θ`)`: Metropolis sampling of the node permutation.

(where normalizing constants cancel out). In particular, suppose that in the Metropolis algorithm (Algorithm 3) we consider a move from permutation σ to a new permutation σ'. Probability of accepting the move to σ' is given by Equation 6 (if $\frac{P(\sigma'|G,\Theta)}{P(\sigma|G,\Theta)} \leq 1$) or 1 otherwise.

Now we have to devise a way to sample permutations σ from the proposal distribution. One way to do this would be to simply generate a random permutation σ' and then check the acceptance condition. This would be very inefficient as we expect the distribution $P(\sigma|G,\Theta)$ to be heavily skewed, that is, there will be a relatively small number of good permutations (node mappings). Even more so as the degree distributions in real networks are skewed there will be many bad permutations with low likelihood, and few good ones that do a good job in matching nodes of high degree.

To make the sampling process "smoother", that is, sample permutations that are not that different (and thus are not randomly jumping across the permutation space) we design a Markov chain. The idea is to stay in high likelihood part of the permutation space longer. We do this by making samples dependent, that is, given σ' we want to generate next candidate permutation σ'' to then evaluate the likelihood ratio. When designing the Markov chain step one has to be careful so that the proposal distribution satisfies the detailed balance condition: $\pi(\sigma')P(\sigma'|\sigma'') = \pi(\sigma'')P(\sigma''|\sigma')$, where $P(\sigma'|\sigma'')$ is the transition probability of obtaining permutation σ' from σ'' and, $\pi(\sigma')$ is the stationary distribution.

In Algorithm 3 we use a simple proposal where given permutation σ' we generate σ'' by swapping elements at two uniformly at random chosen positions of σ'. We refer to this proposal as `SwapNodes`. While this is simple and clearly satisfies the detailed balance condition it is also inefficient in a way that most of the times low degree nodes will get swapped (a direct consequence of heavy tailed degree distributions). This has two consequences, (a) we will slowly converge to good permutations (accurate mappings of high degree nodes), and (b) once we reach a good permutation,

very few permutations will get accepted as most proposed permutations σ' will swap low degree nodes (as they form the majority of nodes).

A possibly more efficient way would be to swap elements of σ biased based on corresponding node degree, so that high degree nodes would get swapped more often. However, doing this directly does not satisfy the detailed balance condition. A way of sampling labels biased by node degrees that at the same time satisfies the detailed balance condition is the following: we pick an edge in G uniformly at random and swap the labels of the nodes at the edge endpoints. Notice this is biased towards swapping labels of nodes with high degrees simply as they have more edges. The detailed balance condition holds as edges are sampled uniformly at random. We refer to this proposal as `SwapEdgeEndpoints`.

However, the issue with this proposal is that if the graph G is disconnected, we will only be swapping labels of nodes that belong to the same connected component. This means that some parts of the permutation space will never get visited. To overcome this problem we execute `SwapNodes` with some probability ω and `SwapEdgeEndpoints` with probability $1-\omega$.

To summarize we consider the following two permutation proposal distributions:

- $\sigma'' = \texttt{SwapNodes}(\sigma')$: we obtain σ'' by taking σ', uniformly at random selecting a pair of elements and swapping their positions.

- $\sigma'' = \texttt{SwapEdgeEndpoints}(\sigma')$: we obtain σ'' from σ' by first sampling an edge (j,k) from G uniformly at random, then we take σ' and swap the labels at positions j and k.

5.3.2 SPEEDING UP THE LIKELIHOOD RATIO CALCULATION

We further speed up the algorithm by using the following observation. As written the Equation 6 takes $O(N^2)$ to evaluate since we have to consider N^2 possible edges. However, notice that permutations σ and σ' differ only at two positions, that is, elements at position j and k are swapped, that is, σ and σ' map all nodes except the two to the same locations. This means those elements of Equation 6 cancel out. Thus to update the likelihood we only need to traverse two rows and columns of matrix $\mathcal{P}$, namely rows and columns j and k, since everywhere else the mapping of the nodes to the adjacency matrix is the same for both permutations. This gives Equation 7 where the products now range only over the two rows/columns of $\mathcal{P}$ where σ and σ' differ.

Graphs we are working with here are too large to allow us to explicitly create and store the stochastic adjacency matrix $\mathcal{P}$ by Kronecker powering the initiator matrix Θ. Every time probability $\mathcal{P}[i,j]$ of edge (i,j) is needed the Equation 1 is evaluated, which takes $O(k)$. So a single iteration of Algorithm 3 takes $O(kN)$.

Observation 3 *Sampling a permutation* σ *from* $P(\sigma|G,\Theta)$ *takes* $O(kN)$.

This is gives us an improvement over the $O(N!)$ complexity of summing over all the permutations. So far we have shown how to obtain a permutation but we still need to evaluate the likelihood and find the gradients that will guide us in finding good initiator matrix. The problem here is that naively evaluating the network likelihood (gradient) as written in Equation 5 takes time $O(N^2)$. This is exactly what we investigate next and how to calculate the likelihood in *linear time*.

5.4 Efficiently Approximating Likelihood and Gradient

We just showed how to efficiently sample node permutations. Now, given a permutation we show how to efficiently evaluate the likelihood and it's gradient. Similarly as evaluating the likelihood ratio, naively calculating the log-likelihood $l(\Theta)$ or its gradient $\frac{\partial}{\partial\Theta}l(\Theta)$ takes time quadratic in the number of nodes. Next, we show how to compute this in linear time $O(E)$.

We begin with the observation that real graphs are sparse, which means that the number of edges is not quadratic but rather almost linear in the number of nodes, $E \ll N^2$. This means that majority of entries of graph adjacency matrix are zero, that is, most of the edges are not present. We exploit this fact. The idea is to first calculate the likelihood (gradient) of an empty graph, that is, a graph with zero edges, and then correct for the edges that actually appear in G.

To naively calculate the likelihood for an empty graph one needs to evaluate every cell of graph adjacency matrix. We consider Taylor approximation to the likelihood, and exploit the structure of matrix $\mathcal{P}$ to devise a constant time algorithm.

First, consider the second order Taylor approximation to log-likelihood of an edge that succeeds with probability x but does not appear in the graph:

$$\log(1-x) \approx -x - \frac{1}{2}x^2.$$

Calculating $l_e(\Theta)$, the log-likelihood of an empty graph, becomes:

$$l_e(\Theta) = \sum_{i=1}^{N}\sum_{j=1}^{N}\log(1-p_{ij}) \approx -\left(\sum_{i=1}^{N_1}\sum_{j=1}^{N_1}\theta_{ij}\right)^k - \frac{1}{2}\left(\sum_{i=1}^{N_1}\sum_{j=1}^{N_1}{\theta_{ij}}^2\right)^k. \tag{8}$$

Notice that while the first pair of sums ranges over N elements, the last pair only ranges over N_1 elements ($N_1 = \log_k N$). Equation 8 holds due to the recursive structure of matrix $\mathcal{P}$ generated by the Kronecker product. We substitute the $\log(1-p_{ij})$ with its Taylor approximation, which gives a sum over elements of $\mathcal{P}$ and their squares. Next, we notice the sum of elements of $\mathcal{P}$ forms a multinomial series, and thus $\sum_{i,j} p_{ij} = (\sum_{i,j}\theta_{ij})^k$, where θ_{ij} denotes an element of Θ, and p_{ij} element of $\Theta^{[k]}$.

Calculating log-likelihood of G now takes $O(E)$: First, we approximate the likelihood of an empty graph in constant time, and then account for the edges that are actually present in G, that is, we subtract "no-edge" likelihood and add the "edge" likelihoods:

$$l(\Theta) = l_e(\Theta) + \sum_{(u,v)\in G} -\log(1-\mathcal{P}[\sigma_u,\sigma_v]) + \log(\mathcal{P}[\sigma_u,\sigma_v]).$$

We note that by using the second order Taylor approximation to the log-likelihood of an empty graph, the error term of the approximation is $\frac{1}{3}(\sum_i {\theta_{ij}}^3)^k$, which can diverge for large k. For typical values of initiator matrix $\mathcal{P}_1$ (that we present in Section 6.5) we note that one needs about fourth or fifth order Taylor approximation for the error of the approximation actually go to zero as k approaches infinity, that is, $\sum_{ij}{\theta_{ij}}^{n+1} < 1$, where n is the order of Taylor approximation employed.

5.5 Calculating the Gradient

Calculation of the gradient of the log-likelihood follows exactly the same pattern as described above. First by using the Taylor approximation we calculate the gradient as if graph G would have no edges. Then we correct the gradient for the edges that are present in G. As in previous section we speed

up the calculations of the gradient by exploiting the fact that two consecutive permutations σ and σ' differ only at two positions, and thus given the gradient from previous step one only needs to account for the swap of the two rows and columns of the gradient matrix $\partial\mathcal{P}/\partial\Theta$ to update to the gradients of individual parameters.

5.6 Determining the Size of Initiator Matrix

The question we answer next is how to determine the right number of parameters, that is, what is the right size of matrix Θ? This is a classical question of model selection where there is a tradeoff between the complexity of the model, and the quality of the fit. Bigger model with more parameters usually fits better, however it is also more likely to overfit the data.

For model selection to find the appropriate value of N_1, the size of matrix Θ, and choose the right tradeoff between the complexity of the model and the quality of the fit, we propose to use the Bayes Information Criterion (BIC) (Schwarz, 1978). Stochastic Kronecker graph model the presence of edges with independent Bernoulli random variables, where the canonical number of parameters is N_1^{2k}, which is a function of a lower-dimensional parameter Θ. This is then a *curved exponential family* (Efron, 1975), and BIC naturally applies:

$$\mathrm{BIC}(N_1) = -l(\hat{\Theta}_{N_1}) + \frac{1}{2} N_1^2 \log(N^2)$$

where $\hat{\Theta}_{N_1}$ are the maximum likelihood parameters of the model with $N_1 \times N_1$ parameter matrix, and N is the number of nodes in G. Note that one could also additional term to the above formula to account for multiple global maxima of the likelihood space but as N_1 is small the additional term would make no real difference.

As an alternative to BIC one could also consider the Minimum Description Length (MDL) (Rissanen, 1978) principle where the model is scored by the quality of the fit plus the size of the description that encodes the model and the parameters.

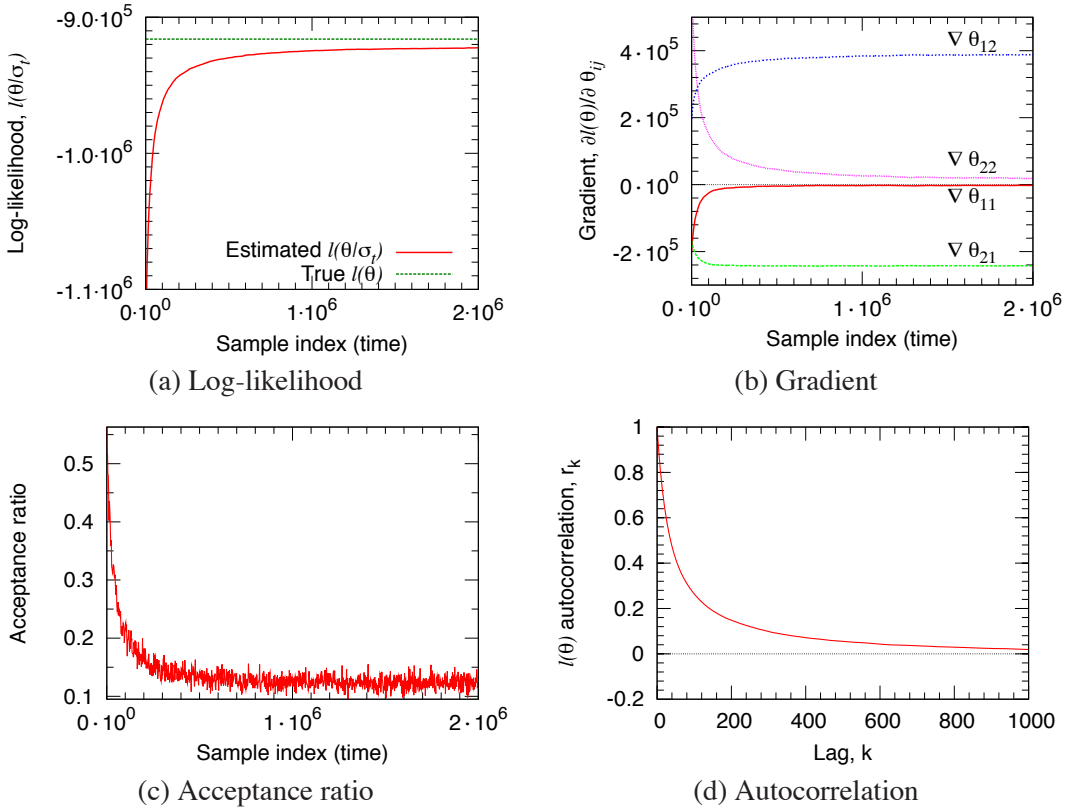
6. Experiments on Real and Synthetic Data

Next we described our experiments on a range of real and synthetic networks. We divide the experiments into several subsections. First we examine the convergence and mixing of the Markov chain of our permutation sampling scheme. Then we consider estimating the parameters of synthetic Kronecker graphs to see whether KRONFIT is able to recover the parameters used to generate the network. Last, we consider fitting Stochastic Kronecker graphs to large real world networks.

KRONFIT code for efficient Kronecker graph parameter estimation can be downloaded from `http://snap.stanford.edu`.

6.1 Permutation Sampling

In our experiments we considered both synthetic and real graphs. Unless mentioned otherwise all synthetic Kronecker graphs were generated using $\mathcal{P}_1^* = [0.8, 0.6; 0.5, 0.3]$, and $k = 14$ which gives us a graph G on N =16,384 nodes and E =115,741 edges. We chose this particular $\mathcal{P}_1^*$ as it resembles the typical initiator for real networks analyzed later in this section.



(a) Log-likelihood (b) Gradient

(c) Acceptance ratio (d) Autocorrelation

Figure 12: Convergence of the log-likelihood and components of the gradient towards their true values for Metropolis permutation sampling (Algorithm 3) with the number of samples.

6.1.1 CONVERGENCE OF THE LOG-LIKELIHOOD AND THE GRADIENT

First, we examine the convergence of Metropolis permutation sampling, where permutations are sampled sequentially. A new permutation is obtained by modifying the previous one which creates a Markov chain. We want to assess the convergence and mixing of the chain. We aim to determine how many permutations one needs to draw to reliably estimate the likelihood and the gradient, and also how long does it take until the samples converge to the stationary distribution. For the experiment we generated a synthetic Stochastic Kronecker graphs using $\mathcal{P}_1^*$ as defined above. Then, starting with a random permutation we run Algorithm 3, and measure how the likelihood and the gradients converge to their true values.

In this particular case, we first generated a Stochastic Kronecker graphs G as described above, but then calculated the likelihood and the parameter gradients for $\Theta' = [0.8, 0.75; 0.45, 0.3]$. We average the likelihoods and gradients over buckets of 1,000 consecutive samples, and plot how the log-likelihood calculated over the sampled permutations approaches the true log-likelihood (that we can compute since G is a Stochastic Kronecker graphs).

First, we present experiments that aim to answer how many samples (i.e., permutations) does one need to draw to obtain a reliable estimate of the gradient (see Equation 5). Figure 12(a) shows how the estimated log-likelihood approaches the true likelihood. Notice that estimated values

quickly converge to their true values, that is, Metropolis sampling quickly moves towards "good" permutations. Similarly, Figure 12(b) plots the convergence of the gradients. Notice that θ_{11} and θ_{22} of Θ' and $\mathcal{P}_1^*$ match, so gradients of these two parameters should converge to zero and indeed they do. On the other hand, θ_{12} and θ_{21} differ between Θ' and $\mathcal{P}_1^*$. Notice, the gradient for one is positive as the parameter θ_{12} of Θ' should be decreased, and similarly for θ_{21} the gradient is negative as the parameter value should be increased to match the Θ'. In summary, this shows that log-likelihood and gradients rather quickly converge to their true values.

In Figures 12(c) and (d) we also investigate the properties of the Markov Chain Monte Carlo sampling procedure, and assess convergence and mixing criteria. First, we plot the fraction of accepted proposals. It stabilizes at around 15%, which is quite close to the rule-of-a-thumb of 25%. Second, Figure 12(d) plots the autocorrelation of the log-likelihood as a function of the lag. Autocorrelation r_k of a signal X is a function of the lag k where r_k is defined as the correlation of signal X at time t with X at $t+k$, that is, correlation of the signal with itself at lag k. High autocorrelations within chains indicate slow mixing and, usually, slow convergence. On the other hand fast decay of autocorrelation implies better the mixing and thus one needs less samples to accurately estimate the gradient or the likelihood. Notice the rather fast autocorrelation decay.

All in all, these experiments show that one needs to sample on the order of tens of thousands of permutations for the estimates to converge. We also verified that the variance of the estimates is sufficiently small. In our experiments we start with a random permutation and use long burn-in time. Then when performing optimization we use the permutation from the previous step to initialize the permutation at the current step of the gradient descent. Intuitively, small changes in parameter space Θ also mean small changes in $P(\sigma|G,\Theta)$.

6.1.2 DIFFERENT PROPOSAL DISTRIBUTIONS

In Section 5.3.1 we defined two permutation sampling strategies: `SwapNodes` where we pick two nodes uniformly at random and swap their labels (node ids), and `SwapEdgeEndpoints` where we pick a random edge in a graph and then swap the labels of the edge endpoints. We also discussed that one can interpolate between the two strategies by executing `SwapNodes` with probability ω, and `SwapEdgeEndpoints` with probability $1-\omega$.

So, given a Stochastic Kronecker graphs G on N =16,384 and E =115,741 generated from $\mathcal{P}_1^* = [0.8, 0.7; 0.5, 0.3]$ we evaluate the likelihood of $\Theta' = [0.8, 0.75; 0.45, 0.3]$. As we sample permutations we observe how the estimated likelihood converges to the true likelihood. Moreover we also vary parameter ω which interpolates between the two permutation proposal distributions. The quicker the convergence towards the true log-likelihood the better the proposal distribution.

Figure 13 plots the convergence of the log-likelihood with the number of sampled permutations. We plot the average over non-overlapping buckets of 1,000 consecutive permutations. Faster convergence implies better permutation proposal distribution. When we use only `SwapNodes` ($\omega = 1$) or `SwapEdgeEndpoints` ($\omega = 0$) convergence is rather slow. We obtain best convergence for ω around 0.6.

Similarly, Figure 14(a) plots the autocorrelation as a function of the lag k for different choices of ω. Faster autocorrelation decay means better mixing of the Markov chain. Again, notice that we get best mixing for $\omega \approx 0.6$. (Notice logarithmic y-axis.)

Last, we diagnose how long the sampling procedure must be run before the generated samples can be considered to be drawn (approximately) from the stationary distribution. We call this the

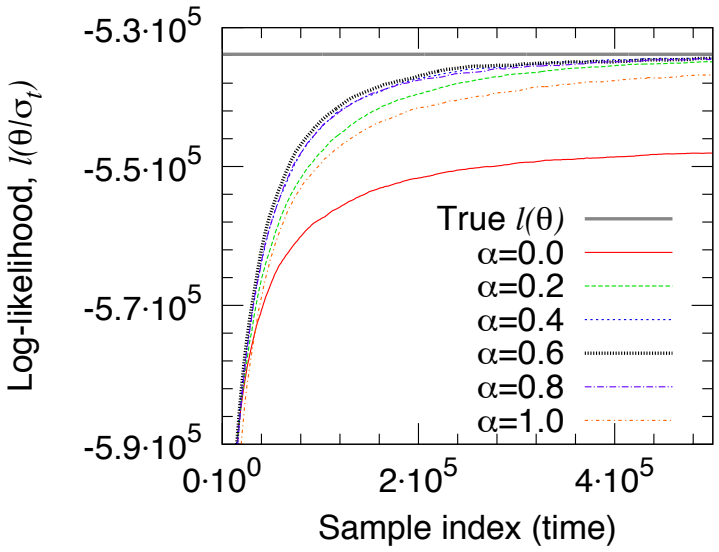


Figure 13: Convergence of the log-likelihood and gradients for Metropolis permutation sampling (Algorithm 3) for different choices of ω that interpolates between the `SwapNodes` ($\omega = 1$) and `SwapEdgeEndpoints` ($\omega = 0$) permutation proposal distributions. Notice fastest convergence of log-likelihood for $\omega = 0.6$.

burn-in time of the chain. There are various procedures for assessing convergence. Here we adopt the approach of Gelman et al. (2003), that is based on running multiple Markov chains each from a different starting point, and then comparing the variance within the chain and between the chains. The sooner the within- and between-chain variances become equal the shorter the burn-in time, that is, the sooner the samples are drawn from the stationary distribution.

Let l be the parameter that is being simulated with J different chains, and then let $l_j^{(k)}$ denote the k^{th} sample of the j^{th} chain, where $j = 1,\ldots,J$ and $k = 1,\ldots,K$. More specifically, in our case we run separate permutation sampling chains. So, we first sample permutation $\sigma_j^{(k)}$ and then calculate the corresponding log-likelihood $l_j^{(k)}$.

First, we compute between and within chain variances $\hat{\sigma}_B^2$ and $\hat{\sigma}_W^2$, where between-chain variance is obtained by

$$\hat{\sigma}_B^2 = \frac{K}{J-1}\sum_{j=1}^{J}(\bar{l}_{\cdot j} - \bar{l}_{\cdot\cdot})^2$$

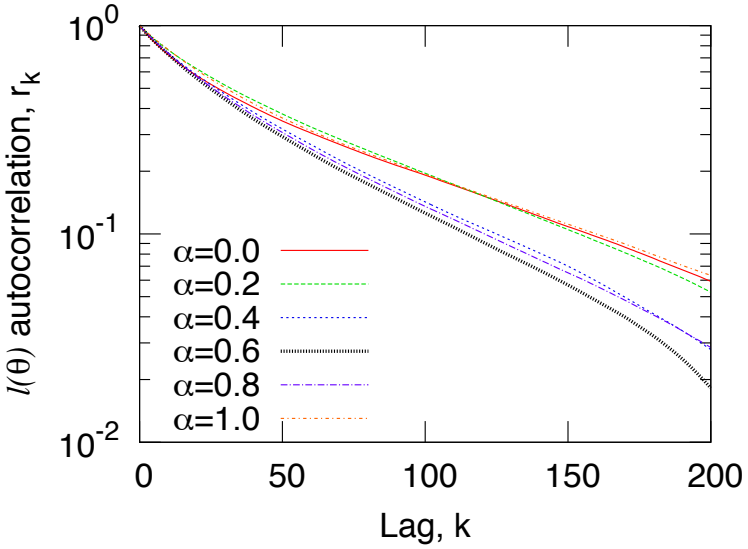
where $\bar{l}_{\cdot j} = \frac{1}{K}\sum_{k=1}^{K} l_j^{(k)}$ and $\bar{l}_{\cdot\cdot} = \frac{1}{J}\sum_{j=1}^{J}\bar{l}_{\cdot j}$.

Similarly the within-chain variance is defined by

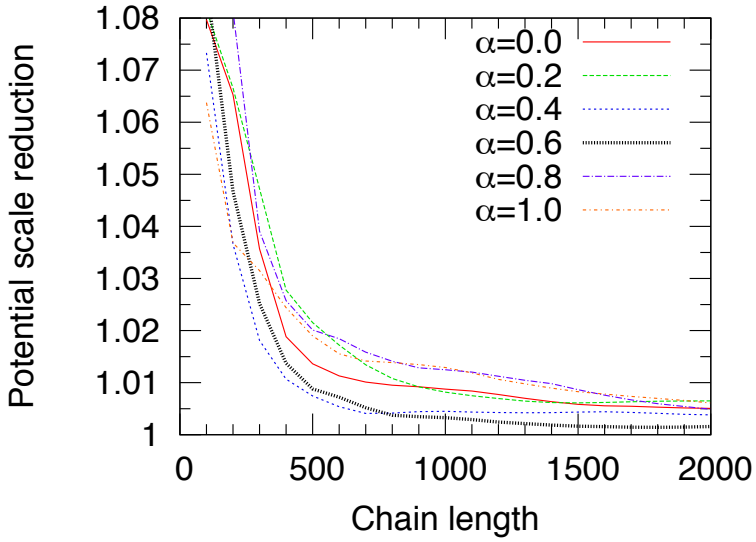
$$\hat{\sigma}_W^2 = \frac{1}{J(K-1)}\sum_{j=1}^{J}\sum_{k=1}^{K}(l_j^{(k)} - \bar{l}_{\cdot j})^2.$$

Then, the marginal posterior variance of $\hat{l}$ is calculated using

$$\hat{\sigma}^2 = \frac{K-1}{K}\hat{\sigma}_W^2 + \frac{1}{K}\hat{\sigma}_B^2.$$



(a) Autocorrelation

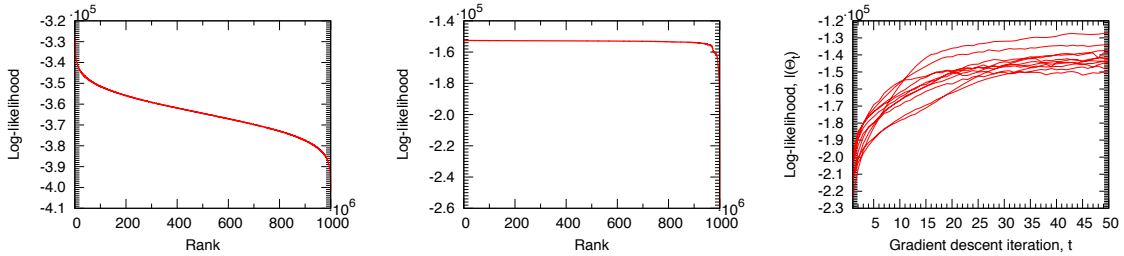


(b) Potential scale reduction

Figure 14: (a) Autocorrelation plot of the log-likelihood for the different choices of parameter ω. Notice we get best mixing with $\omega \approx 0.6$. (b) The potential scale reduction that compares the variance inside- and across- independent Markov chains for different values of parameter ω.

And, finally, we estimate the *potential scale reduction* (Gelman et al., 2003) of l by

$$\sqrt{\hat{R}} = \sqrt{\frac{\hat{\sigma}^2}{\hat{\sigma}_W^2}}.$$



(a) $l(\Theta|\sigma_i)$ where $\sigma_i \sim P(\sigma)$

(b) $l(\Theta|\sigma_i)$ where $\sigma_i \sim P(\sigma|\Theta, G)$

(c) $l(\Theta_t)$ for 10 random gradient descent runs

Figure 15: (a) Distribution of log-likelihood of permutations sampled uniformly at random, and (b) when sampled from $P(\sigma|\Theta, G)$. Notice the space of good permutations is rather small but our sampling quickly finds permutations of high likelihood. (c) Convergence of log-likelihood for 10 runs of gradient descent, each from a different random starting point.

Note that as the length of the chain $K \to \infty$, $\sqrt{\hat{R}}$ converges to 1 from above. The recommendation for convergence assessment from Gelman et al. (2003) is that the potential scale reduction is below 1.2.

Figure 14(b) gives the Gelman-Rubin-Brooks plot, where we plot the potential scale reduction $\sqrt{\hat{R}}$ over the increasing chain length K for different choices of parameter ω. Notice that the potential scale reduction quickly decays towards 1. Similarly as in Figure 14 the extreme values of ω give slow decay, while we obtain the fastest potential scale reduction when $\omega \approx 0.6$.

6.1.3 PROPERTIES OF THE PERMUTATION SPACE

Next we explore the properties of the permutation space. We would like to quantify what fraction of permutations are "good" (have high likelihood), and how quickly are they discovered. For the experiment we took a real network G (AS-ROUTEVIEWS network) and the MLE parameters $\hat{\Theta}$ for it that we estimated before hand ($l(\hat{\Theta}) \approx -150{,}000$). The network G has 6,474 nodes which means the space of all permutations has $\approx 10^{22{,}000}$ elements.

First, we sampled 1 billion (10^9) permutations σ_i uniformly at random, that is, $P(\sigma_i) = 1/(6{,}474!)$ and for each evaluated its log-likelihood $l(\sigma_i|\Theta) = \log P(\Theta|G, \sigma_i)$. We ordered the permutations in deceasing log-likelihood and plotted $l(\sigma_i|\Theta)$ vs. rank. Figure 15(a) gives the plot. Notice that very few random permutations are very bad (i.e., they give low likelihood), similarly few permutations are very good, while most of them are somewhere in between. Notice that best "random" permutation has log-likelihood of $\approx -320{,}000$, which is far below true likelihood $l(\hat{\Theta}) \approx -150{,}000$. This suggests that only a very small fraction of all permutations gives good node labelings.

On the other hand, we also repeated the same experiment but now using permutations sampled from the permutation distribution $\sigma_i \sim P(\sigma|\Theta, G)$ via our Metropolis sampling scheme. Figure 15(b) gives the plot. Notice the radical difference. Now the $l(\sigma|\Theta_i)$ very quickly converges to the true likelihood of $\approx -150{,}000$. This suggests that while the number of "good" permutations (accurate node mappings) is rather small, our sampling procedure quickly converges to the "good" part of the permutation space where node mappings are accurate, and spends the most time there.

6.2 Properties of the Optimization Space

In maximizing the likelihood we use a stochastic approximation to the gradient. This adds variance to the gradient and makes efficient optimization techniques, like conjugate gradient, highly unstable. Thus we use gradient descent, which is slower but easier to control. First, we make the following observation:

Observation 4 *Given a real graph G then finding the maximum likelihood Stochastic Kronecker initiator matrix $\hat{\Theta}$*

$$\hat{\Theta} = \arg\max_{\Theta} P(G|\Theta)$$

is a non-convex optimization problem.

Proof By definition permutations of the Kronecker graphs initiator matrix Θ all have the same log-likelihood. This means that we have several global minima that correspond to permutations of parameter matrix Θ, and then between them the log-likelihood drops. This means that the optimization problem is non-convex. ■

The above observation does not seem promising for estimating $\hat{\Theta}$ using gradient descent as it is prone to finding local minima. To test for this behavir we run the following experiment: we generated 100 synthetic Kronecker graphs on 16,384 (2^{14}) nodes and 1.4 million edges on the average, each with a randomly chosen 2×2 parameter matrix Θ^*. For each of the 100 graphs we run a single trial of gradient descent starting from a random parameter matrix Θ', and try to recover Θ^*. In 98% of the cases the gradient descent converged to the true parameters. Many times the algorithm converged to a different global minima, that is, $\hat{\Theta}$ is a permuted version of original parameter matrix Θ^*. Moreover, the median number of gradient descent iterations was only 52.

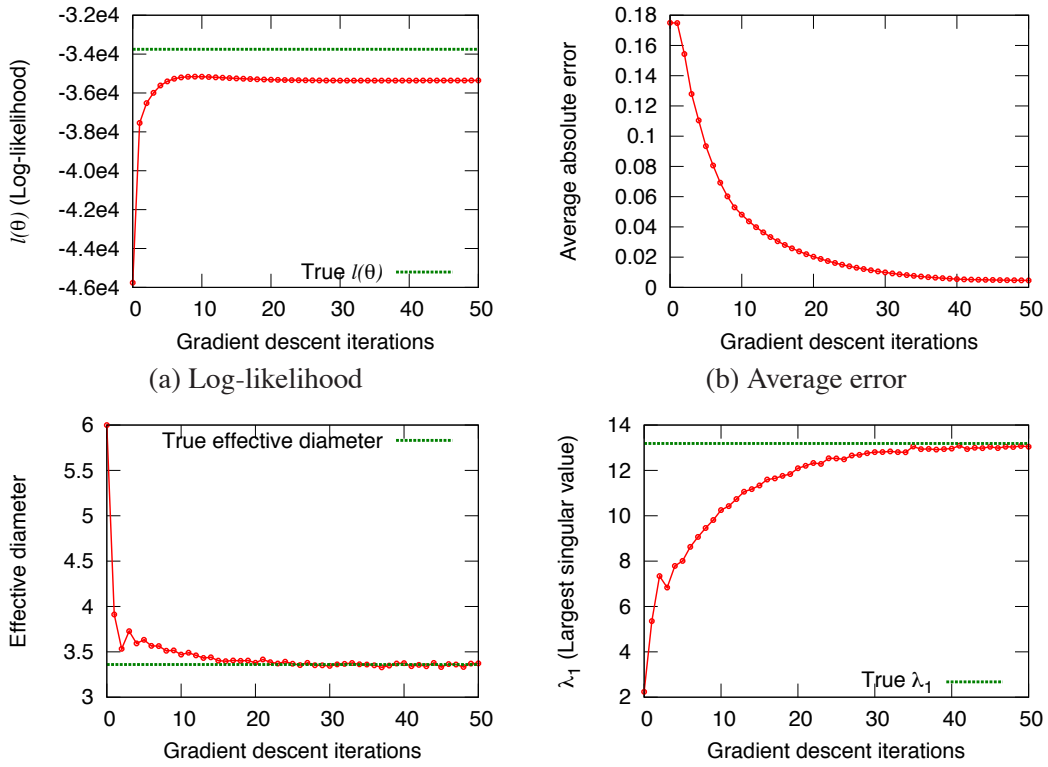
This suggests surprisingly nice structure of our optimization space: it seems to behave like a convex optimization problem with many equivalent global minima. Moreover, this experiment is also a good sanity check as it shows that given a Kronecker graph we can recover and identify the parameters that were used to generate it.

Moreover, Figure 15(c) plots the log-likelihood $l(\Theta_t)$ of the current parameter estimate Θ_t over the iterations t of the stochastic gradient descent. We plot the log-likelihood for 10 different runs of gradient descent, each time starting from a different random set of parameters Θ_0. Notice that in all runs gradient descent always converges towards the optimum, and none of the runs gets stuck in some local maxima.

6.3 Convergence of the Graph Properties

We approached the problem of estimating Stochastic Kronecker initiator matrix Θ by defining the likelihood over the individual entries of the graph adjacency matrix. However, what we would really like is to be given a real graph G and then generate a synthetic graph K that has similar properties as the real G. By properties we mean network statistics that can be computed from the graph, for example, diameter, degree distribution, clustering coefficient, etc. A priori it is not clear that our approach which tries to match individual entries of graph adjacency matrix will also be able to reproduce these global network statistics. However, as show next this is not the case.

To get some understanding of the convergence of the gradient descent in terms of the network properties we performed the following experiment. After every step t of stochastic gradient descent,



(a) Log-likelihood (b) Average error

(c) Effective diameter (d) Largest singular-value

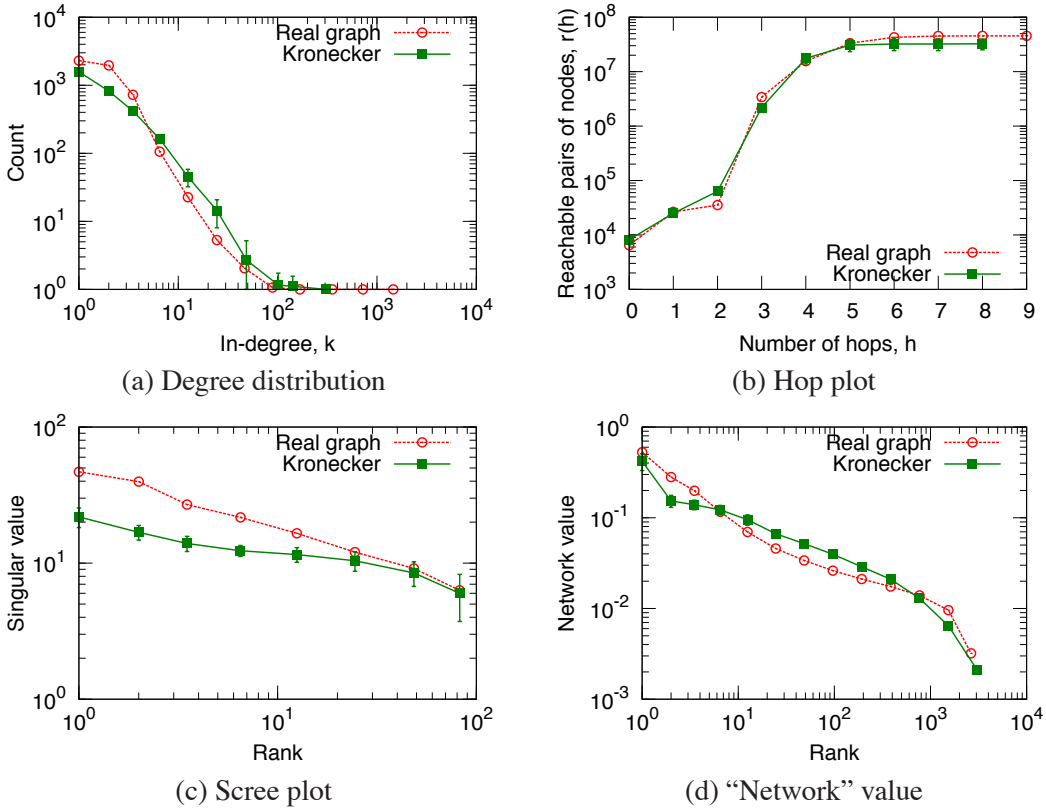
Figure 16: Convergence of graph properties with the number of iterations of gradient descent using the synthetic data set. We start with a random choice of parameters and with steps of gradient descent the Kronecker graph better and better matches network properties of the target graph.

we compare the true graph G with the synthetic Kronecker graph K_t generated using the current parameter estimates $\hat{\Theta}_t$. Figure 16(a) gives the convergence of log-likelihood, and (b) gives absolute error in parameter values ($\sum |\hat{\theta}_{ij} - \theta^*_{ij}|$, where $\hat{\theta}_{ij} \in \hat{\Theta}_t$, and $\theta^*_{ij} \in \Theta^*$). Similarly, Figure 16(c) plots the effective diameter, and (d) gives the largest singular value of graph adjacency matrix K as it converges to largest singular value of G.

Note how with progressing iterations of gradient descent properties of graph K_t quickly converge to those of G even though we are not directly optimizing the similarity in network properties: log-likelihood increases, absolute error of parameters decreases, diameter and largest singular value of K_t both converge to G. This is a nice result as it shows that through maximizing the likelihood the resulting graphs become more and more similar also in their structural properties (even though we are not directly optimizing over them).

6.4 Fitting to Real-world Networks

Next, we present experiments of fitting Kronecker graph model to real-world networks. Given a real network G we aim to discover the most likely parameters $\hat{\Theta}$ that ideally would generate a synthetic



(a) Degree distribution (b) Hop plot

(c) Scree plot (d) "Network" value

Figure 17: *Autonomous Systems (*AS-ROUTEVIEWS*):* Overlayed patterns of real graph and the fitted Kronecker graph. Notice that the fitted Kronecker graph matches patterns of the real graph while using only four parameters (2×2 initiator matrix).

graph K having similar properties as real G. This assumes that Kronecker graphs are a good model of the network structure, and that KRONFIT is able to find good parameters. In previous section we showed that KRONFIT can efficiently recover the parameters. Now we examine how well can Kronecker graph model the structure of real networks.

We consider several different networks, like a graph of connectivity among Internet Autonomous systems (AS-ROUTEVIEWS) with N =6,474 and E =26,467 a who-trusts-whom type social network from Epinions (Richardson et al., 2003) (EPINIONS) with N =75,879 and E =508,960 and many others. The largest network we consider for fitting is FLICKR photo-sharing online social network with 584,207 nodes and 3,555,115 edges.

For the purpose of this section we take a real network G, find parameters $\hat{\Theta}$ using KRONFIT, generate a synthetic graph K using $\hat{\Theta}$, and then compare G and K by comparing their properties that we introduced in Section 2. In all experiments we started from a random point (random initiator matrix) and run gradient descent for 100 steps. At each step we estimate the likelihood and the gradient based on 510,000 sampled permutations where we discard first 10,000 samples to allow the chain to burn-in.

6.4.1 FITTING TO AUTONOMOUS SYSTEMS NETWORK

First, we focus on the Autonomous Systems network obtained from the University of Oregon Route Views project (RouteViews, 1997). Given the AS network G we run KRONFIT to obtain parameter estimates $\hat{\Theta}$. Using the $\hat{\Theta}$ we then generate a synthetic Kronecker graph K, and compare the properties of G and K.

Figure 17 shows properties of AS-ROUTEVIEWS, and compares them with the properties of a synthetic Kronecker graph generated using the fitted parameters $\hat{\Theta}$ of size 2×2. Notice that properties of both graphs match really well. The estimated parameters are $\hat{\Theta} = [0.987, 0.571; 0.571, 0.049]$.

Figure 17(a) compares the degree distributions of the AS-ROUTEVIEWS network and its synthetic Kronecker estimate. In this and all other plots we use the exponential binning which is a standard procedure to de-noise the data when plotting on log-log scales. Notice a very close match in degree distribution between the real graph and its synthetic counterpart.

Figure 17(b) plots the cumulative number of pairs of nodes $g(h)$ that can be reached in $\leq h$ hops. The hop plot gives a sense about the distribution of the shortest path lengths in the network and about the network diameter. Last, Figures 17(c) and (d) plot the spectral properties of the graph adjacency matrix. Figure 17(c) plots largest singular values vs. rank, and (d) plots the components of left singular vector (the network value) vs. the rank. Again notice the good agreement with the real graph while using only four parameters.

Moreover, on all plots the error bars of two standard deviations show the variance of the graph properties for different realizations $R(\hat{\Theta}^{[k]})$. To obtain the error bars we took the same $\hat{\Theta}$, and generated 50 realizations of a Kronecker graph. As for the most of the plots the error bars are so small to be practically invisible, this shows that the variance of network properties when generating a Stochastic Kronecker graph is indeed very small.

Also notice that the AS-ROUTEVIEWS is an undirected graph, and that the fitted parameter matrix $\hat{\Theta}$ is in fact symmetric. This means that without a priori biasing the fitting towards undirected graphs, the recovered parameters obey this aspect of the network. Fitting AS-ROUTEVIEWS graph from a random set of parameters, performing gradient descent for 100 iterations and at each iteration sampling half a million permutations, took less than 10 minutes on a standard desktop PC. This is a significant speedup over Bezáková et al. (2006), where by using a similar permutation sampling approach for calculating the likelihood of a preferential attachment model on similar AS-ROUTEVIEWS graph took about two days on a cluster of 50 machines.

6.4.2 CHOICE OF THE INITIATOR MATRIX SIZE N_1

As mentioned earlier for finding the optimal number of parameters, that is, selecting the size of initiator matrix, BIC criterion naturally applies to the case of Kronecker graphs. Figure 23(b) shows BIC scores for the following experiment: We generated Kronecker graph with N =2,187 and E =8,736 using $N_1 = 3$ (9 parameters) and $k = 7$. For $1 \leq N_1 \leq 9$ we find the MLE parameters using gradient descent, and calculate the BIC scores. The model with the lowest score is chosen. As Figure 23(b) shows we recovered the true model, that is, BIC score is the lowest for the model with the true number of parameters, $N_1 = 3$.

Intuitively we expect a more complex model with more parameters to fit the data better. Thus we expect larger N_1 to generally give better likelihood. On the other hand the fit will also depend on the size of the graph G. Kronecker graphs can only generate graphs on N_1^k nodes, while real graphs do not necessarily have N_1^k nodes (for some, preferably small, integers N_1 and k). To solve this

N_1	$l(\hat{\Theta})$	N_1^k	E_1^k	$\lvert\{\deg(u) > 0\}\rvert$	BIC score
2	−152,499	8,192	25,023	5,675	152,506
3	−127,066	6,561	28,790	5,683	127,083
4	−153,260	16,384	24,925	8,222	153,290
5	−149,949	15,625	29,111	9,822	149,996
6	−128,241	7,776	26,557	6,623	128,309
AS-ROUTEVIEWS			26,467	6,474	

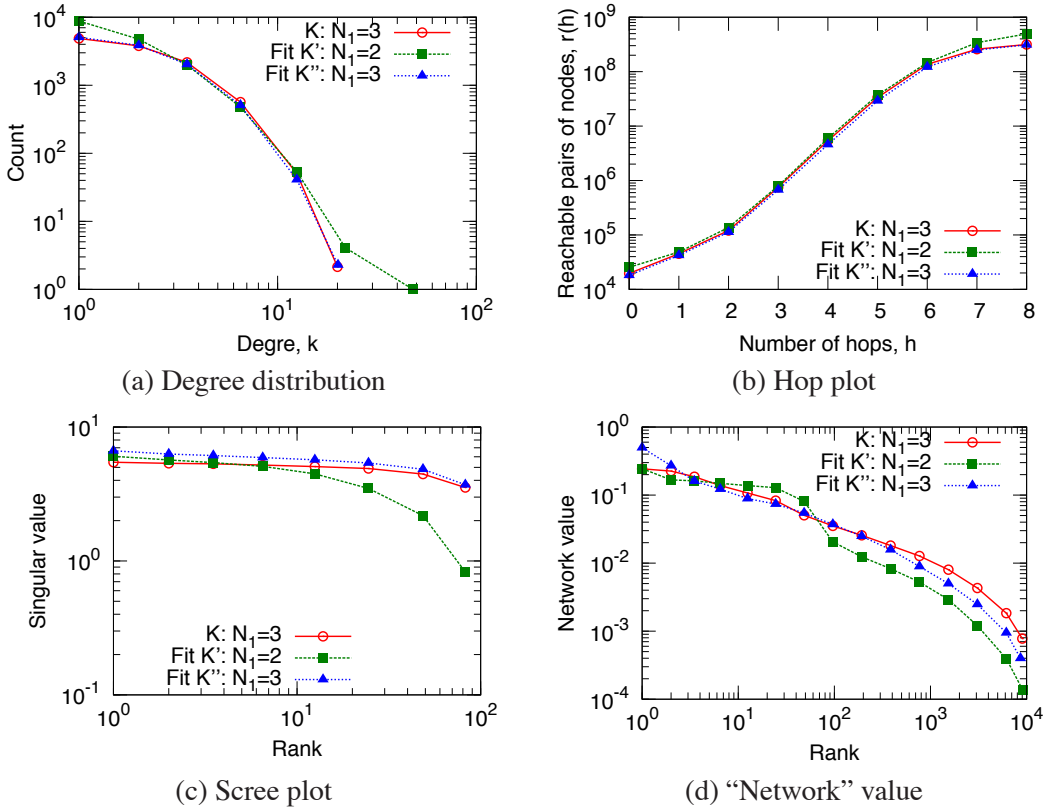
Table 2: Log-likelihood at MLE for different choices of the size of the initiator matrix N_1 for the AS-ROUTEVIEWS graph. Notice the log-likelihood $l(\hat{\theta})$ generally increases with the model complexity N_1. Also notice the effect of zero-padding, that is, for $N_1 = 4$ and $N_1 = 5$ the constraint of the number of nodes being an integer power of N_1 decreases the log-likelihood. However, the column $\lvert\{\deg(u) > 0\}\rvert$ gives the number of non-isolated nodes in the network which is much less than N_1^k and is in fact very close to the true number of nodes in the AS-ROUTEVIEWS. Using the BIC scores we see that $N_1 = 3$ or $N_1 = 6$ are best choices for the size of the initiator matrix.

problem we choose k so that $N_1^{k-1} < N(G) \leq N_1^k$, and then augment G by adding $N_1^k - N$ isolated nodes. Or equivalently, we pad the adjacency matrix of G with zeros until it is of the appropriate size, $N_1^k \times N_1^k$. While this solves the problem of requiring the integer power of the number of nodes it also makes the fitting problem harder as when $N \ll N_1^k$ we are basically fitting G plus a large number of isolated nodes.

Table 2 shows the results of fitting Kronecker graphs to AS-ROUTEVIEWS while varying the size of the initiator matrix N_1. First, notice that in general larger N_1 results in higher log-likelihood $l(\hat{\Theta})$ at MLE. Similarly, notice (column N_1^k) that while AS-ROUTEVIEWS has 6,474 nodes, Kronecker estimates have up to 16,384 nodes (16,384$= 4^7$, which is the first integer power of 4 greater than 6,474). However, we also show the number of non-zero degree (non-isolated) nodes in the Kronecker graph (column $\lvert\{\deg(u) > 0\}\rvert$). Notice that the number of non-isolated nodes well corresponds to the number of nodes in AS-ROUTEVIEWS network. This shows that KRONFIT is actually fitting the graph well and it successfully fits the structure of the graph plus a number of isolated nodes. Last, column E_1^k gives the number of edges in the corresponding Kronecker graph which is close to the true number of edges of the AS-ROUTEVIEWS graph.

Last, comparing the log-likelihood at the MLE and the BIC score in Table 2 we notice that the log-likelihood heavily dominates the BIC score. This means that the size of the initiator matrix (number of parameters) is so small that overfitting is not a concern. Thus we can just choose the initiator matrix that maximizes the likelihood. A simple calculation shows that one would need to take initiator matrices with thousands of entries before the model complexity part of BIC score would start to play a significant role.

We further examine the sensitivity of the choice of the initiator size by the following experiment. We generate a Stochastic Kronecker graphs K on 9 parameters ($N_1 = 3$), and then fit a Kronecker graph K' with a smaller number of parameters (4 instead of $9, N_1' = 2$). And also a Kronecker graph K'' of the same complexity as K ($N_1'' = 3$).



(a) Degree distribution (b) Hop plot

(c) Scree plot (d) "Network" value

Figure 18: *3 by 3 Stochastic Kronecker graphs:* Given a Stochastic Kronecker graphs G generated from $N_1 = 3$ (red curve), we fit a Kronecker graph K' with $N_1' = 2$ (green) and K'' with $N_1'' = 3$ (blue). Not surprisingly K'' fits the properties of K perfectly as the model is the of same complexity. On the other hand K' has only 4 parameters (instead of 9 as in K and K'') and still fits well.

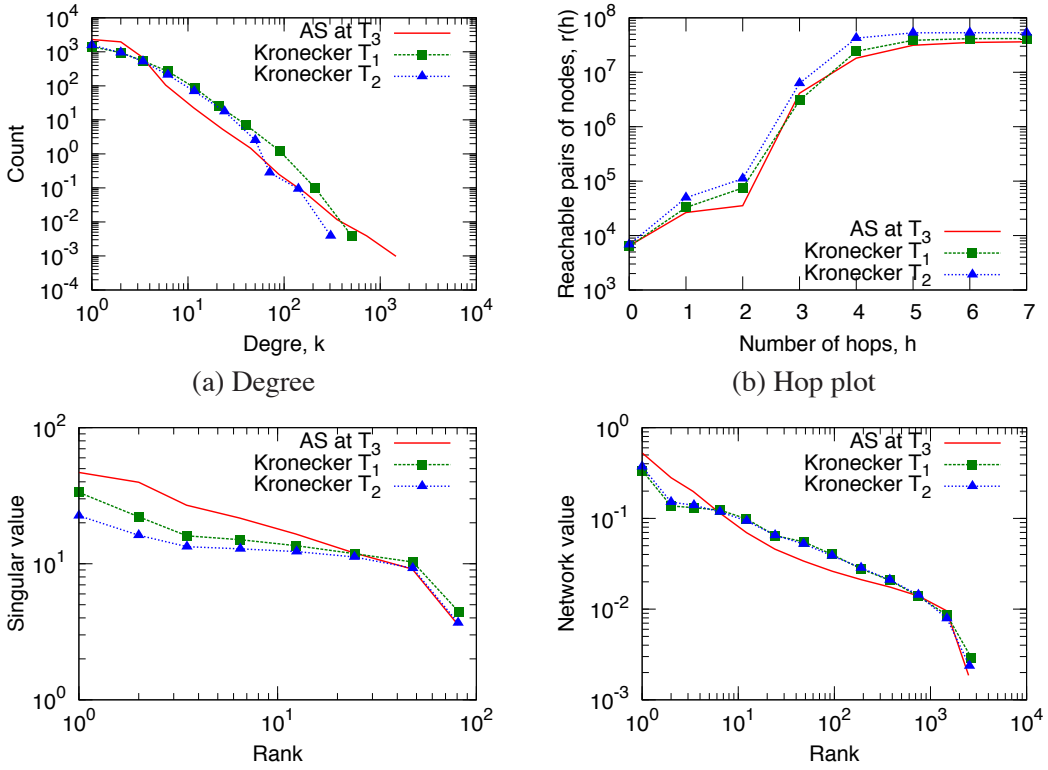
Figure 18 plots the properties of all three graphs. Not surprisingly K'' (blue) fits the properties of K (red) perfectly as the initiator is of the same size. On the other hand K' (green) is a simpler model with only 4 parameters (instead of 9 as in K and K'') and still generally fits well: hop plot and degree distribution match well, while spectral properties of graph adjacency matrix, especially scree plot, are not matched that well. This shows that nothing drastic happens and that even a bit too simple model still fits the data well. In general we observe empirically that by increasing the size of the initiator matrix one does not gain radically better fits for degree distribution and hop plot. On the other hand there is usually an improvement in the scree plot and the plot of network values when one increases the initiator size.

6.4.3 NETWORK PARAMETERS OVER TIME

Next we briefly examine the evolution of the Kronecker initiator for a temporally evolving graph. The idea is that given parameter estimates of a real-graph G_t at time t, we can forecast the future structure of the graph G_{t+x} at time $t+x$, that is, using parameters obtained from G_t we can generate a larger synthetic graph K that will be similar to G_{t+x}.

Snapshot at time	N	E	$l(\hat{\Theta})$	Estimates at MLE, $\hat{\Theta}$
T_1	2,048	8,794	−40,535	[0.981, 0.633; 0.633, 0.048]
T_2	4,088	15,711	−82,675	[0.934, 0.623; 0.622, 0.044]
T_3	6,474	26,467	−152,499	[0.987, 0.571; 0.571, 0.049]

Table 3: Parameter estimates of the three temporal snapshots of the AS-ROUTEVIEWS network. Notice that estimates stay remarkably stable over time.



(a) Degree (b) Hop plot

(c) Scree plot (d) "Network" value

Figure 19: *Autonomous systems network over time* (AS-ROUTEVIEWS): Overlayed patterns of real AS-ROUTEVIEWS network at time T_3 and the Kronecker graphs with parameters estimated from AS-ROUTEVIEWS at time T_1 and T_2. Notice good fits which means that parameters estimated on historic snapshots can be used to estimate the graph in the future.

As we have the information about the evolution of the AS-ROUTEVIEWS network, we estimated parameters for three snapshots of the network when it had about 2^k nodes. Table 3 gives the results of the fitting for the three temporal snapshots of the AS-ROUTEVIEWS network. Notice the parameter estimates $\hat{\Theta}$ remain remarkably stable over time. This means that Kronecker graphs can be used to estimate the structure of the networks in the future, that is, parameters estimated from the historic data can extrapolate the graph structure in the future.

Network	N	E	Estimated MLE parameters $\hat{\Theta}$	$l(\hat{\Theta})$	Time
AS-ROUTEVIEWS	6,474	26,467	[0.987, 0.571; 0.571, 0.049]	−152,499	8m15s
ATP-GR-QC	19,177	26,169	[0.902, 0.253; 0.221, 0.582]	−242,493	7m40s
BIO-PROTEINS	4,626	29,602	[0.847, 0.641; 0.641, 0.072]	−185,130	43m41s
EMAIL-INSIDE	986	32,128	[0.999, 0.772; 0.772, 0.257]	−107,283	1h07m
CA-GR-QC	5,242	28,980	[0.999, 0.245; 0.245, 0.691]	−160,902	14m02s
AS-NEWMAN	22,963	96,872	[0.954, 0.594; 0.594, 0.019]	−593,747	28m48s
BLOG-NAT05-6M	31,600	271,377	[0.999, 0.569; 0.502, 0.221]	−1,994,943	47m20s
BLOG-NAT06ALL	32,443	318,815	[0.999, 0.578; 0.517, 0.221]	−2,289,009	52m31s
CA-HEP-PH	12,008	237,010	[0.999, 0.437; 0.437, 0.484]	−1,272,629	1h22m
CA-HEP-TH	9,877	51,971	[0.999, 0.271; 0.271, 0.587]	−343,614	21m17s
CIT-HEP-PH	30,567	348,721	[0.994, 0.439; 0.355, 0.526]	−2,607,159	51m26s
CIT-HEP-TH	27,770	352,807	[0.990, 0.440; 0.347, 0.538]	−2,507,167	15m23s
EPINIONS	75,879	508,837	[0.999, 0.532; 0.480, 0.129]	−3,817,121	45m39s
GNUTELLA-25	22,687	54,705	[0.746, 0.496; 0.654, 0.183]	−530,199	16m22s
GNUTELLA-30	36,682	88,328	[0.753, 0.489; 0.632, 0.178]	−919,235	14m20s
DELICIOUS	205,282	436,735	[0.999, 0.327; 0.348, 0.391]	−4,579,001	27m51s
ANSWERS	598,314	1,834,200	[0.994, 0.384; 0.414, 0.249]	−20,508,982	2h35m
CA-DBLP	425,957	2,696,489	[0.999, 0.307; 0.307, 0.574]	−26,813,878	3h01m
FLICKR	584,207	3,555,115	[0.999, 0.474; 0.485, 0.144]	−32,043,787	4h26m
WEB-NOTREDAME	325,729	1,497,134	[0.999, 0.414; 0.453, 0.229]	−14,588,217	2h59m

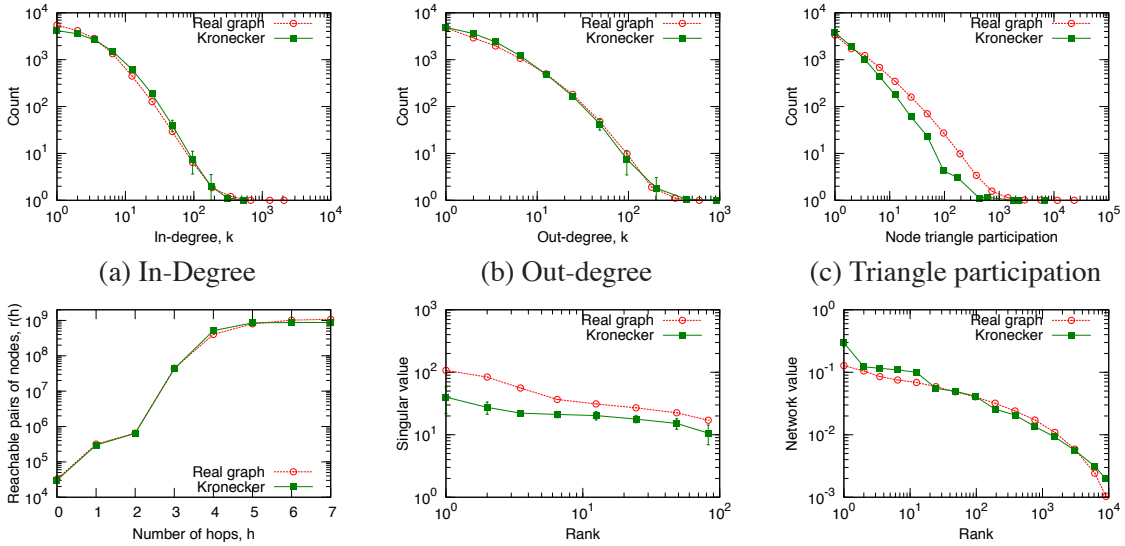
Table 4: Results of parameter estimation for 20 different networks. Table 5 gives the description and basic properties of the above network data sets. Networks and KRONFIT code are available for download at `http://snap.stanford.edu`.

Figure 19 further explores this. It overlays the graph properties of the real AS-ROUTEVIEWS network at time T_3 and the synthetic graphs for which we used the parameters obtained on historic snapshots of AS-ROUTEVIEWS at times T_1 and T_2. The agreements are good which demonstrates that Kronecker graphs can forecast the structure of the network in the future.

Moreover, this experiments also shows that parameter estimates do not suffer much from the zero padding of graph adjacency matrix (i.e., adding isolated nodes to make G have N_1^k nodes). Snapshots of AS-ROUTEVIEWS at T_1 and T_2 have close to 2^k nodes, while we had to add 26% (1,718) isolated nodes to the network at T_3 to make the number of nodes be 2^k. Regardless of this we see the parameter estimates $\hat{\Theta}$ remain basically constant over time, which seems to be independent of the number of isolated nodes added. This means that the estimated parameters are not biased too much from zero padding the adjacency matrix of G.

6.5 Fitting to Other Large Real-world Networks

Last, we present results of fitting Stochastic Kronecker graphs to 20 large real-world networks: large online social networks, like EPINIONS, FLICKR and DELICIOUS, web and blog graphs (WEB-NOTREDAME, BLOG-NAT05-6M, BLOG-NAT06ALL), internet and peer-to-peer networks (AS-NEWMAN, GNUTELLA-25, GNUTELLA-30), collaboration networks of co-authorships from DBLP (CA-DBLP) and various areas of physics (CA-HEP-TH, CA-HEP-PH, CA-GR-QC), physics citation networks (CIT-HEP-PH, CIT-HEP-TH), an email network (EMAIL-INSIDE), a protein inter-



(a) In-Degree (b) Out-degree (c) Triangle participation

(d) Hop plot (e) Scree plot (f) "Network" value

Figure 20: *Blog network* (BLOG-NAT06ALL): Overlayed patterns of real network and the estimated Kronecker graph using 4 parameters (2×2 initiator matrix). Notice that the Kronecker graph matches all properties of the real network.

action network BIO-PROTEINS, and a bipartite affiliation network (authors-to-papers, ATP-GR-QC). Refer to Table 5 in the appendix for the description and basic properties of these networks. They are available for download at `http://snap.stanford.edu`.

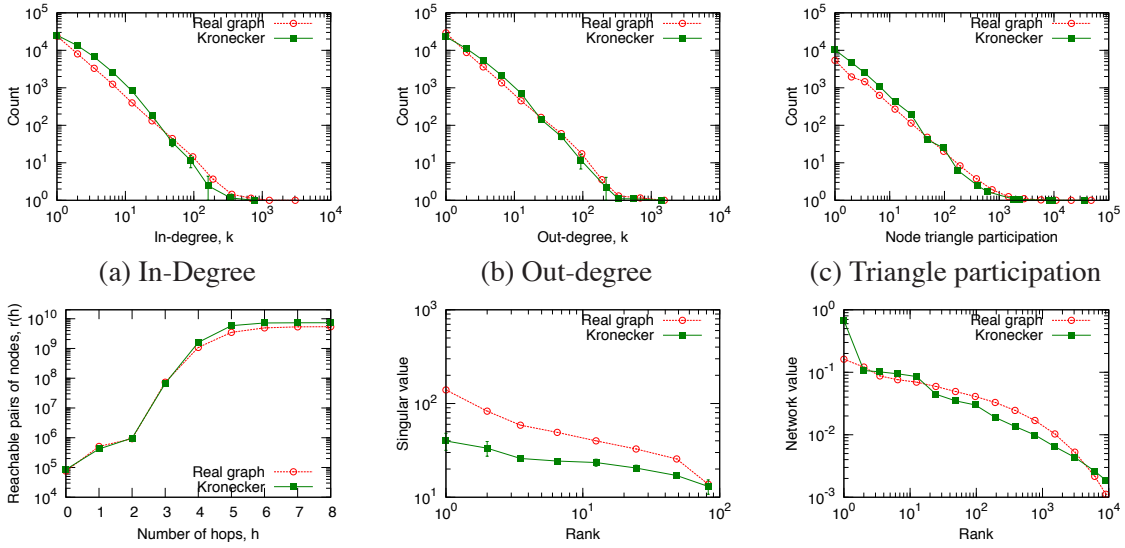
For each data set we started gradient descent from a random point (random initiator matrix) and ran it for 100 steps. At each step we estimate the likelihood and the gradient based on 510,000 sampled permutations where we discard first 10,000 samples to allow the chain to burn-in.

Table 4 gives the estimated parameters, the corresponding log-likelihoods and the wall clock times. All experiments were carried out on standard desktop computer. Notice that the estimated initiator matrices $\hat{\Theta}$ seem to have almost universal structure with a large value in the top left entry, a very small value at the bottom right corner and intermediate values in the other two corners. We further discuss the implications of such structure of Kronecker initiator matrix on the global network structure in next section.

Last, Figures 20 and 21 show overlays of various network properties of real and the estimated synthetic networks. In addition to the network properties we plotted in Figure 18, we also separately plot in- and out-degree distributions (as both networks are directed) and plot the node triangle participation in panel (c), where we plot the number of triangles a node participates in versus the number of such nodes. (Again the error bars show the variance of network properties over different realizations $R(\hat{\Theta}^{[k]})$ of a Stochastic Kronecker graph.)

Notice that for both networks and in all cases the properties of the real network and the synthetic Kronecker coincide really well. Using Stochastic Kronecker graphs with just 4 parameters we match the scree plot, degree distributions, triangle participation, hop plot and network values.

Given the previous experiments from the Autonomous systems graph we only present the results for the simplest model with initiator size $N_1 = 2$. Empirically we also observe that $N_1 = 2$ gives



(a) In-Degree (b) Out-degree (c) Triangle participation

(d) Hop plot (e) Scree plot (f) "Network" value

Figure 21: EPINIONS *who-trusts-whom social network:* Overlayed patterns of real network and the fitted Kronecker graph using only 4 parameters (2×2 initiator matrix). Again, the synthetic Kronecker graph matches all the properties of the real network.

surprisingly good fits and the estimation procedure is the most robust and converges the fastest. Using larger initiator matrices $N_1 > 2$ generally helps improve the likelihood but not dramatically. In terms of matching the network properties we also gent a slight improvement by making the model more complex. Figure 22 gives the percent improvement in log-likelihood as we make the model more complex. We use the log-likelihood of a 2×2 model as a baseline and estimate the log-likelihood at the MLE for larger initiator matrices. Again, models with more parameters tend to fit better. However, sometimes due to zero-padding of a graph adjacency matrix they actually have lower log-likelihood (as for example seen in Table 2).

6.6 Scalability

Last we also empirically evaluate the scalability of the KRONFIT. The experiment confirms that KRONFIT runtime scales linearly with the number of edges E in a graph G. More precisely, we performed the following experiment.

We generated a sequence of increasingly larger synthetic graphs on N nodes and $8N$ edges, and measured the time of one iteration of gradient descent, that is, sample 1 million permutations and evaluate the gradients. We started with a graph on 1,000 nodes, and finished with a graph on 8 million nodes, and 64 million edges. Figure 23(a) shows KRONFIT scales *linearly* with the size of the network. We plot wall-clock time vs. size of the graph. The dashed line gives a linear fit to the data points.

7. Discussion

Here we discuss several of the desirable properties of the proposed Kronecker graphs.

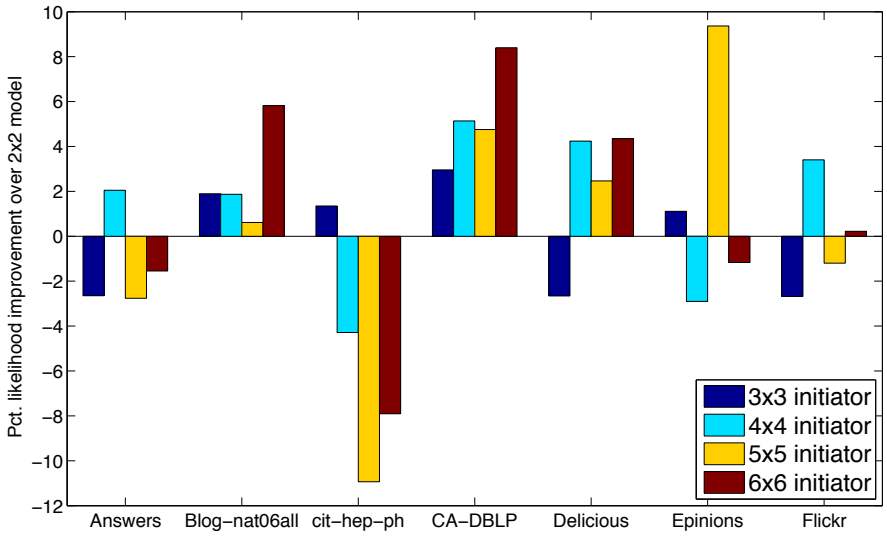
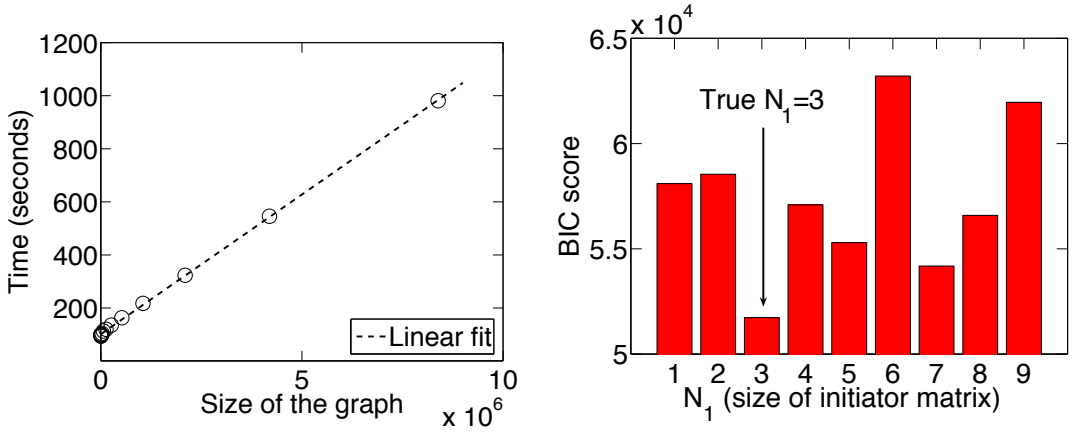


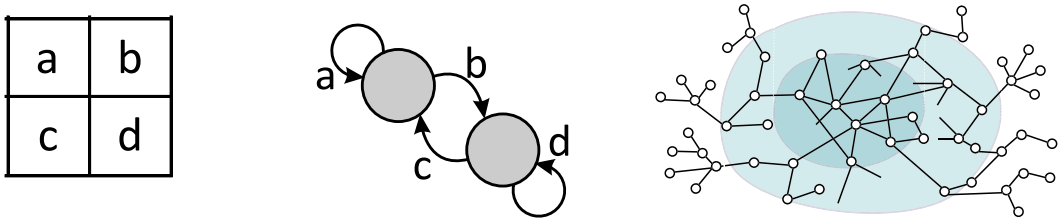
Figure 22: Percent improvement in log-likelihood over the 2×2 model as we increase the model complexity (size of initiator matrix). In general larger initiator matrices that have more degrees of freedom help improving the fit of the model.



(a) Scalability (b) Model selection

Figure 23: (a) Processor time to sample 1 million gradients as the graph grows. Notice the algorithm scales linearly with the graph size. (b) BIC score for model selection.

Generality: Stochastic Kronecker graphs include several other generators as special cases: For $\theta_{ij} = c$, we obtain the classical Erdős-Rényi random graph model. For $\theta_{i,j} \in \{0,1\}$, we obtain a deterministic Kronecker graph. Setting the K_1 matrix to a 2×2 matrix, we obtain the R-MAT generator (Chakrabarti et al., 2004). In contrast to Kronecker graphs, the RMAT cannot extrapolate into the future, since it needs to know the number of edges to insert. Thus, it is incapable of obeying the densification power law.



(a) 2×2 initiator matrix (b) Two recursive communities (c) Core-periphery

Figure 24: 2×2 Kronecker initiator matrix (a) can be thought of as two communities where there are a and d edges inside each of the communities and b and c edges crossing the two communities as illustrated in (b). Each community can then be recursively divided using the same pattern. (c) The onion like core-periphery structure where the network gets denser and denser as we move towards the center of the network.

Phase transition phenomena: The Erdős-Rényi graphs exhibit phase transitions (Erdős and Rényi, 1960). Several researchers argue that real systems are "at the edge of chaos" or phase transition (Bak, 1996; Sole and Goodwin, 2000). Stochastic Kronecker graphs also exhibit phase transitions (Mahdian and Xu, 2007) for the emergence of the giant component and another phase transition for connectivity.

Implications to the structure of the large-real networks: Empirically we found that 2×2 initiator ($N_1 = 2$) fits well the properties of real-world networks. Moreover, given a 2×2 initiator matrix, one can look at it as a recursive expansion of two groups into sub-groups. We introduced this recursive view of Kronecker graphs back in Section 3. So, one can then interpret the diagonal values of Θ as the proportion of edges inside each of the groups, and the off-diagonal values give the fraction of edges connecting the groups. Figure 24 illustrates the setting for two groups.

For example, as shown in Figure 24, large a, d and small b, c would imply that the network is composed of hierarchically nested communities, where there are many edges inside each community and few edges crossing them (Leskovec, 2009). One could think of this structure as some kind of organizational or university hierarchy, where one expects the most friendships between people within same lab, a bit less between people in the same department, less across different departments, and the least friendships to be formed across people from different schools of the university.

However, parameter estimates for a wide range of networks presented in Table 4 suggests a very different picture of the network structure. Notice that for most networks $a \gg b > c \gg d$. Moreover, $a \approx 1, b \approx c \approx 0.6$ and $d \approx 0.2$. We empirically observed that the same structure of initiator matrix $\hat{\Theta}$ also holds when fitting 3×3 or 4×4 models. Always the top left element is the largest and then the values on the diagonal decay faster than off the diagonal (Leskovec, 2009).

This suggests a network structure which is also known as *core-periphery* (Borgatti and Everett, 2000; Holme, 2005), the *jellyfish* (Tauro et al., 2001; Siganos et al., 2006), or the *octopus* (Chung and Lu, 2006) structure of the network as illustrated in Figure 24(c).

All of the above basically say that the network is composed of a densely linked network core and the periphery. In our case this would imply the following structure of the initiator matrix. The core is modeled by parameter a and the periphery by d. Most edges are inside the core (large a), and very few between the nodes of periphery (small d). Then there are many more edges between the core and the periphery than inside the periphery ($b, c > d$) (Leskovec, 2009). This is exactly what

we see as well. And in spirit of Kronecker graphs the structure repeats recursively—the core again has the dense core and the periphery, and so on. And similarly the periphery itself has the core and the periphery.

This suggest an "onion" like *nested core-periphery* (Leskovec et al., 2008a,b) network structure as illustrated in Figure 24(c), where the network is composed of denser and denser layers as one moves towards the center of the network. We also observe similar structure of the Kronecker initiator when fitting 3×3 or 4×4 initiator matrix. The diagonal elements have large but decreasing values with off diagonal elements following same decreasing pattern.

One of the implications of this is that networks do not break nicely into hierarchically organized sets of communities that lend themselves to graph partitioning and community detection algorithms. On contrary, this suggests that large networks can be decomposed into a densely linked core with many small periphery pieces hanging off the core. This is in accordance with our recent results (Leskovec et al., 2008a,b), that make similar observation (but based on a completely different methodology based on graph partitioning) about the clustering and community structure of large real-world networks.

8. Conclusion

In conclusion, the main contribution of this work is a family of models of network structure that uses a non-traditional matrix operation, the *Kronecker product*. The resulting graphs (a) have all the static properties (heavy-tailed degree distribution, small diameter, etc.), (b) all the temporal properties (densification, shrinking diameter) that are found in real networks. And in addition, (c) we can formally prove all of these properties.

Several of the proofs are extremely simple, thanks to the rich theory of Kronecker multiplication. We also provide proofs about the diameter and effective diameter, and we show that Stochastic Kronecker graphs can mimic real graphs well.

Moreover, we also presented KRONFIT, a fast, scalable algorithm to estimate Stochastic Kronecker initiator, which can be then used to create a synthetic graph that mimics the properties of a given real network.

In contrast to earlier work, our work has the following novelties: (a) it is among the few that estimates the parameters of the chosen generator in a principled way, (b) it is among the few that has a concrete measure of goodness of the fit (namely, the likelihood), (c) it avoids the quadratic complexity of computing the likelihood by exploiting the properties of the Kronecker graphs, and (d) it avoids the factorial explosion of the node correspondence problem, by using the Metropolis sampling.

The resulting algorithm matches well all the known properties of real graphs, as we show with the Epinions graph and the AS graph, it scales linearly on the number of edges, and it is orders of magnitudes faster than earlier graph-fitting attempts: 20 minutes on a commodity PC, versus 2 days on a cluster of 50 workstations (Bezáková et al., 2006).

The benefits of fitting a Kronecker graph model into a real graph are several:

- *Extrapolation*: Once we have the Kronecker generator Θ for a given real matrix G (such that G is mimicked by $\Theta^{[k]}$), a larger version of G can be generated by $\Theta^{[k+1]}$.
- *Null-model*: When analyzing a real network G one often needs to asses the significance of the observation. $\Theta^{[k]}$ that mimics G can be used as an accurate model of G.

- *Network structure*: Estimated parameters give insight into the global network and community structure of the network.

- *Forecasting*: As we demonstrated one can obtain Θ from a graph G_t at time t such that G is mimicked by $\Theta^{[k]}$. Then Θ can be used to model the structure of G_{t+x} in the future.

- *Sampling*: Similarly, if we want a realistic sample of the real graph, we could use a smaller exponent in the Kronecker exponentiation, like $\Theta^{[k-1]}$.

- *Anonymization*: Since $\Theta^{[k]}$ mimics G, we can publish $\Theta^{[k]}$, without revealing information about the nodes of the real graph G.

Future work could include extensions of Kronecker graphs to evolving networks. We envision formulating a dynamic Bayesian network with first order Markov dependencies, where parameter matrix at time t depends on the graph G_t at current time t and the parameter matrix at time $t-1$. Given a series of network snapshots one would then aim to estimate initiator matrices at individual time steps and the parameters of the model governing the evolution of the initiator matrix. We expect that based on the evolution of initiator matrix one would gain greater insight in the evolution of large networks.

Second direction for future work is to explore connections between Kronecker graphs and Random Dot Product graphs (Young and Scheinerman, 2007; Nickel, 2008). This also nicely connects with the "attribute view" of Kronecker graphs as described in Section 3.5. It would be interesting to design methods to estimate the individual node attribute values as well as the attribute-attribute similarity matrix (i.e., the initiator matrix). As for some networks node attributes are already given one could then try to infer "hidden" or missing node attribute values and this way gain insight into individual nodes as well as individual edge formations. Moreover, this would be interesting as one could further evaluate how realistic is the "attribute view" of Kronecker graphs.

Last, we also mention possible extensions of Kronecker graphs for modeling weighted and labeled networks. Currently Stochastic Kronecker graphs use a Bernoulli edge generation model, that is, an entry of big matrix $\mathcal{P}$ encodes the parameter of a Bernoulli coin. In similar spirit one could consider entries of $\mathcal{P}$ to encode parameters of different edge generative processes. For example, to generate networks with weights on edges an entry of $\mathcal{P}$ could encode the parameter of an exponential distribution, or in case of labeled networks one could use several initiator matrices in parallel and this way encode parameters of a multinomial distribution over different node attribute values.

Acknowledgments

Research supported by generous gifts form Microsoft, Yahoo! and IBM.

Appendix A. Table of Networks

Table 5 lists all the network data sets that were used in this paper. We also computed some of the structural network properties. Most of the networks can be obtained at
`http://snap.stanford.edu`.

Network	N	E	N_c	N_c/N	$\bar{C}$	D	$\bar{D}$	Description
Social networks								
ANSWERS	598,314	1,834,200	488,484	0.82	0.11	22	5.72	Yahoo! Answers social network (Leskovec et al., 2008a)
DELICIOUS	205,282	436,735	147,567	0.72	0.3	24	6.28	del.icio.us social network (Leskovec et al., 2008a)
EMAIL-INSIDE	986	32,128	986	1.00	0.45	7	2.6	European research organization email network (Leskovec et al., 2007a)
EPINIONS	75,879	508,837	75,877	1.00	0.26	15	4.27	Who-trusts-whom graph of epinions.com (Richardson et al., 2003)
FLICKR	584,207	3,555,115	404,733	0.69	0.4	18	5.42	Flickr photo sharing social network (Kumar et al., 2006)
Information (citation) networks								
BLOG-NAT05-6M	31,600	271,377	29,150	0.92	0.24	10	3.4	Blog-to-blog citation network (6 months of data) (Leskovec et al., 2007b)
BLOG-NAT06ALL	32,443	318,815	32,384	1.00	0.2	18	3.94	Blog-to-blog citation network (1 year of data) (Leskovec et al., 2007b)
CIT-HEP-PH	30,567	348,721	34,401	1.13	0.3	14	4.33	Citation network of ArXiv hep-th papers (Gehrke et al., 2003)
CIT-HEP-TH	27,770	352,807	27,400	0.99	0.33	15	4.2	Citations network of ArXiv hep-ph papers (Gehrke et al., 2003)
Collaboration networks								
CA-DBLP	425,957	2,696,489	317,080	0.74	0.73	23	6.75	DBLP co-authorship network (Backstrom et al., 2006)
CA-GR-QC	5,242	28,980	4,158	0.79	0.66	17	6.1	Co-authorship network in gr-qc category of ArXiv (Leskovec et al., 2005b)
CA-HEP-PH	12,008	237,010	11,204	0.93	0.69	13	4.71	Co-authorship network in hep-ph category of ArXiv (Leskovec et al., 2005b)
CA-HEP-TH	9,877	51,971	8,638	0.87	0.58	18	5.96	Co-authorship network in hep-th category of ArXiv (Leskovec et al., 2005b)
Web graphs								
WEB-NOTREDAME	325,729	1,497,134	325,729	1.00	0.47	46	7.22	Web graph of University of Notre Dame (Albert et al., 1999)
Internet networks								
AS-NEWMAN	22,963	96,872	22,963	1.00	0.35	11	3.83	AS graph from new (July 16, 2007)
AS-ROUTEVIEWS	6,474	26,467	6,474	1.00	0.4	9	3.72	AS from Oregon Route View (Leskovec et al., 2005b)
GNUTELLA-25	22,687	54,705	22,663	1.00	0.01	11	5.57	Gnutella P2P network on 3/25 2000 (Ripeanu et al., 2002)
GNUTELLA-30	36,682	88,328	36,646	1.00	0.01	11	5.75	Gnutella P2P network on 3/30 2000 (Ripeanu et al., 2002)
Bi-partite networks								
ATP-GR-QC	19,177	26,169	14,832	0.77	0	35	11.08	Affiliation network of gr-qc category in ArXiv (Leskovec et al., 2007b)
Biological networks								
BIO-PROTEINS	4,626	29,602	4,626	1.00	0.12	12	4.24	Yeast protein interaction network (Colizza et al., 2005)

Table 5: Network data sets we analyzed. Statistics of networks we consider: number of nodes N; number of edges E, number of nodes in largest connected component N_c, fraction of nodes in largest connected component N_c/N, average clustering coefficient $\bar{C}$; diameter D, and average path length $\bar{D}$. Networks are available for download at http://snap.stanford.edu.

References

Network data. http://www-personal.umich.edu/~mejn/netdata, July 16, 2007.

R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

R. Albert, H. Jeong, and A.-L. Barabási. Diameter of the world-wide web. *Nature*, 401:130–131, September 1999.

L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 44–54, 2006.

P. Bak. *How Nature Works: The Science of Self-Organized Criticality*. Springer, September 1996.

A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

A.-L. Barabási, E. Ravasz, and T. Vicsek. Deterministic scale-free networks. *Physica A*, 299: 559–564, 2001.

I. Bezáková, A. Kalai, and R. Santhanam. Graph model selection using maximum likelihood. In *ICML '06: Proceedings of the 23rd International Conference on Machine Learning*, pages 105–112, 2006.

Z. Bi, C. Faloutsos, and F. Korn. The DGX distribution for mining massive, skewed data. In *KDD '01: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 17–26, 2001.

A. Blum, H. Chan, and M. Rwebangira. A random-surfer web-graph model. In *ANALCO '06: Proceedings of the 3rd Workshop on Analytic Algorithmics and Combinatorics*, 2006.

S. P. Borgatti and M. G. Everett. Models of core/periphery structures. *Social Networks*, 21(4): 375–395, 2000.

A. Bottreau and Y. Metivier. Some remarks on the kronecker product of graphs. *Information Processing Letters*, 68(2):55 – 61, 1998.

A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. In *WWW '00: Proceedings of the 9th International Conference on World Wide Web*, 2000.

T. Bu and D. F. Towsley. On distinguishing between internet power law topology generators. In *INFOCOM*, 2002.

C. T. Butts. Permutation models for relational data. (tech. rep. mbs 05-02, Univ. of California, Irvine, 2005.

J. M. Carlson and J. Doyle. Highly optimized tolerance: a mechanism for power laws in designed systems. *Physical Review E*, 60(2):1412–1427, 1999.

D. Chakrabarti, Y. Zhan, and C. Faloutsos. R-mat: A recursive model for graph mining. In *SDM '04: SIAM Conference on Data Mining*, 2004.

T. Chow. The Q-spectrum and spanning trees of tensor products of bipartite graphs. *Proc. Amer. Math. Soc.*, 125:3155–3161, 1997.

F. R. K. Chung and L. Lu. *Complex Graphs and Networks*, volume 107 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 2006.

F. R. K. Chung, L. Lu, and V. Vu. Eigenvalues of random power law graphs. *Annals of Combinatorics*, 7:21–33, 2003.

A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *ArXiv*, ArXiv:0706.1062, Jun 2007.

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

V. Colizza, A. Flammini, M. A. Serrano, and A. Vespignani. Characterization and modeling of protein protein interaction networks. *Physica A Statistical Mechanics and its Applications*, 352: 1–27, 2005.

M. E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE /ACM Transactions on Networking*, 5(6):835–846, 1997.

S. Dill, R. Kumar, K. S. Mccurley, S. Rajagopalan, D. Sivakumar, and A. Tomkins. Self-similarity in the web. *ACM Trans. Interet Technology*, 2(3):205–223, 2002.

S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. Pseudofractal scale-free web. *Physical Review E*, 65(6):066122, Jun 2002.

B. Efron. Defining the curvature of a statistical problem (with applications to second order efficiency). *Ann. Statist.*, 3(6):1189–1242, 1975.

P. Erdős and A. Rényi. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Acadamy of Science*, 5:17–67, 1960.

A. Fabrikant, E. Koutsoupias, and C. H. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the internet. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages, and Programming*, volume 2380, 2002.

M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM '99: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 251–262, 1999.

I. Farkas, I. Deréni, A.-L. Barabási, and T. Vicsek. Spectra of "real-world" graphs: beyond the semicircle law. *Physical Review E*, 64(026704), 2001.

A. D. Flaxman, A. M. Frieze, and J. Vera. A geometric preferential attachment model of networks II. In *WAW '07: Proceedings of the 5th Workshop On Algorithms And Models For The Web-Graph*, pages 41–55, 2007.

D. Gamerman. *Markov Chain Monte Carlo, Stochastic Simulation for Bayesian Inference*. Chapman & Hall, London, 1997.

J. Gehrke, P. Ginsparg, and J. M. Kleinberg. Overview of the 2003 kdd cup. *SIGKDD Explorations*, 5(2):149–151, 2003.

A. Gelman, J. B. Carlin, H. S. Stern, and D.B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall, London, July 2003.

R. H. Hammack. Proof of a conjecture concerning the direct product of bipartite graphs. *European Journal of Combinatorics*, 30(5):1114 – 1118, 2009. Part Special Issue on Metric Graph Theory.

P. Holme. Core-periphery organization of complex networks. *Physical Review E*, 72:046111, 2005.

W. Imrich. Factoring cardinal product graphs in polynomial time. *Discrete Mathematics*, 192(1-3): 119–144, 1998.

W. Imrich and S. Klavžar. *Product Graphs: Structure and Recognition*. Wiley, 2000.

J. M. Kleinberg. The small-world phenomenon: an algorithmic perspective. Technical Report 99-1776, Cornell Computer Science Department, 1999.

J. M. Kleinberg, S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. The web as a graph: Measurements, models and methods. In *COCOON '99: Proceedings of the International Conference on Combinatorics and Computing*, 1999.

K. Klemm and V. M. Eguíluz. Highly clustered scale-free networks. *Phys. Rev. E*, 65(3):036123, Feb 2002.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 57, 2000.

R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 611–617, 2006.

S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. In *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 1999.

A. N. Langville and W. J. Stewart. The Kronecker product and stochastic automata networks. *Journal of Computation and Applied Mathematics*, 167:429–447, 2004.

J. Leskovec. Networks, communities and kronecker products. In *CNIKM '09: Complex Networks in Information and Knowledge Management*, 2009.

J. Leskovec and C. Faloutsos. Scalable modeling of real graphs using kronecker multiplication. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, 2007.

J. Leskovec, D. Chakrabarti, J. M. Kleinberg, and C. Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *PKDD '05: Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 133–145, 2005a.

J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD '05: Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 177–187, 2005b.

J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007a.

J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs. In *SDM '07: Proceedings of the SIAM Conference on Data Mining*, 2007b.

J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08: Proceedings of the 17th International Conference on World Wide Web*, 2008a.

J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *ArXiv*, arXiv:0810.1355, Oct 2008b.

C. F. Van Loan. The ubiquitous Kronecker product. *Journal of Computation and Applied Mathematics*, 123:85–100, 2000.

M. Mahdian and Y. Xu. Stochastic kronecker graphs. In *WAW '07: Proceedings of the 5th Workshop On Algorithms And Models For The Web-Graph*, pages 179–186, 2007.

M. Mihail and C. H. Papadimitriou. On the eigenvalue power law. In *RANDOM*, pages 254–262, 2002.

S. Milgram. The small-world problem. *Psychology Today*, 2:60–67, 1967.

C. L. M. Nickel. Random dot product graphs: A model for social networks. Ph.D. Thesis, Dept. of Applied Mathematics and Statistics, Johns Hopkins University, 2008.

C. R. Palmer, P. B. Gibbons, and C. Faloutsos. Anf: a fast and scalable tool for data mining in massive graphs. In *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 81–90, 2002.

D. M. Pennock, G. W. Flake, S. Lawrence, E. J. Glover, and C. L. Giles. Winners don't take all: Characterizing the competition for links on the Web. *Proceedings of the National Academy of Sciences*, 99(8):5207–5211, 2002.

V. V. Petrov. *Limit Theorems of Probability Theory*. Oxford University Press, 1995.

E. Ravasz and A.-L. Barabási. Hierarchical organization in complex networks. *Physical Review E*, 67(2):026112, 2003.

E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, 2002.

S. Redner. How popular is your paper? an empirical study of the citation distribution. *European Physical Journal B*, 4:131–134, 1998.

M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *ISWC*, 2003.

M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6 (1):50–57, Jan/Feb 2002.

J. Rissanen. Modelling by the shortest data description. *Automatica*, 14:465–471, 1978.

RouteViews. University of Oregon Route Views Project. Online data and reports. `http://www.routeviews.org`, 1997.

M. Sales-Pardo, R. Guimera, A. A. Moreira, and L. A. Amaral. Extracting the hierarchical organization of complex systems. *Proceedings of the National Academy of Sciences*, 104(39): 15224–15229, September 2007.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

G. Siganos, S. L. Tauro, and M. Faloutsos. Jellyfish: A conceptual model for the as internet topology. *Journal of Communications and Networks*, 8:339–350, 2006.

R. Sole and B. Goodwin. *Signs of Life: How Complexity Pervades Biology*. Perseus Books Group, New York, NY, 2000.

S. L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. A simple conceptual model for the internet topology. In *GLOBECOM '01: Global Telecommunications Conference*, volume 3, pages 1667 – 1671, 2001.

C. E. Tsourakakis. Fast counting of triangles in large real networks, without counting: Algorithms and laws. In *ICDM '08 : IEEE International Conference on Data Mining*, 2008.

A. Vázquez. Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations. *Phys. Rev. E*, 67(5):056104, May 2003.

S. Wasserman and P. Pattison. Logit models and logistic regressions for social networks. *Psychometrika*, 60:401–425, 1996.

D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

B. M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, December 1988.

P. M. Weichsel. The kronecker product of graphs. In *American Mathematical Society*, volume 13, pages 37–52, 1962.

J. Winick and S. Jamin. Inet-3.0: Internet Topology Generator. Technical Report CSE-TR-456-02, University of Michigan, Ann Arbor, 2002. URL `http://topology.eecs.umich.edu/inet/`.

C. Wiuf, M. Brameier, O. Hagberg, and M. P. Stumpf. A likelihood approach to analysis of network data. *Proceedings of the National Academy of Sciences*, 103(20):7566–7570, 2006.

S. J. Young and E. R. Scheinerman. Random dot product graph models for social networks. In *WAW '07: Proceedings of the 5th Workshop On Algorithms And Models For The Web-Graph*, pages 138–149, 2007.

E. Zheleva, H. Sharara, and L. Getoor. Co-evolution of social and affiliation networks. In *KDD*, pages 1007–1016, 2009.

Journal of Machine Learning Research 11 (2010) 1043-1080 Submitted 10/08; Revised 12/09; Published 3/10

Message-passing for Graph-structured Linear Programs: Proximal Methods and Rounding Schemes

Pradeep Ravikumar PRADEEPR@STAT.BERKELEY.EDU
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720

Alekh Agarwal ALEKH@CS.BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720

Martin J. Wainwright WAINWRIG@STAT.BERKELEY.EDU
Department of Statistics∗
University of California, Berkeley
Berkeley, CA 94720

Editor: Yair Weiss

Abstract

The problem of computing a maximum a posteriori (MAP) configuration is a central computational challenge associated with Markov random fields. There has been some focus on "tree-based" linear programming (LP) relaxations for the MAP problem. This paper develops a family of super-linearly convergent algorithms for solving these LPs, based on proximal minimization schemes using Bregman divergences. As with standard message-passing on graphs, the algorithms are distributed and exploit the underlying graphical structure, and so scale well to large problems. Our algorithms have a double-loop character, with the outer loop corresponding to the proximal sequence, and an inner loop of cyclic Bregman projections used to compute each proximal update. We establish convergence guarantees for our algorithms, and illustrate their performance via some simulations. We also develop two classes of rounding schemes, deterministic and randomized, for obtaining integral configurations from the LP solutions. Our deterministic rounding schemes use a "re-parameterization" property of our algorithms so that when the LP solution is integral, the MAP solution can be obtained even before the LP-solver converges to the optimum. We also propose graph-structured randomized rounding schemes applicable to iterative LP-solving algorithms in general. We analyze the performance of and report simulations comparing these rounding schemes.

Keywords: graphical models, MAP Estimation, LP relaxation, proximal minimization, rounding schemes

1. Introduction

A key computational challenge that arises in applications of discrete graphical models is to compute the most probable configuration(s), often referred to as the *maximum a posteriori* (MAP) problem. Although the MAP problem can be solved exactly in polynomial time on trees (and more generally, graphs with bounded treewidth) using the max-product algorithm, it is computationally challenging

∗. Also in the Department of Electrical Engineering and Computer Sciences.

for general graphs. Indeed, the MAP problem for general discrete graphical models includes a large number of classical NP-complete problems as special cases, including MAX-CUT, independent set, and various satisfiability problems.

This intractability motivates the development and analysis of methods for obtaining approximate solutions, and there is a long history of approaches to the problem. One class of methods is based on simulated annealing (Geman and Geman, 1984), but the cooling schedules required for theoretical guarantees are often prohibitively slow. Besag (1986) proposed the iterated conditional modes algorithm, which performs a sequence of greedy local maximizations to approximate the MAP solution, but may be trapped by local maxima. Greig et al. (1989) observed that for binary problems with attractive pairwise interactions (the ferromagnetic Ising model in statistical physics terminology), the MAP configuration can be computed in polynomial-time by reduction to a max-flow problem. The ordinary max-product algorithm, a form of non-serial dynamic-programming (Bertele and Brioschi, 1972), computes the MAP configuration exactly for trees, and is also frequently applied to graphs with cycles. Despite some local optimality results (Freeman and Weiss, 2001; Wainwright et al., 2004), it has no general correctness guarantees for graph with cycles, and even worse, it can converge rapidly to non-MAP configurations (Wainwright et al., 2005), even for problems that are easily solved in polynomial time (e.g., ferromagnetic Ising models). For certain graphical models arising in computer vision, Boykov et al. (2001) proposed graph-cut based search algorithms that compute a local maximum over two classes of moves. A broad class of methods are based on the principle of convex relaxation, in which the discrete MAP problem is relaxed to a convex optimization problem over continuous variables. Examples of this convex relaxation problem include linear programming relaxations (Koval and Schlesinger, 1976; Chekuri et al., 2005; Wainwright et al., 2005), as well as quadratic, semidefinite and other conic programming relaxations (for instance, (Ravikumar and Lafferty, 2006; Kumar et al., 2006; Wainwright and Jordan, 2004)).

Among the family of conic programming relaxations, linear programming (LP) relaxation is the least expensive computationally, and also the best understood. The primary focus of this paper is a well-known LP relaxation of the MAP estimation problem for pairwise Markov random fields, one which has been independently proposed by several groups (Koval and Schlesinger, 1976; Chekuri et al., 2005; Wainwright et al., 2005). This LP relaxation is based on optimizing over a set of locally consistent pseudomarginals on edges and vertices of the graph. It is an exact method for any tree-structured graph, so that it can be viewed naturally as a tree-based LP relaxation.[1] The first connection between max-product message-passing and LP relaxation was made by Wainwright et al. (2005), who connected the tree-based LP relaxation to the class of tree-reweighted max-product (TRW-MP) algorithms, showing that TRW-MP fixed points satisfying a strong "tree agreement" condition specify optimal solutions to the LP relaxation.

For general graphs, this first-order LP relaxation could be solved—at least in principle—by various standard algorithms for linear programming, including the simplex and interior-point methods (Bertsimas and Tsitsikilis, 1997; Boyd and Vandenberghe, 2004). However, such generic methods fail to exploit the graph-structured nature of the LP, and hence do not scale favorably to large-scale problems (Yanover et al., 2006). A body of work has extended the connection between the LP relaxation and message-passing algorithms in various ways. Kolmogorov (2005) developed a serial form of TRW-MP updates with certain convergence guarantees; he also showed that there exist fixed points of the TRW-MP algorithm, not satisfying strong tree agreement, that do not cor-

1. In fact, this LP relaxation is the first in a hierarchy of relaxations, based on the treewidth of the graph (Wainwright et al., 2005).

respond to optimal solutions of the LP. This issue has a geometric interpretation, related to the fact that coordinate ascent schemes (to which TRW-MP is closely related), need not converge to the global optima for convex programs that are not strictly convex, but can become trapped in corners. Kolmogorov and Wainwright (2005) showed that this trapping phenomena does not arise for graphical models with binary variables and pairwise interactions, so that TRW-MP fixed points are always LP optimal. Globerson and Jaakkola (2007b) developed a related but different dual-ascent algorithm, which is guaranteed to converge but is not guaranteed to solve the LP. Weiss et al. (2007) established connections between convex forms of the sum-product algorithm, and exactness of reweighted max-product algorithms; Johnson et al. (2007) also proposed algorithms related to convex forms of sum-product. Various authors have connected the ordinary max-product algorithm to the LP relaxation for special classes of combinatorial problems, including matching (Bayati et al., 2005; Huang and Jebara, 2007; Bayati et al., 2007) and independent set (Sanghavi et al., 2007). For general problems, max-product does *not* solve the LP; Wainwright et al. (2005) describe a instance of the MIN-CUT problem on which max-product fails, even though LP relaxation is exact. Other authors (Feldman et al., 2002a; Komodakis et al., 2007) have implemented subgradient methods which are guaranteed to solve the linear program, but such methods typically have sub-linear convergence rates (Bertsimas and Tsitsikilis, 1997).

This paper makes two contributions to this line of work. Our first contribution is to develop and analyze a class of message-passing algorithms with the following properties: their only fixed points are LP-optimal solutions, they are provably convergent with at least a geometric rate, and they have a distributed nature, respecting the graphical structure of the problem. All of the algorithms in this paper are based on the well-established idea of *proximal minimization*: instead of directly solving the original linear program itself, we solve a sequence of so-called proximal problems, with the property that the sequence of associated solutions is guaranteed to converge to the LP solution. We describe different classes of algorithms, based on different choices of the proximal function: quadratic, entropic, and tree-reweighted entropies. For all choices, we show how the intermediate proximal problems can be solved by forms of message-passing on the graph that are similar to but distinct from the ordinary max-product or sum-product updates. An additional desirable feature, given the wide variety of lifting methods for further constraining LP relaxations (Wainwright and Jordan, 2003), is that new constraints can be incorporated in a relatively seamless manner, by introducing new messages to enforce them.

Our second contribution is to develop various types of rounding schemes that allow for early termination of LP-solving algorithms. There is a substantial body of past work (e.g., Raghavan and Thompson, 1987) on rounding fractional LP solutions so as to obtain integral solutions with approximation guarantees. Our use of rounding is rather different: instead, we consider rounding schemes applied to problems for which the LP solution is integral, so that rounding would be unnecessary if the LP were solved to optimality. In this setting, the benefit of certain rounding procedures (in particular, those that we develop) is allowing an LP-solving algorithm to be terminated *before* it has solved the LP, while still returning the MAP configuration, either with a deterministic or high probability guarantee. Our deterministic rounding schemes apply to a class of algorithms which, like the proximal minimization algorithms that we propose, maintain a certain invariant of the original problem. We also propose and analyze a class of graph-structured randomized rounding procedures that apply to any algorithm that approaches the optimal LP solution from the interior of the relaxed polytope. We analyze these rounding schemes, and give finite bounds on the number of iterations required for the rounding schemes to obtain an integral MAP solution.

The remainder of this paper is organized as follows. We begin in Section 2 with background on Markov random fields, and the first-order LP relaxation. In Section 3, we introduce the notions of proximal minimization and Bregman divergences, then derive various of message-passing algorithms based on these notions, and finally discuss their convergence properties. Section 4 is devoted to the development and analysis of rounding schemes, both for our proximal schemes as well as other classes of LP-solving algorithms. We provide experimental results in Section 5, and conclude with a discussion in Section 6.

2. Background

We begin by introducing some background on Markov random fields, and the LP relaxations that are the focus of this paper. Given a discrete space $\mathcal{X} = \{0, 1, 2, \ldots, m-1\}$, let $X = (X_1, \ldots, X_N) \in \mathcal{X}^N$ denote a N-dimensional discrete random vector. (While we have assumed the variables take values in the same set $\mathcal{X}$, we note that our results easily generalize to the case where the variables take values in different sets with differing cardinalities.) We assume that the distribution $\mathbb{P}$ of the random vector is a Markov random field, meaning that it factors according to the structure of an undirected graph $G = (V, E)$, with each variable X_s associated with one node $s \in V$, in the following way. Letting $\theta_s : \mathcal{X} \to \mathbb{R}$ and $\theta_{st} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be singleton and edgewise potential functions respectively, we assume that the distribution takes the form

$$\mathbb{P}(x; \theta) \quad \propto \quad \exp\Big\{ \sum_{s \in V} \theta_s(x_s) + \sum_{(s,t) \in E} \theta_{st}(x_s, x_t) \Big\}.$$

The problem of *maximum a posteriori* (MAP) estimation is to compute a configuration with maximum probability—that is, an element

$$x^* \quad \in \quad \arg\max_{x \in \mathcal{X}^N} \Big\{ \sum_{s \in V} \theta_s(x_s) + \sum_{(s,t) \in E} \theta_{st}(x_s, x_t) \Big\}, \tag{1}$$

where the arg max operator extracts the configurations that achieve the maximal value. This problem is an integer program, since it involves optimizing over the discrete space $\mathcal{X}^N$. For future reference, we note that the functions $\theta_s(\cdot)$ and $\theta_{st}(\cdot)$ can always be represented in the form

$$\theta_s(x_s) = \sum_{j \in \mathcal{X}} \theta_{s;j} \mathbb{I}[x_s = j],$$
$$\theta_{st}(x_s, x_t) = \sum_{j,k \in \mathcal{X}} \theta_{st;jk} \mathbb{I}[x_s = j; x_t = k],$$

where the m-vectors $\{\theta_{s;j}, j \in \mathcal{X}\}$ and $m \times m$ matrices $\{\theta_{st;jk},\ (j,k) \in \mathcal{X} \times \mathcal{X}\}$ parameterize the problem.

The first-order linear programming (LP) relaxation (Koval and Schlesinger, 1976; Chekuri et al., 2005; Wainwright et al., 2005) of this problem is based on a set of pseudomarginals μ_s and μ_{st}, associated with the nodes and vertices of the graph. These pseudomarginals are constrained to be non-negative, as well to normalize and be locally consistent in the following sense:

$$\sum_{x_s \in \mathcal{X}} \mu_s(x_s) \quad = \quad 1, \qquad \text{for all } s \in V, \text{ and} \tag{2}$$

$$\sum_{x_t \in \mathcal{X}} \mu_{st}(x_s, x_t) \quad = \quad \mu_s(x_s) \qquad \text{for all } (s,t) \in E,\ x_s \in \mathcal{X}. \tag{3}$$

The polytope defined by the non-negativity constraints $\mu \geq 0$, the normalization constraints (2) and the marginalization constraints (3), is denoted by $\mathbb{L}(G)$. The LP relaxation is based on maximizing the linear function

$$\langle \theta, \mu \rangle \quad := \quad \sum_{s \in V} \sum_{x_s} \theta_s(x_s)\mu_s(x_s) + \sum_{(s,t) \in E} \sum_{x_s, x_t} \theta_{st}(x_s, x_t)\mu_{st}(x_s, x_t), \tag{4}$$

subject to the constraint $\mu \in \mathbb{L}(G)$.

In the sequel, we write the linear program (4) more compactly in the form $\max_{\mu \in \mathbb{L}(G)} \langle \theta, \mu \rangle$. By construction, this relaxation is guaranteed to be exact for any problem on a tree-structured graph (Wainwright et al., 2005), so that it can be viewed as a tree-based relaxation. The main goal of this paper is to develop efficient and distributed algorithms for solving this LP relaxation,[2] as well as strengthenings based on additional constraints. For instance, one natural strengthening is by "lifting": view the pairwise MRF as a particular case of a more general MRF with higher order cliques, define higher-order pseudomarginals on these cliques, and use them to impose higher-order consistency constraints. This particular progression of tighter relaxations underlies the Bethe to Kikuchi (sum-product to generalized sum-product) hierarchy (Yedidia et al., 2005); see Wainwright and Jordan (2003) for further discussion of such LP hierarchies.

3. Proximal Minimization Schemes

We begin by defining the notion of a proximal minimization scheme, and various types of divergences (among them Bregman) that we use to define our proximal sequences. Instead of dealing with the maximization problem $\max_{\mu \in \mathbb{L}(G)} \langle \theta, \mu \rangle$, it is convenient to consider the equivalent minimization problem,

$$\min_{\mu \in \mathbb{L}(G)} -\langle \theta, \mu \rangle.$$

3.1 Proximal Minimization

The class of methods that we develop are based on the notion of proximal minimization (Bertsekas and Tsitsiklis, 1997). Instead of attempting to solve the LP directly, we solve a sequence of problems of the form

$$\mu^{n+1} = \arg\min_{\mu \in \mathbb{L}(G)} \left\{ -\langle \theta, \mu \rangle + \frac{1}{\omega^n} D_f(\mu \,\|\, \mu^n) \right\}, \tag{5}$$

where for iteration numbers $n = 0, 1, 2, \ldots$, the vector μ^n denotes current iterate, the quantity ω^n is a positive weight, and D_f is a generalized distance function, known as the proximal function. (Note that we are using superscripts to represent the iteration number, *not* for the power operation.)

The purpose of introducing the proximal function is to convert the original LP—which is convex but not strictly so—into a strictly convex problem. The latter property is desirable for a number of reasons. First, for strictly convex programs, coordinate descent schemes are guaranteed to converge to the global optimum; note that they may become trapped for non-strictly convex problems, such as the piecewise linear surfaces that arise in linear programming. Moreover, the dual of a strictly convex problem is guaranteed to be differentiable (Bertsekas, 1995); a guarantee which need not hold

2. The relaxation could fail to be exact though, in which case the optimal solution to the relaxed problem will be suboptimal on the original MAP problem

for non-strictly convex problems. Note that differentiable dual functions can in general be solved more easily than non-differentiable dual functions. In the sequel, we show how for appropriately chosen generalized distances, the proximal sequence $\{\mu^n\}$ can be computed using message passing updates derived from cyclic projections.

We note that the proximal scheme (5) is similar to an annealing scheme, in that it involves perturbing the original cost function, with a choice of weights $\{\omega^n\}$. While the weights $\{\omega^n\}$ can be adjusted for faster convergence, they can also be set to a constant, unlike for standard annealing procedures in which the annealing weight is taken to 0. The reason is that $D_f(\mu \,\|\, \mu^{(n)})$, as a generalized distance, itself converges to zero as the algorithm approaches the optimum, thus providing an "adaptive" annealing. For appropriate choice of weights and proximal functions, these proximal minimization schemes converge to the LP optimum with at least geometric and possibly superlinear rates (Bertsekas and Tsitsiklis, 1997; Iusem and Teboulle, 1995).

In this paper, we focus primarily on proximal functions that are Bregman divergences (Censor and Zenios, 1997), a class that includes various well-known divergences, among them the squared ℓ_2-distance and norm, and the Kullback-Leibler divergence. We say that a function $f : S \mapsto \mathbb{R}$, with domain $S \subseteq \mathbb{R}^p$, is a *Bregman function* if int $S \neq \emptyset$ and it is continuously differentiable and strictly convex on int S. Any such function induces a *Bregman divergence* $D_f : S \times \text{int}S \mapsto \mathbb{R}$ as follows:

$$D_f(\mu' \,\|\, \nu) \quad := \quad f(\mu') - f(\nu) - \langle \nabla f(\nu), \mu' - \nu \rangle. \tag{6}$$

Figure 1 illustrates the geometric interpretation of this definition in terms of the tangent approximation. A Bregman divergence satisfies $D_f(\mu' \,\|\, \nu) \geq 0$ with equality if and only if $\mu' = \nu$, but need not be symmetric or satisfy the triangle inequality, so it is only a generalized distance. Further restrictions on the inducing function f are thus required for the divergence to be "well-behaved," for instance that it satisfy the property that for any sequence $\nu^n \to \nu^*$, where $\nu^n \in \text{int } S$, $\nu^* \in S$, then $D_f(\nu^* \,\|\, \nu^n) \to 0$. Censor and Zenios (1997) impose such technical conditions explicitly in their definition of a Bregman function; in this paper, we impose the stronger yet more easily stated condition that the Bregman function f (as defined above) be of Legendre type (Rockafellar, 1970). In this case, in addition to the Bregman function properties, it satisfies the following property: for any sequence $\mu^n \to \mu^*$ where $\mu^n \in \text{int } S$, $\mu^* \in \partial S$, it holds that $\|\nabla f(\mu^n)\| \to +\infty$. Further, we assume that the range $\nabla f(\text{int } S) = \mathbb{R}^p$.

Let us now look at some choices of divergences, proximal minimizations (5) based on which we will be studying in the sequel.

3.1.1 QUADRATIC DIVERGENCE

This choice is the simplest, and corresponds to setting the inducing Bregman function f in (6) to be the quadratic function

$$q(\mu) \quad := \quad \frac{1}{2}\Big\{ \sum_{s \in V} \sum_{x_s \in \mathcal{X}} \mu_s^2(x_s) + \sum_{(s,t) \in E} \sum_{(x_s, x_t) \in \mathcal{X} \times \mathcal{X}} \mu_{st}^2(x_s, x_t) \Big\},$$

defined over nodes and edges of the graph. The divergence is then simply the quadratic norm across nodes and edges

$$Q(\mu \,\|\, \nu) := \frac{1}{2} \sum_{s \in V} \|\mu_s - \nu_s\|^2 + \frac{1}{2} \sum_{(s,t) \in E} \|\mu_{st} - \nu_{st}\|^2, \tag{7}$$

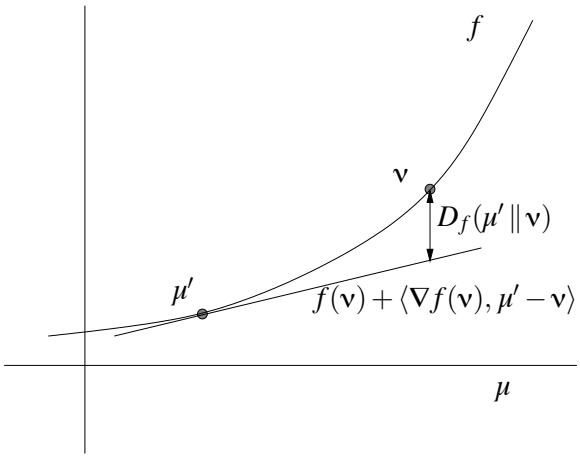


Figure 1: Graphical illustration of a Bregman divergence.

where we have used the shorthand $\|\mu_s - \nu_s\|^2 = \sum_{x_s \in \mathcal{X}} |\mu_s(x_s) - \nu_s(x_s)|^2$, with similar notation for the edges.

3.1.2 WEIGHTED ENTROPIC DIVERGENCE

Another choice for the inducing Bregman function is the weighted sum of negative entropies

$$\bar{H}_\alpha(\mu) = \sum_{s \in V} \alpha_s \bar{H}_s(\mu_s) + \sum_{(s,t) \in E} \alpha_{st} \bar{H}_{st}(\mu_{st}), \tag{8}$$

where $\bar{H}_s$ and $\bar{H}_{st}$ are defined by

$$\begin{aligned} \bar{H}_s(\mu_s) &:= \sum_{x_s \in \mathcal{X}} \mu_s(x_s) \log \mu_s(x_s), \text{ and} \\ \bar{H}_{st}(\mu_{st}) &:= \sum_{(x_s, x_t) \in \mathcal{X} \times \mathcal{X}} \mu_{st}(x_s, x_t) \log \mu_{st}(x_s, x_t), \end{aligned}$$

corresponding to the node-based and edge-based negative entropies, respectively. The corresponding Bregman divergence is a weighted sum of Kullback-Leibler (KL) divergences across the nodes and edges. In particular, letting $\alpha_s > 0$ and $\alpha_{st} > 0$ be positive weights associated with node s and edge (s,t) respectively, we define

$$D_\alpha(\mu \| \nu) \quad = \quad \sum_{s \in V} \alpha_s D(\mu_s \| \nu_s) + \sum_{(s,t) \in E} \alpha_{st} D(\mu_{st} \| \nu_{st}), \tag{9}$$

where $D(p \| q) := \sum_x \left(p(x) \log \frac{p(x)}{q(x)} - \left[p(x) - q(x) \right] \right)$ is the KL divergence. An advantage of the KL divergence, relative to the quadratic norm, is that it automatically acts to enforce non-negativity constraints on the pseudomarginals in the proximal minimization problem. (See Section 3.4 for a more detailed discussion of this issue.)

3.1.3 TREE-REWEIGHTED ENTROPIC DIVERGENCE

Our last example is based on a *tree-reweighted* entropy. The notion of a tree-reweighted entropy was first proposed by Wainwright et al. (2002). Their entropy function however while a Bregman

function is not of the Legendre type. Nonetheless let us first describe their proposed function. Given a set $\mathcal{T}$ of spanning trees $T = (V, E(T))$, and a probability distribution ρ over $\mathcal{T}$, we can obtain edge weights $\rho_{st} \in (0,1]$ for each edge (s,t) of the graph G as $\rho_{st} = \sum_{T \in \mathcal{T}} \mathbb{I}((s,t) \in E)$. Given such edge weights, define

$$f_{\text{trw}}(\mu) \quad := \quad \sum_{s \in V} \bar{H}_s(\mu_s) + \sum_{(s,t) \in E} \rho_{st} I_{st}(\mu_{st}), \tag{10}$$

where $\bar{H}$ is the negative entropy as defined earlier, while the quantity I_{st} defined as

$$I_{st}(\mu_{st}) \quad := \quad \sum_{(x_s, x_t) \in \mathcal{X} \times \mathcal{X}} \mu_{st}(x_s, x_t) \log \frac{\mu_{st}(x_s, x_t)}{[\sum_{x'_t} \mu_{st}(x_s, x'_t)][\sum_{x'_s} \mu_{st}(x'_s, x_t)]},$$

is the mutual information associated with edge (s,t). It can be shown that the function f_{trw} is strictly convex and continuously differentiable when restricted to $\mu \in \mathbb{L}(G)$; and in particular that it is a Bregman function with domain $\mathbb{L}(G)$. Within its domain $\mathbb{L}(G)$, the function can be re-expressed as a weighted negative entropy family (8),

$$f_{\text{trw}}(\mu) \quad = \quad \sum_{s \in V} (1 - \sum_{t:(s,t) \in E} \rho_{st}) \bar{H}_t(\mu_t) + \sum_{(s,t) \in E} \rho_{st} \bar{H}_{st}(\mu_{st}),$$

but where the node entropy weights $\alpha_s := 1 - \sum_{t:(s,t) \in E} \rho_{st}$ are not always positive. The corresponding Bregman divergence belongs to the weighted entropic family (9), with node weights α_s defined above, and edge-weights $\alpha_{st} = \rho_{st}$. However as stated above, this tree-reweighted entropy function is not of Legendre type, and hence is not admissible for our proximal minimization procedure (5).

However, Globerson and Jaakkola (2007a) proposed an alternative tree reweighted entropy that while equal to $f_{\text{trw}}(\mu)$ for $\mu \in \mathbb{L}(G)$ is yet convex for all μ (not just when restricted to $\mathbb{L}(G)$). Their proposed function is described as follows. For each undirected edge in E, construct two oriented edges in both directions; denote the set of these oriented edges by $\bar{E}$. Then given node weights $\rho_{os} \in (0,1]$ for each node $s \in V$, and edge weights $\rho_{s|t} \in (0,1]$ for oriented edges $(t \to s) \in \bar{E}$, define

$$f_{\text{otw}}(\mu) \quad := \quad \sum_{s \in V} \rho_{os} \bar{H}_s(\mu_s) + \sum_{(t \to s) \in \bar{E}} \rho_{s|t} \bar{H}_{s|t}(\mu_{st}), \tag{11}$$

where the quantity $\bar{H}_{s|t}$ defined as

$$\bar{H}_{s|t}(\mu_{st}) \quad := \quad \sum_{(x_s, x_t) \in \mathcal{X} \times \mathcal{X}} \mu_{st}(x_s, x_t) \log \frac{\mu_{st}(x_s, x_t)}{\sum_{x'_s} \mu_{st}(x'_s, x_t)},$$

is the conditional entropy of X_s given X_t with respect to the joint distribution μ_{st}. It can be shown that this oriented tree-reweighted entropy is not only a Bregman function with domain the non-negative orthant $\mathbb{R}^p_+$, but is also of Legendre type, so that it is indeed admissible for our proximal minimization procedure. The corresponding divergence is given as,

$$D_\rho(\mu \,\|\, \nu) \quad = \quad \sum_{s \in V} \rho_{os} D(\mu_s \,\|\, \nu_s) + \sum_{t \to s \in \bar{E}} \rho_{s|t} (D(\mu_{st} \,\|\, \nu_{st}) + \tilde{D}(\mu_{st} \,\|\, \nu_{st})),$$

where $D(p \,\|\, q)$ is the KL divergence, and $\tilde{D}(\cdot \,\|\, \cdot)$ is a KL divergence like term, defined as

$$\begin{aligned} \tilde{D}(\mu_{st} \,\|\, \nu_{st}) := & \sum_{(x_s, x_t) \in \mathcal{X} \times \mathcal{X}} \mu_{st}(x_s, x_t) \log \frac{[\sum_{x'_s} \nu_{st}(x'_s, x_t)]}{[\sum_{x'_s} \mu_{st}(x'_s, x_t)]} \\ & + \quad \frac{\nu_{st}(x_s, x_t)}{[\sum_{x'_s} \nu_{st}(x'_s, x_t)]} [\mu_{st}(x_s, x_t) - \nu_{st}(x_s, x_t)]. \end{aligned}$$

3.2 Proximal Sequences via Bregman Projection

The key in designing an efficient proximal minimization scheme is ensuring that the proximal sequence $\{\mu^n\}$ can be computed efficiently. In this section, we first describe how sequences of proximal minimizations (when the proximal function is a Bregman divergence) can be reformulated as a particular Bregman projection. We then describe how this Bregman projection can itself be computed iteratively, in terms of a sequence of cyclic Bregman projections (Censor and Zenios, 1997) based on a decomposition of the constraint set $\mathbb{L}(G)$. In the sequel, we then show how these cyclic Bregman projections reduce to very simple message-passing updates.

Given a Bregman divergence D, the *Bregman projection* of a vector ν onto a convex set C is given by

$$\widehat{\mu} \quad := \quad \arg\min_{\mu \in C} D_f(\mu \,\|\, \nu). \tag{12}$$

That this minimum is achieved and is unique follows from our assumption that the function f is of Legendre type and from Theorem 3.12 in Bauschke and Borwein (1997), so that the projection is well-defined. We define the projection operator

$$\Pi_C(\nu) := \arg\min_{\mu \in C} D_f(\mu \,\|\, \nu), \tag{13}$$

where we have suppressed the dependence on the Bregman function f in the notation. When the constraint set $C = \cap_{i=1}^{M} C_i$ is an intersection of simpler constraint sets, then a candidate algorithm for the Bregman projection is to compute it in a *cyclic manner*: by iteratively projecting onto the simple constraint sets $\{C_i\}$ (Censor and Zenios, 1997). Define the sequence

$$\mu^{t+1} = \Pi_{C_{i(t)}}(\mu^t),$$

for some control sequence parameter $i : \mathbb{N} \mapsto \{1, \ldots, M\}$ that takes each output value an infinite number of times, for instance $i(t) = t \bmod M$. It can be shown that *when the constraints are affine* then such cyclic Bregman projections μ^t converge to the projection $\widehat{\mu}$ onto the entire constraint set as defined in (12) so that $\mu^t \to \widehat{\mu}$ (Censor and Zenios, 1997). But when a constraint C_i is non-affine, the individual projection would have to be followed by a correction (Dykstra, 1985; Han, 1988; Censor and Zenios, 1997) in order for such convergence to hold. In Appendix A we have outlined these corrections briefly for the case where the constraints are linear inequalities. For ease of notation, we will now subsume these corrections into the iterative projection notation, $\mu^{t+1} = \Pi_{C_{i(t)}}(\mu^t)$, so that the notation assumes that the Bregman projections are suitably corrected when the constraints $C_{i(t)}$ are non-affine. In this paper, other than positivity constraints, we will be concerned only with affine constraints, for which no corrections are required.

Let us now look at the stationary condition characterizing the optimum $\widehat{\mu}$ of (12). As shown in for instance Bertsekas (1995), the optimum $\widehat{\mu}$ of any constrained optimization problem $\min_{\mu \in C} g(\mu)$ is given by the stationary condition,

$$\langle \nabla g(\widehat{\mu}), \mu - \widehat{\mu} \rangle \geq 0, \tag{14}$$

for all $\mu \in C$. For the projection problem (12), the gradient of the objective $D_f(\mu \,\|\, \nu) := f(\mu) - f(\nu) - \langle \nabla f(\nu), \mu - \nu \rangle$ with respect to the first argument μ is given by $\nabla f(\mu) - \nabla f(\nu)$, which when substituted in (14) yields the stationary condition of the optimum $\widehat{\mu}$ as

$$\langle \nabla f(\widehat{\mu}) - \nabla f(\nu), \mu - \widehat{\mu} \rangle \quad \geq \quad 0, \tag{15}$$

for all $\mu \in C$. Now consider the proximal minimization problem to be solved at step n, namely the strictly convex problem

$$\min_{\mu \in \mathbb{L}(G)} \Big\{ -\langle \theta, \mu \rangle + \frac{1}{\omega^n} D_f(\mu \| \mu^n) \Big\}. \tag{16}$$

Solving for the derivative of this objective with respect to μ as $-\theta + \frac{1}{\omega^n}(\nabla f(\mu) - \nabla f(\mu^n))$, and substituting in (14), we obtain the conditions defining the optimum μ^{n+1} as

$$\langle \nabla f(\mu^{n+1}) - \nabla f(\mu^n) - \omega^n \theta, \mu - \mu^{n+1} \rangle \quad \geq \quad 0,$$

for all $\mu \in \mathbb{L}(G)$. Comparing these with the conditions for Bregman projection (15), we see that if there exists a vector ν such that

$$\nabla f(\nu) = \nabla f(\mu^n) + \omega^n \theta, \tag{17}$$

then the proximal iterate μ^{n+1} is the Bregman projection of this vector ν onto the set $\mathbb{L}(G)$. As shown in Bauschke and Borwein (1997), for any function f of Legendre type with domain S, the gradient ∇f is a one-to-one function with domain int S, so that its inverse $(\nabla f)^{-1}$ is a well-defined function on the range $\nabla f(\text{int } S)$ of ∇f. Since we have assumed that this range is $\mathbb{R}^p$, we can thus obtain the unique ν which satisfies the condition in (17) as $\nu = (\nabla f)^{-1}(\nabla f(\mu) + \omega^n \theta)$ (Note that the range constraint could be relaxed to only require that the range of ∇f be a cone containing θ). Accordingly, we set up the following notation: for any Bregman function f, induced divergence D_f, and convex set C, we define the operator

$$\mathrm{J}_f(\mu, \nu) := (\nabla f)^{-1}(\nabla f(\mu) + \nu).$$

We can then write the proximal update (16) in a compact manner as the compounded operation

$$\mu^{n+1} \quad = \quad \Pi_{\mathbb{L}(G)}\Big(\mathrm{J}_f(\mu^n, \omega^n \theta)\Big).$$

Consequently, efficient algorithms for computing the Bregman projection (12) can be leveraged to compute the proximal update (16). In particular, we consider a decomposition of the constraint set as an intersection—$\mathbb{L}(G) = \cap_{k=1}^{M} \mathbb{L}_k(G)$—and then apply the method of cyclic Bregman projections discussed above. Initializing $\mu^{n,0} = \mu^n$ and updating from $\mu^{n,\tau} \mapsto \mu^{n,\tau+1}$ by projecting $\mu^{n,\tau}$ onto constraint set $\mathbb{L}_{i(\tau)}(G)$, where $i(\tau) = \tau \mod M$, for instance, we obtain the meta-algorithm summarized in Algorithm 1.

As shown in the following sections, by using a decomposition of $\mathbb{L}(G)$ over the edges of the graph, the inner loop steps correspond to local message-passing updates, slightly different in nature depending on the choice of Bregman distance. Iterating the inner and outer loops yields a provably convergent message-passing algorithm for the LP. Convergence follows from the convergence properties of proximal minimization (Bertsekas and Tsitsiklis, 1997), combined with convergence guarantees for cyclic Bregman projections (Censor and Zenios, 1997). In the following section, we derive the message-passing updates corresponding to various Bregman functions of interest.

3.3 Quadratic Projections

Consider the proximal sequence with the quadratic distance Q from Equation (7); the Bregman function inducing this distance is the quadratic function $q(y) = \frac{1}{2}y^2$, with gradient $\nabla q(y) = y$. A little calculation shows that the operator J_q takes the form

$$\mathrm{J}_q(\mu, \omega\theta) \quad = \quad \mu + \omega\theta,$$

Algorithm 1 Basic proximal-Bregman LP solver

Given a Bregman distance D, weight sequence $\{\omega^n\}$ and problem parameters θ:

- Initialize μ^0 to the uniform distribution: $\mu_s^{(0)}(x_s) = \frac{1}{m}, \mu_{st}^{(0)}(x_s, x_t) = \frac{1}{m^2}$.
- **Outer Loop:** For iterations $n = 0, 1, 2, \ldots$, update $\mu^{n+1} = \Pi_{\mathbb{L}(G)}\Big(\mathrm{J}_f(\mu^n, \omega^n\theta)\Big)$.
  - Solve Outer Loop via **Inner Loop:**
    (a) Inner initialization $\mu^{n,0} = \mathrm{J}_f(\mu^n, \omega^n\theta)$.
    (b) For $t = 0, 1, 2, \ldots$, set $i(t) = t \mod M$.
    (c) Update $\mu^{n,t+1} = \Pi_{\mathbb{L}_{i(t)}(G)}\big(\mu^{n,t}\big)$.

whence we obtain the initialization in Equation (19).

We now turn to the projections $\mu^{n,\tau+1} = \Pi_q(\mu^{n,\tau}, \mathbb{L}_i(G))$ onto the individual constraints $\mathbb{L}_i(G)$. For each such constraint, the local update is based on the solving the problem

$$\mu^{n,\tau+1} \quad = \quad \arg\min_{\nu \in \mathbb{L}_i(G)} \Big\{ q(\nu) - \langle \nu, \nabla q(\mu^{n,\tau}) \rangle \Big\}. \tag{18}$$

In Appendix B.1, we show how the solution to these inner updates takes the form (20) given in Algorithm 2. The $\{Z_s, Z_{st}\}$ variables correspond to the dual variables used to correct the Bregman projections for positivity (and hence inequality) constraints, as outlined in (33) in Section 3.2.

3.4 Entropic Projections

Consider the proximal sequence with the Kullback-Leibler distance $D(\mu \,\|\, \nu)$ defined in Equation (9). The Bregman function h_α inducing the distance is a sum of negative entropy functions $f(\mu) = \mu \log \mu$, and its gradient is given by $\nabla f(\mu) = \log(\mu) + \vec{1}$. In this case, some calculation shows that the map $\nu = \mathrm{J}_f(\mu, \omega\theta)$ is given by

$$\nu \quad = \quad \mu \exp(\omega\,\theta/\alpha),$$

whence we obtain the initialization Equation (21). In Appendix B.2, we derive the message-passing updates summarized in Algorithm 3.

3.5 Tree-reweighted Entropy Proximal Sequences

In the previous sections, we saw how to solve the proximal sequences following the algorithmic template 1 and using message passing updates derived from cyclic Bregman projections. In this section, we show that for the tree-reweighted entropic divergences (11), in addition to the cyclic Bregman projection recipe of the earlier sections, we can also use tree-reweighted sum-product or related methods (Globerson and Jaakkola, 2007b; Hazan and Shashua, 2008) to compute the proximal sequence.

Algorithm 2 Quadratic Messages for μ^{n+1}

Initialization:

$$\begin{aligned} \mu_{st}^{(n,0)}(x_s,x_t) &= \mu_{st}^{(n)}(x_s,x_t) + w^n\theta_{st}(x_s,x_t), \\ \mu_{s}^{(n,0)}(x_s) &= \mu_{s}^{(n)}(x_s) + w^n\theta_{s}(x_s). \end{aligned} \tag{19}$$

$$\begin{aligned} Z_s(x_s) &= \mu_s^{(n,0)}(x_s), \\ Z_{st}(x_s,x_t) &= \mu_{st}^{(n)}(x_s,x_t). \end{aligned}$$

repeat

for each edge $(s,t) \in E$ **do**

$$\begin{aligned} \mu_{st}^{(n,\tau+1)}(x_s,x_t) &= \mu_{st}^{(n,\tau)}(x_s,x_t) + \left(\frac{1}{m+1}\right)\left(\mu_s^{(n,\tau)}(x_s) - \sum_{x_t}\mu_{st}^{(n,\tau)}(x_s,x_t)\right), \\ \mu_{s}^{(n,\tau+1)}(x_s) &= \mu_{s}^{(n,\tau)}(x_s) + \left(\frac{1}{m+1}\right)\left(-\mu_s^{(n,\tau)}(x_s) + \sum_{x_t}\mu_{st}^{(n,\tau)}(x_s,x_t)\right), \end{aligned} \tag{20}$$

$$\begin{aligned} C_{st}(x_s,x_t) &= \min\{Z_{st}(x_s,x_t), \mu_{st}^{(n,\tau+1)}(x_s,x_t)\}, \\ Z_{st}(x_s,x_t) &= Z_{st}(x_s,x_t) - C_{st}(x_s,x_t), \\ \mu_{st}^{(n,\tau+1)}(x_s,x_t) &= \mu_{st}^{(n,\tau+1)}(x_s,x_t) - C_{st}(x_s,x_t). \end{aligned}$$

end for

for each node $s \in V$ **do**

$$\begin{aligned} \mu_{s}^{(n,\tau+1)}(x_s) &= \mu_{s}^{(n,\tau)}(x_s) + \frac{1}{m}\left(1 - \sum_{x_s}\mu_s^{(n,\tau)}(x_s)\right), \\ C_s(x_s) &= \min\{Z_s(x_s), \mu_s^{(n,\tau+1)}(x_s)\}, \\ Z_s(x_s) &= Z_s(x_s) - C_s(x_s), \\ \mu_s^{(n,\tau+1)}(x_s) &= \mu_s^{(n,\tau+1)}(x_s) - C_s(x_s). \end{aligned}$$

end for

until convergence

Recall the proximal sequence optimization problem (5) written as

$$\begin{aligned} \mu^{n+1} &= \arg\min_{\nu\in\mathbb{L}(G)}\left\{ -\langle\theta,\nu\rangle + \frac{1}{\omega^n}D_f(\nu\,\|\,\mu^n)\right\} \\ &= \arg\min_{\nu\in\mathbb{L}(G)}\left\{ -\langle\theta,\nu\rangle + \frac{1}{\omega^n}(f(\nu) - f(\mu^n) - \langle\nabla f(\mu^n), \nu-\mu^n\rangle)\right\}. \end{aligned} \tag{24}$$

Let us denote $\theta^n := \omega^n\theta + \nabla f(\mu^n)$, and set the Bregman function f to the tree-reweighted entropy f_{trw} defined in (10) (or equivalently the oriented tree-reweighted entropy f_{otw} (11) since both are

Algorithm 3 Entropic Messages for μ^{n+1}

Initialization:

$$\begin{aligned} \mu_{st}^{(n,0)}(x_s,x_t) &= \mu_{st}^{(n)}(x_s,x_t)\exp(\omega^n\theta_{st}(x_s,x_t)/\alpha_{st}), \qquad \text{and} \\ \mu_s^{(n,0)}(x_s) &= \mu_s^{(n)}(x_s)\exp(\omega^n\,\theta_s(x_s)/\alpha_s). \end{aligned} \tag{21}$$

repeat

for each edge $(s,t) \in E$ **do**

$$\begin{aligned} \mu_{st}^{(n,\tau+1)}(x_s,x_t) &= \mu_{st}^{(n,\tau)}(x_s,x_t)\left(\frac{\mu_s^{(n,\tau)}(x_s)}{\sum_{x_t}\mu_{st}^{(n,\tau)}(x_s,x_t)}\right)^{\frac{\alpha_s}{\alpha_s+\alpha_{st}}}, \qquad \text{and} \\ \mu_s^{(n,\tau+1)}(x_s) &= \mu_s^{(n,\tau)}(x_s)^{\frac{\alpha_s}{\alpha_s+\alpha_{st}}}\left(\sum_{x_t}\mu_{st}^{(n,\tau)}(x_s,x_t)\right)^{\frac{\alpha_{st}}{\alpha_s+\alpha_{st}}}. \end{aligned} \tag{22}$$

end for

for each node $s \in V$ **do**

$$\mu_s^{(n,\tau+1)}(x_s) = \frac{\mu_s^{(n,\tau)}(x_s)}{\sum_{x_s}\mu_s^{(n,\tau)}(x_s)}. \tag{23}$$

end for

until convergence

equivalent over the constraint set $\mathbb{L}(G)$). The proximal optimization problem as stated above (24) reduces to,

$$\mu^{n+1} = \arg\min_{\nu\in\mathbb{L}(G)}\{\langle\theta^n,\nu\rangle + f_{\text{trw}}(\nu)\}.$$

But this is precisely the optimization problem solved by the tree-reweighted sum-product (Wainwright and Jordan, 2003), as well as other related methods (Globerson and Jaakkola, 2007b; Hazan and Shashua, 2008), for a graphical model with parameters θ^n.

Computing the gradient of the function f_{trw}, and performing some algebra yields the algorithmic template of Algorithm 4.

3.6 Convergence

We now turn to the convergence of the message-passing algorithms that we have proposed. At a high-level, for any Bregman proximal function, convergence follows from two sets of known results: (a) convergence of proximal algorithms; and (b) convergence of cyclic Bregman projections.

For completeness, we re-state the consequences of these results here. For any positive sequence $\omega^n > 0$, we say that it satisfies the *infinite travel condition* if $\sum_{n=1}^{\infty}(1/\omega^n) = +\infty$. We let $\mu^* \in \mathbb{L}(G)$ denote an optimal solution (not necessarily unique) of the LP, and use $f^* = f(\mu^*) = \langle\theta,\mu^*\rangle$ to denote

Algorithm 4 TRW proximal solver

- For outer iterations $n = 0, 1, 2, \ldots,$

  (a) Update the parameters:

$$\begin{aligned} \theta^n_s(x_s) &= \omega^n \theta_s(x_s) + \log(\mu^n(x_s)) + 1, \\ \theta^n_{st}(x_s, x_t) &= \omega^n \theta_{st}(x_s, x_t) + \rho_{st} \left(\log \frac{\mu^n_{st}(x_s, x_t)}{\sum_{x'_s} \mu^n_{st}(x'_s, x_t) \sum_{x'_t} \mu^n_{st}(x_s, x'_t)} - 1 \right). \end{aligned}$$

  (b) Run a convergent TRW-solver on a graphical model with parameters θ^n, so as to compute

$$\mu^{n+1} = \arg\min_{\nu \in \mathbb{L}(G)} \Big\{ -\langle \theta^n, \nu \rangle + f_{\mathrm{trw}}(\nu) \Big\}.$$

the LP optimal value. We say that the convergence rate is *superlinear* if

$$\lim_{n \to +\infty} \frac{|f(\mu^{n+1}) - f^*|}{|f(\mu^n) - f^*|} = 0,$$

and *linear* if

$$\lim_{n \to +\infty} \frac{|f(\mu^{n+1}) - f^*|}{|f(\mu^n) - f^*|} \leq \gamma,$$

for some $\gamma \in (0,1)$. We say the convergence is *geometric* if there exists some constant $C > 0$ and $\gamma \in (0,1)$ such that for all n,

$$|f(\mu^n) - f^*| \leq C\gamma^n.$$

Proposition 1 (Rate of outer loop convergence) *Consider the sequence of iterates produced by a proximal algorithm* (5) *for LP-solving.*

(a) Using the quadratic proximal function and positive weight sequence $\omega^n \to +\infty$ satisfying infinite travel, the proximal sequence $\{\mu^n\}$ converges superlinearly.

(b) Using the entropic proximal function and positive weight sequence ω^n satisfying infinite travel, the proximal sequence $\{\mu^n\}$ converges:

(i) superlinearly if $\omega^n \to 0$, and

(ii) at least linearly if $1/\omega^n \geq c$ for some constant $c > 0$.

The quadratic case is covered in Bertsekas and Tsitsiklis (1997), whereas the entropic case was analyzed by Tseng and Bertsekas (1993), and Iusem and Teboulle (1995).

Our inner loop message updates use cyclic Bregman projections, for which there is also a substantial literature on convergence. Censor and Zenios (1997) show that with dual feasibility correction, cyclic projections onto general convex sets are convergent. For Euclidean projections with

linear constraints, Deutsch and Hundal (2006) establish a linear rate of convergence, with the rate dependent on angles between the half-spaces defining the constraints. The intuition is that the more orthogonal the half-spaces, the faster the convergence; for instance, a single iteration suffices for completely orthogonal constraints. Our inner updates thus converge linearly to the solution within each outer proximal step.

We note that the rate-of-convergence results for the outer proximal loops assume that the proximal update (computed within each inner loop) has been performed exactly. In practice, the inner loop iterations do not converge finitely (though they have a linear rate of convergence), so that an early stopping entails that the solution to each proximal update would be computed only approximately, up to some accuracy ε. That is, if the proximal optimization function at outer iteration n is $h^n(\mu)$ with minimum μ^{n+1}, then the computed proximal update $\underline{\mu}^{n+1}$ is sub-optimal, with $h^n(\underline{\mu}^{n+1}) - h^n(\mu^{n+1}) \leq \varepsilon$. Some recent theory has addressed whether superlinear convergence can still be obtained in such a setting; for instance, Solodov and Svaiter (2001) shows that that under mild conditions superlinear rates still hold for proximal iterates with inner-loop solutions that are ε-suboptimal. In practice, we cannot directly use ε-suboptimality as the stopping criterion for the inner loop iterations since we do not have the optimal solution μ^{n+1}. However, since we are trying to solve a feasibility problem, it is quite natural to check for violation in the constraints defining $\mathbb{L}(G)$. We terminate our inner iterations when the violation in all the constraints below a tolerance ε. As we show in Section 5, our experiments show that setting this termination threshold to $\underline{\varepsilon} = 10^{-4}$ is small enough for sub-optimality to be practically irrelevant and that superlinear convergence still occurs.

3.6.1 REMARKS

The quadratic proximal updates turn out to be equivalent to solving the primal form of the LP by the projected subgradient method (Bertsekas, 1995) for constrained optimization. (This use of the subgradient method should be contrasted with other work Feldman et al. (2002b); Komodakis et al. (2007) which performed subgradient descent to the dual of the LP.) For any constrained optimization problem:

$$\begin{aligned} \min_{\mu} \quad & f_0(\mu) \\ s.t. \quad & f_j(\mu) \leq 0,\ j = 1,\ldots,m, \end{aligned} \tag{25}$$

the projected subgradient method performs subgradient descent iteratively on (i) the objective function f_0, as well as on (ii) the constraint functions $\{f_j\}_{j=1}^m$ till the constraints are satisfied. Casting it in the notation of Algorithm 1; over outer loop iterations $n = 1,\ldots,$ it sets

$$\mu^{n,0} = \mu^n - \alpha_n \nabla f_0(\mu^n),$$

and computes, over inner loop iterations $t = 1,\ldots,$

$$\begin{aligned} j(t) &= t \mod m, \\ \mu^{n,t+1} &= \mu^{n,t} - \alpha_{n,t} \nabla f_{j(t)}(\mu^{n,t}), \end{aligned}$$

and sets $\mu^{n+1} = \mu^{n,\infty}$, the converged estimate of the inner loops of outer iteration n. The constants $\{\alpha_n, \alpha_{n,t}\}$ are step-sizes for the corresponding subgradient descent steps.

The constraint set in our LP problem, $\mathbb{L}(G)$, has equality constraints so that it is not directly in the form of Equation (25). However any equality constraint $h(\mu) = 0$ can be rewritten equivalently as two inequality constraints $h(\mu) \leq 0$, and $-h(\mu) \leq 0$; so that one could cast our constrained LP in the form of (25) and solve it using the constrained subgradient descent method. As regards the step-sizes, suppose we set $\alpha_n = \omega^n$, and $\alpha_{n,t}$ according to Polyak's step-size (Bertsekas, 1995) so that $\alpha_{n,t} = \frac{f_{j(t)}(\mu^{n,t}) - f_{j(t)}(\mu^*)}{\|\nabla f_{j(t)}(\mu^{n,t})\|_2^2}$, where μ^* is the constrained optimum. Since μ^* is feasible by definition, $f_j(\mu^*) = 0$. Further, for the normalization constraints $C_{ss}(\mu) \gtreqless 1$ where $C_{ss}(\mu) := \sum_{x_s \in \mathcal{X}} \mu_s(x_s) - 1$, we have $\|\nabla C_{ss}(\mu)\|^2 = m$, while for the marginalization constraints $C_{st}(\mu) \gtreqless 0$, where $C_{st}(\mu) := \sum_{x_t \in \mathcal{X}} \mu_{st}(x_s, x_t) = \mu_s(x_s)$, we have $\|\nabla C_{st}(\mu)\|^2 = (m+1)$. It can then be seen that the subgradient method for constrained optimization applied to our constrained LP with the above step-sizes yields the same updates as our quadratic proximal scheme.

4. Rounding Schemes with Optimality Guarantees

The graph-structured LP in (4) was a relaxation of the MAP integer program (1), so that there are two possible outcomes to solving the LP: either an integral vertex is obtained, which is then guaranteed to be a MAP configuration, or a fractional vertex is obtained, in which case the relaxation is loose. In the latter case, a natural strategy is to "round" the fractional solution, so as to obtain an integral solution (Raghavan and Thompson, 1987). Such rounding schemes may either be randomized or deterministic. A natural measure of the quality of the rounded solution is in terms of its value relative to the optimal (MAP) value. There is now a substantial literature on performance guarantees of various rounding schemes, when applied to particular sub-classes of MAP problems (e.g., Raghavan and Thompson, 1987; Kleinberg and Tardos, 1999; Chekuri et al., 2005).

In this section, we show that rounding schemes can be useful even when the LP optimum is integral, since they may permit an LP-solving algorithm to be *finitely terminated*—that is, before it has actually solved the LP—while retaining the same optimality guarantees about the final output. An attractive feature of our proximal Bregman procedures is the existence of precisely such rounding schemes—namely, that under certain conditions, rounding pseudomarginals at intermediate iterations yields the integral LP optimum. We describe these rounding schemes in the following sections, and provide two kinds of results. We provide certificates under which the rounded solution is guaranteed to be MAP optimal; moreover, we provide upper bounds on the number of outer iterations required for the rounding scheme to obtain the LP optimum.

In the next Section 4.1, we describe and analyze deterministic rounding schemes that are specifically tailored to the proximal Bregman procedures that we have described. Then in the following Section 4.2, we propose and analyze a graph-structured randomized rounding scheme, which applies not only to our proximal Bregman procedures, but more broadly to any algorithm that generates a sequence of iterates contained within the local polytope $\mathbb{L}(G)$.

4.1 Deterministic Rounding Schemes

We begin by describing three deterministic rounding schemes that exploit the particular structure of the Bregman proximal updates.

4.1.1 NODE-BASED ROUNDING

This method is the simplest of the deterministic rounding procedures, and applies to the quadratic and entropic updates. It operates as follows: given the vector μ^n of pseudomarginals at iteration n, obtain an integral configuration $x^n(\mu^n) \in \mathcal{X}^N$ by choosing

$$x_s^n \quad \in \quad \arg\max_{x_s' \in \mathcal{X}} \mu^n(x_s'), \qquad \text{for each } s \in V.$$

We say that the node-rounded solution x^n is *edgewise-consistent* if

$$(x_s^n, x_t^n) \quad \in \quad \arg\max_{(x_s', x_t') \in \mathcal{X} \times \mathcal{X}} \mu_{st}^n(x_s', x_t') \qquad \text{for all edges } (s,t) \in E. \tag{26}$$

4.1.2 NEIGHBORHOOD-BASED ROUNDING

This rounding scheme applies to all three proximal schemes. For each node $s \in V$, denote its star-shaped neighborhood graph by $N_s = \{(s,t) | t \in N(s)\}$, consisting of edges between node s and its neighbors. Let $\{\text{QUA}, \text{ENT}, \text{TRW}\}$ refer to the quadratic, entropic, and tree-reweighted schemes respectively.

(a) Define the neighborhood-based energy function

$$F_s(x; \mu^n) \quad := \quad \begin{cases} 2\mu^n(x_s) + \sum_{t \in N(s)} \mu^n(x_s, x_t) & \text{for QUA} \\ 2\alpha_s \log \mu_s^n(x_s) + \sum_{t \in N(s)} \alpha_{st} \log \mu_{st}^n(x_s, x_t) & \text{for ENT} \\ 2 \log \mu^n(x_s) + \sum_{t \in N(s)} \rho_{st} \log \frac{\mu_{st}^n(x_s, x_t)}{\mu_s^n(x_s) \mu_t^n(x_t)} & \text{for TRW.} \end{cases} \tag{27}$$

(b) Compute a configuration $x^n(N_s)$ maximizing the function $F_s(x; \mu^n)$ by running two rounds of ordinary max-product on the star graph.

Say that such a rounding is *neighborhood-consistent* if the neighborhood MAP solutions $\{x^n(N_s), s \in V\}$ agree on their overlaps.

4.1.3 TREE-BASED ROUNDING

This method applies to all three proximal schemes, but most naturally to the TRW proximal method. Let $T_1, \ldots, T_K$ be a set of spanning trees that cover the graph (meaning that each edge appears in at least one tree), and let $\{\rho(T_i), i = 1, \ldots, K\}$ be a probability distribution over the trees. For each edge (s,t), define the *edge appearance probability* $\rho_{st} = \sum_{i=1}^K \rho(T_i)\, \mathbb{I}[(s,t) \in T_i]$. Then for each tree $i = 1, \ldots, K$:

(a) Define the tree-structured energy function

$$F_i(x; \mu^n) \quad := \quad \begin{cases} \sum_{s \in V} \log \mu^n(x_s) + \sum_{(s,t) \in E(T_i)} \frac{1}{\rho_{st}} \log \mu_{st}^n(x_s, x_t) & \text{for QUA} \\ \sum_{s \in V} \alpha_s \log \mu^n(x_s) + \sum_{(s,t) \in E(T_i)} \frac{\alpha_{st}}{\rho_{st}} \log \mu_{st}^n(x_s, x_t) & \text{for ENT} \\ \sum_{s \in V} \log \mu^n(x_s) + \sum_{(s,t) \in E(T_i)} \log \frac{\mu_{st}^n(x_s, x_t)}{\mu_s^n(x_s) \mu_t^n(x_t)} & \text{for TRW.} \end{cases} \tag{28}$$

(b) Run the ordinary max-product problem on energy $F_i(x;\mu^n)$ to find a MAP-optimal configuration $x^n(T_i)$.

Say that such a rounding is *tree-consistent* if the tree MAP solutions $\{x^n(T_i), i = 1,\ldots,M\}$ are all equal. This notion of tree-consistency is similar to the underlying motivation of the tree-reweighted max-product algorithm (Wainwright et al., 2005).

4.1.4 OPTIMALITY CERTIFICATES FOR DETERMINISTIC ROUNDING

The following result characterizes the optimality guarantees associated with these rounding schemes, when they are consistent respectively in the *edge-consistency*, *neighborhood-consistency* and *tree-consistency* senses defined earlier.

Theorem 2 (Deterministic rounding with MAP certificate) *Consider a sequence of iterates $\{\mu^n\}$ generated by the quadratic or entropic proximal schemes. For any $n = 1,2,3,\ldots$, any consistent rounded solution x^n obtained from μ^n via any of the node, neighborhood or tree-rounding schemes (when applicable) is guaranteed to be a MAP-optimal solution. For the iterates of TRW proximal scheme, the guarantee holds for both neighborhood and tree-rounding methods.*

We prove this claim in Section 4.1.6. It is important to note that such deterministic rounding guarantees do *not* apply to an arbitrary algorithm for solving the linear program. At a high-level, there are two key properties required to ensure guarantees in the rounding. First, the algorithm must maintain some representation of the cost function that (up to possible constant offsets) is equal to the cost function of the original problem, so that the set of maximizers of the invariance would be equivalent to the set of maximizers of the original cost function, and hence the MAP problem. Second, given a rounding scheme that maximizes tractable sub-parts of the reparameterized cost function, the rounding is said to be admissible if these partial solutions agree with one another. Our deterministic rounding schemes and optimality guarantees follow this approach, as we detail in the proof of Theorem 2.

We note that the invariances maintained by the proximal updates in this paper are closely related to the reparameterization condition satisfied by the sum-product and max-product algorithms (Wainwright et al., 2003). Indeed, each sum-product (or max-product) update can be shown to compute a new set of parameters for the Markov random field that preserves the probability distribution. A similar but slightly different notion of reparameterization underlies the tree-reweighted sum-product and max-product algorithms (Wainwright et al., 2005); for these algorithms, the invariance is preserved in terms of convex combinations over tree-structured graphs. The tree-reweighted max-product algorithm attempts to produce MAP optimality certificates that are based on verifying consistency of MAP solutions on certain tree-structured components whose convex combination is equal to the LP cost. The sequential TRW-S max-product algorithm of Kolmogorov (2006) is a version of tree-reweighted max-product using a clever scheduling of the messages to guarantee monotonic changes in a dual LP cost function. Finally, the elegant work of Weiss et al. (2007) exploits similar reparameterization arguments to derive conditions under which their convex free-energy based sum-product algorithms yield the optimal MAP solution.

An attractive feature of all the rounding schemes that we consider is their relatively low computational cost. The node-rounding scheme is trivial to implement. The neighborhood-based scheme requires running two iterations of max-product for each neighborhood of the graph. Finally, the tree-rounding scheme requires $O(KN)$ iterations of max-product, where K is the number of trees

that cover the graph, and N is the number of nodes. Many graphs with cycles can be covered with a small number K of trees; for instance, the lattice graph in 2-dimensions can be covered with two spanning trees, in which case the rounding cost is linear in the number of nodes.

4.1.5 BOUNDS ON ITERATIONS FOR DETERMINISTIC ROUNDING

Of course, the natural question is how many iterations are sufficient for a a given rounding scheme to succeed. The following result provides a way of deriving such upper bounds:

Corollary 3 *Suppose that the LP optimum is uniquely attained at an integral vertex μ^*, and consider algorithms generating sequence $\{\mu^n\}$ converging to μ^*. Then we have the following guarantees:*

(a) for quadratic and entropic schemes, all three types of rounding recover the MAP solution once $\|\mu^n - \mu\|_\infty \leq 1/2$.

(b) for the TRW-based proximal method, tree-based rounding recovers the MAP solution once $\|\mu^n - \mu\|_\infty \leq \frac{1}{4N}$.

Proof We first claim that if the ℓ_∞-bound $\|\mu^n - \mu^*\|_\infty < \frac{1}{2}$ is satisfied, then the node-based rounding returns the (unique) MAP configuration, and moreover this MAP configuration x^* is edge-consistent with respect to μ^n. To see these facts, note that the ℓ_∞ bound implies, in particular, that at every node $s \in V$, we have

$$|\mu_s^n(x_s^*) - \mu_s^*(x_s^*)| \;=\; |\mu_s^n(x_s^*) - 1| \;<\; \frac{1}{2},$$

which implies that $\mu_s^n(x_s^*) > 1/2$ as $\mu_s^*(x_s^*) = 1$. Due to the non-negativity constraints and marginalization constraint $\sum_{x_s \in \mathcal{X}} \mu^n(x_s) = 1$, at most one configuration can have mass above $1/2$. Thus, node-based rounding returns x_s^* at each node s, and hence overall, it returns the MAP configuration x^*. The same argument also shows that the inequality $\mu_{st}^n(x_s^*, x_t^*) > \frac{1}{2}$ holds, which implies that $(x_s^*, x_t^*) = \arg\max_{x_s, x_t} \mu^n(x_s, x_t)$ for all $(s,t) \in E$. Thus, we have shown x^* is edge-consistent for μ_{st}^n, according to the definition (26).

Next we turn to the performance of neighborhood and tree-rounding for the quadratic and entropic updates. For $n \geq n^*$, we know that x^* achieves the unique maximum of $\mu_s^n(x_s)$ at each node, and $\mu_{st}^n(x_s, x_t)$ on each edge. From the form of the neighborhood and tree energies (27),(28), this node- and edge-wise optimality implies that $x^*(N(s)) := \{x_t^*, t \in s \cup N(s)\}$ maximizes the neighborhood-based and tree-based cost functions as well, which implies success of neighborhood and tree-rounding. (Note that the positivity of the weights α_s and α_{st} is required to make this assertion.)

For the TRW algorithm in part (b), we note that when $\|\mu^n - \mu\|_\infty \leq 1/(4N)$, then we must have $\mu_s^n(x_s^*) \geq 1 - 1/(4N)$ for every node. We conclude that these inequalities imply that $x^* = (x_1^*, \ldots, x_N^*)$ must be the unique MAP on every tree. Indeed, consider the set $S = \{x \in \mathcal{X}^N \mid x \neq x^*\}$. By union bound, we have

$$\begin{aligned}
\mathbb{P}(S) &= \mathbb{P}[\exists s \in V \mid x_s \neq x_s^*] \\
&\leq \sum_{s=1}^N \mathbb{P}(x_s \neq x_s^*) \\
&= \sum_{s=1}^N (1 - \mu_s(x_s^*)) \leq \frac{1}{4},
\end{aligned}$$

showing that we have $\mathbb{P}(x^*) \geq 3/4$, so that x^* must be the MAP configuration.

To conclude the proof, note that the tree-rounding scheme computes the MAP configuration on each tree T_i, under a distribution with marginals μ_s and μ_{st}. Consequently, under the stated conditions, the configuration x^* must be the unique MAP configuration on each tree, so that tree rounding is guaranteed to find it. ■

Using this result, it is possible to bound the number of iterations required to achieve the ℓ_∞-bounds. In particular, suppose that the algorithm has a linear rate of convergence—say that $|f(\mu^n) - f(\mu^*)| \leq |f(\mu^0) - f(\mu^*)|\gamma^n$ for some $\gamma \in (0,1)$. For the quadratic or entropic methods, it suffices to show that $\|\mu^n - \mu^*\|_2 < 1/2$. For the entropic method, there exists some constant $C > 0$ such that $\|\mu^n - \mu^*\|_2 \leq \frac{1}{2C}|f(\mu^n) - f(\mu^*)|$ (cf. Prop. 8, Iusem and Teboulle, 1995). Consequently, we have

$$\|\mu^n - \mu^*\|_2 \quad \leq \quad \frac{|f(\mu^0) - f(\mu^*)|}{2C}\gamma^n.$$

Consequently, after $n^* := \frac{\log C|f(\mu^0)-f(\mu^*)|}{\log(1/\gamma)}$ iterations, the rounding scheme would be guaranteed to configuration for the entropic proximal method. Similar finite iteration bounds can also be obtained for the other proximal methods, showing finite convergence through use of our rounding schemes.

Note that we proved correctness of the neighborhood and tree-based rounding schemes by leveraging the correctness of the node-based rounding scheme. In practice, it is possible for neighborhood- or tree-based rounding to succeed even if node-based rounding fails; however, we currently do not have any sharper sufficient conditions for these rounding schemes.

4.1.6 PROOF OF THEOREM 2

We now turn to the proof of Theorem 2. At a high level, the proof consists of two main steps. First, we show that each proximal algorithm maintains a certain invariant of the original MAP cost function $F(x;\theta)$; in particular, the iterate μ^n induces a reparameterization $F(x;\mu^n)$ of the cost function such that the set of maximizers is preserved—viz.:

$$\arg\max_{x\in\mathcal{X}^N} F(x;\theta) := \arg\max_{x\in\mathcal{X}^N} \sum_{s\in V, x_s\in\mathcal{X}} \theta_s(x_s) + \sum_{(s,t)\in E, x_s, x_t\in\mathcal{X}} \theta_{st}(x_s,x_t) = \arg\max_{x\in\mathcal{X}^N} F(x;\mu^n). \tag{29}$$

Second, we show that the consistency conditions (edge, neighborhood or tree, respectively) guarantee that the rounded solution belongs to $\arg\max_{x\in\mathcal{X}^N} F(x;\mu^n)$

We begin with a lemma on the invariance property:

Lemma 4 (Invariance of maximizers) *Define the function*

$$F(x;\mu) \quad := \quad \begin{cases} \sum_{s\in V} \mu_s(x_s) + \sum_{(s,t)\in E} \mu_{st}(x_s,x_t) & \text{for QUA} \\ \sum_{s\in V} \alpha_s \log\mu_s(x_s) + \sum_{(s,t)\in E} \alpha_{st}\log\mu_{st}(x_s,x_t) & \text{for ENT} \\ \sum_{s\in V} \log\mu_s(x_s) + \sum_{(s,t)\in E} \rho_{st}\log\frac{\mu_{st}(x_s,x_t)}{\mu_s(x_s)\mu_t(x_t)} & \text{for TRW.} \end{cases} \tag{30}$$

At each iteration $n = 1,2,3,\ldots$ for which $\mu^n > 0$, the function $F(x;\mu^n)$ preserves the set of maximizers (29).

The proof of this claim, provided in Appendix C, is based on exploiting the necessary (Lagrangian) conditions defined by the optimization problems characterizing the sequence of iterations $\{\mu^n\}$.

For the second part of the proof, we show how a solution x^*, obtained by a rounding procedure, is guaranteed to maximize the function $F(x;\mu^n)$, and hence (by Lemma 4) the original cost function $F(x;\theta)$. In particular, we state the following simple lemma:

Lemma 5 *The rounding procedures have the following guarantees:*

(a) Any edge-consistent configuration from node rounding maximizes $F(x;\mu^n)$ for the quadratic and entropic schemes.

(b) Any neighborhood-consistent configuration from neighborhood rounding maximizes $F(x;\mu^n)$ for the quadratic and entropic schemes.

(c) Any tree-consistent configuration from tree rounding maximizes $F(x;\mu^n)$ for all three schemes.

Proof We begin by proving statement (a). Consider an edge-consistent integral configuration x^* obtained from node rounding. By definition, it maximizes $\mu^n(x_s)$ for all $s \in V$, and $\mu^n_{st}(x_s,x_t)$ for all $(s,t) \in E$, and so by inspection, also maximizes $F(x;\mu^n)$ for the quadratic and proximal cases.

We next prove statement (b) on neighborhood rounding. Suppose that neighborhood rounding outputs a single neighborhood-consistent integral configuration x^*. Since $x^*_{N(s)}$ maximizes the neighborhood energy (27) at each node $s \in V$, it must also maximize the sum $\sum_{s\in V} F_s(x;\mu^n)$. A little calculation shows that this sum is equal to $2F(x;\mu^n)$, the factor of two arising since the term on edge (s,t) arises twice, one for neighborhood rooted at s, and once for t.

Turning to claim (c), let x^* be a tree-consistent configuration obtained from tree rounding. Then for each $i = 1,\ldots,K$, the configuration x^* maximizes the tree-structured function $F_i(x;\mu^n)$, and hence also maximizes the convex combination $\sum_{i=1}^K \rho(T_i)F_i(x;\mu^n)$. By definition of the edge appearance probabilities ρ_{st}, this convex combination is equal to the function $F(x;\mu^n)$. ■

4.2 Randomized Rounding Schemes

The schemes considered in the previous section were all deterministic, since (disregarding any possible ties), the output of the rounding procedure was a deterministic function of the given pseudomarginals $\{\mu^n_s,\mu^n_{st}\}$. In this section, we consider randomized rounding procedures, in which the output is a random variable.

Perhaps the most naive randomized rounding scheme is the following: for each node $r \in V$, assign it value $x_r \in \mathcal{X}$ with probability $\mu^n_v(x_r)$. We propose a graph-structured generalization of this naive randomized rounding scheme, in which we perform the rounding in a dependent way across sub-groups of nodes, and establish guarantees for its success. In particular, we show that when the LP relaxation has a unique integral optimum that is well-separated from the second best configuration, then the rounding scheme succeeds with high probability after a pre-specified number of iterations.

4.2.1 THE RANDOMIZED ROUNDING SCHEME

Our randomized rounding scheme is based on any given subset E' of the edge set E. Consider the subgraph $G(E\backslash E')$, with vertex set V, and edge set $E\backslash E'$. We assume that E' is chosen such that

the subgraph $G(E\backslash E')$ is a forest. That is, we can decompose $G(E\backslash E')$ into a union of disjoint trees, $\{T_1,\ldots,T_K\}$, where $T_i = (V_i,E_i)$, such that the vertex subsets V_i are all disjoint and $V = V_1 \cup V_2 \cup \ldots \cup V_K$. We refer to the edge subset as *forest-inducing* when it has this property. Note that such a subset always exists, since $E' = E$ is trivially forest-inducing. In this case, the "trees" simply correspond to individual nodes, without any edges; $V_i = \{i\}$, $E_i = \emptyset$, $i = 1,\ldots,N$.

For any forest-inducing subset $E' \subseteq E$, Algorithm 5 defines our randomized rounding scheme.

Algorithm 5 RANDOMIZED ROUNDING SCHEME

for subtree indices $i = 1,\ldots,K$ **do**

Sample a sub-configuration X_{V_i} from the probability distribution

$$p(x_{V_i};\mu(T_i)) \quad = \quad \prod_{s\in V_i} \mu^n(x_s) \prod_{(s,t)\in E_i} \frac{\mu^n(x_s,x_t)}{\mu^n(x_s)\mu^n(x_t)}.$$

end for

Form the global configuration $X \in \mathcal{X}^N$ by concatenating all the local random samples:

$$X \quad := \quad \Big(X_{V_1},\ldots,X_{V_K}\Big).$$

To be clear, the randomized solution X is a function of both the pseudomarginals μ^n, and the choice of forest-inducing subset E', so that we occasionally use the notation $X(\mu^n;E')$ to reflect explicitly this dependence. Note that the simplest rounding scheme of this type is obtained by setting $E' = E$. Then the "trees" simply correspond to individual nodes without any edges, and the rounding scheme is the trivial node-based scheme.

The randomized rounding scheme can be "derandomized" so that we obtain a deterministic solution $x^d(\mu^n;E')$ that does at least well as the randomized scheme does in expectation. This derandomization scheme is shown in Algorithm 6, and its correctness is guaranteed in the following theorem, proved in Appendix D.

Theorem 6 *Let $(G = (V,E),\theta)$ be the given MAP problem instance, and let $\mu^n \in \mathbb{L}(G)$ be any set of pseudomarginals in the local polytope $\mathbb{L}(G)$. Then, for any subset $E' \subseteq E$ of the graph G, the (E',μ^n)-randomized rounding scheme in Algorithm 5, when derandomized as in Algorithm 6 satisfies,*

$$F(x^d(\mu^n;E');\theta) \geq \mathbb{E}\Big(F(X(\mu^n;E');\theta)\Big),$$

where $X(\mu^n;E')$ and $x^d(\mu^n;E')$ denote the outputs of the randomized and derandomized schemes respectively.

4.2.2 OSCILLATION AND GAPS

In order to state some theoretical guarantees on our randomized rounding schemes, we require some notation. For any edge $(s,t) \in E$, we define the *edge-based oscillation*

$$\delta_{st}(\theta) \quad := \quad \max_{x_s,x_t}[\theta_{st}(x_s,x_t)] - \min_{x_s,x_t}[\theta_{st}(x_s,x_t)].$$

Algorithm 6 DERANDOMIZED ROUNDING SCHEME

Initialize: $\bar{\mu} = \mu^n$.

for subtree indices $i = 1, \ldots, K$ **do**

Solve

$$x^d_{V_i} = \arg\max_{x_{V_i}} \sum_{s \in V_i} \left\{ \theta_s(x_s) + \sum_{t:(s,t)\in E'} \sum_{x_t} \bar{\mu}_t(x_t)\theta_{st}(x_s,x_t) \right\} + \sum_{(s,t)\in E_i} \theta_{st}(x_s,x_t).$$

Update $\bar{\mu}$:

$$\bar{\mu}_s(x_s) = \begin{cases} \bar{\mu}_s(x_s) & \text{if } s \notin V_i \\ 0 & \text{if } s \in V_i, x^d_s \neq x_s \\ 1 & \text{if } s \in V_i, x^d_s = x_s. \end{cases}$$

$$\bar{\mu}_{st}(x_s,x_t) = \begin{cases} \bar{\mu}_{st}(x_s,x_t) & \text{if } (s,t) \notin E_i \\ \bar{\mu}_s(x_s)\bar{\mu}_t(x_t) & \text{if } (s,t) \in E_i. \end{cases}$$

end for

Form the global configuration $x^d \in \mathcal{X}^N$ by concatenating all the subtree configurations:

$$x^d := \left(x^d_{V_1}, \ldots, x^d_{V_K} \right).$$

We define the *node-based oscillation* $\delta_s(\theta)$ in the analogous manner. The quantities $\delta_s(\theta)$ and $\delta_{st}(\theta)$ are measures of the strength of the potential functions.

We extend these measures of interaction strength to the full graph in the natural way

$$\delta_G(\theta) := \max\left\{ \max_{(s,t)\in E} \delta_{st}(\theta), \max_{s\in V} \delta_s(\theta) \right\}.$$

Using this oscillation function, we now define a measure of the quality of a unique MAP optimum, based on its separation from the second most probable configuration. In particular, letting $x^* \in \mathcal{X}^N$ denote a MAP configuration, and recalling the notation $F(x;\theta)$ for the LP objective, we define the *graph-based gap*

$$\Delta(\theta;G) := \frac{\min_{x\neq x^*} \left[F(x^*;\theta) - F(x;\theta) \right]}{\delta_G(\theta)}.$$

This gap function is a measure of how well-separated the MAP optimum x^* is from the remaining integral configurations. By definition, the gap $\Delta(\theta;G)$ is always non-negative, and it is strictly positive whenever the MAP configuration x^* is unique. Finally, note that the gap is invariant to the translations ($\theta \mapsto \theta' = \theta + C$) and rescalings ($\theta \mapsto \theta' = c\theta$) of the parameter vector θ. These invariances are appropriate for the MAP problem since the optima of the energy function $F(x;\theta)$ are not affected by either transformation (i.e., $\arg\max_x F(x;\theta) = \arg\max_x F(x;\theta')$ for both $\theta' = \theta + C$ and $\theta' = c\theta$).

Finally, for any forest-inducing subset, we let $d(E')$ be the maximum degree of any node with respect to edges in E'—namely,

$$d(E') \quad := \quad \max_{s \in V} |t \in V \ \mid \ (s,t) \in E'|.$$

4.2.3 OPTIMALITY GUARANTEES FOR RANDOMIZED ROUNDING

We show, in this section, that when the pseudomarginals μ^n are within a specified ℓ_1 norm ball around the unique MAP optimum μ^*, the randomized rounding scheme outputs the MAP configuration with high probability.

Theorem 7 *Consider a problem instance (G, θ) for which the MAP optimum x^* is unique, and let μ^* be the associated vertex of the polytope $\mathbb{L}(G)$. For any $\varepsilon \in (0,1)$, if at some iteration n, we have $\mu^n \in \mathbb{L}(G)$, and*

$$\|\mu^n - \mu^*\|_1 \ \leq \ \frac{\varepsilon \, \Delta(\theta; G)}{1 + d(E')}, \tag{31}$$

then (E', μ^n)-randomized rounding succeeds with probability greater than $1 - \varepsilon$,

$$\mathbb{P}[X(\mu^n; E') = x^*] \quad \geq \quad 1 - \varepsilon.$$

We provide the proof of this claim in Appendix E. It is worthwhile observing that the theorem applies to any algorithm that generates a sequence $\{\mu^n\}$ of iterates contained within the local polytope $\mathbb{L}(G)$. In addition to the proximal Bregman updates discussed in this paper, it also applies to interior-point methods (Boyd and Vandenberghe, 2004) for solving LPs. For the naive rounding based on $E' = E$, the sequence $\{\mu^n\}$ need not belong to $\mathbb{L}(G)$, but instead need only satisfy the milder conditions $\mu^n_s(x_s) \geq 0$ for all $s \in V$ and $x_s \in \mathcal{X}$, and $\sum_{x_s} \mu^n_s(x_s) = 1$ for all $s \in V$.

The derandomized rounding scheme enjoys a similar guarantee, as shown in the following theorem, proved in Appendix F.

Theorem 8 *Consider a problem instance (G, θ) for which the MAP optimum x^* is unique, and let μ^* be the associated vertex of the polytope $\mathbb{L}(G)$. If at some iteration n, we have $\mu^n \in \mathbb{L}(G)$, and*

$$\|\mu^n - \mu^*\|_1 \ \leq \ \frac{\Delta(\theta; G)}{1 + d(E')},$$

then the (E', μ^n)-derandomized rounding scheme in Algorithm 6 outputs the MAP solution,

$$x^d(\mu^n; E') = x^*.$$

4.2.4 BOUNDS ON ITERATIONS FOR RANDOMIZED ROUNDING

Although Theorems 7 and 8 apply even for sequences $\{\mu^n\}$ that need not converge to μ^*, it is most interesting when the LP relaxation is tight, so that the sequence $\{\mu^n\}$ generated by any LP-solver satisfies the condition $\mu^n \to \mu^*$. In this case, we are guaranteed that for any fixed $\varepsilon \in (0,1)$, the bound (31) will hold for an iteration number n that is "large enough". Of course, making this intuition precise requires control of convergence rates. Recall that N is the number of nodes in the graph, and m is cardinality of the set $\mathcal{X}$ from which all variables takes their values. With this notation, we have the following.

Corollary 9 *Under the conditions of Theorem 7, suppose that the sequence of iterates $\{\mu^n\}$ converge to the LP (and MAP) optimum at a linear rate: $\|\mu^n - \mu^*\|_2 \leq \gamma^n \|\mu^0 - \mu^*\|_2$. Then:*

(a) The randomized rounding in Algorithm 5 succeeds with probability at least $1-\varepsilon$ for all iterations greater than

$$n^* \quad := \quad \frac{\frac{1}{2}\log\left(Nm + N^2m^2\right) + \log\left(\|\mu^0 - \mu^*\|_2\right) + \log\left(\frac{1+d(E')}{\Delta(\theta;G)}\right) + \log(1/\varepsilon)}{\log(1/\gamma)}.$$

(b) The derandomized rounding in Algorithm 6 yields the MAP solution for all iterations greater than

$$n^* \quad := \quad \frac{\frac{1}{2}\log\left(Nm + N^2m^2\right) + \log\left(\|\mu^0 - \mu^*\|_2\right) + \log\left(\frac{1+d(E')}{\Delta(\theta;G)}\right)}{\log(1/\gamma)}.$$

This corollary follows by observing that the vector $(\mu^n - \mu^*)$ has less than $Nm + N^2m^2$ elements, so that $\|\mu^n - \mu^*\|_1 \leq \sqrt{Nm + N^2m^2}\,\|\mu^n - \mu^*\|_2$. Moreover, Theorems 7 and 8 provide an ℓ_1-ball radius such that the rounding schemes succeed (either with probability greater than $1-\varepsilon$, or deterministically) for all pseudomarginal vectors within these balls.

5. Experiments

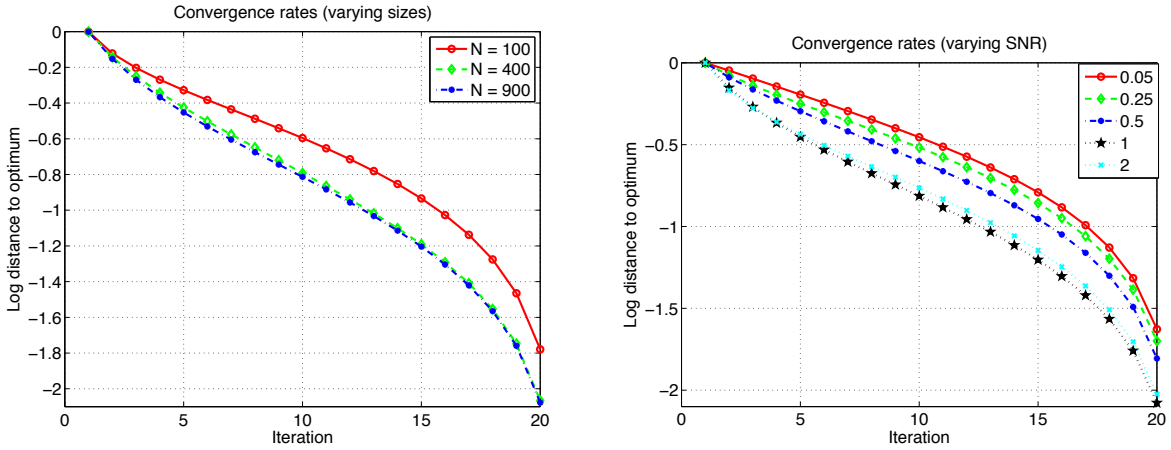
In this section, we provide the results of several experiments to illustrate the behavior of our methods on different problems. We performed experiments on 4-nearest neighbor grid graphs with sizes varying from $N = 100$ to $N = 900$, using models with either $m = 3$ or $m = 5$ labels. The edge potentials were set to Potts functions, of the form

$$\theta_{st}(x_s, x_t) = \begin{cases} \beta_{st} & \text{if } x_s = x_t \\ 0 & \text{otherwise.} \end{cases}$$

for a parameter $\beta_{st} \in \mathbb{R}$. These potential functions penalize disagreement of labels if $\beta_{st} > 0$, and penalize agreement if $\beta_{st} < 0$. The Potts weights on edges β_{st} were chosen randomly as $\text{Uniform}(-1,+1)$. We set the node potentials as $\theta_s(x_s) \sim \text{Uniform}(-\text{SNR}, \text{SNR})$, for some signal-to-noise parameter $\text{SNR} \geq 0$ that controls the ratio of node to edge strengths. In applying all of the proximal procedures, we set the proximal weights as $\omega^n = n$.

5.1 Rates of Convergence

We begin by reporting some results on the convergence rates of proximal updates. Figure 2(a) plots the logarithmic distance $\log\|\mu^n - \mu^*\|_2$ versus the number of iterations for grids of different sizes (node numbers $N \in \{100, 400, 900\}$). Here μ^n is the iterate at step n entropic proximal method and μ^* is the LP optimum. In all cases, note how the curves have an inverted quadratic shape, corresponding to a superlinear rate of convergence, which is consistent with Proposition 1. On other hand, Figure 2(b) provides plots of the logarithmic distance versus iteration number for problem sizes $N = 900$, and over a range of signal-to-noise ratios SNR (in particular, $\text{SNR} \in \{0.05, 0.25, 0.50, 1.0, 2.0\}$). Notice how the plots still show the same inverted quadratic shape, but that the rate of convergence slows down as the SNR decreases, as is to be expected.



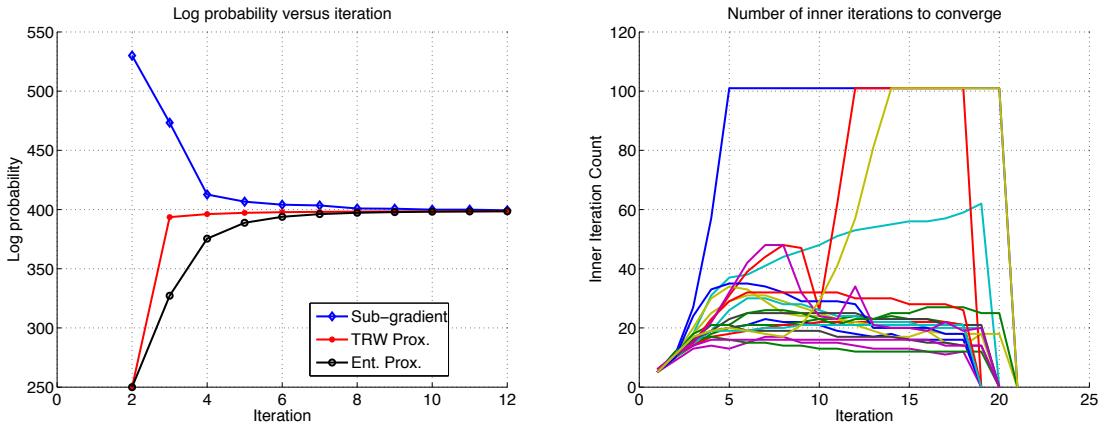
(a) (b)

Figure 2: (a) Plot of distance $\log_{10} \|\mu^n - \mu^*\|_2$ between the current entropic proximal iterate μ^n and the LP optimum μ^* versus iteration number for Potts models on grids with $N \in \{100, 400, 900\}$ vertices, $m = 5$ labels and SNR $= 1$. Note the superlinear rate of convergence, consistent with Proposition 1. (b) Plot of distance $\log_{10} \|\mu^n - \mu^*\|_2$ between the current entropic proximal iterate μ^n and the LP optimum μ^* versus iteration number for Potts models on grids with $m = 5$ labels, $N = 900$ vertices, and a range of signal-to-noise ratios SNR $\in \{0.05, 0.25, 0.50, 1.0, 2.0\}$. The rate of convergence remains superlinear but slows down as the SNR is decreased.

In Figure 3, we compare two of our proximal schemes—the entropic and the quadratic schemes—with a subgradient descent method, as previously proposed (Feldman et al., 2002a; Komodakis et al., 2007). For the comparison, we used a Potts model on a grid of 400 nodes, with each node taking three labels. The Potts weights were set as earlier, with SNR $= 2$. Plotted in Figure 3(a) are the log probabilities of the solutions from the TRW-proximal and entropic proximal methods, compared to the dual upper bound that is provided by the sub-gradient method. Each step on the horizontal axis is a single outer iteration for the proximal methods, and five steps of the subgradient method. (We note that it is slower to perform five subgradient steps than a single proximal outer iteration.) Both the primal proximal methods and the dual subgradient method converge to the same point. The TRW-based proximal scheme converges the fastest, essentially within four outer iterations, whereas the entropic scheme requires a few more iterations. The convergence rate of the subgradient ascent method is slower than both of these proximal schemes, even though we allowed it to take more steps per "iteration". In Figure 3(b), we plot a number of traces showing the number of inner iterations (vertical axis) required as a function of outer iteration (horizontal axis). The average number of inner iterations is around 20, and only rarely does the algorithm require substantially more.

5.2 Comparison of Rounding Schemes

In Figure 4, we compare five of our rounding schemes on a Potts model on grid graphs with $N = 400$, $m = 3$ labels and SNR $= 2$. For the graph-structured randomized rounding schemes, we used the node-based rounding scheme (so that $E \backslash E' = \emptyset$), and the chain-based rounding scheme (so that



(a) (b)

Figure 3: (a) Plots of the function value (for fractional iterates μ^n) versus number of iterations for a Potts model with $N = 400$ vertices, $m = 3$ labels and SNR $= 2$. Three methods are compared: a subgradient method (Feldman et al., 2002b; Komodakis et al., 2007), the entropic proximal method (Ent. Prox.), and the TRW-based proximal method (TRW Prox.). (b) Traces of different algorithm runs showing the number of inner iterations (vertical axis) versus the outer iteration number (horizontal axis). Typically around 20 inner iterations are required.

$E \backslash E'$ is the set of horizontal chains in the grid). For the deterministic rounding schemes, we used the node-based, neighborhood-based and the tree-based rounding schemes. Panel (a) of Figure 4 shows rounding schemes as applied to the entropic proximal algorithm, whereas panel (b) shows rounding schemes applied to the TRW proximal scheme. In both plots, the tree-based and star-based deterministic schemes are the first to return an optimal solution, whereas the node-based randomized scheme is the slowest in both plots. Of course, this type of ordering is to be expected, since the tree and star-based schemes look over larger neighborhoods of the graph, but incur larger computational cost.

6. Discussion

In this paper, we have developed distributed algorithms, based on the notion of proximal sequences, for solving graph-structured linear programming (LP) relaxations. Our methods respect the graph structure, and so can be scaled to large problems, and they exhibit a superlinear rate of convergence. We have also developed a series of graph-structured rounding schemes that can be used to generate integral solutions along with a certificate of optimality. These optimality certificates allow the algorithm to be terminated in a finite number of iterations.

The structure of our algorithms naturally lends itself to incorporating additional constraints, both linear and other types of conic constraints. It would be interesting to develop an adaptive version of our algorithm, which selectively incorporated new constraints as necessary, and then used the same proximal schemes to minimize the new conic program. Our algorithms for solving the LP are primal-based, so that the updates are in terms of the pseudo-marginals μ that are the primal parameters of the LP. This is contrast to typical message-passing algorithms such as tree-reweighted

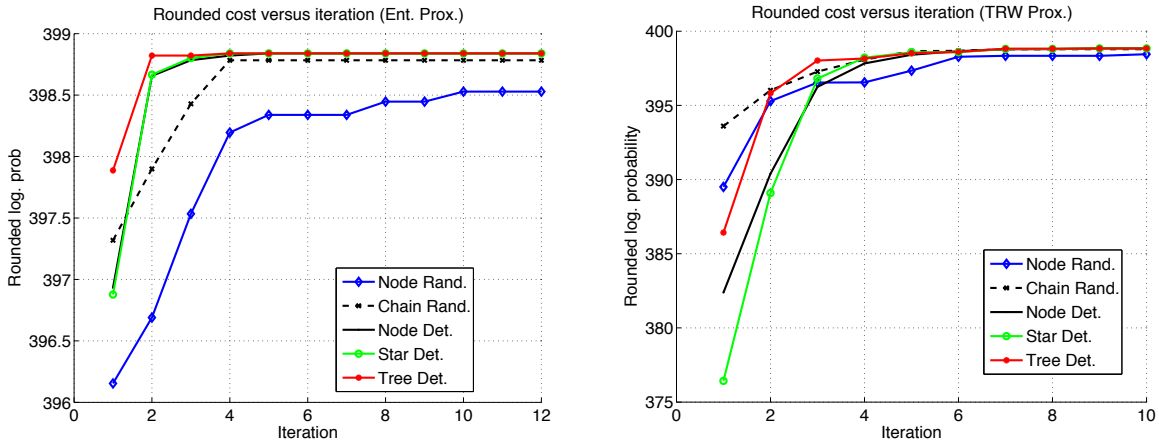


Figure 4: Plots of the log probability of rounded solutions versus the number of iterations for the entropic proximal scheme (panel (a)), and the TRW proximal scheme (panel (b)). In both cases, five different rounding schemes are compared: node-based randomized rounding (Node Rand.), chain-based randomized rounding (Chain Rand.), node-based deterministic rounding (Node. Det.), star-based deterministic rounding (Star Det.), and tree-based deterministic rounding (Tree Det.).

max-product, which are dual-based and where the updates are entirely in terms of *message* parameters that are the dual parameters of the LP. However, the dual of the LP is non-differentiable, so that these dual-based updates could either get trapped in local minima (dual co-ordinate ascent) or have sub-linear convergence rates (dual sub-gradient ascent). On the one hand, our primal-based algorithm converges to the LP minimum, and has at least linear convergence rates. On the other, it is more memory-intensive because of the need to maintain $O(|E|)$ edge pseudo-marginal parameters. It would be interesting to modify our algorithms so that maintaining these explicitly could be avoided; note that our derandomized rounding scheme (Algorithm 4.2.1) does not make use of the edge pseudo-marginal parameters.

Acknowledgments

This work was partially supported by NSF grants CCF-0545862 and DMS-0528488 to MJW. AA is partially supported by NSF award DMS-0830410, DARPA award HR0011-08-2-0002 and MSR PhD Fellowship.

Appendix A. Corrections to Bregman Projections

We briefly outline the corrections needed to cyclic Bregman projections for the case where the constraints are linear inequalities. It is useful, in order to characterize these needed corrections, to first note that these cyclic projections are equivalent to co-ordinate ascent steps on the dual of the Bregman projection problem (13). Let the linear constraint set for the Bregman projection

problem (13) be $C \equiv \cap_i \{\langle a_i, \mu \rangle \leq b_i\}$. Its Lagrangian can be written as

$$\mathcal{L}(\mu, z) = D_f(\mu \,\|\, \nu) + \sum_i z_i (\langle a_i, \mu \rangle - b_i),$$

where $z \geq 0$ are the Lagrangian or dual parameters. The dual function is given as $g(z) = \min_\mu \mathcal{L}(\mu, z)$, so that the dual problem can be written as

$$\min_{z \geq 0} \; g(z).$$

If the constraints were linear *equalities*, the dual variables $\{z\}$ would be unconstrained, and iterative co-ordinate ascent—which can be verified to be equivalent to cyclic projections of the primal variables onto individual constraints—would suffice to solve the dual problem. However, when the constraints have inequalities, the dual problem is no longer unconstrained: the dual variables are constrained to be positive. We would thus need to constrain the co-ordinate ascent steps. This can also be understood as the following primal-dual algorithmic scheme. Note that a necessary KKT condition for optimality of a primal-dual pair (μ, z) for (13) is

$$\nabla f(\mu) = \nabla f(\nu) - \sum_i z_i a_i. \tag{32}$$

The primal-dual algorithmic scheme then consists of maintaining primal-dual iterates (μ^t, z^t) which satisfy the equality (32), are dual feasible with $z^t \geq 0$, and which entail co-ordinate ascent on the dual problem, so that $g(z^{t+1}) \geq g(z^t)$ with at most one co-ordinate of μ^t updated in μ^{t+1}. We can now write down the corrected-projection update of μ^t given the single constraint $C_i \equiv \{\langle a_i, \mu \rangle \leq b_i\}$. According to the primal-dual algorithmic scheme this corresponds to co-ordinate ascent on the i-th co-ordinate of z^t so that (32) is maintained, whereby

$$\begin{aligned} \nabla f(\mu^{t+1}) &= \nabla f(\mu^t) + C\, a_i, \\ z^{t+1} &= z^t - C\, e_i, \\ C &:= \min\{z_i^t, \beta\}, \end{aligned} \tag{33}$$

where e_i is the co-ordinate vector with one in the i-th co-ordinate and zero elsewhere, and β is the i-th dual parameter setting corresponding to an unconstrained co-ordinate ascent update,

$$\begin{aligned} \nabla f(\mu) &= \nabla f(\mu^n) + \beta\, a_i, \\ \langle \mu, a_i \rangle &= b_i. \end{aligned} \tag{34}$$

One could derive such corrections corresponding to constrained dual ascent for general convex constraints (Dykstra, 1985; Han, 1988).

Appendix B. Detailed Derivation of Message-passing Updates

In this appendix, we provided detailed derivation of the message-passing updates for the inner loops of the algorithms.

B.1 Derivation of Algorithm 2

Consider the edge marginalization constraint for edge (s,t), $\mathbb{L}_i(G) \equiv \sum_{x_t} \mu_{st}(x_s,x_t) = \mu_s(x_s)$. Denoting the dual (Lagrange) parameter corresponding to the constraint by $\lambda_{st}(x_s)$, the Karush-Kuhn-Tucker conditions for the quadratic update (18) are given by

$$\begin{aligned}
\nabla q(\mu_{st}^{n,\tau+1}(x_s,x_t)) &= \nabla q(\mu_{st}^{n,\tau}(x_s,x_t)) + \lambda_{st}(x_s), \\
\nabla q(\mu_{s}^{n,\tau+1}(x_s)) &= \nabla q(\mu_{s}^{n,\tau}(x_s)) - \lambda_{st}(x_s), \\
\mu_{st}^{n,\tau+1}(x_s,x_t) &= \mu_{st}^{n,\tau}(x_s,x_t) + \lambda_{st}(x_s), \\
\mu_{s}^{n,\tau+1}(x_s) &= \mu_{s}^{n,\tau}(x_s) - \lambda_{st}(x_s),
\end{aligned}$$

while the constraint itself gives

$$\sum_{x_t} \mu_{st}^{n,\tau+1}(x_s,x_t) = \mu_s^{n,\tau}(x_s).$$

Solving for $\lambda_{st}(x_s)$ yields Equation (20). The node marginalization follows similarly.

The only inequalities are the positivity constraints, requiring that the node and edge pseudo-marginals be non-negative. Following the correction procedure for Bregman projections in (33), we maintain Lagrange dual variables corresponding to these constraints. We use $Z_s(x_s)$ as the Lagrange variables for the node positivity constraints $\mu_s(x_s) \geq 0$, and $Z_{st}(x_s,x_t)$ for the edge-positivity constraints $\mu_{st}(x_s,x_t) \geq 0$.

Consider the projection of $\{\mu^{n,\tau+1}\}$ onto the constraint $\mu_s(x_s) \geq 0$. Following (34), we first solve for $\beta_s(x_s)$ that satisfies

$$\begin{aligned}
\mu_s(x_s) &= \mu_s^{n,\tau+1}(x_s) - \beta_s(x_s), \\
\mu_s(x_s) &= 0,
\end{aligned}$$

so that $\beta_s(x_s) = \mu_s^{n,\tau+1}(x_s)$. Substituting in (33), we obtain the update

$$\begin{aligned}
C_s(x_s) &= \min\{Z_s(x_s), \mu_s^{(n,\tau+1)}(x_s)\}, \\
Z_s(x_s) &= Z_s(x_s) - C_s(x_s), \\
\mu_s^{(n,\tau+1)}(x_s) &= \mu_s^{(n,\tau+1)}(x_s) - C_s(x_s).
\end{aligned}$$

The edge positivity constraint updates follow similarly.

Thus overall, we obtain message-passing Algorithm 2 for the inner loop.

B.2 Derivation of Algorithm 3

Note that we do not need to explicitly impose positivity constraints in this case. Because the domain of the entropic Bregman function is the positive orthant, if we start from a positive point, any further Bregman projections would also result in a point in the positive orthant.

The projection $\mu^{n,\tau+1} = \Pi_h(\mu^{n,\tau}, \mathbb{L}_i(G))$ onto the individual constraint $\mathbb{L}_i(G)$ is defined by the optimization problem:

$$\mu^{n,\tau+1} = \min_{\mathbb{L}_i(G)} \{h(\mu) - \mu^\top \nabla h(\mu^{n,\tau})\}.$$

Consider the subset $\mathbb{L}_i(G)$ defined by the marginalization constraint along edge (s,t), namely $\sum_{x_t' \in \mathcal{X}} \mu_{st}(x_s, x_t') = \mu_s(x_s)$ for each $x_s \in \mathcal{X}$. Denoting the dual (Lagrange) parameters corresponding to these constraint by $\lambda_{st}(x_s)$, the KKT conditions are given by

$$\begin{aligned} \nabla h(\mu_{st}^{n,\tau+1}(x_s,x_t)) &= \nabla h(\mu_{st}^{n,\tau}(x_s,x_t)) + \lambda_{st}(x_s), \quad \text{and} \\ \nabla h(\mu_s^{n,\tau+1}(x_s)) &= \nabla h(\mu_s^{n,\tau}(x_s)) - \lambda_{st}(x_s). \end{aligned}$$

Computing the gradient ∇h and performing some algebra yields the relations

$$\begin{aligned} \mu_{st}^{(n,\tau+1)}(x_s,x_t) &= \mu_{st}^{(n,\tau)}(x_s,x_t)\exp(\lambda_{st}^{(n,\tau+1)}(x_s)), \\ \mu_s^{(n,\tau+1)}(x_s) &= \mu_s^{(n,\tau)}(x_s)\exp(-\lambda_{st}^{(n,\tau+1)}(x_s)), \quad \text{and} \\ \exp(2\lambda_{st}^{(n,\tau+1)}(x_s)) &= \frac{\mu_s^{(n,\tau)}(x_s)}{\sum_{x_t}\mu_{st}^{(n,\tau)}(x_s,x_t)}, \end{aligned}$$

from which the updates (22) follow.

Similarly, for the constraint set defined by the node marginalization constraint $\sum_{x_s \in \mathcal{X}} \mu_s(x_s) = 1$, we have $\nabla h(\mu_s^{(n,\tau+1)}(x_s)) = \nabla h(\mu_s^{(n,\tau)}(x_s)) + \lambda_s^{(n,\tau+1)}$, from which

$$\begin{aligned} \mu_s^{(n,\tau+1)}(x_s) &= \mu_s^{(n,\tau)}(x_s)\exp(\lambda_s^{(n,\tau+1)}), \qquad \text{and} \\ \exp(\lambda_s^{(n,\tau+1)}) &= 1/\sum_{x_s \in \mathcal{X}} \mu_s^{(n,\tau)}(x_s). \end{aligned}$$

The updates in Equation (23) follow.

Appendix C. Proof of Lemma 4

We provide a detailed proof for the entropic scheme; the arguments for other proximal algorithms are analogous. The key point is the following: regardless of how the proximal updates are computed, they must satisfy the necessary Lagrangian conditions for optimal points over the set $\mathbb{L}(G)$. Accordingly, we define the following sets of Lagrange multipliers:

$$\begin{aligned} \lambda_{ss} &\quad \text{for the normalization constraint } C_{ss}(\mu_s) = \textstyle\sum_{x_s'} \mu_s(x_s') - 1 = 0, \\ \lambda_{st}(x_s) &\quad \text{for the marginalization constraint } C_{ts}(x_s) = \textstyle\sum_{x_t'} \mu_{st}(x_s, x_t') - \mu_s(x_s) = 0, \\ \gamma_{st}(x_s,x_t) &\quad \text{for the non-negativity constraint } \mu_{st}(x_s,x_t) \geq 0. \end{aligned}$$

(There is no need to enforce the non-negativity constraint $\mu_s(x_s) \geq 0$ directly, since it is implied by the non-negativity of the joint pseudo-marginals and the marginalization constraints.)

With this notation, consider the Lagrangian associated with the entropic proximal update at step n:

$$L(x;\lambda,\gamma) = C(\mu;\theta,\mu^n) + \langle \gamma, \mu \rangle + \sum_{s \in V} \lambda_{ss} C_{ss}(x_s) + \sum_{(s,t) \in E} \left[\lambda_{ts}(x_s)C_{ts}(x_s) + \lambda_{st}(x_t)C_{st}(x_t)\right],$$

where $C(\mu;\theta,\mu^n)$ is shorthand for the cost component $-\langle \theta, \mu \rangle + \frac{1}{\omega^n} D_\alpha(\mu \,\|\, \mu^n)$. Using C, C' to denote constants (whose value can change from line to line), we now take derivatives to find the necessary

Lagrangian conditions:

$$\begin{aligned}
\frac{\partial L}{\partial \mu_s(x_s)} &= -\theta_s(x_s) + \frac{2\alpha_s}{\omega^n}\log\frac{\mu_s(x_s)}{\mu_s^n(x_s)} + C + \lambda_{ss} + \sum_{t\in N(s)} \lambda_{ts}(x_s), \qquad \text{and} \\
\frac{\partial L}{\partial \mu_{st}(x_s,x_t)} &= -\theta_{st}(x_s,x_t) + \frac{2\alpha_{st}}{\omega^n}\log\frac{\mu_{st}(x_s,x_t)}{\mu_{st}^n(x_s,x_t)} + C' + \gamma_{st}(x_s,x_t) - \lambda_{ts}(x_s) - \lambda_{st}(x_t).
\end{aligned}$$

Solving for the optimum $\mu = \mu^{n+1}$ yields

$$\begin{aligned}
\frac{2\alpha_s}{\omega^n}\log\mu_s^{n+1}(x_s) &= \theta_s(x_s) + \frac{2\alpha_s}{\omega^n}\log\mu_s^n(x_s) - \sum_{t\in N(s)} \lambda_{ts}(x_s) + C, \\
\frac{2\alpha_{st}}{\omega^n}\log\mu_{st}^{n+1}(x_s,x_t) &= \theta_{st}(x_s,x_t) + \frac{2\alpha_{st}}{\omega^n}\log\mu_{st}^n(x_s,x_t) - \gamma_{st}(x_s,x_t) \\
&\qquad + \lambda_{ts}(x_s) + \lambda_{st}(x_t) + C'.
\end{aligned}$$

From these conditions, we can compute the energy invariant (30):

$$\begin{aligned}
\frac{2}{\omega^n}F(x;\mu^{n+1}) &= \sum_{s\in V}\frac{2\alpha_s}{\omega^n}\log\mu_s^{n+1}(x_s) + \sum_{(s,t)\in E}\frac{2\alpha_{st}}{\omega^n}\log\mu_{st}^{n+1}(x_s,x_t) + C \\
&= F(x;\theta) + \frac{2}{\omega^n}\left\{\sum_{s\in V}\alpha_s\log\mu^n(x_s) + \sum_{(s,t)\in E}\alpha_{st}\log\mu_{st}^n(x_s,x_t)\right\} \\
&\qquad - \sum_{(s,t)\in E}\gamma_{st}(x_s,x_t) + C \\
&= F(x;\theta) + \frac{2}{\omega^n}F(x;\mu^n) - \sum_{(s,t)\in E}\gamma_{st}(x_s,x_t) + C.
\end{aligned}$$

Now since $\mu^n > 0$, by complementary slackness, we must have $\gamma_{st}(x_s,x_t) = 0$, which implies that

$$\frac{2}{\omega^n}F(x;\mu^{n+1}) = F(x;\theta) + \frac{2}{\omega^n}F(x;\mu^n) + C. \tag{35}$$

From this equation, it is a simple induction to show for some constants $\gamma_n > 0$ and $C_n \in \mathbb{R}$, we have $F(x;\mu^n) = \gamma_n F(x;\theta) + C_n$ for all iterations $n = 1,2,3,\ldots$, which implies preservation of the maximizers. If at iteration $n = 0$, we initialize $\mu^0 = 0$ to the all-uniform distribution, then we have $\frac{2}{\omega^1}F(x;\mu^1) = F(x;\theta) + C'$, so the statement follows for $n = 1$. Suppose that it holds at step n; then $\frac{2}{\omega^n}F(x;\mu^n) = \frac{2}{\omega^n}\gamma_n F(x;\theta) + \frac{2C_n}{\omega_n}$, and hence from the induction step (35), we have $F(x;\mu^{n+1}) = \gamma_{n+1}F(x;\theta) + C_{n+1}$, where $\gamma_{n+1} = \frac{\omega^n}{2}\gamma_n$.

Appendix D. Proof of Theorem 6

Consider the expected cost of the configuration $X(\mu^n;E')$ obtained from the randomized rounding procedure of Algorithm 5. A simple computation shows that

$$\mathbb{E}[F(X(\mu^n;E');\theta)] = G(\bar{\mu}) := \sum_{i=1}^{K} H(\mu^n;T_i) + H(\mu^n;E'),$$

where

$$\begin{aligned} H(\mu^n;T_i) &:= \sum_{s\in V_i}\sum_{x_s}\mu_s^n(x_s)\theta_s(x_s)+\sum_{(s,t)\in E_i}\sum_{x_s,x_t}\mu_{st}^n(x_s,x_t)\theta_{st}(x_s,x_t), \\ H(\mu^n;E') &:= \sum_{(u,v)\in E'}\sum_{x_s,x_t}\mu_u^n(x_u)\mu_v^n(x_v)\theta_{st}(x_u,x_v). \end{aligned} \tag{36}$$

We now show by induction that the de-randomized rounding scheme achieves cost at least as large as this expected value. Let $\bar{\mu}^{(i)}$ denote the updated pseudomarginals at the end of the i-th iteration. Since we initialize with $\bar{\mu}^{(0)}=\mu^n$, we have $G(\bar{\mu}^{(0)})=\mathbb{E}[F(X(\mu^n;E');\theta)]$. Consider the i-th step of the algorithm; the algorithm computes the portion of the de-randomized solution $x_{V_i}^d$ over the $i-$th tree. It will be convenient to use the decomposition $G=G_i+G_{\setminus i}$, where

$$\begin{aligned} G_i(\bar{\mu}) := \sum_{s\in V_i}\sum_{x_s}\bar{\mu}_s(x_s)\left\{\theta_s(x_s)+\sum_{\{t\,|\,(s,t)\in E'\}}\sum_{x_t}\bar{\mu}_t(x_t)\theta_{st}(x_s,x_t)\right\}+ \\ \sum_{(s,t)\in E_i}\sum_{x_s,x_t}\bar{\mu}_{st}(x_s,x_t)\,\theta_{st}(x_s,x_t), \end{aligned}$$

and $G_{\setminus i}=G-G_i$. If we define

$$\bar{F}_i(x_{V_i}) := \sum_{s\in V_i}\left\{\theta_s(x_s)+\sum_{t:(s,t)\in E'}\sum_{x_t}\bar{\mu}_t^{(i-1)}(x_t)\theta_{st}(x_s,x_t)\right\}+\sum_{(s,t)\in E_i}\theta_{st}(x_s,x_t),$$

it can be seen that $G_i(\bar{\mu}^{(i-1)})=\mathbb{E}[\bar{F}_i(x_{V_i})]$ where the expectation is under the tree-structured distribution over X_{V_i} given by

$$p(x_{V_i};\bar{\mu}^{(i-1)}(T_i)) = \prod_{s\in V_i}\bar{\mu}^{(i-1)}(x_s)\prod_{(s,t)\in E_i}\frac{\bar{\mu}^{(i-1)}(x_s,x_t)}{\bar{\mu}^{(i-1)}(x_s)\bar{\mu}^{(i-1)}(x_t)}.$$

Thus when the algorithm makes the choice $x_{V_i}^d=\arg\max_{x_{V_i}}\bar{F}_i(x_{V_i})$, it holds that

$$G_i(\bar{\mu}^{(i-1)}) \;=\; \mathbb{E}[\bar{F}_i(x_{V_i})] \;\leq\; \bar{F}_i(x_{V_i}^d).$$

The updated pseudomarginals $\bar{\mu}^{(i)}$ at the end the i-th step of the algorithm are given by,

$$\bar{\mu}_s^{(i)}(x_s) = \left\{\begin{array}{cc} \bar{\mu}_s^{(i-1)}(x_s) & \text{if } s\notin V_i \\ 0 & \text{if } s\in V_i, X_{d,s}\neq x_s \\ 1 & \text{if } s\in V_i, X_{d,s}=x_s. \end{array}\right\},$$

$$\bar{\mu}_{st}^{(i)}(x_s,x_t) = \left\{\begin{array}{ll} \bar{\mu}_{st}^{(i-1)}(x_s,x_t) & \text{if } (s,t)\notin E_i \\ \bar{\mu}_s^{(i)}(x_s)\bar{\mu}_t^{(i)}(x_t) & \text{if } (s,t)\in E_i. \end{array}\right.$$

In other words, $\bar{\mu}^{(i)}(T_i)$ is the indicator vector of the maximum energy sub-configuration $x_{V_i}^d$. Consequently, we have

$$G_i(\bar{\mu}^{(i)}) \;=\; \bar{F}_i(x_{V_i}^d) \;\geq\; G_i(\bar{\mu}^{(i-1)}),$$

and $G_{\setminus i}(\bar{\mu}^{(i)})=G_{\setminus i}(\bar{\mu}^{(i-1)})$, so that at the end of the i-th step, $G(\bar{\mu}^{(i)})\geq G(\bar{\mu}^{(i-1)})$. By induction, we conclude that $G(\bar{\mu}^{(K)})\geq G(\bar{\mu}^{(0)})$, where K is the total number of trees in the rounding scheme.

At the end of K steps, the quantity $\bar{\mu}^{(K)}$ is the indicator vector for $x^d(\mu^n;E')$ so that $G(\bar{\mu}^{(K)})=F(X_d(\mu^n;E');\theta)$. We have also shown that $G(\bar{\mu}^{(0)})=\mathbb{E}[F(X(\mu^n;E');\theta)]$. Combining these pieces, we conclude that $F(x^d(\mu^n;E');\theta)\geq\mathbb{E}[F(X(\mu^n;E');\theta)]$, thereby completing the proof.

Appendix E. Proof of Theorem 7

Let $p_{\text{succ}} = \mathbb{P}[X(\mu^n; E') = x^*]$, and let $R(\mu^n; E')$ denote the (random) integral vertex of $\mathbb{L}(G)$ that is specified by the random integral solution $X(\mu^n; E')$. (Since E' is some fixed forest-inducing subset, we frequently shorten this notation to $R(\mu^n)$.) We begin by computing the expected cost of the random solution, where the expectation is taken over the rounding procedure. A simple computation shows that $\mathbb{E}[\langle \theta, R(\mu^n)\rangle] := \sum_{i=1}^K H(\mu^n; T_i) + H(\mu^n; E')$, where $H(\mu^n; T_i)$ and $H(\mu^n; E')$ were defined previously (36).

We now upper bound the difference $\langle \theta, \mu^*\rangle - \mathbb{E}[\langle \theta, R(\mu^n)\rangle]$. For each subtree $i = 1, \ldots, K$, the quantity $D_i := H(\mu^*; T_i) - H(\mu^n; T_i)$ is upper bounded as

$$\begin{aligned} D_i &= \sum_{s \in V_i} \sum_{x_s} \Big[\mu_s^*(x_s) - \mu_s^n(x_s)\Big] \theta_s(x_s) + \sum_{(s,t) \in E_i} \sum_{x_s, x_t} \Big[\mu_s^*(x_s)\mu_t^*(x_t) - \mu_{st}^n(x_s, x_t)\Big] \theta_{st}(x_s, x_t) \\ &\leq \sum_{s \in V_i} \delta_s(\theta) \sum_{x_s} |\mu_s^*(x_s) - \mu_s^n(x_s)| + \sum_{(s,t) \in E_i} \delta_{st}(\theta) \sum_{x_s, x_t} |\mu_{st}^*(x_s, x_t) - \mu^n(x_s, x_t)|. \end{aligned}$$

In asserting this inequality, we have used the fact that that the matrix with entries given by $\mu_s^*(x_s)\mu_t^*(x_t) - \mu_{st}^n(x_s, x_t)$ is a difference of probability distributions, meaning that all its entries are between -1 and 1, and their sum is zero.

Similarly, we can upper bound the difference $D(E') = H(\mu^*; E') - H(\mu^n; E')$ associated with E':

$$\begin{aligned} D(E') &= \sum_{(u,v) \in E'} \sum_{x_u, x_v} \Big[\mu_u^*(x_u)\mu_v^*(x_v) - \mu_u^n(x_u)\mu_v^n(x_v)\Big] \theta_{uv}(x_u, x_v) \\ &\leq \sum_{(u,v) \in E'} \delta_{uv}(\theta) \sum_{x_u, x_v} \Big|\mu_u^*(x_u)\mu_v^*(x_v) - \mu_u^n(x_u)\mu_v^n(x_v)\Big| \\ &\leq \sum_{(u,v) \in E'} \delta_{uv}(\theta) \sum_{x_u, x_v} \Big\{ \Big|\mu_u^*(x_u)[\mu_v^*(x_v) - \mu_v^n(x_v)]\Big| + \Big|\mu_v^n(x_v)[\mu_u^*(x_u) - \mu_u^n(x_u)]\Big| \Big\} \\ &\leq \sum_{(u,v) \in E'} \delta_{uv}(\theta) \Big\{ \sum_{x_u} |\mu_u^n(x_u) - \mu_u^*(x_u)| + \sum_{x_u} |\mu_v^n(x_v) - \mu_v^*(x_v)| \Big\}. \end{aligned}$$

Combining the pieces, we obtain

$$\begin{aligned} \langle \theta, \mu^*\rangle - \mathbb{E}[\langle \theta, R(\mu^n)\rangle] &\leq \delta_G(\theta) \Big\{ \|\mu^n - \mu^*\|_1 + \sum_{s \in V} d(s; E') \sum_{x_s} |\mu_s^n(x_s) - \mu_s^*(x_s)| \Big\} \\ &\leq (1 + d(E'))\delta_G(\theta)\|\mu^n - \mu^*\|_1. \end{aligned} \tag{37}$$

In the other direction, we note that when the rounding fails, then we have

$$\langle \theta, \mu^*\rangle - \langle \theta, R(\mu^n)\rangle \geq \max_{x \neq x^*} [F(x^*; \theta) - F(x; \theta)].$$

Consequently, conditioning on whether the rounding succeeds or fails, we have

$$\begin{aligned} \langle \theta, \mu^*\rangle - \mathbb{E}[\langle \theta, R(\mu^n)\rangle] &\geq p_{\text{succ}} \big[\langle \theta, \mu^*\rangle - \langle \theta, \mu^*\rangle\big] + (1 - p_{\text{succ}}) \max_{x \neq x^*} [F(x^*; \theta) - F(x; \theta)] \\ &= (1 - p_{\text{succ}}) \max_{x \neq x^*} [F(x^*; \theta) - F(x; \theta)]. \end{aligned}$$

Combining this lower bound with the upper bound (37), performing some algebra, and using the definition of the gap $\Delta(\theta; G)$ yields that the probability of successful rounding is at least

$$p_{\text{succ}} \quad \geq \quad 1 - \frac{(1 + d(E'))}{\Delta(\theta; G)} \|\mu^n - \mu^*\|_1.$$

If the condition (31) holds, then this probability is at least $1 - \varepsilon$, as claimed.

Appendix F. Proof of Theorem 8

The proof follows that of Theorem 7 until Equation (37), which gives

$$\langle \theta, \mu^* \rangle - \mathbb{E}[\langle \theta, R(\mu^n) \rangle] \quad \leq (1 + d(E'))\, \delta_G(\theta)\, \|\mu^n - \mu^*\|_1.$$

Let $v^d(\mu^n; E')$ denote the integral vertex of $\mathbb{L}(G)$ that is specified by the de-randomized integral solution $x^d(\mu^n; E')$. Since E' is some fixed forest-inducing subset, we frequently shorten this notation to $v^d(\mu^n)$. Theorem 6 shows that

$$\mathbb{E}[\langle \theta, R(\mu^n) \rangle] \leq \langle \theta, v^d(\mu^n) \rangle.$$

Suppose the de-randomized solution is not optimal so that $v^d(\mu^n) \neq \mu^*$. Then, from the definition of the graph-based gap $\Delta(\theta; G)$, we obtain

$$\langle \theta, \mu^* \rangle - \langle \theta, v^d(\mu^n) \rangle \geq \delta_G(\theta)\, \Delta(\theta; G).$$

Combining the pieces, we obtain

$$\begin{aligned} \delta_G(\theta)\, \Delta(\theta; G) &\leq \langle \theta, \mu^* \rangle - \langle \theta, v^d(\mu^n) \rangle \\ &\leq \langle \theta, \mu^* \rangle - \mathbb{E}[\langle \theta, R(\mu^n) \rangle] \\ &\leq (1 + d(E'))\delta_G(\theta) \|\mu^n - \mu^*\|_1, \end{aligned}$$

which implies $\|\mu^n - \mu^*\|_1 \geq \frac{\Delta(\theta; G)}{1 + d(E')}$. However, this conclusion is a contradiction under the given assumption on $\|\mu^n - \mu^*\|_1$ in the theorem. It thus holds that the de-randomized solution $v^d(\mu^n)$ is equal to the MAP optimum μ^*, thereby completing the proof.

References

H. H. Bauschke and J. M. Borwein. Legendre functions and the method of random bregman projections. *Journal of Convex Analysis*, 4(1):27–67, 1997.

M. Bayati, D. Shah, and M. Sharma. Maximum weight matching for max-product belief propagation. In *Int. Symp. Info. Theory*, Adelaide, Australia, September 2005.

M. Bayati, C. Borgs, J. Chayes, and R. Zecchina. Belief-propagation for weighted b-matchings on arbitrary graphs and its relation to linear programs with integer solutions. Technical Report arxiv:0709.1190, Microsoft Research, September 2007.

U. Bertele and F. Brioschi. *Nonserial Dynamic Programming*. Academic Press, New York, 1972.

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Boston, MA, 1997.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

D. Bertsimas and J. Tsitsikilis. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, 1997.

J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48(3):259–279, 1986.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

Y. Boykov, O. Veksler, and R .Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.

Y. Censor and S. A. Zenios. *Parallel Optimization - Theory, Algorithms and Applications*. Oxford University Press, 1997.

C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM Journal on Discrete Mathematics*, 18(3):608–625, 2005.

F. Deutsch and H. Hundal. The rate of convergence for the cyclic projection algorithm I: Angles between convex sets. *Journal of Approximation Theory*, 142:36 55, 2006.

R. L. Dykstra. An iterative procedure for obtaining i-projections onto the intersection of convex sets. *Annals of Probability*, 13(3):975–984, 1985.

J. Feldman, D. R. Karger, and M. J. Wainwright. Linear programming-based decoding of turbo-like codes and its relation to iterative approaches. In *Proc. 40th Annual Allerton Conf. on Communication, Control, and Computing*, October 2002a.

J. Feldman, M. J. Wainwright, and D. R. Karger. Linear programming-based decoding of turbo-like codes and its relation to iterative approaches. In *Proc. Allerton Conf. Comm., Control and Computing*, October 2002b.

W. T. Freeman and Y. Weiss. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. Info. Theory*, 47:736–744, 2001.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. PAMI*, 6:721–741, 1984.

A. Globerson and T. Jaakkola. Convergent propagation algorithms via oriented trees. In *Proc. Uncertainty in Artificial Intelligence*, Vancouver, Canada, July 2007a.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Neural Information Processing Systems*, Vancouver, Canada, December 2007b.

D.M. Greig, B.T. Porteous, and A.H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51, 1989.

S.-P. Han. A successive projection method. *Math. Programming*, 40:114, 1988.

T. Hazan and A. Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energy. In *The 24th Conference on Uncertainty in Artificial Intelligence (UAI)*, Helsinki, 2008.

B. Huang and T. Jebara. Loopy belief propagation for bipartite maximum weight b-matching. In *AISTATS*, San Juan, Puerto Rico, March 2007.

A. N. Iusem and M. Teboulle. Convergence rate analysis of nonquadratic proximal methods for convex and linear programming. *Mathematics of Operations Research*, 20(3):657–677, 1995.

J. K. Johnson, D. M. Malioutov, and A. S. Willsky. Lagrangian relaxation for MAP estimation in graphical models. In *Proc. 45th Allerton Conf. Comm. Cont. Comp.*, Urbana-Champaign, IL, September 2007.

J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *IEEE Symp. Found. Comp. Science*, pages 14–24, 1999.

V. Kolmogorov. Convergent tree-reweighted message-passing for energy minimization. In *International Workshop on Artificial Intelligence and Statistics*, January 2005.

V. Kolmogorov. Convergent tree-reweighted message-passing for energy minimization. *IEEE Trans. PAMI*, 28(10):1568–1583, October 2006.

V. Kolmogorov and M. J. Wainwright. On optimality properties of tree-reweighted message-passing. In *Uncertainty in Artificial Intelligence*, July 2005.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, Rio di Janeiro, Brazil, 2007.

V. K. Koval and M. I. Schlesinger. Two-dimensional programming in image analysis problems. *USSR Academy of Science, Automatics and Telemechanics*, 8:149–168, 1976. In Russian.

P. Kumar, P.H.S. Torr, and A. Zisserman. Solving markov random fields using second order cone programming. *IEEE Conference of Computer Vision and Pattern Recognition*, 2006.

P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

P. Ravikumar and J. Lafferty. Quadratic programming relaxations for metric labeling and markov random field map estimation. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 737–744, 2006.

G. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, 1970.

S. Sanghavi, D. Shah, and A. Willsky. Message-passing for max-weight independent set. In *Neural Information Processing Systems*, Vancouver, Canada, December 2007.

M.V. Solodov and B.F. Svaiter. A unified framework for some inexact proximal point algorithms. *Numerical Functional Analysis and Optimization*, 22:1013–1035, 2001.

P. Tseng and D. P. Bertsekas. On the convergence of the exponential multiplier method for convex programming. *Math. Programming*, 60:1—19, 1993.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical report, UC Berkeley, Department of Statistics, No. 649, September 2003.

M. J. Wainwright and M. I. Jordan. Treewidth-based conditions for exactness of the Sherali-Adams and Lasserre relaxations. Technical report, UC Berkeley, Department of Statistics, No. 671, September 2004.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. In *Uncertainty in Artificial Intelligence*, volume 18, August 2002.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Trans. Info. Theory*, 49(5):1120–1146, May 2003.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree consistency and bounds on the max-product algorithm and its generalizations. *Statistics and Computing*, 14:143–166, April 2004.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Exact MAP estimates via agreement on (hyper)trees: Linear programming and message-passing. *IEEE Trans. Information Theory*, 51 (11):3697–3717, November 2005.

Y. Weiss, C. Yanover, and T. Meltzer. Map estimation, linear programming and belief propagation with convex free energies. In *Uncertainty in Artificial Intelligence*, 2007.

C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation: An empirical study. *Journal of Machine Learning Research*, 7:1887–1907, September 2006.

J.S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Trans. Info. Theory*, 51(7):2282–2312, July 2005.

Journal of Machine Learning Research 11 (2010) 1081-1107 Submitted 5/09; Revised 1/10; Published 3/10

Analysis of Multi-stage Convex Relaxation for Sparse Regularization

Tong Zhang * TZHANG@STAT.RUTGERS.EDU
Statistics Department
110 Frelinghuysen Road
Rutgers University
Piscataway, NJ 08854

Editor: Francis Bach

Abstract

We consider learning formulations with non-convex objective functions that often occur in practical applications. There are two approaches to this problem:

- Heuristic methods such as gradient descent that only find a local minimum. A drawback of this approach is the lack of theoretical guarantee showing that the local minimum gives a good solution.
- Convex relaxation such as L_1-regularization that solves the problem under some conditions. However it often leads to a sub-optimal solution in reality.

This paper tries to remedy the above gap between theory and practice. In particular, we present a multi-stage convex relaxation scheme for solving problems with non-convex objective functions. For learning formulations with sparse regularization, we analyze the behavior of a specific multi-stage relaxation scheme. Under appropriate conditions, we show that the local solution obtained by this procedure is superior to the global solution of the standard L_1 convex relaxation for learning sparse targets.

Keywords: sparsity, non-convex optimization, convex relaxation, multi-stage convex relaxation

1. Introduction

We consider the general regularization framework for machine learning, where a loss function is minimized, subject to a regularization condition on the model parameter. For many natural machine learning problems, either the loss function or the regularization condition can be non-convex. For example, the loss function is non-convex for classification problems, and the regularization condition is non-convex in problems with sparse parameters.

A major difficulty with nonconvex formulations is that the global optimal solution cannot be efficiently computed, and the behavior of a local solution is hard to analyze. In practice, convex relaxation (such as support vector machine for classification or L_1 regularization for sparse learning) has been adopted to remedy the problem. The choice of convex formulation makes the solution unique and efficient to compute. Moreover, the solution is easy to analyze theoretically. That is, it can be shown that the solution of the convex formulation approximately solves the original problem under appropriate assumptions. However, for many practical problems, such simple convex relaxation schemes can be sub-optimal.

∗. Supported by the following grants: NSF DMS-0706805, NSA-081024, and AFOSR-10097389.

Because of the above gap between practice and theory, it is important to study direct solutions of non-convex optimization problems beyond the standard convex relaxation. Our goal is to design a numerical procedure that leads to a *reproducible solution* which is better than the standard convex relaxation solution. In order to achieve this, we present a general framework of multi-stage convex relaxation, which iteratively refine the convex relaxation formulation to give better solutions. The method is derived from concave duality, and involves solving a sequence of convex relaxation problems, leading to better and better approximations to the original nonconvex formulation. It provides a unified framework that includes some previous approaches (such as LQA Jianqing Fan, 2001, LLA Zou and Li, 2008, CCCP Yuille and Rangarajan, 2003) as special cases. The procedure itself may be regarded as a special case of alternating optimization, which automatically ensures its convergence. Since each stage of multi-stage convex relaxation is a convex optimization problem, the approach is also computationally efficient. Although the method only leads to a local optimal solution for the original nonconvex problem, this local solution is a refinement of the global solution for the initial convex relaxation. Therefore intuitively one expects that the local solution is better than the standard one-stage convex relaxation. In order to prove this observation more rigorously, we consider least squares regression with nonconvex sparse regularization terms, for which we can analyze the effectiveness of the multi-stage convex relaxation. It is shown that under appropriate assumptions, the (local) solution computed by the multi-stage convex relaxation method using nonconvex regularization achieves better parameter estimation performance than the standard convex relaxation with L_1 regularization.

The main contribution of this paper is the analysis of sparse regularized least squares regression presented in Section 3, where we derive theoretical results showing that under appropriate conditions, it is beneficial to use multi-stage convex relaxation with nonconvex regularization as opposed to the standard convex L_1 regularization. This demonstrates the effectiveness of multi-stage convex relaxation for a specific but important problem. Although without theoretical analysis, we shall also present the general idea of multi-stage convex relaxation in Section 2, because it can be applied to other potential application examples as illustrated in Appendix C. The gist of our analysis can be applied to those examples (e.g., the multi-task learning problem in the setting of matrix completion, which has drawn significant attention recently) as well. However, the detailed derivation will be specific to each application and the analysis will not be trivial. Therefore while we shall present a rather general form of multi-stage convex relaxation formulation in order to unify various previous approaches, and put this work in a broader context, the detailed theoretical analysis (and empirical studies) for other important applications will be left to future work.

2. Multi-stage Convex Relaxation

This section presents the general idea of multi-stage convex relaxation which can be applied to various optimization problems. It integrates a number of existing ideas into a unified framework.

2.1 Regularized Learning Formulation

The multi-stage convex relaxation approach considered in the paper can be applied to the following optimization problem, which can be motivated from supervised learning formulation. As back-

ground information, its connection to regularized learning formula (15) is given in Appendix B.

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} R(\mathbf{w}),$$
$$R(\mathbf{w}) = R_0(\mathbf{w}) + \sum_{k=1}^{K} R_k(\mathbf{w}), \tag{1}$$

where $\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_d] \in \mathbb{R}^d$ is a d-dimensional parameter vector, and $R(\mathbf{w})$ is the general form of a regularized objective function. Moreover, for convenience, we assume that $R_0(\mathbf{w})$ is convex in $\mathbf{w}$, and each $R_k(\mathbf{w})$ is non-convex. In the proposed work, we shall employ convex/concave duality to derive convex relaxations of (1) that can be efficiently solved.

Related to (1), one may also consider the constrained formulation

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} R_0(\mathbf{w}) \quad \text{subject to } \sum_{k=1}^{K} R_k(\mathbf{w}) \le A, \tag{2}$$

where A is a constant. One may also mix (1) and (2).

2.2 Concave Duality

In the following discussion, we consider a single nonconvex component $R_k(\mathbf{w})$ in (1), which we shall rewrite using concave duality. Let $\mathbf{h}_k(\mathbf{w}) : R^d \to \Omega_k \subset R^{d_k}$ be a vector function with Ω_k being the convex hull of its range. It may not be a one-to-one map. However, we assume that there exists a function $\bar{R}_k$ defined on Ω_k so that we can express $R_k(\mathbf{w})$ as

$$R_k(\mathbf{w}) = \bar{R}_k(\mathbf{h}_k(\mathbf{w})).$$

Assume that we can find $\mathbf{h}_k$ so that the function $\bar{R}_k(\mathbf{u}_k)$ is concave on $\mathbf{u}_k \in \Omega_k$. Under this assumption, we can rewrite the regularization function $R_k(\mathbf{w})$ as:

$$R_k(\mathbf{w}) = \inf_{\mathbf{v}_k \in R^{d_k}} \left[\mathbf{v}_k^\top \mathbf{h}_k(\mathbf{w}) - R_k^*(\mathbf{v}_k) \right] \tag{3}$$

using concave duality (Rockafellar, 1970). In this case, $R_k^*(\mathbf{v}_k)$ is the concave dual of $\bar{R}_k(\mathbf{u}_k)$ given below

$$R_k^*(\mathbf{v}_k) = \inf_{\mathbf{u}_k \in \Omega_k} \left[\mathbf{v}_k^\top \mathbf{u}_k - \bar{R}_k(\mathbf{u}_k) \right].$$

Note that using the convention in convex analysis (Rockafellar, 1970), we may assume that $R_k^*(\mathbf{v}_k)$ is defined on R^{d_k} but may take $-\infty$ value. Equivalently, we may consider the subset $\{\mathbf{v}_k : R_k^*(\mathbf{v}_k) > -\infty\}$ as the feasible region of the optimization problem (3), and assume that $R_k^*(\mathbf{v}_k)$ is only defined on this feasible region.

It is well-known that the minimum of the right hand side of (3) is achieved at

$$\hat{\mathbf{v}}_k = \nabla_{\mathbf{u}} \bar{R}_k(\mathbf{u})|_{\mathbf{u}=\mathbf{h}_k(\mathbf{w})}. \tag{4}$$

This is the general framework we suggest in the paper. For illustration, some example non-convex problems encountered in machine learning are included in Appendix C.

2.3 Penalized Formulation

Let $\mathbf{h}_k(\mathbf{w})$ be a vector function with convex components, so that (3) holds. Given an appropriate vector $\mathbf{v}_k \in R^{d_k}$, a simple convex relaxation of (1) becomes

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}\in R^d} \left[R_0(\mathbf{w}) + \sum_{k=1}^{K} \mathbf{h}_k(\mathbf{w})^\top \mathbf{v}_k \right]. \tag{5}$$

This simple relaxation yields a solution that is different from the solution of (1). However, if each $\mathbf{h}_k$ satisfies the condition of Section 2.2, then it is possible to write $R_k(\mathbf{w})$ using (3). Now, with this new representation, we can rewrite (1) as

$$[\hat{\mathbf{w}}, \hat{\mathbf{v}}] = \arg\min_{\mathbf{w},\{\mathbf{v}_k\}} \left[R_0(\mathbf{w}) + \sum_{k=1}^{K} (\mathbf{h}_k(\mathbf{w})^\top \mathbf{v}_k - R_k^*(\mathbf{v}_k)) \right]. \tag{6}$$

This is clearly equivalent to (1) because of (3). If we can find a good approximation of $\hat{\mathbf{v}} = \{\hat{\mathbf{v}}_k\}$ that improves upon the initial value of $\hat{\mathbf{v}}_k = \mathbf{1}$, then the above formulation can lead to a refined convex problem in $\mathbf{w}$ that is a better convex relaxation than (5).

Our numerical procedure exploits the above fact, which tries to improve the estimation of $\mathbf{v}_k$ over the initial choice of $\mathbf{v}_k = \mathbf{1}$ in (5) using an iterative algorithm. This can be done using an alternating optimization procedure, which repeatedly applies the following two steps:

- First we optimize $\mathbf{w}$ with $\mathbf{v}$ fixed: this is a convex problem in $\mathbf{w}$ with appropriately chosen $\mathbf{h}(\mathbf{w})$.
- Second we optimize $\mathbf{v}$ with $\mathbf{w}$ fixed: although non-convex, it has a closed form solution that is given by (4).

Initialize $\hat{\mathbf{v}} = \mathbf{1}$
Repeat the following two steps until convergence:

- Let
$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \sum_{k=1}^{K} \mathbf{h}_k(\mathbf{w})^\top \hat{\mathbf{v}}_k \right]. \tag{7}$$
- Let $\hat{\mathbf{v}}_k = \nabla_{\mathbf{u}} \bar{R}_k(\mathbf{u})|_{\mathbf{u}=\mathbf{h}_k(\hat{\mathbf{w}})}$ $(k = 1, \ldots, K)$

Figure 1: Multi-stage Convex Relaxation Method

The general procedure for solving (6) is presented in Figure 1. It can be regarded as a generalization of CCCP (concave-convex programming) (Yuille and Rangarajan, 2003), which takes $\mathbf{h}(\mathbf{w}) = \mathbf{w}$. It is also more general than LQA (local quadratic approximation) (Jianqing Fan, 2001) or LLA (local linear approximation) (Zou and Li, 2008). Specifically LQA takes $\mathbf{h}_j(\hat{\mathbf{w}}) = \mathbf{w}_j^2$ and LLA takes $\mathbf{h}_j(\hat{\mathbf{w}}) = |\mathbf{w}_j|$. The justification of those procedures rely on the so-called MM (majorization-minimization) principle, where an upper bound of the objective function is minimized at each step (see Zou and Li, 2008 and references therein). However, in order to apply MM, for each particular choice of $\mathbf{h}$, one has to demonstrate that the convex relaxation is indeed an upper bound, which is

necessary to show convergence. In the concave relaxation formulation adopted in this work, the justification of convergence is automatically embedded in (6), which becomes a joint optimization problem. Figure 1 is simply an alternating optimization procedure for solving (6), which is equivalent to (1). Since convex duality of many interesting objective functions (including matrix functions) are familiar to many machine learning researchers, the concave duality derivation presented here can be automatically applied to various applications without the need to worry about convergence justification. This will be especially useful for complex formulations such as structured or matrix regularization, where the more traditional MM idea cannot be easily applied. One may also regard our framework as a principled method to design a class of algorithms that may be interpreted as MM procedures. Some examples illustrate its applications are presented in Appendix C.

Note that by repeatedly refining the parameter $\mathbf{v}$, we can potentially obtain better and better convex relaxation in Figure 1, leading to a solution superior to that of the initial convex relaxation. Since at each step the procedure decreases the objective function in (6), its convergence to a local minimum is easy to show. In fact, in order to achieve convergence, one only needs to approximately minimize (7) and reasonably decrease the objective value at each step. We skip the detailed analysis here, because in the general case, a local solution is not necessarily a good solution, and there are other approaches (such as gradient descent) that can compute a local solution. In order to demonstrate the effectiveness of multi-stage convex relaxation, we shall include a more careful analysis for the special case of sparse regularization in Section 3.1. Our theory shows that the local solution of multi-stage relaxation with a nonconvex sparse regularizer is superior to the convex L_1 regularization solution (under appropriate conditions).

2.4 Constrained Formulation

The multi-stage convex relaxation idea can also be used to solve the constrained formulation (2). The one-stage convex relaxation of (2), given fixed relaxation parameter $\mathbf{v}_k$, becomes

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}\in R^d} R_0(\mathbf{w}) \quad \text{subject to } \sum_{k=1}^{K} \mathbf{h}_k(\mathbf{w})^\top \mathbf{v}_k \le A + \sum_{k=1}^{K} R_k^*(\mathbf{v}_k).$$

Because of (3), the above formulation is equivalent to (2) if we optimize over $\mathbf{v}$. This means that by optimizing $\mathbf{v}$ in addition to $\mathbf{w}$, we obtain the following algorithm:

- Initialize $\hat{\mathbf{v}} = \mathbf{1}$
- Repeat the following two steps until convergence:
  - Let
$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} R_0(\mathbf{w}) \quad \text{subject to } \sum_{k=1}^{K} \mathbf{h}_k(\mathbf{w})^\top \hat{\mathbf{v}}_k \le A + \sum_{k=1}^{K} R_k^*(\hat{\mathbf{v}}_k).$$
  - Let $\hat{\mathbf{v}}_k = \nabla_{\mathbf{u}} \bar{R}_k(\mathbf{u})|_{\mathbf{u}=\mathbf{h}_k(\hat{\mathbf{w}})}$ $(k = 1,\ldots,K)$

If an optimization problem includes both nonconvex penalization and nonconvex constrains, then one may use the above algorithm with Figure 1.

3. Multi-stage Convex Relaxation for Sparse Regularization

The multi-stage convex relaxation method described in the previous section tries to obtain better approximations of the original nonconvex problem by refining the convex relaxation formulation. Since the local solution found by the algorithm is the global solution of a refined convex relaxation formulation, it should be closer to the desired solution than that of the standard one-stage convex relaxation method. Although this high level intuition is appealing, it is still necessarily to present a more rigorous theoretical result which can precisely demonstrate the advantage of the multi-stage approach over the standard single stage method. Unless we can develop a theory to show the effectiveness of the multi-stage procedure in Figure 1, our proposal is yet another local minimum finding scheme that may potentially get stuck at a bad local solution.

In order to obtain some strong theoretical results that can demonstrate the advantage of the multi-stage approach, we consider the special case of sparse learning. This is because this problem has been well-studied in recent years, and the behavior of convex relaxation (L_1 regularization) is well-understood.

3.1 Theory of Sparse Regularization

For a non-convex but smooth regularization condition such as capped-L_1 or smoothed-L_p with $p \in (0,1)$, standard numerical techniques such as gradient descent lead to a local minimum solution. Unfortunately, it is difficult to find the global optimum, and it is also difficult to analyze the quality of the local minimum. Although in practice, such a local minimum solution may outperform the Lasso solution, the lack of theoretical (and practical) performance guarantee prevents the more wide-spread applications of such algorithms. As a matter of fact, results with non-convex regularization are difficult to reproduce because different numerical optimization procedures can lead to different local minima. Therefore the quality of the solution heavily depend on the numerical procedure used.

The situation is very different for a convex relaxation formulation such as L_1-regularization (Lasso). The global optimum can be easily computed using standard convex programming techniques. It is known that in practice, 1-norm regularization often leads to sparse solutions (although often suboptimal). Moreover, its performance has been theoretically analyzed recently. For example, it is known from the compressed sensing literature that under certain conditions, the solution of L_1 relaxation may be equivalent to L_0 regularization asymptotically (e.g., Candes and Tao, 2005). If the target is truly sparse, then it was shown in Zhao and Yu (2006) that under some restrictive conditions referred to as *irrepresentable conditions*, 1-norm regularization solves the feature selection problem. The prediction performance of this method has been considered in Koltchinskii (2008), Zhang (2009a), Bickel et al. (2009) and Bunea et al. (2007).

In spite of its success, L_1-regularization often leads to suboptimal solutions because it is not a good approximation to L_0 regularization. Statistically, this means that even though it converges to the true sparse target when $n \to \infty$ (consistency), the rate of convergence can be suboptimal. The only way to fix this problem is to employ a non-convex regularization condition that is closer to L_0 regularization. In the following, we formally prove a result for multi-stage convex relaxation with non-convex sparse regularization that is superior to the Lasso result. In essence, we establish a performance guarantee for non-convex formulations when they are solved by using the multi-stage convex relaxation approach which is more sophisticated than the standard one-stage convex relaxation.

In supervised learning, we observe a set of input vectors $\mathbf{x}_1,\ldots,\mathbf{x}_n \in R^d$, with corresponding desired output variables $y_1,\ldots,y_n$. In general, we may assume that there exists a target $\bar{\mathbf{w}} \in R^d$ such that

$$y_i = \bar{\mathbf{w}}^\top \mathbf{x}_i + \varepsilon_i \qquad (i = 1,\ldots,n), \tag{8}$$

where ε_i are zero-mean independent random noises (but not necessarily identically distributed). Moreover, we assume that the target vector $\bar{\mathbf{w}}$ is sparse. That is, there exists $\bar{k} = \|\bar{\mathbf{w}}\|_0$ is small. This is the standard statistical model for sparse learning.

Let $\mathbf{y}$ denote the vector of $[y_i]$ and X be the $n \times d$ matrix with each row a vector $\mathbf{x}_i$. We are interested in recovering $\bar{\mathbf{w}}$ from noisy observations using the following sparse regression method:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[\frac{1}{n}\|X\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \sum_{j=1}^{d} g(|\mathbf{w}_j|) \right], \tag{9}$$

where $g(|\mathbf{w}_j|)$ is a regularization function. Here we require that $g'(u)$ is non-negative which means we penalize larger u more significantly. Moreover, we assume $u^{1-q}g'(u)$ is a non-increasing function when $u > 0$, which means that $[g(|\mathbf{w}_1|),\ldots,g(|\mathbf{w}_d|)]$ is concave with respect to $\mathbf{h}(\mathbf{w}) = [|\mathbf{w}_1|^q,\ldots,|\mathbf{w}_d|^q]$ for some $q \geq 1$. It follows that (9) can be solved using the multi-stage convex relaxation algorithm in Figure 2, which we will analyze. Although this algorithm was mentioned in Zou and Li (2008) as LLA when $q = 1$, they only presented a one-step low-dimensional asymptotic analysis. We present a true multi-stage analysis in high dimension. Our analysis also focuses on $q = 1$ (LLA) for convenience because the Lasso analysis in Zhang (2009a) can be directly adapted; however in principle, one can also analyze the more general case of $q > 1$.

Initialize $\lambda_j^{(0)} = \lambda$ for $j = 1,\ldots,d$
For $\ell = 1, 2, \ldots$

- Let
$$\hat{\mathbf{w}}^{(\ell)} = \arg\min_{\mathbf{w}\in R^d} \left[\frac{1}{n}\|X\mathbf{w} - \mathbf{y}\|_2^2 + \sum_{j=1}^{d} \lambda_j^{(\ell-1)} |\mathbf{w}_j|^q \right]. \tag{10}$$
- Let $\lambda_j^{(\ell)} = \lambda q^{-1} |\hat{\mathbf{w}}_j|^{1-q} g'(|\hat{\mathbf{w}}_j^{(\ell)}|)$ $(j = 1,\ldots,d)$

Figure 2: Multi-stage Convex Relaxation for Sparse Regularization

For convenience, we consider fixed design only, where X is fixed and the randomness is with respect to $\mathbf{y}$ only. We require some technical conditions for our analysis. First we assume sub-Gaussian noise as follows.

Assumption 3.1 *Assume that $\{\varepsilon_i\}_{i=1,\ldots,n}$ in (8) are independent (but not necessarily identically distributed) sub-Gaussians: there exists $\sigma \geq 0$ such that $\forall i$ and $\forall t \in R$,*

$$\mathbf{E}_{\varepsilon_i} e^{t\varepsilon_i} \leq e^{\sigma^2 t^2/2}.$$

Both Gaussian and bounded random variables are sub-Gaussian using the above definition. For example, if a random variable $\xi \in [a,b]$, then $\mathbf{E}_\xi e^{t(\xi - \mathbf{E}\xi)} \leq e^{(b-a)^2 t^2/8}$. If a random variable is Gaussian: $\xi \sim N(0,\sigma^2)$, then $\mathbf{E}_\xi e^{t\xi} \leq e^{\sigma^2 t^2/2}$.

We also introduce the concept of sparse eigenvalue, which is standard in the analysis of L_1 regularization.

Definition 1 *Given k, define*

$$\rho_+(k) = \sup\left\{\frac{1}{n}\|X\mathbf{w}\|_2^2/\|\mathbf{w}\|_2^2 : \|\mathbf{w}\|_0 \le k\right\},$$
$$\rho_-(k) = \inf\left\{\frac{1}{n}\|X\mathbf{w}\|_2^2/\|\mathbf{w}\|_2^2 : \|\mathbf{w}\|_0 \le k\right\}.$$

Our main result is stated as follows. The proof is in the appendix.

Theorem 2 *Let Assumption 3.1 hold. Assume also that the target $\bar{\mathbf{w}}$ is sparse, with $\mathbf{E}y_i = \bar{\mathbf{w}}^\top \mathbf{x}_i$, and $\bar{k} = \|\bar{\mathbf{w}}\|_0$. Choose λ such that*

$$\lambda \ge 20\sigma\sqrt{2\rho_+(1)\ln(2d/\eta)/n}.$$

Assume that $g'(z) \ge 0$ is a non-increasing function such that $g'(z) = 1$ when $z \le 0$. Moreover, we require that $g'(\theta) \ge 0.9$ with $\theta = 9\lambda/\rho_-(2\bar{k}+s)$. Assume that $\rho_+(s)/\rho_-(2\bar{k}+2s) \le 1 + 0.5s/\bar{k}$ for some $s \ge 2\bar{k}$, then with probability larger than $1-\eta$:

$$\|\hat{\mathbf{w}}^{(\ell)} - \bar{\mathbf{w}}\|_2 \le \frac{17}{\rho_-(2\bar{k}+s)}\left[2\sigma\sqrt{\rho_+(\bar{k})}\left(\sqrt{7.4\bar{k}/n} + \sqrt{2.7\ln(2/\eta)/n}\right)\right.$$
$$\left. + \lambda\left(\sum_{j:\bar{\mathbf{w}}_j \ne 0} g'(|\bar{\mathbf{w}}_j| - \theta)^2\right)^{1/2}\right] + 0.7^\ell \frac{10}{\rho_-(2\bar{k}+s)}\sqrt{\bar{k}}\lambda,$$

where $\hat{\mathbf{w}}^{(\ell)}$ is the solution of (10) with $q = 1$.

Note that the theorem allows the situation $d \gg n$, which is what we are interested in. This is the first general analysis of multi-stage convex relaxation for high dimensional sparse learning, although some simpler asymptotic results for low dimensional two-stage procedures were obtained in Zou (2006) and Zou and Li (2008), they are not comparable to ours.

Results most comparable to what we have obtained here are that of the FoBa procedure in Zhang (2009b) and that of the MC+ procedure in Zhang (2010). The former is a forward backward greedy algorithm, which does not optimize (9), while the latter is a path-following algorithm for solving (9). Although results in Zhang (2010) are comparable to ours, we should note that efficient path-following computation in MC+ requires specialized regularizers $g(\cdot)$. Moreover, unlike our procedure, which is efficient because of convex optimization, there is no proof showing that the path-following strategy in Zhang (2010) is always efficient (in the sense that there may be exponentially many switching points). However, empirical experience in Zhang (2010) does indicate its efficiency for a class of regularizers that can be relatively easily handled by path-following. Therefore we are not claiming here that our approach will always be superior to Zhang (2010) in practice. Nevertheless, our result suggests that different local solution procedures can be used to solve the same nonconvex formulation with valid theoretical guarantees. This opens the door for additional theoretical studies of other numerical procedures.

The condition $\rho_+(s)/\rho_-(2\bar{k}+2s) \le 1 + 0.5s/\bar{k}$ requires the eigenvalue ratio $\rho_+(s)/\rho_-(s)$ to grow sub-linearly in s. Such a condition, referred to as *sparse eigenvalue condition*, is also needed in

the standard analysis of L_1 regularization (Zhang and Huang, 2008; Zhang, 2009a). It is related but weaker than the *restricted isometry property* (RIP) in compressive sensing (Candes and Tao, 2005). Note that in the traditional low-dimensional statistical analysis, one assumes that $\rho_+(s)/\rho_-(2\bar{k}+2s) < \infty$ as $s \to \infty$, which is significantly stronger than the condition we use here. Although in practice it is often difficult to verify the sparse eigenvalue condition for real problems, Theorem 2 nevertheless provides important theoretical insights for multi-stage convex relaxation.

Since in standard Lasso, $g'(|\mathbf{w}_j|) \equiv 1$, we obtain the following bound from Theorem 2

$$\|\hat{\mathbf{w}}_{L_1} - \bar{\mathbf{w}}\|_2 = O(\sqrt{k}\lambda),$$

where $\hat{\mathbf{w}}_{L_1}$ is the solution of the standard L_1 regularization. This bound is tight for Lasso, in the sense that the right hand side cannot be improved except for the constant—this can be easily verified with an orthogonal design matrix. It is known that in order for Lasso to be effective, one has to pick λ no smaller than the order $\sigma\sqrt{\ln d/n}$. Therefore, the parameter estimation error of the standard Lasso is of the order $\sigma\sqrt{\bar{k}\ln d/n}$, which cannot be improved.

In comparison, if we consider an appropriate regularization condition $g(|\mathbf{w}_j|)$ that is concave in $|\mathbf{w}_j|$. Since $g'(|\mathbf{w}_j|) \approx 0$ when $|\mathbf{w}_j|$ is large, the bound in Theorem 2 can be significantly better when most non-zero coefficients of $\bar{\mathbf{w}}$ are relatively large in magnitude. For example, consider the capped-L_1 regularizer $g(|\mathbf{w}_j|) = \min(\alpha, |\mathbf{w}_j|)$ with $\alpha \geq \theta$; in the extreme case where $\min_j |\mathbf{w}_j| > \alpha + \theta$ (which can be achieved when all nonzero components of $\bar{\mathbf{w}}$ are larger than the order $\sigma\sqrt{\ln d/n}$), we obtain the better bound

$$\|\hat{\mathbf{w}}^{(\ell)} - \bar{\mathbf{w}}\|_2 = O(\sqrt{\bar{k}/n} + \sqrt{\ln(1/\eta)/n})$$

for the multi-stage procedure for a sufficiently large ℓ at the order of $\ln\ln d$. This bound is superior to the standard one-stage L_1 regularization bound $\|\hat{\mathbf{w}}_{L_1} - \bar{\mathbf{w}}\|_2 = O(\sqrt{\bar{k}\ln(d/\eta)/n})$, which is tight for Lasso. The difference can be significant when $\ln d$ is large.

Generally speaking, with a regularization condition $g(|\mathbf{w}_j|)$ that is concave in $|\mathbf{w}_j|$, the dependency on λ is through $g'(|\bar{\mathbf{w}}_j|)$ which decreases as $|\bar{\mathbf{w}}_j|$ increases. This removes the bias of the Lasso and leads to improved performance. Specifically, if $\bar{\mathbf{w}}_j$ is large, then $g'(|\bar{\mathbf{w}}_j|) \approx 0$. In comparison, the Lasso bias is due to the fact that $g'(|\bar{\mathbf{w}}_j|) \equiv 1$. For illustration, the derivative $g'(\cdot)$ of some sparse regularizers are plotted in Figure 3.

Note that our theorem only applies to regularizers with finite derivative at zero. That is, $g'(0) < \infty$. The result doesn't apply to L_p regularization with $p < 1$ because $g'(0) = \infty$. Although a weaker result can be obtained for such regularizers, we do not include it here. We only include an intuitive example below to illustrate why the condition $g'(0) < \infty$ is necessary for stronger results presented in the paper. Observe that the multi-stage convex relaxation method only computes a local minimum, and the regularization update rule is given by $\lambda_j^{(\ell-1)} = g'(\hat{\mathbf{w}}_j^{(\ell-1)})$. If $g'(0) = \infty$, then $\lambda_j^{(\ell-1)} = \infty$ when $\hat{\mathbf{w}}_j^{(\ell-1)} = 0$. This means that if a feature accidentally becomes zero in some stage, it will always remain zero. This is why only weaker results can be obtained for L_p regularizers ($p < 1$): we need to further assume that $\hat{\mathbf{w}}_j^{(\ell)}$ never becomes close to zero when $\bar{\mathbf{w}}_j \neq 0$. A toy example is presented in Table 1 to demonstrate this point. The example is a simulated regression problem with $d = 500$ variables and $n = 100$ training data. The first five variables of the target $\bar{\mathbf{w}}$ are non-zeros, and the remaining variables are zeros. For both capped-L_1 and L_p regularizers, the first stage is the standard L_1 regularization, which misses the correct feature #2 and wrongly selects some incorrect ones. For capped-L_1 regularization, in the second stage, because most correct features are identified,

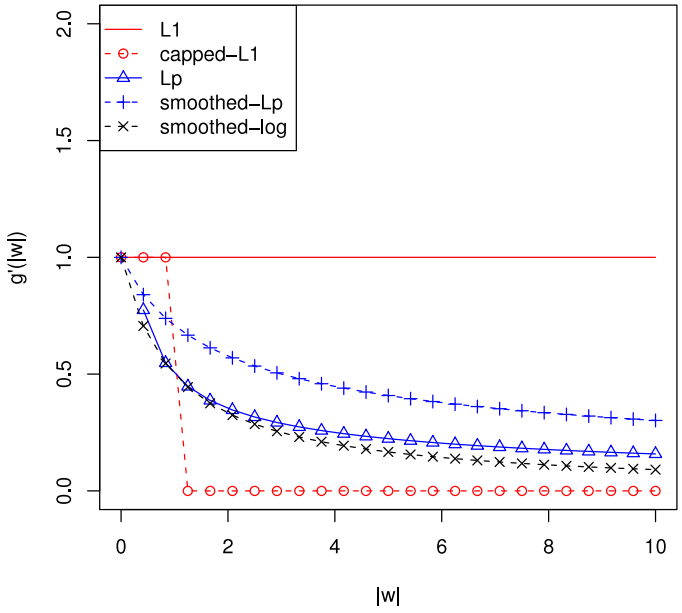


Figure 3: Derivative $g'(|\mathbf{w}_j|)$ of some sparse regularizers

Stage ℓ	coefficients	$\|\hat{\mathbf{w}}^{(\ell)} - \bar{\mathbf{w}}\|_2$
multi-stage capped-L_1		
1	$[6.0, 0.0, 4.7, 4.8, 3.9, 0.6, 0.7, 1.2, 0.0, \ldots]$	4.4
2	$[7.7, 0.4, 5.7, 6.3, 5.7, 0.0, 0.0, 0.2, 0.0, \ldots]$	1.6
3	$[7.8, 1.2, 5.7, 6.6, 5.7, 0.0, 0.0, 0.0, 0.0, \ldots]$	0.98
4	$[7.8, 1.2, 5.7, 6.6, 5.7, 0.0, 0.0, 0.0, 0.0, \ldots]$	0.98
multi-stage $L_{0.5}$		
1	$[6.0, 0.0, 4.7, 4.8, 3.9, 0.6, 0.7, 1.2, 0.0, \ldots]$	4.4
2	$[7.3, 0.0, 5.4, 5.9, 5.3, 0.0, 0.3, 0.3, 0.0, 0.0, \ldots]$	2.4
3	$[7.5, 0.0, 5.6, 6.1, 5.7, 0.0, 0.1, 0.0, 0.0, 0.0, \ldots]$	2.2
4	$[7.5, 0.0, 5.6, 6.2, 5.7, 0.0, 0.1, 0.0, 0.0, 0.0, \ldots]$	2.1
target $\bar{\mathbf{w}}$	$[8.2, 1.7, 5.4, 6.9, 5.7, 0.0, 0.0, 0.0, 0.0, \ldots]$	

Table 1: An Illustrative Example for Multi-stage Sparse Regularization

the corresponding "bias" is reduced by not penalizing the corresponding variables. This leads to improved performance. Since the correct feature #2 shows up in stage 2, we are able to identify it and further improve the convex relaxation in stage 3. After stage 3, the procedure stabilizes because it computes exactly the same relaxation. For L_p regularization, since feature #2 becomes zero in stage 1, it will remain zero thereafter because $\lambda_2^{(\ell)} = \infty$ when $\ell \geq 1$. In order to remedy this problem, one has to use a regularizer with $g'(0) < \infty$ such as the smoothed L_p regularizer.

3.2 Empirical Study

Although this paper focuses on the development of the general multi-stage convex relaxation framework as well as its theoretical understanding (in particular the major result given in Theorem 2), we include two simple numerical examples to verify our theory. More comprehensive empirical comparisons can be found in other related work such as Candes et al. (2008), Zou (2006) and Zou and Li (2008).

In order to avoid cluttering, we only present results with capped-L_1 and L_p ($p = 0.5$) regularization methods. Note that based on Theorem 2, we may tune α in capped-L_1 by using a formula $\alpha = \alpha_0\lambda$ where λ is the regularization parameter. We choose $\alpha_0 = 10$ and $\alpha_0 = 100$.

In the first experiment, we generate an $n \times d$ random matrix with its column j corresponding to $[\mathbf{x}_{1,j}, \ldots, \mathbf{x}_{n,j}]$, and each element of the matrix is an independent standard Gaussian $N(0,1)$. We then normalize its columns so that $\sum_{i=1}^{n} \mathbf{x}_{i,j}^2 = n$. A truly sparse target $\bar{\beta}$, is generated with k nonzero elements that are uniformly distributed from $[-10, 10]$. The observation $\mathbf{y}_i = \bar{\beta}^\top \mathbf{x}_i + \varepsilon_i$, where each $\varepsilon_i \sim N(0, \sigma^2)$. In this experiment, we take $n = 50, d = 200, k = 5, \sigma = 1$, and repeat the experiment 100 times. The average training error and 2-norm parameter estimation error are reported in Figure 4. We compare the performance of multi-stage methods with different regularization parameter λ. As expected, the training error for the multi-stage algorithms are smaller than that of L_1, due to the smaller bias. Moreover, substantially smaller parameter estimation error is achieved by the multi-stage procedures, which is consistent with Theorem 2. This can be regarded as an empirical verification of the theoretical result.

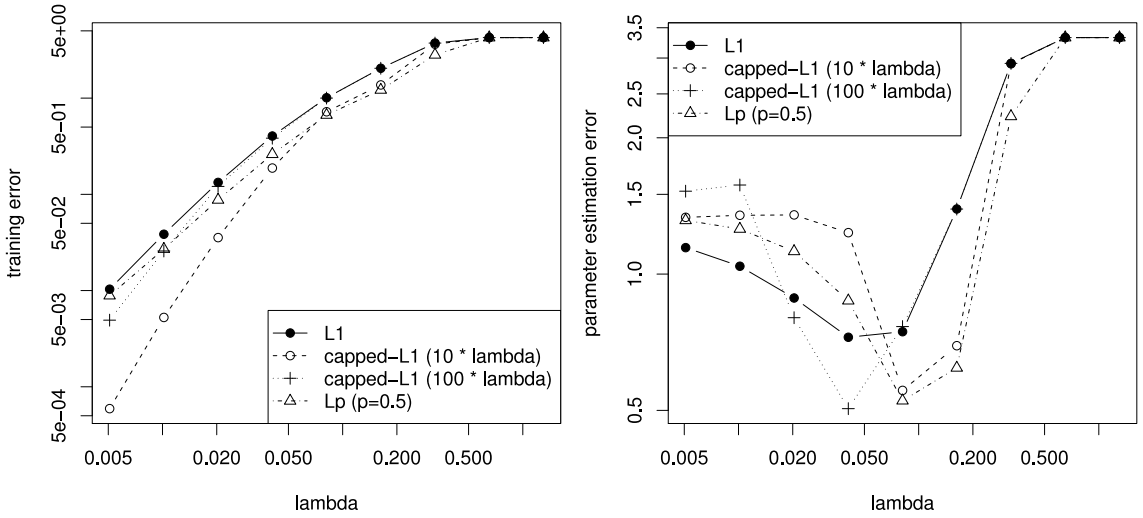


Figure 4: Performance of multi-stage convex relaxation on simulation data. Left: average training squared error versus λ; Right: parameter estimation error versus λ.

In the second experiment, we use the *Boston Housing* data to illustrate the effectiveness of multi-stage convex relaxation. This data set contains 506 census tracts of Boston from the 1970 census, available from the *UCI Machine Learning Database Repository*: `http://archive.ics.`

uci.edu/ml/. Each census tract is a data-point, with 13 features (we add a constant offset on e as the 14th feature), and the desired output is the housing price. In this example, we randomly partition the data into 20 training plus 486 test points. We perform the experiments 100 times, and report training and test squared error versus the regularization parameter λ for different q. The results are plotted in Figure 5. In this case, $L_{0.5}$ is not effective, while capped-L_1 regularization with $\alpha = 100\lambda$ is slightly better than Lasso. Note that this data set contains only a small number ($d = 14$) features, which is not the case where we can expect significant benefit from the multi-stage approach (most of other UCI data similarly contain only small number of features). In order to illustrate the advantage of the multi-stage method more clearly, we also report results on a modified Boston Housing data, where we append 20 random features (similar to the simulation experiments) to the original Boston Housing data, and rerun the experiments. The results are shown in Figure 6. As expected from Theorem 2 and the discussion thereafter, since d becomes large, the multi-stage convex relaxation approach with capped-L_1 regularization and $L_{0.5}$ regularization perform significantly better than the standard Lasso.

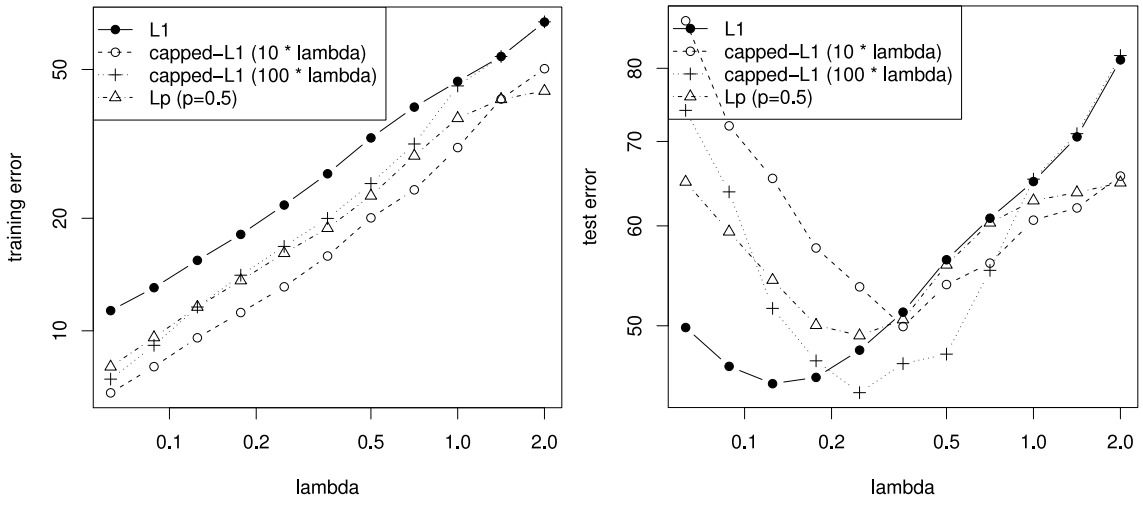


Figure 5: Performance of multi-stage convex relaxation on the original Boston Housing data. Left: average training squared error versus λ; Right: test squared error versus λ.

4. Discussion

Many machine learning applications require solving nonconvex optimization problems. There are two approaches to this problem:

- Heuristic methods such as gradient descent that only find a local minimum. A drawback of this approach is the lack of theoretical guarantee showing that the local minimum gives a good solution.

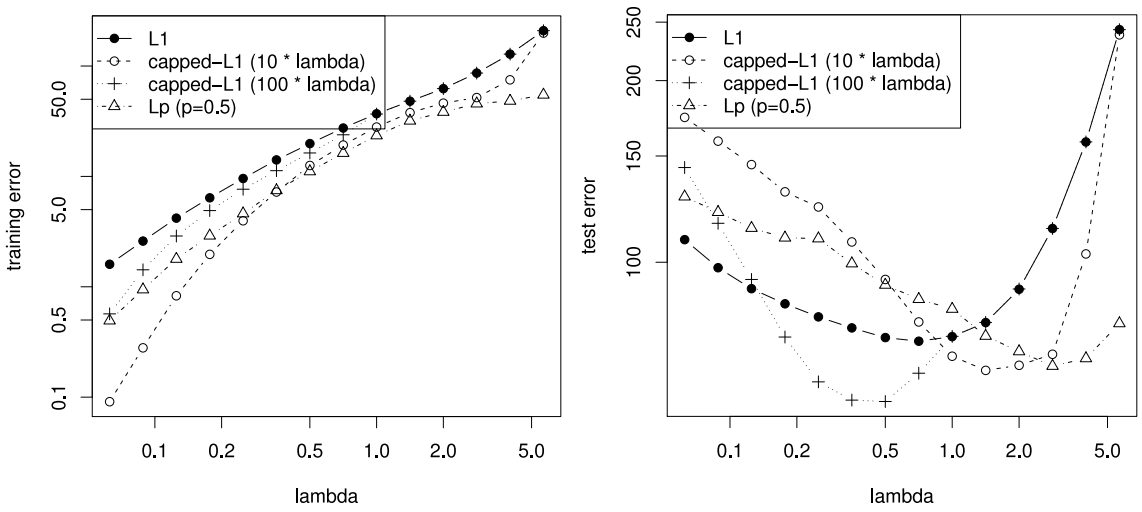


Figure 6: Performance of multi-stage convex relaxation on the modified Boston Housing data. Left: average training squared error versus λ; Right: test squared error versus λ.

- Convex relaxation such as L_1-regularization that solves the problem under some conditions. However it often leads to a sub-optimal solution in reality.

The goal of this paper is to remedy the above gap between theory and practice. In particular, we investigated a multi-stage convex relaxation scheme for solving problems with non-convex objective functions. The general algorithmic technique is presented first, which can be applied to a wide range of problems. It unifies a number of earlier approaches. The intuition is to refine convex relaxation iteratively by using solutions obtained from earlier stages. This leads to better and better convex relaxation formulations, and thus better and better solutions.

Although the scheme only finds a local minimum, the above argument indicates that the local minimum it finds should be closer to the original nonconvex problem than the standard convex relaxation solution. In order to prove the effectiveness of this approach theoretically, we considered the sparse learning problem where the behavior of convex relaxation (Lasso) has been well studied in recent years. We showed that under appropriate conditions, the local solution from the multi-stage convex relaxation algorithm is superior to the global solution of the standard L_1 convex relaxation for learning sparse targets. Experiments confirmed the effectiveness of this method.

We shall mention that our theory only shows that nonconvex regularization behaves better than Lasso under appropriate sparse eigenvalue conditions. When such conditions hold, multi-stage convex relaxation is superior. On the other hand, when such conditions fail, neither Lasso nor (the local solution of) multi-stage convex relaxation can be shown to work well. However, in such case, some features will become highly correlated, and local solutions of non-convex formulations may become unstable. In order to improve stability, it may be helpful to employ ensemble methods such as bagging. Our empirical experience suggests that when features are highly correlated, convex formulations may perform better than (non-bagged) nonconvex formulations due to the added sta-

bility. However, since our analysis doesn't yield any insights in this scenario, further investigation is necessary to theoretically compare convex formulations to bagged nonconvex formulations.

Finally, multi-stage convex relaxation is not the only numerical method that can solve nonconvex formulations with strong theoretical guarantee. For example, the MC+ procedure in Zhang (2010) offers a different method with similar guarantee. This opens the possibility of investigating other local solution methods for nonconvex optimization such as modified gradient descent algorithms that may be potentially more efficient.

Appendix A. Proof of Theorem 2

The analysis is an adaptation of Zhang (2009a). We first introduce some definitions. Consider the positive semi-definite matrix $A = n^{-1}X^\top X \in \mathbb{R}^{d\times d}$. Given $s,k \geq 1$ such that $s+k \leq d$. Let I,J be disjoint subsets of $\{1,\ldots,d\}$ with k and s elements respectively. Let $A_{I,I} \in R^{k\times k}$ be the restriction of A to indices I, $A_{I,J} \in R^{k\times s}$ be the restriction of A to indices I on the left and J on the right. Similarly we define restriction $\mathbf{w}_I$ of a vector $\mathbf{w} \in R^d$ on I; and for convenience, we allow either $\mathbf{w}_I \in R^k$ or $\mathbf{w}_I \in R^d$ (where components not in I are zeros) depending on the context.

We also need the following quantity in our analysis:

$$\pi(k,s) = \sup_{\mathbf{v}\in R^k, \mathbf{u}\in R^s, I, J} \frac{\mathbf{v}^\top A_{I,J}\mathbf{u}\|\mathbf{v}\|_2}{\mathbf{v}^\top A_{I,I}\mathbf{v}\|\mathbf{u}\|_\infty}.$$

The following two lemmas are taken from Zhang (2009a). We skip the proof.

Lemma 3 *The following inequality holds:*

$$\pi(k,s) \leq \frac{s^{1/2}}{2}\sqrt{\rho_+(s)/\rho_-(k+s)-1},$$

Lemma 4 *Consider $k,s>0$ and $G \subset \{1,\ldots,d\}$ such that $|G^c| = k$. Given any $\mathbf{w} \in R^d$. Let J be the indices of the s largest components of $\mathbf{w}_G$ (in absolute values), and $I = G^c \cup J$. Then*

$$\max(0, \mathbf{w}_I^\top A\mathbf{w}) \geq \rho_-(k+s)(\|\mathbf{w}_I\|_2 - \pi(k+s,s)\|\mathbf{w}_G\|_1/s)\|\mathbf{w}_I\|_2.$$

The following lemma gives bounds for sub-Gaussian noise needed in our analysis.

Lemma 5 *Define $\hat{\varepsilon} = \frac{1}{n}\sum_{i=1}^n(\bar{\mathbf{w}}^\top\mathbf{x}_i - \mathbf{y}_i)\mathbf{x}_i$. Under the conditions of Assumption 3.1, with probability larger than $1-\eta$:*

$$\|\hat{\varepsilon}\|_\infty^2 \leq 2\sigma^2\rho_+(1)\ln(2d/\eta)/n. \tag{11}$$

Moreover, for any fixed F, with probability larger than $1-\eta$:

$$\|\hat{\varepsilon}_F\|_2^2 \leq \rho_+(|F|)\sigma^2[7.4|F| + 2.7\ln(2/\eta)]/n. \tag{12}$$

Proof The proof relies on two propositions. The first proposition is a simple application of large deviation bound for sub-Gaussian random variables.

Proposition 6 *Consider a fixed vector $\mathbf{u} = [\mathbf{u}_1,\ldots,\mathbf{u}_n] \in \mathbb{R}^n$, and a random vector $\mathbf{y} = [\mathbf{y}_1,\ldots,\mathbf{y}_n] \in \mathbb{R}^n$ with independent sub-Gaussian components: $\mathbf{E}e^{t(\mathbf{y}_i - \mathbf{E}\mathbf{y}_i)} \leq e^{\sigma^2t^2/2}$ for all t and i, then $\forall\varepsilon>0$:*

$$\Pr\left(\left|\mathbf{u}^\top(\mathbf{y}-\mathbf{E}\mathbf{y})\right| \geq \varepsilon\right) \leq 2e^{-\varepsilon^2/(2\sigma^2\|\mathbf{u}\|_2^2)}.$$

Proof (of Proposition 6). Let $s_n = \sum_{i=1}^n \mathbf{u}_i(\mathbf{y}_i - \mathbf{E}\mathbf{y}_i)$; then by assumption, $\mathbf{E}(e^{ts_n} + e^{-ts_n}) \leq 2e^{\sum_i \mathbf{u}_i^2\sigma^2 t^2/2}$, which implies that $\Pr(|s_n| \geq \varepsilon)e^{t\varepsilon} \leq 2e^{\sum_i \mathbf{u}_i^2\sigma^2 t^2/2}$. Now let $t = \varepsilon/(\sum_i \mathbf{u}_i^2\sigma^2)$, we obtain the desired bound. ■

The second proposition is taken from Pisier (1989).

Proposition 7 *Consider the unit sphere $S^{k-1} = \{\mathbf{u} : \|\mathbf{u}\|_2 = 1\}$ in $\mathbb{R}^k$ ($k \geq 1$). Given any $\varepsilon > 0$, there exists an ε-cover $Q \subset S^{k-1}$ such that $\min_{q\in Q} \|\mathbf{u} - q\|_2 \leq \varepsilon$ for all $\|\mathbf{u}\|_2 = 1$, with $|Q| \leq (1+2/\varepsilon)^k$.*

Now are are ready to prove (11). Let $\mathbf{x}_{i,j}$ be the j-th component of $\mathbf{x}_i$, then by definition, we have $\sum_{i=1}^n \mathbf{x}_{i,j}^2 \leq n\rho_+(1)$ for all $j = 1,\ldots,d$. It follows from Proposition 6 that for all $\varepsilon > 0$ and j: $\Pr(|\hat{\varepsilon}_j| \geq \varepsilon) \leq 2e^{-n\varepsilon^2/(2\sigma^2\rho_+(1))}$. Taking union bound for $j = 1,\ldots,d$, we obtain $\Pr(\|\hat{\varepsilon}\|_\infty \geq \varepsilon) \leq 2de^{-n\varepsilon^2/(2\sigma^2\rho_+(1))}$, which is equivalent to (11).

Next we are ready to prove (12). Let P be the projection matrix to the column spanned by X_F, and let k be the dimension of P, then $k \leq |F|$.

According to Proposition 7, given $\varepsilon_1 > 0$, there exists a finite set $Q = \{q_i\}$ with $|Q| \leq (1+2/\varepsilon_1)^k$ such that $\|Pq_i\|_2 = 1$ for all i, and $\min_i \|P\mathbf{z} - Pq_i\|_2 \leq \varepsilon_1$ for all $\|P\mathbf{z}\|_2 = 1$. To see the existence of Q, we consider a rotation of the coordinate system (which does not change 2-norm) so that $P\mathbf{z}$ is the projection of $\mathbf{z} \in \mathbb{R}^n$ to its first k coordinates in the new coordinate system. Proposition 7 can now be directly applied to the first k coordinates in the new system, implying that we can pick q_i such that $Pq_i = q_i$.

For each i, Proposition 6 implies that $\forall \varepsilon_2 > 0$:

$$\Pr\left(\left|q_i^\top P(\mathbf{y} - \mathbf{E}\mathbf{y})\right| \geq \varepsilon_2\right) \leq 2e^{-\varepsilon_2^2/(2\sigma^2)}.$$

Taking union bound for all $q_i \in Q$, we obtain with probability exceeding $1 - 2(1+2/\varepsilon_1)^k e^{-\varepsilon_2^2/2\sigma^2}$:

$$\left|q_i^\top P(\mathbf{y} - \mathbf{E}\mathbf{y})\right| \leq \varepsilon_2$$

for all i.

Let $\mathbf{z} = P(\mathbf{y} - \mathbf{E}\mathbf{y})/\|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2$, then there exists i such that $\|P\mathbf{z} - Pq_i\|_2 \leq \varepsilon_1$. We have

$$\begin{aligned}\|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 =& \mathbf{z}^\top P(\mathbf{y} - \mathbf{E}\mathbf{y}) \\ \leq& \|P\mathbf{z} - Pq_i\|_2 \|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 + |q_i^\top P(\mathbf{y} - \mathbf{E}\mathbf{y})| \\ \leq& \varepsilon_1 \|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 + \varepsilon_2.\end{aligned}$$

Therefore

$$\|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 \leq \varepsilon_2/(1 - \varepsilon_1).$$

Let $\varepsilon_1 = 2/15$, and $\eta = 2(1+2/\varepsilon_1)^k e^{-\varepsilon_2^2/2\sigma^2}$, we have

$$\varepsilon_2^2 = 2\sigma^2[(4k+1)\ln 2 - \ln\eta],$$

and thus

$$\begin{aligned}\rho_+(|F|)^{-1/2}\|\hat{\varepsilon}_F\|_2 =& \rho_+(|F|)^{-1/2}\|X_F^\top(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 \\ \leq& \|P(\mathbf{y} - \mathbf{E}\mathbf{y})\|_2 \leq \frac{15}{13}\sigma\sqrt{2(4k+1)\ln 2 - 2\ln\eta}.\end{aligned}$$

This simplifies to the desired bound.

■

Lemma 8 *Consider $\bar{\mathbf{w}}$ such that $\{j : \bar{\mathbf{w}}_j \neq 0\} \subset F$ and $F \cap G = \emptyset$. Let $\hat{\mathbf{w}} = \hat{\mathbf{w}}^{(\ell)}$ be the solution of (10) with $q = 1$, and let $\Delta\hat{\mathbf{w}} = \hat{\mathbf{w}} - \bar{\mathbf{w}}$. Let $\lambda_G = \min_{j\in G}\lambda_j^{(\ell-1)}$ and $\lambda_0 = \max_j \lambda_j^{(\ell-1)}$. Then*

$$\sum_{j\in G}|\hat{\mathbf{w}}_j| \leq \frac{2\|\hat{\varepsilon}\|_\infty}{\lambda_G - 2\|\hat{\varepsilon}\|_\infty}\sum_{j\notin F\cup G}|\hat{\mathbf{w}}_j| + \frac{2\|\hat{\varepsilon}\|_\infty + \lambda_0}{\lambda_G - 2\|\hat{\varepsilon}\|_\infty}\sum_{j\in F}|\Delta\hat{\mathbf{w}}_j|.$$

Proof For simplicity, let $\lambda_j = \lambda_j^{(\ell-1)}$. The first order equation implies that

$$\frac{1}{n}\sum_{i=1}^n 2(\mathbf{x}_i^\top \mathbf{w} - y_i)\mathbf{x}_{i,j} + \lambda_j \mathrm{sgn}(\mathbf{w}_j) = 0,$$

where $\mathrm{sgn}(\mathbf{w}_j) = 1$ when $\mathbf{w}_j > 0$, $\mathrm{sgn}(\mathbf{w}_j) = -1$ when $\mathbf{w}_j < 0$, and $\mathrm{sgn}(\mathbf{w}_j) \in [-1,1]$ when $\mathbf{w}_j = 0$. This implies that for all $\mathbf{v} \in \mathbb{R}^d$, we have

$$2\mathbf{v}^\top A\Delta\hat{\mathbf{w}} \leq -2\mathbf{v}^\top\hat{\varepsilon} - \sum_{j=1}^d \lambda_j \mathbf{v}_j \mathrm{sgn}(\hat{\mathbf{w}}_j). \tag{13}$$

Now, let $\mathbf{v} = \Delta\hat{\mathbf{w}}$ in (13), we obtain

$$\begin{aligned}
0 \leq & 2\Delta\hat{\mathbf{w}}^\top A\Delta\hat{\mathbf{w}} \leq 2|\Delta\hat{\mathbf{w}}^\top\hat{\varepsilon}| - \sum_{j=1}^d \lambda_j\Delta\hat{\mathbf{w}}_j\mathrm{sgn}(\hat{\mathbf{w}}_j)\\
\leq & 2\|\Delta\hat{\mathbf{w}}\|_1\|\hat{\varepsilon}\|_\infty - \sum_{j\in F}\lambda_j\Delta\hat{\mathbf{w}}_j\mathrm{sgn}(\hat{\mathbf{w}}_j) - \sum_{j\notin F}\lambda_j\Delta\hat{\mathbf{w}}_j\mathrm{sgn}(\hat{\mathbf{w}}_j)\\
\leq & 2\|\Delta\hat{\mathbf{w}}\|_1\|\hat{\varepsilon}\|_\infty + \sum_{j\in F}\lambda_j|\Delta\hat{\mathbf{w}}_j| - \sum_{j\notin F}\lambda_j|\hat{\mathbf{w}}_j|\\
\leq & \sum_{j\in G}(2\|\hat{\varepsilon}\|_\infty - \lambda_G)|\hat{\mathbf{w}}_j| + \sum_{j\notin G\cup F} 2\|\hat{\varepsilon}\|_\infty|\hat{\mathbf{w}}_j| + \sum_{j\in F}(2\|\hat{\varepsilon}\|_\infty + \lambda_0)|\Delta\hat{\mathbf{w}}_j|.
\end{aligned}$$

By rearranging the above inequality, we obtain the desired bound. ■

Lemma 9 *Using the notations of Lemma 8, and let J be the indices of the largest s coefficients (in absolute value) of $\hat{\mathbf{w}}_G$. Let $I = G^c \cup J$ and $k = |G^c|$. If $(\lambda_0 + 2\|\hat{\varepsilon}\|_\infty)/(\lambda_G - 2\|\hat{\varepsilon}\|_\infty) \leq 3$, then*

$$\|\Delta\hat{\mathbf{w}}\|_2 \leq (1 + (3k/s)^{0.5})\|\Delta\hat{\mathbf{w}}_I\|_2.$$

Proof Using $(\lambda_0 + 2\|\hat{\varepsilon}\|_\infty)/(\lambda_G - 2\|\hat{\varepsilon}\|_\infty) \leq 3$, we obtain from Lemma 8

$$\|\hat{\mathbf{w}}_G\|_1 \leq 3\|\Delta\hat{\mathbf{w}} - \hat{\mathbf{w}}_G\|_1.$$

Therefore $\|\Delta\hat{\mathbf{w}} - \Delta\hat{\mathbf{w}}_I\|_\infty \leq \|\Delta\hat{\mathbf{w}}_G\|_1/s \leq 3\|\Delta\hat{\mathbf{w}} - \hat{\mathbf{w}}_G\|_1/s$, which implies that

$$\begin{aligned}
\|\Delta\hat{\mathbf{w}} - \Delta\hat{\mathbf{w}}_I\|_2 \leq & (\|\Delta\hat{\mathbf{w}} - \Delta\hat{\mathbf{w}}_I\|_1\|\Delta\hat{\mathbf{w}} - \Delta\hat{\mathbf{w}}_I\|_\infty)^{1/2}\\
\leq & 3^{1/2}\|\Delta\hat{\mathbf{w}} - \hat{\mathbf{w}}_G\|_1 s^{-1/2} \leq (3k/s)^{1/2}\|\Delta\hat{\mathbf{w}}_I\|_2.
\end{aligned}$$

By rearranging this inequality, we obtain the desired bound. ■

Lemma 10 *Let the conditions of Lemma 8 hold, and let $k = |G^c|$. If $t = 1 - \pi(k+s,s)k^{1/2}s^{-1} > 0$, and $(\lambda_0 + 2\|\hat{\varepsilon}\|_\infty)/(\lambda_G - 2\|\hat{\varepsilon}\|_\infty) \leq (4-t)/(4-3t)$, then*

$$\|\Delta\hat{\mathbf{w}}\|_2 \leq \frac{1+(3k/s)^{0.5}}{t\rho_-(k+s)}\left[2\|\hat{\varepsilon}_{G^c}\|_2 + \left(\sum_{j\in F}(\lambda_j^{(\ell-1)})^2\right)^{1/2}\right].$$

Proof Let J be the indices of the largest s coefficients (in absolute value) of $\hat{\mathbf{w}}_G$, and $I = G^c \cup J$. The conditions of the lemma imply that

$$\begin{aligned}\max(0, \Delta\hat{\mathbf{w}}_I^\top A\Delta\hat{\mathbf{w}}) \geq& \rho_-(k+s)[\|\Delta\hat{\mathbf{w}}_I\|_2 - \pi(k+s,s)\|\hat{\mathbf{w}}_G\|_1/s]\|\Delta\hat{\mathbf{w}}_I\|_2 \\ \geq& \rho_-(k+s)[1-(1-t)(4-t)(4-3t)^{-1}]\|\Delta\hat{\mathbf{w}}_I\|_2^2 \\ \geq& 0.5t\rho_-(k+s)\|\Delta\hat{\mathbf{w}}_I\|_2^2.\end{aligned}$$

In the above derivation, the first inequality is due to Lemma 4; the second inequality is due to the conditions of this lemma plus Lemma 8, which implies that

$$\|\hat{\mathbf{w}}_G\|_1 \leq 2\frac{\|\hat{\varepsilon}\|_\infty + \lambda_0}{\lambda_G - 2\|\hat{\varepsilon}\|_\infty}\|\hat{\mathbf{w}}_{G^c}\|_1 \leq \frac{\|\hat{\varepsilon}\|_\infty + \lambda_0}{\lambda_G - 2\|\hat{\varepsilon}\|_\infty}\sqrt{k}\|\hat{\mathbf{w}}_I\|_2;$$

and the last inequality follows from $1-(1-t)(4-t)(4-3t)^{-1} \geq 0.5t$.

If $\Delta\hat{\mathbf{w}}_I^\top \hat{A}\Delta\hat{\mathbf{w}} \leq 0$, then the above inequality, together with Lemma 9, imply the lemma. Therefore in the following, we can assume that

$$\Delta\hat{\mathbf{w}}_I^\top A\Delta\hat{\mathbf{w}} \geq 0.5t\rho_-(k+s)\|\Delta\hat{\mathbf{w}}_I\|_2^2.$$

Moreover, let $\lambda_j = \lambda_j^{(\ell-1)}$. We obtain from (13) with $\mathbf{v} = \Delta\hat{\mathbf{w}}_I$ the following:

$$\begin{aligned}2\Delta\hat{\mathbf{w}}_I^\top \hat{A}\Delta\hat{\mathbf{w}} \leq& -2\Delta\hat{\mathbf{w}}_I^\top\hat{\varepsilon} - \sum_{j\in I}\lambda_j\Delta\hat{\mathbf{w}}_j\mathrm{sgn}(\hat{\mathbf{w}}_j) \\ \leq& 2\|\Delta\hat{\mathbf{w}}_I\|_2\|\hat{\varepsilon}_{G^c}\|_2 + 2\|\hat{\varepsilon}_G\|_\infty\sum_{j\in G}|\Delta\hat{\mathbf{w}}_j| + \sum_{j\in F}\lambda_j|\Delta\hat{\mathbf{w}}_j| - \sum_{j\in G}\lambda_j|\Delta\hat{\mathbf{w}}_j| \\ \leq& 2\|\Delta\hat{\mathbf{w}}_I\|_2\|\hat{\varepsilon}_{G^c}\|_2 + (\sum_{j\in F}\lambda_j^2)^{1/2}\|\Delta\hat{\mathbf{w}}_I\|_2,\end{aligned}$$

where $\lambda_j \geq \lambda_G \geq 2\|\hat{\varepsilon}_G\|_\infty$ is used to derive the last inequality. Now by combining the above two estimates, we obtain

$$\|\Delta\hat{\mathbf{w}}_I\|_2 \leq \frac{1}{t\rho_-(k+s)}\left[2\|\hat{\varepsilon}_{G^c}\|_2 + (\sum_{j\in F}\lambda_j^2)^{1/2}\right].$$

The desired bound now follows from Lemma 9. ■

Lemma 11 *Consider $g(\cdot)$ that satisfies the conditions of Theorem 2. Let $\lambda_j = \lambda g'(|\tilde{\mathbf{w}}_j|)$ for some $\tilde{\mathbf{w}} \in R^d$, then*

$$\left(\sum_{j\in F}\lambda_j^2\right)^{1/2} \le \lambda\left(\sum_{j\in F} g'(|\bar{\mathbf{w}}_j|-\theta)^2\right)^{1/2} + \lambda\theta^{-1}\left(\sum_{j\in F}|\bar{\mathbf{w}}_j-\tilde{\mathbf{w}}_j|^2\right)^{1/2}.$$

Proof By assumption, if $|\bar{\mathbf{w}}_j - \tilde{\mathbf{w}}_j| \ge \theta$, then

$$g'(|\tilde{\mathbf{w}}_j|) \le 1 \le \theta^{-1}|\bar{\mathbf{w}}_j - \tilde{\mathbf{w}}_j|;$$

otherwise, $g'(|\tilde{\mathbf{w}}_j|) \le g'(|\bar{\mathbf{w}}_j|-\theta)$. It follows that the following inequality always holds:

$$g'(|\tilde{\mathbf{w}}_j|) \le g'(|\bar{\mathbf{w}}_j|-\theta) + \theta^{-1}|\bar{\mathbf{w}}_j - \tilde{\mathbf{w}}_j|.$$

The desired bound is a direct consequence of the above result and the 2-norm triangle inequality $(\sum_j (x_j+\Delta x_j)^2)^{1/2} \le (\sum_j x_j^2)^{1/2} + (\sum_j \Delta x_j^2)^{1/2}$. ■

Lemma 12 *Under the conditions of Theorem 2, we have for all $s \ge 2\bar{k}$:*

$$\|\hat{\mathbf{w}}^{(\ell)} - \bar{\mathbf{w}}\|_2 \le \frac{7}{\rho_-(2\bar{k}+s)}\sqrt{|F|}\lambda.$$

Proof Let $t = 0.5$, then using Lemma 5, the condition of the theorem implies that

$$\frac{\lambda + 2\|\hat{\varepsilon}\|_\infty}{\lambda g'(\theta) - 2\|\hat{\varepsilon}\|_\infty} \le \frac{4-t}{4-3t}.$$

Moreover, Lemma 3 implies that the condition

$$t = 0.5 \le 1 - \pi(2\bar{k}+s, s)(2\bar{k})^{0.5}/s$$

is also satisfied.

Now, if we assume that at some $\ell \ge 1$ that

$$|G_\ell^c| \le 2\bar{k}, \quad \text{where } G_\ell = \{j \notin F : \lambda_j^{(\ell-1)} \ge \lambda g'(\theta)\}, \tag{14}$$

then we can obtain from Lemma 10 that

$$\|\hat{\mathbf{w}}^{(\ell)} - \bar{\mathbf{w}}\|_2 \le \frac{1+\sqrt{3}}{t\rho_-(2\bar{k}+s)}\left[2\sqrt{|G_\ell^c|}\|\hat{\varepsilon}\|_\infty + \sqrt{|F|}\lambda\right] \le \frac{3.2}{t\rho_-(2\bar{k}+s)}\sqrt{|F|}\lambda,$$

where we have used the fact that $|G_\ell^c| \le 2\bar{k} \le 2|F|$ and $\lambda \ge 20\|\hat{\varepsilon}\|_\infty$ in the derivation of the second inequality. This shows that (14) implies the lemma.

Therefore next we only need to prove by induction on ℓ that (14) holds for all $\ell = 1, 2, \ldots$. When $\ell = 1$, we have $G_1 = F^c$, which implies that (14) holds.

Now assume that (14) holds at $\ell-1$ for some $\ell>1$. Since $j\in G_\ell^c-F$ implies that $j\notin F$ and $\lambda g'(|\hat{\mathbf{w}}_j^{(\ell-1)}|)=\lambda_j^{(\ell)}<\lambda g'(\theta)$ by definition, and since $g'(z)$ is non-increasing when $z\geq 0$ (theorem assumption), we know that $|\hat{\mathbf{w}}_j^{(\ell-1)}|\geq\theta$. Therefore by induction hypothesis we obtain

$$\begin{aligned}\sqrt{|G_\ell^c-F|}\leq&\sqrt{\sum_{j\in G_\ell^c-F}|\hat{\mathbf{w}}_j^{(\ell-1)}|^2/\theta^2}\leq\frac{\|\hat{\mathbf{w}}^{(\ell-1)}-\bar{\mathbf{w}}\|_2}{\theta}\\ \leq&\frac{7\lambda}{\rho_-(2\bar{k}+s)\theta}\sqrt{|F|}\leq\sqrt{|F|},\end{aligned}$$

where the second to the last inequality is due to the fact that (14) implies the lemma at $\ell-1$. The last inequality uses the definition of θ in the theorem. This inequality implies that $|G_\ell^c|\leq 2|F|\leq 2\bar{k}$, which completes the induction step. ■

A.1 Proof of Theorem 2

As in the proof of Lemma 12, if we let $t=0.5$, then using Lemma 5, the condition of the theorem implies that

$$\frac{\lambda+2\|\hat{\varepsilon}\|_\infty}{\lambda g'(\theta)-2\|\hat{\varepsilon}\|_\infty}\leq\frac{4-t}{4-3t}.$$

Moreover, Lemma 3 implies that the condition

$$t=0.5\leq 1-\pi(2\bar{k}+s,s)(2\bar{k})^{0.5}/s$$

is also satisfied.

We prove by induction: for $\ell=1$, the result follows from Lemma 12. For $\ell>1$, we let $G^c=F\cup\{j:|\hat{\mathbf{w}}_j^{(\ell-1)}|\geq\theta\}$. From the proof of Lemma 12, we know that

$$k=|G^c|\leq 2\bar{k}.$$

Let $u=\sqrt{\rho_+(\bar{k})}\sigma[\sqrt{7.4\bar{k}/n}+\sqrt{2.7\ln(2/\eta)/n}]$. We know from Lemma 5, and $\lambda\geq 20\|\hat{\varepsilon}\|_\infty$ that with probability $1-2\eta$,

$$\begin{aligned}\|\hat{\varepsilon}_{G^c}\|_2\leq&\|\hat{\varepsilon}_F\|_2+\sqrt{|G^c-F|}\|\hat{\varepsilon}\|_\infty\\ \leq&u+\sqrt{|G^c-F|}\lambda/20\\ \leq&u+(\lambda/20)\sqrt{\sum_{j\in G^c-F}|\hat{\mathbf{w}}_j^{(\ell-1)}|^2/\theta^2}\\ \leq&u+\lambda(20\theta)^{-1}\|\hat{\mathbf{w}}^{(\ell-1)}-\bar{\mathbf{w}}\|_2.\end{aligned}$$

Now, using Lemma 10 and Lemma 11, we obtain

$$\begin{aligned}
&\|\Delta\hat{\mathbf{w}}^{(\ell)}\|_2 \\
\leq&\frac{1+\sqrt{3}}{t\rho_-(k+s)}\left[2\|\hat{\varepsilon}_{G^c}\|_2+(\sum_{j\in F}(\lambda_j^{(\ell-1)})^2)^{1/2}\right] \\
\leq&\frac{1+\sqrt{3}}{t\rho_-(k+s)}\left[2\|\hat{\varepsilon}_{G^c}\|_2+\lambda\left(\sum_{j\in F}g'(|\bar{\mathbf{w}}_j|-\theta)^2\right)^{1/2}+\lambda\theta^{-1}\left(\sum_{j\in F}|\bar{\mathbf{w}}_j-\hat{\mathbf{w}}_j^{(\ell-1)}|^2\right)^{1/2}\right] \\
\leq&\frac{1+\sqrt{3}}{t\rho_-(k+s)}\left[2u+\lambda\left(\sum_{j\in F}g'(|\bar{\mathbf{w}}_j|-\theta)^2\right)^{1/2}+1.1\lambda\theta^{-1}\|\bar{\mathbf{w}}-\hat{\mathbf{w}}^{(\ell-1)}\|_2\right] \\
\leq&\frac{1+\sqrt{3}}{t\rho_-(k+s)}\left[2u+\lambda\left(\sum_{j\in F}g'(|\bar{\mathbf{w}}_j|-\theta)^2\right)^{1/2}\right]+0.67\|\bar{\mathbf{w}}-\hat{\mathbf{w}}^{(\ell-1)}\|_2.
\end{aligned}$$

The desired bound can now be obtained by solving this recursion with respect to

$$\|\Delta\hat{\mathbf{w}}^{(\ell)}\|_2=\|\bar{\mathbf{w}}-\hat{\mathbf{w}}^{(\ell)}\|_2$$

for $\ell=2,3,\ldots$, where $\|\Delta\hat{\mathbf{w}}^{(1)}\|_2$ is given by Lemma 12.

Appendix B. Some Non-convex Formulations in Machine Learning

Consider a set of input vectors $\mathbf{x}_1,\ldots,\mathbf{x}_n\in R^d$, with corresponding desired output variables $y_1,\ldots,y_n$. The task of supervised learning is to estimate the functional relationship $y\approx f(\mathbf{x})$ between the input $\mathbf{x}$ and the output variable y from the training examples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n)\}$. The quality of prediction is often measured through a loss function $\phi(f(\mathbf{x}),y)$.

Now, consider linear prediction model $f(\mathbf{x})=\mathbf{w}^\top\mathbf{x}$. As in boosting or kernel methods, nonlinearity can be introduced by including nonlinear features in $\mathbf{x}$. For linear models, we are mainly interested in the scenario that $d\gg n$. That is, there are many more features than the number of samples. In this case, an unconstrained empirical risk minimization is inadequate because the solution overfits the data. The standard remedy for this problem is to impose a constraint on $\mathbf{w}$ to obtain a *regularized* problem. This leads to the following regularized empirical risk minimization method:

$$\hat{\mathbf{w}}=\arg\min_{\mathbf{w}\in R^d}\left[\sum_{i=1}^n\phi(\mathbf{w}^\top\mathbf{x}_i,y_i)+\lambda g(\mathbf{w})\right], \tag{15}$$

where $\lambda>0$ is an appropriately chosen regularization condition. This is the motivation for the general problem formulation (1) in Section 2.

B.1 Loss Function

Examples of loss function $\phi(\mathbf{w}^\top\mathbf{x},y)$ in (15) include least squares for regression: $\phi(\mathbf{w}^\top\mathbf{x},y)=(\mathbf{w}^\top\mathbf{x}-y)^2$, and 0-1 binary classification error: $\phi(\mathbf{w}^\top\mathbf{x},y)=I(\mathbf{w}^\top\mathbf{x}y\leq 0)$, where $y\in\{\pm 1\}$ are the class labels, and $I(\cdot)$ is the set indicator function. The latter is nonconvex. In practice, for computational reasons, a convex relaxation such as the SVM loss $\phi(\mathbf{w}^\top\mathbf{x},y)=\max(0,1-\mathbf{w}^\top\mathbf{x}y)$

is often used to substitute the classification error loss. Such a convex loss is often referred to as a surrogate loss function, and the resulting method becomes a convex relaxation method for solving binary classification. This class of methods have been theoretically analyzed in Bartlett et al. (2006) and Zhang (2004). While asymptotically, convex surrogate methods are consistent (that is, they can be used to obtain Bayes optimal classifiers when the sample size approaches infinity), for finite data, these methods can be more sensitive to outliers. In order to alleviate the effect of outliers, one may consider the smoothed classification error loss function $\phi(\mathbf{w}^\top \mathbf{x}, y) = \min(\alpha, \max(0, 1 - \mathbf{w}^\top \mathbf{x} y))$ ($\alpha \geq 1$). This loss function is bounded, and thus more robust to outliers than SVMs under finite sample size; moreover, it is piece-wise differentiable, and thus easier to handle than the discontinuous classification error loss. For comparison purpose, the three loss functions are plotted in Figure 7.

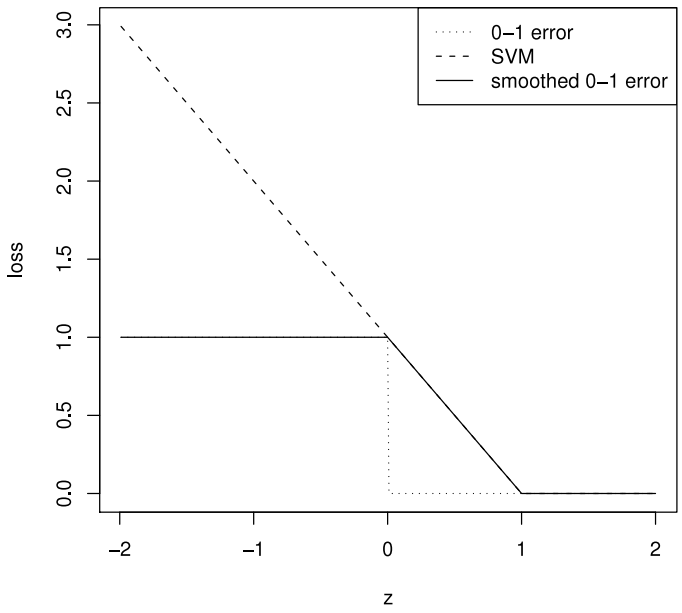


Figure 7: Loss Functions: classification error versus smoothed classification error ($\alpha = 1$) and SVM

B.2 Regularization Condition

Some examples of regularization conditions in (15) include squared regularization $g(\mathbf{w}) = \mathbf{w}^\top \mathbf{w}$, and 1-norm regularization $g(\mathbf{w}) = \|\mathbf{w}\|_1$. The former can be generalized to kernel methods, and the latter leads to sparse solutions. Sparsity is an important regularization condition, which corresponds to the (non-convex) L_0 regularization, defined as $\|\mathbf{w}\|_0 = |\{j : \mathbf{w}_j \neq 0\}| = k$. That is, the measure of complexity is the number of none zero coefficients. If we know the sparsity parameter k for the target vector, then a good learning method is L_0 regularization:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in R^d} \frac{1}{n} \sum_{i=1}^{n} \phi(\mathbf{w}^\top \mathbf{x}_i, y_i) \quad \text{subject to } \|\mathbf{w}\|_0 \leq k, \tag{16}$$

which applies the standard empirical risk minimization formulation to learning L_0 constrained sparse targets.

If k is not known, then one may regard k as a tuning parameter, which can be selected through cross-validation. This method is often referred to as *subset selection* in the literature. Sparse learning is an essential topic in machine learning, which has attracted considerable interests recently. It can be shown that the solution of the L_0 regularization problem in (16) achieves good prediction accuracy if the target function can be approximated by a sparse $\bar{\mathbf{w}}$. However, a fundamental difficulty with this method is the computational cost, because the number of subsets of $\{1,\ldots,d\}$ of cardinality k (corresponding to the nonzero components of $\mathbf{w}$) is exponential in k.

Due to the computational difficult, in practice, it is necessary to replace (16) by some easier to solve formulations in (15). Specifically, L_0 regularization is equivalent to (15) by choosing the regularization function as $g(\mathbf{w}) = \|\mathbf{w}\|_0$. However, this function is discontinuous. For computational reasons, it is helpful to consider a continuous approximation with $g(\mathbf{w}) = \|\mathbf{w}\|_p^p$, where $p > 0$. If $p \geq 1$, the resulting formulation is convex. In particular, by choosing the closest approximation with $p = 1$, one obtain *Lasso*, which is the standard convex relaxation formulation for sparse learning. With $p \in (0,1)$, the L_p regularizer $\|\mathbf{w}\|_p^p$ is non-convex but continuous.

Supervised learning can be solved using general empirical risk minimization formulation in (15). Both ϕ and g can be non-convex in application problems. The traditional approach is to use convex relaxation to approximate it, leading to a single stage convex formulation. In this paper, we try to extend the idea by looking at a more general multi-stage convex relaxation method, which leads to more accurate approximations.

For illustration, we consider the following examples which will be used in our later discussion.

- Smoothed classification error loss: formulation (15) with convex regularization $g(\mathbf{w})$ and nonconvex loss function (with $\alpha \geq 1$)

$$\phi(\mathbf{w}^\top \mathbf{x}, y) = \min(\alpha, \max(0, 1 - \mathbf{w}^\top \mathbf{x} y)).$$

  This corresponds to $R_0(\mathbf{w}) = \lambda g(\mathbf{w})$, and $R_k(\mathbf{w}) = \phi(\hat{\mathbf{w}}^\top \mathbf{x}_k, y_k)$ for $k = 1,\ldots,n$ in (1).

- L_p regularization ($0 \leq p \leq 1$): formulation (15) with nonconvex regularization $g(\mathbf{w}) = \|\mathbf{w}\|_p^p$ and a loss function $\phi(\cdot,\cdot)$ that is convex in $\mathbf{w}$. This corresponds to $R_0(\mathbf{w}) = n^{-1}\sum_{i=1}^n \phi(\mathbf{w}^\top \mathbf{x}_i, y_i)$, and $R_k(\mathbf{w}) = \lambda |\mathbf{w}_k|^p$ for $k = 1,\ldots,d$ in (1).

- Smoothed L_p regularization (with parameters $\alpha > 0$ and $0 \leq p \leq 1$): formulation (15) with nonconvex regularization $g(\mathbf{w}) = \sum_k [(\alpha + |\mathbf{w}_k|)^p - \alpha^p]/(p\alpha^{p-1})$, and a loss function $\phi(\cdot,\cdot)$ that is convex in $\mathbf{w}$. This corresponds to $R_0(\mathbf{w}) = n^{-1}\sum_{i=1}^n \phi(\mathbf{w}^\top \mathbf{x}_i, y_i)$, and $R_k(\mathbf{w}) = \lambda[(\alpha + |\mathbf{w}_k|)^p - \alpha^p]/(p\alpha^{p-1})$ for $k = 1,\ldots,d$ in (1). The main difference between standard L_p and smoothed L_p is at $|\mathbf{w}_k| = 0$, where the smoothed L_p regularization is differentiable, with derivative 1. This difference is theoretically important as discussed in Section 3.1.

- Smoothed log regularization (with parameter $\alpha > 0$): formulation (15) with nonconvex regularization $g(\mathbf{w}) = \sum_k \alpha \ln(\alpha + |\mathbf{w}_k|)$, and a loss function $\phi(\cdot,\cdot)$ that is convex in $\mathbf{w}$. This corresponds to $R_0(\mathbf{w}) = n^{-1}\sum_{i=1}^n \phi(\mathbf{w}^\top \mathbf{x}_i, y_i)$, and $R_k(\mathbf{w}) = \lambda\alpha \ln(\alpha + |\mathbf{w}_k|)$ for $k = 1,\ldots,d$ in (1). Similar to the smoothed L_p regularization, the smoothed log-loss has derivative 1 at $|\mathbf{w}_k| = 0$.

- Capped-L_1 regularization (with parameter $\alpha > 0$): formulation (15) with nonconvex regularization $g(\mathbf{w}) = \sum_{j=1}^{d} \min(|\mathbf{w}_j|, \alpha)$, and a loss function $\phi(\cdot,\cdot)$ that is convex in $\mathbf{w}$. This corresponds to $R_0(\mathbf{w}) = n^{-1}\sum_{i=1}^{n}\phi(\mathbf{w}^\top \mathbf{x}_i, y_i)$, and $R_k(\mathbf{w}) = \lambda \min(|\mathbf{w}_k|, \alpha)$ for $k = 1,\ldots,d$ in (1). The capped-L_1 regularization is a good approximation to L_0 because as $\alpha \to 0$, $\sum_k \min(|\mathbf{w}_k|,\alpha)/\alpha \to \|\mathbf{w}\|_0$. Therefore when $\alpha \to 0$, this regularization condition is equivalent to the sparse L_0 regularization up to a rescaling of λ. Capped-L_1 regularization is a simpler but less smooth version of the SCAD regularization (Jianqing Fan, 2001). SCAD is more complicated, but its advantage cannot be shown through our analysis.

Appendix C. Some Examples of Multi-stage Convex Relaxation Methods

The multi-stage convex relaxation method can be used with examples in Section 2.2 to obtain concrete algorithms for various formulations. We describe some examples here.

C.1 Smoothed Classification Loss

We consider a loss term of the form $R_k(\mathbf{w}) = \min(\alpha, \max(0, 1 - \mathbf{w}^\top \mathbf{x}_k y_k))$ for $k = 1,\ldots,n$ (with $\alpha \geq 1$), and relax it to the SVM loss $\mathbf{h}_k(\mathbf{w}) = \max(0, 1 - \mathbf{w}^\top \mathbf{x}_k y_k)$.

The optimization problem is

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{n} \min(\alpha, \max(0, 1 - \mathbf{w}^\top \mathbf{x}_i y_i)) + \lambda g(\mathbf{w}) \right],$$

where we assume that $g(\mathbf{w})$ is a convex regularization condition such as $g(\mathbf{w}) = \lambda\|\mathbf{w}\|_2^2$.

Consider concave duality in Section 2.2. Each $\mathbf{u}_k$ is a scalar in the range $\Omega_k = [0,\infty)$, and $\bar{R}_k(\mathbf{u}_k) = \min(\alpha, \mathbf{u}_k)$. We have $R_k^*(\mathbf{v}_k) = \alpha(\mathbf{v}_k - 1)I(\mathbf{v}_k \in [0,1])$, defined on the domain $\mathbf{v}_k \geq 0$. The solution in (4) is given by $\hat{\mathbf{v}}_k = I(\mathbf{w}^\top \mathbf{x}_k y_k \geq 1 - \alpha)$ for $k = 1,\ldots,n$. Therefore Section 2.2 implies that the multi-stage convex relaxation solves the weighted SVM formulation

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{n} \hat{\mathbf{v}}_i \max(0, 1 - \mathbf{w}^\top \mathbf{x}_i y_i) + \lambda g(\mathbf{w}) \right],$$

where the relaxation parameter $\mathbf{v}$ is updated as

$$\hat{\mathbf{v}}_i = I(\hat{\mathbf{w}}^\top \mathbf{x}_i y_i \geq 1 - \alpha) \quad (i = 1,\ldots,n).$$

Intuitively, the mis-classified points $\hat{\mathbf{w}}^\top \mathbf{x}_i y_i < 1 - \alpha$ are considered as outliers, and ignored.

C.2 L_p and Smoothed L_p Regularization

In sparse regularization, we may consider a regularization term $R_k(\mathbf{w}) = \lambda|\mathbf{w}_k|^p/p$ ($k = 1,\ldots,d$) for some $p \in (0,1)$. Given any $q > p$, (3) holds with $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = |\mathbf{w}_k|^q \in [0,\infty)$, and $\bar{R}_k(\mathbf{u}_k) = \lambda|\mathbf{u}_k|^{p/q}/p$, where $\mathbf{u}_k \in \Omega_k = [0,\infty)$. The dual is $R_k^*(\mathbf{v}_k) = -\lambda c(p,q)(\mathbf{v}_k/\lambda)^{p/(p-q)}$, defined on the domain $\mathbf{v}_k \geq 0$, where $c(p,q) = (q-p)p^{-1}q^{q/(p-q)}$. The solution in (4) is given by $\hat{\mathbf{v}}_k = (\lambda/q)|\mathbf{w}_k|^{p-q}$.

An extension is to consider a regularization term $R_k(\mathbf{w}) = \lambda[(\alpha + |\mathbf{w}_k|)^p - \alpha^p]/(p\alpha^{p-1})$ ($k = 1,\ldots,d$) for some $p \in (0,1)$ and $\alpha > 0$. Given any $q > p$, (3) holds with $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = (\alpha +$

$|\mathbf{w}_k|)^q \in [\alpha^q, \infty)$, and $\bar{R}_k(\mathbf{u}_k) = \lambda[\mathbf{u}_k^{p/q} - \alpha^p]/(p\alpha^{p-1})$, where $\mathbf{u}_k \in \Omega_k = [0,\infty)$. The dual is $R_k^*(\mathbf{v}_k) = -\lambda c(p,q)\alpha^{p/(p-q)}(\mathbf{v}_k/\lambda)^{p/(p-q)} + \lambda\alpha/p$, defined on the domain $\mathbf{v}_k \geq 0$, where $c(p,q) = (q-p)p^{p/(q-p)}q^{q/(p-q)}$. The solution in (4) is given by $\hat{\mathbf{v}}_k = \lambda/(q\alpha^{p-1})(\alpha + |\mathbf{w}_k|)^{p-q}$.

An alternative is to relax smoothed L_p regularization ($p \in (0,1)$) directly to L_q regularization for $q \geq 1$ (one usually takes either $q = 1$ or $q = 2$). In this case, $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = |\mathbf{w}_k|^q \in [0,\infty)$, and $\bar{R}_k(\mathbf{u}_k) = \lambda[(\alpha + \mathbf{u}_k^{1/q})^p - \alpha^p]/(p\alpha^{p-1})$. Although it is not difficult to verify that $\bar{R}_k(\mathbf{u}_k)$ is concave, we do not have a simple closed form for $R_k^*(\mathbf{v}_k)$. However, it is easy to check that the solution in (4) is given by $\hat{\mathbf{v}}_k = \lambda/(q\alpha^{p-1})(\alpha + |\mathbf{w}_k|)^{p-1}|\mathbf{w}_k|^{1-q}$.

In summary, for L_p and smoothed L_p, we consider the following optimization formulation for some $\alpha \geq 0$ and $p \in (0,1]$:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \lambda \sum_{j=1}^{d} (\alpha + |\mathbf{w}_j|)^p \right],$$

where we assume that $R_0(\mathbf{w})$ is a convex function of $\mathbf{w}$.

From previous discussion, the multi-stage convex relaxation method in Section 2.2 becomes a weighted L_q regularization formulation for $q \geq 1$:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \sum_{j=1}^{d} \hat{\mathbf{v}}_j |\mathbf{w}_j|^q \right],$$

where the relaxation parameter $\mathbf{v}$ is updated as

$$\hat{\mathbf{v}}_j = \lambda(p/q)(\alpha + |\hat{\mathbf{w}}_j|)^{p-1}|\hat{\mathbf{w}}_j|^{1-q} \quad (j = 1,\ldots,d).$$

The typical choices of q are $q = 1$ or $q = 2$. That is, we relax L_p regularization to L_1 or L_2 regularization.

Finally, we note that the two stage version of L_p regularization, relaxed to L_q with $q = 1$, is referred to Adaptive-Lasso (Zou, 2006).

C.3 Smoothed log Regularization

This is a different sparse regularization condition, where we consider a regularization term $R_k(\mathbf{w}) = \lambda\alpha\ln(\alpha + |\mathbf{w}_k|)$ for some $\alpha > 0$. Given any $q > 0$, (3) holds with $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = (\alpha + |\mathbf{w}_k|)^q \in [\alpha^q, \infty)$, and $\bar{R}_k(\mathbf{u}_k) = \lambda(\alpha/q)\ln(\mathbf{u}_k)$, where $\mathbf{u}_k \in \Omega_k = [0,\infty)$. The dual is $R_k^*(\mathbf{v}_k) = \lambda(\alpha/q)[\ln\mathbf{v}_k + 1 - \ln(\lambda\alpha/q)]$, defined on the domain $\mathbf{v}_k \geq 0$. The solution in (4) is given by $\hat{\mathbf{v}}_k = \lambda(\alpha/q)(\alpha + |\mathbf{w}_k|)^{-q}$.

Similar to smoothed L_p, we may relax directly to L_q, with $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = |\mathbf{w}_k|^q \in [0,\infty)$. $\bar{R}_k(\mathbf{u}_k) = \lambda\alpha\ln(\alpha + \mathbf{u}_k^{1/q})$, where The solution in (4) is given by $\hat{\mathbf{v}}_k = \lambda(\alpha/q)(\alpha + |\mathbf{w}_k|)^{-1}|\mathbf{w}_k|^{1-q}$.

Similar to smoothed log regularization, the multi-stage convex relaxation method in Section 2.2 becomes a weighted L_q regularization formulation for $q \geq 1$:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \sum_{j=1}^{d} \hat{\mathbf{v}}_j |\mathbf{w}_j|^q \right],$$

where the relaxation parameter $\mathbf{v}$ is updated as

$$\hat{\mathbf{v}}_j = \lambda(\alpha/q)(\alpha + |\mathbf{w}_j|)^{-1}|\mathbf{w}_j|^{1-q} \quad (j = 1,\ldots,d).$$

This resulting procedure is the same as the one empirically studied in Candes et al. (2008).

C.3.1 CAPPED L_1 REGULARIZATION

We consider another sparse regularization term with $R_k(\mathbf{w}) = \lambda \min(|\mathbf{w}_k|, \alpha)$ ($k = 1, \ldots, d$) for some $\alpha > 0$. In this case, (3) holds with $\mathbf{u}_k = \mathbf{h}_k(\mathbf{w}) = |\mathbf{w}_k| \in [0, \infty)$, and $\bar{R}_k(\mathbf{u}_k) = \lambda \min(\mathbf{u}_k, \alpha)$, where $\mathbf{u}_k \in \Omega_k = [0, \infty)$. The dual is $R_k^*(\mathbf{v}_k) = \lambda\alpha(-1 + \mathbf{v}_k/\lambda) I(\mathbf{v}_k \in [0, \lambda])$ defined $[0, \infty)$, where $I(\cdot)$ is the set indicator function. The solution in (4) is given by $\hat{\mathbf{v}}_k = \lambda I(|\mathbf{w}_k| \leq \alpha)$.

In capped L_1 regularization, we consider the optimization problem

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \lambda \sum_{j=1}^{d} \min(\alpha, |\mathbf{w}_j|) \right],$$

where we assume that $R_0(\mathbf{w})$ is a convex function of $\mathbf{w}$.

From Section 2.2, the multi-stage convex relaxation becomes a weighted L_1 regularization formulation:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left[R_0(\mathbf{w}) + \sum_{j=1}^{d} \hat{\mathbf{v}}_j |\mathbf{w}_j| \right],$$

where the relaxation parameter $\mathbf{v}$ is updated as

$$\hat{\mathbf{v}}_j = \lambda I(|\hat{\mathbf{w}}_j| \leq \alpha) \quad (j = 1, \ldots, d).$$

This method has an intuitive interpretation: in order to achieve sparsity, the standard L_1 regularization not only shrinks small coefficients to zero, but also shrinks large coefficients. This causes a bias. The capped-L_1 formulation removes the bias by adaptively adjusting the relaxation parameter $\hat{\mathbf{v}}_j$ so that if $|\hat{\mathbf{w}}_j|$ is large, then we do not penalize the corresponding variable j.

C.3.2 SPARSE EIGENVALUE PROBLEM

We use a simple example to illustrate that the multi-stage convex relaxation idea does not only apply to formulations with convex risks. Consider the sparse eigenvalue problem, where we are interested in finding the largest eigenvalue of a positive semi-definite matrix A. One formulation is

$$\hat{\mathbf{w}} = \arg\max_{\|\mathbf{w}\|_2 \leq 1} \left[\mathbf{w}^\top A \mathbf{w} - \lambda \sum_{j=1}^{d} (\alpha + |\mathbf{w}_j|)^p \right],$$

with parameter $p \in (0, 1)$ and a small parameter $\alpha > 0$ to encourage sparsity. If $\lambda = 0$, then it is the standard eigenvalue problem without sparsity constraints. Although the standard eigenvalue problem is not convex in $\mathbf{w}$, it has a convex relaxation to a semi-definite programming problem, and thus can be efficiently solved. For convenience, we think of the standard eigenvalue problem as "convex" for the purpose of this paper. The multi-stage convex relaxation becomes:

$$\hat{\mathbf{w}} = \arg\max_{\|\mathbf{w}\|_2 \leq 1} \left[\mathbf{w}^\top A \mathbf{w} - \sum_{j=1}^{d} \mathbf{v}_j \mathbf{w}_j^2 \right],$$

which is a standard eigenvalue problem. The relaxation parameter is updated as

$$\hat{\mathbf{v}}_j = \lambda(p/2)(\alpha + |\hat{\mathbf{w}}_j|)^{p-1} |\hat{\mathbf{w}}_j|^{-1} \quad (j = 1, \ldots, d).$$

C.3.3 MATRIX REGULARIZATION

Our final example is multi-task learning with matrix regularization, also considered in Argyriou et al. (2008). In this case, $\mathbf{w}$ is not a vector, but a matrix, with columns (tasks) $\mathbf{w}^\ell$. We solve a problem of the following form:

$$\mathbf{w} = \arg\min_{\mathbf{w}} \left[\sum_{\ell=1}^{m} R^\ell(\mathbf{w}^\ell) + \lambda \mathrm{tr}((\alpha I + \mathbf{w}\mathbf{w}^\top)^{p/2}) \right].$$

In the above formulation, R^ℓ is the risk function for task ℓ. The matrix regularization used here is the counterpart of L_p regularization for vectors. It encourages low-rank if $p < 2$. In particular, the case of $p = 1$ is often called trace norm (or nuclear norm). It is convex and frequently used in the literature. The parameter $\alpha > 0$ gives some smoothness, similar to the vector case.

The case of $p < 1$ gives better low-rank approximation, similar to the vector regularization case. Again, this problem can be solved with multi-stage convex relaxation method. In this case, the relaxation parameter $\mathbf{v}$ is a positive semi-definite matrix, and we relax the regularization term to $\mathbf{h}(\mathbf{w}) = (\alpha I + \mathbf{w}\mathbf{w}^\top)$ as a matrix. Thus the relaxed regularization term becomes $\mathrm{tr}(\mathbf{v}(\alpha I + \mathbf{w}\mathbf{w}^\top))$. This regularization decouples the problems as follows, which allows us to solve each task ℓ separately:

$$\hat{\mathbf{w}}^\ell = \arg\min_{\mathbf{w}^\ell} \left[R^\ell(\mathbf{w}^\ell) + (\mathbf{w}^\ell)^\top \hat{\mathbf{v}} \mathbf{w}^\ell \right] \qquad (\ell = 1, 2, \ldots, m).$$

This is a key advantage of the method. Similar to the vector case, we have the following update formula for the relaxation parameter:

$$\hat{\mathbf{v}} = \lambda(p/2)(\alpha I + \hat{\mathbf{w}}\hat{\mathbf{w}}^\top)^{(p-2)/2}.$$

References

Andreas Argyriou, Charles A. Micchelli, Massimiliano Pontil, and Yiming Ying. A spectral regularization framework for multi-task structure learning. In *NIPS'07*, 2008.

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Peter Bickel, Yaacov Ritov, and Alexandre Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

Florentina Bunea, Alexandre Tsybakov, and Marten H. Wegkamp. Sparsity oracle inequalities for the Lasso. *Electronic Journal of Statistics*, 1:169–194, 2007.

Emmanuel J. Candes and Terence Tao. Decoding by linear programming. *IEEE Trans. on Information Theory*, 51:4203–4215, 2005.

Emmanuel J. Candes, Michael B. Wakin, and Stephen P. Boyd. Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

Runze Li Jianqing Fan. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2001.

Vladimir Koltchinskii. Sparsity in penalized empirical risk minimization. *Annales de l'Institut Henri Poincaré*, 2008.

Gilles Pisier. *The Volume of Convex Bodies and Banach Space Geometry*. Cambridge University Press, 1989.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, 1970.

Alan L. Yuille and Anand Rangarajan. The concave-convex procedure. *Neural Computation*, 15: 915–936, 2003.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 2010. to appear.

Cun-Hui Zhang and Jian Huang. The sparsity and bias of the Lasso selection in high-dimensional linear regression. *Annals of Statistics*, 36(4):1567–1594, 2008.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 32:56–85, 2004. with discussion.

Tong Zhang. Some sharp performance bounds for least squares regression with L_1 regularization. *Ann. Statist.*, 37(5A):2109–2144, 2009a. ISSN 0090-5364. doi: 10.1214/08-AOS659.

Tong Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. In *NIPS'08*, 2009b.

Peng Zhao and Bin Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2567, 2006.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

Hui Zou and Runze Li. One-step sparse estimates in nonconcave penalized likelihood models. *Annals of Statistics*, 36(4):1509–1533, 2008.

Journal of Machine Learning Research 11 (2010) 1109-1135 Submitted 2/09; Revised 9/09; Published 3/10

Large Scale Online Learning of Image Similarity Through Ranking

Gal Chechik GAL@GOOGLE.COM
Google
1600 Amphitheatre Parkway
Mountain View CA, 94043

Varun Sharma* VASHARMA@GOOGLE.COM
Google, RMZ Infinity
Old Madras Road, Bengalooru
Karnataka 560016, India

Uri Shalit*† URI.SHALIT@MAIL.HUJI.AC.IL
The Gonda Brain Research Center
Bar Ilan University
52900, Israel

Samy Bengio BENGIO@GOOGLE.COM
Google
1600 Amphitheatre Parkway
Mountain View CA, 94043

Editor: Soeren Sonnenburg, Vojtech Franc, Elad Yom-Tov, Michele Sebag

Abstract

Learning a measure of similarity between pairs of objects is an important generic problem in machine learning. It is particularly useful in large scale applications like searching for an image that is similar to a given image or finding videos that are relevant to a given video. In these tasks, users look for objects that are not only visually similar but also semantically related to a given object. Unfortunately, the approaches that exist today for learning such semantic similarity do not scale to large data sets. This is both because typically their CPU and storage requirements grow quadratically with the sample size, and because many methods impose complex positivity constraints on the space of learned similarity functions.

The current paper presents OASIS, an *Online Algorithm for Scalable Image Similarity* learning that learns a bilinear similarity measure over sparse representations. OASIS is an online dual approach using the passive-aggressive family of learning algorithms with a large margin criterion and an efficient hinge loss cost. Our experiments show that OASIS is both fast and accurate at a wide range of scales: for a data set with thousands of images, it achieves better results than existing state-of-the-art methods, while being an order of magnitude faster. For large, web scale, data sets, OASIS can be trained on more than two million images from 150K text queries within 3 days on a single CPU. On this large scale data set, human evaluations showed that 35% of the ten nearest neighbors of a given test image, as found by OASIS, were semantically relevant to that image. This suggests that query independent similarity could be accurately learned even for large scale data sets that could not be handled before.

Keywords: large scale, metric learning, image similarity, online learning

∗. Varun Sharma and Uri Shalit contributed equally to this work.
†. Also at ICNC, The Hebrew University of Jerusalem, 91904, Israel.

1. Introduction

Large scale learning is sometimes defined as the regime where learning is limited by computational resources rather than by availability of data (Bottou, 2008). Learning a pairwise similarity measure is a particularly challenging large scale task: since pairs of samples have to be considered, the large scale regime is reached even for fairly small data sets, and learning similarity for large data sets becomes exceptionally hard to handle.

At the same time, similarity learning is a well studied problem with multiple real world applications. It is particularly useful for applications that aim to discover new and relevant data for a user. For instance, a user browsing a photo in her album may ask to find similar or related images. Another user may search for additional data while viewing an online video or browsing text documents. In all these applications, similarity could have different flavors: a user may search for images that are similar visually, or semantically, or anywhere in between.

Many similarity learning algorithms assume that the available training data contains real-valued pairwise similarities or distances. However, in all the above examples, the precise numerical value of pairwise similarity between objects is usually not available. Fortunately, one can often obtain information about the *relative* similarity of different pairs (Frome et al., 2007), for instance, by presenting people with several object pairs and asking them to select the pair that is most similar. For large scale data, where man-in-the-loop experiments are prohibitively costly, relative similarities can be extracted from analyzing pairs of images that are returned in response to the same text query (Schultz and Joachims, 2004). For instance, the images that are ranked highly by one of the image search engines for the query "cute kitty" are likely to be semantically more similar than a random pair of images. The current paper focuses on this setting: similarity information is extracted from pairs of images that share a common label or are retrieved in response to a common text query.

Similarity learning has an interesting reciprocal relation with classification. On one hand, pairwise similarity can be used in classification algorithms like nearest neighbors or kernel methods. On the other hand, when objects can be classified into (possibly overlapping) classes, the inferred labels induce a notion of similarity across object pairs. Importantly however, similarity learning assumes a form of supervision that is weaker than in classification, since no labels are provided. OASIS is designed to learn a *class-independent* similarity measure with no need for class labels.

A large number of previous studies have focused on learning a similarity measure that is also a metric, like in the case of a positive semidefinite matrix that defines a Mahalanobis distance (Yang, 2006). However, similarity learning algorithms are often evaluated in a context of ranking. For instance, the learned metric is typically used together with a nearest-neighbor classifier (Weinberger et al., 2006; Globerson and Roweis, 2006). When the amount of training data available is very small, adding positivity constraints for enforcing metric properties is useful for reducing over fitting and improving generalization. However, when sufficient data is available, as in many modern applications, adding positive semi-definitiveness constraints consumes considerable computation time, and its benefit in terms of generalization are limited. With this view, we take here an approach that avoids imposing positivity or symmetry constraints on the learned similarity measure.

The current paper presents an approach for learning semantic similarity that scales up to an order of magnitude larger than current published approaches. Three components are combined to make this approach fast and scalable: First, our approach uses an unconstrained bilinear similarity. Given two images p_1 and p_2 we measure similarity through a bilinear form $p_1^T \mathbf{W} p_2$, where the matrix $\mathbf{W}$ is not required to be positive, or even symmetric. Second we use a sparse representation

of the images, which allows to compute similarities very fast. Finally, the training algorithm that we developed, OASIS, *Online Algorithm for Scalable Image Similarity learning*, is an online dual approach based on the passive-aggressive algorithm (Crammer et al., 2006). It minimizes a large margin target function based on the hinge loss, and already converges to high quality similarity measures after being presented with a small fraction of the training pairs.

We find that OASIS is both fast and accurate at a wide range of scales: for a standard benchmark with thousands of images, it achieves better (but comparable) results than existing state-of-the-art methods, with computation times that are shorter by orders of magnitude. For web-scale data sets, OASIS can be trained on more than two million images within three days on a single CPU, and its training time grows linearly with the size of the data. On this large scale data set, human evaluations of OASIS learned similarity show that 35% of the ten nearest neighbors of a given image are semantically relevant to that image.

The paper is organized as follows. We first present our online algorithm, OASIS, based on the Passive-aggressive family of algorithms. We then present the sparse feature extraction technique used in the experiments. We continue by describing experiments with OASIS on problems of image similarity, at two different scales: a large scale academic benchmark with tens of thousands of images, and a web-scale problem with millions of images. The paper ends with a discussion on properties of OASIS.

2. Learning Relative Similarity

We consider the problem of learning a pairwise similarity function S, given data on the relative similarity of pairs of images.

Formally, let $\mathcal{P}$ be a set of images, and $r_{ij} = r(p_i, p_j) \in \mathbb{R}$ be a pairwise relevance measure which states how strongly $p_j \in \mathcal{P}$ is related to $p_i \in \mathcal{P}$. This relevance measure could encode the fact that two images belong to the same category or were appropriate for the same query. We do not assume that we have full access to all the values of r. Instead, we assume that we can compare some pairwise relevance scores (for instance $r(p_i, p_j)$ and $r(p_i, p_k)$) and decide which pair is more relevant. We also assume that when $r(p_i, p_j)$ is not available, its value is zero (since the vast majority of images are not related to each other). Our goal is to learn a similarity function $S(p_i, p_j)$ that assigns higher similarity scores to pairs of more relevant images,

$$S(p_i, p_i^+) > S(p_i, p_i^-)\ , \quad \forall p_i, p_i^+, p_i^- \in \mathcal{P} \text{ such that } r(p_i, p_i^+) > r(p_i, p_i^-). \tag{1}$$

In this paper we overload notation by using p_i to denote both the image and its representation as a column vector $p_i \in \mathbb{R}^d$. We consider a parametric similarity function that has a bi-linear form,

$$S_{\mathbf{W}}(p_i, p_j) \equiv p_i^T\ \mathbf{W}\ p_j \tag{2}$$

with $\mathbf{W} \in \mathbb{R}^{d \times d}$. Importantly, if the images p_i are represented as sparse vectors, namely, only a number $k_i \ll d$ of the d entries in the vector p_i are non-zeroes, then the value of Equation (2) can be computed very efficiently even when d is large. Specifically, $S_{\mathbf{W}}$ can be computed with complexity of $O(k_i k_j)$ regardless of the dimensionality d.

2.1 An Online Algorithm

We propose an online algorithm based on the Passive-Aggressive (PA) family of learning algorithms introduced by Crammer et al. (2006). Here we consider an algorithm that uses triplets of images $p_i, p_i^+, p_i^- \in \mathcal{P}$ such that $r(p_i, p_i^+) > r(p_i, p_i^-)$.

We aim to find a parametric similarity function S such that all triplets obey

$$S_{\mathbf{W}}(p_i, p_i^+) > S_{\mathbf{W}}(p_i, p_i^-) + 1 \tag{3}$$

which means that it fulfills Equation (1) with a safety margin of 1. We define the following hinge loss function for the triplet:

$$l_{\mathbf{W}}(p_i, p_i^+, p_i^-) = \max\left\{0, 1 - S_{\mathbf{W}}(p_i, p_i^+) + S_{\mathbf{W}}(p_i, p_i^-)\right\}. \tag{4}$$

Our goal is to minimize a global loss $L_{\mathbf{W}}$ that accumulates hinge losses (4) over all possible triplets in the training set:

$$L_{\mathbf{W}} = \sum_{(p_i, p_i^+, p_i^-) \in \mathcal{P}} l_{\mathbf{W}}(p_i, p_i^+, p_i^-)\ .$$

In order to minimize this loss, we apply the Passive-Aggressive algorithm iteratively over triplets to optimize $\mathbf{W}$. First, $\mathbf{W}$ is initialized to some value $\mathbf{W}^0$. Then, at each training iteration i, we randomly select a triplet (p_i, p_i^+, p_i^-), and solve the following convex problem with soft margin:

$$\begin{aligned} \mathbf{W}^i &= \operatorname{argmin}_{\mathbf{W}} \frac{1}{2}\|\mathbf{W} - \mathbf{W}^{i-1}\|_{Fro}^2 + C\xi \\ \text{s.t.} \quad & l_{\mathbf{W}}(p_i, p_i^+, p_i^-) \le \xi \quad \text{and} \quad \xi \ge 0 \end{aligned} \tag{5}$$

where $\|\cdot\|_{Fro}$ is the Frobenius norm (point-wise L_2 norm). Therefore, at each iteration i, $\mathbf{W}^i$ is selected to optimize a trade-off between remaining close to the previous parameters $\mathbf{W}^{i-1}$ and minimizing the loss on the current triplet $l_{\mathbf{W}}(p_i, p_i^+, p_i^-)$. The *aggressiveness* parameter C controls this trade-off.

OASIS

Initialization:

Initialize $\mathbf{W}^0 = I$

Iterations

repeat

Sample three images p, p_i^+, p_i^-, such that $r(p_i, p_i^+) > r(p_i, p_i^-)$.

Update $\mathbf{W}^i = \mathbf{W}^{i-1} + \tau_i \mathbf{V}^i$

where $\tau_i = \min\left\{C, \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}^i\|^2}\right\}$

and $\mathbf{V}^i = [p_i^1(p_k^+ - p_k^-), \ldots, p_i^d(p_k^+ - p_k^-)]^T$

until (stopping criterion)

Figure 1: Pseudo-code of the OASIS algorithm.

We follow Crammer et al. (2006) to solve the problem in Equation (5). When $l_{\mathbf{W}}(p_i, p_i^+, p_i^-) = 0$, it is clear that $\mathbf{W}^i = \mathbf{W}^{i-1}$ satisfies Equation (5) directly. Otherwise, we define the Lagrangian

$$\mathcal{L}(\mathbf{W}, \tau, \xi, \lambda) = \frac{1}{2}\|\mathbf{W} - \mathbf{W}^{i-1}\|^2 + C\xi + \tau(1 - \xi - p_i^T \mathbf{W}(p_i^+ - p_i^-)) - \lambda\xi \tag{6}$$

where $\tau \geq 0$ and $\lambda \geq 0$ are Lagrange multipliers. The optimal solution is such that the gradient vanishes $\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \mathbf{W}} = 0$, hence

$$\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \mathbf{W}} = \mathbf{W} - \mathbf{W}^{i-1} - \tau \mathbf{V}_i = 0$$

where the gradient matrix $\mathbf{V}_i = \frac{\partial \mathcal{L}_W}{\partial \mathbf{W}} = [p_i^1(p_i^+ - p_i^-), \ldots, p_i^d(p_i^+ - p_i^-)]^T$. The optimal new $\mathbf{W}$ is therefore

$$\mathbf{W} = \mathbf{W}^{i-1} + \tau \mathbf{V}_i \tag{7}$$

where we still need to estimate τ. Differentiating the Lagrangian with respect to ξ and setting it to zero also yields:

$$\frac{\partial \mathcal{L}(\mathbf{W}, \tau, \xi, \lambda)}{\partial \xi} = C - \tau - \lambda = 0 \tag{8}$$

which, knowing that $\lambda \geq 0$, means that $\tau \leq C$. Plugging Equations (7) and (8) back into the Lagrangian in Equation (6), we obtain

$$\mathcal{L}(\tau) = \frac{1}{2}\tau^2 \|\mathbf{V}_i\|^2 + \tau(1 - p_i^T(\mathbf{W}^{i-1} + \tau \mathbf{V}_i)(p_i^+ - p_i^-)) \,.$$

Regrouping the terms we obtain

$$\mathcal{L}(\tau) = -\frac{1}{2}\tau^2 \|\mathbf{V}_i\|^2 + \tau(1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)) \,.$$

Taking the derivative of this second Lagrangian with respect to τ and setting it to 0, we have

$$\frac{\partial \mathcal{L}(\tau)}{\partial \tau} = -\tau \|\mathbf{V}_i\|^2 + (1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)) = 0$$

which yields

$$\tau = \frac{1 - p_i^T \mathbf{W}^{i-1}(p_i^+ - p_i^-)}{\|\mathbf{V}_i\|^2} = \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}_i\|^2} \,.$$

Finally, Since $\tau \leq C$, we obtain

$$\tau = \min\left\{C, \frac{l_{\mathbf{W}^{i-1}}(p_i, p_i^+, p_i^-)}{\|\mathbf{V}_i\|^2}\right\} \,. \tag{9}$$

Equations (7) and (9) summarize the update needed for every triplets (p_i, p_i^+, p_i^-). It has been shown (Crammer et al., 2006) that applying such an iterative algorithm yields a cumulative online loss that is likely to be small. It was furthermore shown that selecting the best $\mathbf{W}_i$ during training using a hold-out validation set achieves good generalization. We also show below that multiple runs of the algorithm converge to provide similar precision (see Figure 7).

2.2 Loss Bounds

Following closely the analysis of loss bounds for passive aggressive (PA) algorithms developed by Crammer et al. (2006) we state similar *relative* bounds for the OASIS framework. We do this by rewriting OASIS as a straightforward linear classification problem. Denote by $\overrightarrow{w_i}$ the vector obtained by"unfolding" the matrix $\mathbf{W}$ (concatenating all its columns into a single vector) and similarly $\overrightarrow{x_i}$ the unfolded matrix $p_i(p_i^+ - p_i^-)^T$. Using this notation, the constraint in Equation (3) becomes

$$\overrightarrow{w_i} \cdot \overrightarrow{x_i} > 1 \quad ,$$

with $\cdot$ denoting the standard inner product. This is equivalent to the formulation of PA when the label y_i is always 1. The introduction of slack variables in Equation (5) brings us to the variant denoted by Crammer et al. (2006) as PA-I.

The loss bounds in Crammer et al. (2006) rely on $\overrightarrow{w_0}$ being the zero vector. Since here we initialize with $W^0 = I$ (the identity matrix) we need to adapt the analysis slightly. Let $\overrightarrow{u}$ be a vector in $\mathbb{R}^{d^2}$ obtained by unfolding an arbitrary matrix $\mathbf{U}$. We define

$$l_i = 1 - \overrightarrow{w_i} \cdot \overrightarrow{x_i} \quad \text{and} \quad l_i^* = 1 - \overrightarrow{u} \cdot \overrightarrow{x_i} \quad ,$$

where l_i is the instantaneous loss at round i, and l_i^* is the loss suffered by the arbitrary vector $\overrightarrow{u}$. The following two theorems rely on Lemma 1 of Crammer et al. (2006), which we restate without proof:

$$\sum \tau_i (2l_i - \tau_i \|x_i\|^2 - 2l_i^*) \leq \|\overrightarrow{u} - \overrightarrow{w_0}\|^2 \, .$$

While in Crammer et al. (2006) $\overrightarrow{w_0}$ is the zero vector, in our case $\overrightarrow{w_0}$ is the unfolded *identity matrix*. We therefore have

$$\|\overrightarrow{u} - \overrightarrow{w_0}\|^2 = \|\mathbf{U}\|_{Fro}^2 - 2trace(\mathbf{U}) + n \, .$$

Using this modified lemma we can restate the relevant bound:

Theorem 1 *Let $(\overrightarrow{x_1}),...,(\overrightarrow{x_M})$ be a sequence of examples where $\overrightarrow{x_i} \in \mathbb{R}^{d^2}$, $\|\overrightarrow{x_i}\| \leq R$ for all $i = 1...M$. Then, for any matrix $\mathbf{U} \in \mathbb{R}^{n^2}$, the number of prediction mistakes made by OASIS on this sequence of examples is bounded from above by,*

$$\max\{R^2, 1/C\}\left(\|\mathbf{U}\|_{Fro}^2 - 2trace(\mathbf{U}) + n + 2C\sum_{i=1}^{M} l_i^*\right)$$

where C is the aggressiveness parameter provided to OASIS.

2.3 Sampling Strategy

For real world data sets, the actual number of triplets (p_i, p_i^+, p_i^-) is typically very large and cannot be stored in memory. Instead, we use the fact that the number of relevant images for a category or a query is typically small, and keep a list of relevant images for each query or category. For the case of single-labeled images, we can efficiently retrieve an image that is relevant to a given image, by first finding its class, and then finding another image from that class. The case of multi-labeled images is described in Section 5.2.

Specifically, to sample a triplet (p_i, p_i^+, p_i^-) during training, we first uniformly sample an image p_i from $\mathcal{P}$. Then we uniformly sample an image p_i^+ from the images sharing the same categories

or queries as p_i. Finally, we uniformly sample an image p_i^- from the images that share no category or query with p_i. When the set $\mathcal{P}$ is very large and the number of categories or queries is also very large, one does not need to maintain the set of non-relevant images for each image: sampling directly from $\mathcal{P}$ instead only adds a small amount of noise to the training procedure and is not really harmful.

When relevance feedbacks $r(p_i, p_j)$ are provided as real numbers and not just $\in \{0,1\}$, one could use these number to bias training towards those pairs that have a higher relevance feedback value. This can be done by considering $r(p_i, p_j)$ as frequencies of appearance, and sampling pairs according to the distribution of these frequencies.

3. Image Representation

The problem of selecting an informative representation of images is still an unsolved computer vision challenge, and an ongoing research topic. Different approaches for image representation have been proposed including by Feng et al. (2004); Takala et al. (2005) and Tieu and Viola (2004). In the information retrieval community there is wide agreement that a bag-of-words representation is a very useful representation for handling text documents in a wide range of applications. For image representation, there is still no such approach that would be adequate for a wide variety of image processing problems. However, among the proposed representations, a consensus is emerging on using *local descriptors* for various tasks, for example, Lowe (2004); Quelhas et al. (2005). This type of representation segments the image into *regions of interest*, and extracts visual features from each region. The segmentation algorithm as well as the region features vary among approaches, but, in all cases, the image is then represented as a set of feature vectors describing the regions of interest. Such a set is often called a *bag-of-local-descriptors*.

In this paper we take the approach of creating a sparse representation based on the framework of local descriptors. Our features are extracted by dividing each image into overlapping square blocks, and each block is then described with edge and color histograms. For edge histograms, we rely on *uniform Local Binary Patterns* (uLBPs) proposed by Ojala et al. (2002). These texture descriptors have shown to be effective on various tasks in the computer vision literature (Ojala et al., 2002; Takala et al., 2005), certainly due to their robustness with respect to changes in illumination and other photometric transformations (Ojala et al., 2002). Local Binary Patterns estimate a texture histogram of a block by considering differences in intensity at circular neighborhoods centered on each pixel. Precisely, we use $LBP_{8,2}$ patterns, which means that a circle of radius 2 is considered centered on each block. For each circle, the intensity of the center pixel is compared to the interpolated intensities located at 8 equally-spaced locations on the circle, as shown on Figure 2, left. These eight binary tests (lower or greater intensity) result in an 8-bit sequence, see Figure 2, right. Hence, each block pixel is mapped to a sequence among $2^8 = 256$ possible sequences and each block can therefore be represented as a 256-bin histogram. In fact, it has been observed that the bins corresponding to non-uniform sequences (sequences with more than 2 transitions $1 \rightarrow 0$ or $0 \rightarrow 1$) can be merged, yielding more compact 59-bin histograms without performance loss (Ojala et al., 2002).

Color histograms are obtained by K-means clustering. We first select a palette or typical colors by training a color codebook from the Red-Green-Blue pixels of a large training set of images using K-means. The color histogram of a block is then obtained by mapping each block pixel to the closest color in the codebook palette.

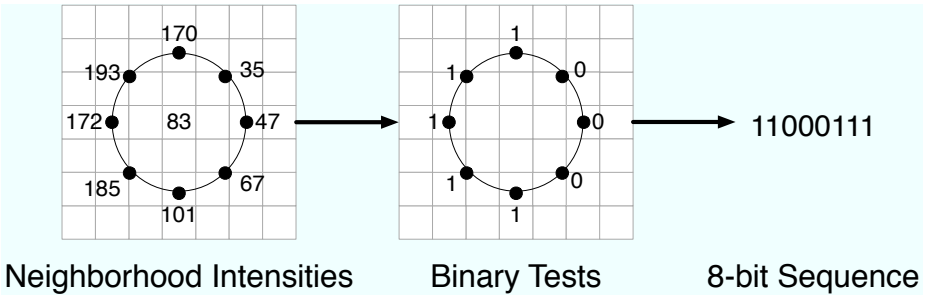


Figure 2: An example of Local Binary Pattern ($LBP_{8,2}$). For a given pixel, the Local Binary Pattern is an 8-bit code obtained by verifying whether the intensity of the pixel is greater or lower than its 8 neighbors.

Finally, the histograms describing color and edge statistics of each block are concatenated, which yields a single vector descriptor per block. Our local descriptor representation is therefore simple, relying on both a basic segmentation approach and simple features. Naturally, alternative representations could also be used with OASIS, (Feng et al., 2004; Grangier et al., 2006; Tieu and Viola, 2004) However, this paper focuses on the learning model, and a benchmark of image representations is beyond the scope of the current paper.

As a final step, we use the representation of blocks to obtain a representation for an image. For computation efficiency we aim at a high dimensional and sparse vector space. For this purpose, each local descriptor of an image p is represented as a discrete index, called *visual term* or *visterm*, and, like for text data, the image is represented as a *bag-of-visterms* vector, in which each component p_i is related to the presence or absence of visterm i in p.

The mapping of the descriptors to discrete indexes is performed according to a codebook C, which is typically learned from the local descriptors of the training images through k-means clustering (Duygulu et al., 2002; Jeon and Manmatha, 2004; Quelhas et al., 2005). The assignment of the weight p_i of visterm i in image p is as follows:

$$p_i = \frac{f_i \, d_i}{\sqrt{\sum_{j=1}^{d} (f_j \, d_j)^2}} \, ,$$

where f_i is the term frequency of i in p, which refers to the number of occurrences of i in p, while d_j is the inverse document frequency of j, which is defined as $-log(r_j)$, r_j being the fraction of training images containing at least one occurrence of visterm j. This approach has been found successful for the task of content based image ranking described by Grangier and Bengio (2008).

In the experiments described below, we used a large set of images collected from the web to train the features. This set is described in more detail in Section 5.2. We used a set of 20 typical RGB colors (hence the number of clusters used in the k-means for colors was 20), the block vocabulary size $d = 10000$ and our image blocks were of size 64x64 pixels, overlapping every 32 pixels. Furthermore, in order to be robust to scale, we extracted blocks at various scales by successively down scaling images by a factor of 1.25 and extracting the features at each level, until there were less than 10 blocks in the resulting image. There was on average around 70 non-zero

values (out of 10000) describing a single image. Note that no other information (such as meta-data) was added in the input vector representation each image.

4. Related Work

Similarity learning can be considered in two main setups, depending on the type of available training labels. First, a regression setup, where the training set consists of pairs of objects x_i^1, x_i^2 and their pairwise similarity $y_i \in \mathbf{R}$. In many cases however, precise similarities are not available, but rather a weaker notion of similarity order. In one such setup, the training set consists of triplets of objects x_i^1, x_i^2, x_i^3 and a ranking similarity function, that can tell which of the two pairs (x^1, x^2) or (x^1, x^3) is more similar. Finally, multiple similarity learning studies assume that a binary measure of similarity is available $y_i \in \{+1, -1\}$, indicating whether a pair of objects is similar or not.

For small-scale data, there are two main groups of similarity learning approaches. The first approach, learning Mahalanobis distances, can be viewed as learning a linear projection of the data into another space (often of lower dimensionality), where a Euclidean distance is defined among pairs of objects. Such approaches include Fisher's Linear Discriminant Analysis (LDA), relevant component analysis (RCA) (Bar-Hillel et al., 2003), supervised global metric learning (Xing et al., 2003), large margin nearest neighbor (LMNN) (Weinberger et al., 2006) and Metric Learning by Collapsing Classes (Globerson and Roweis, 2006). A Mahalanobis distance learning algorithm which uses a supervision signal identical to the one we employ in OASIS is Rosales and Fung (2006), which learns a special kind of PSD matrix via linear programming. See also a review by Yang (2006) for more details.

The second family of approaches, learning kernels, is used to improve performance of kernel based classifiers. Learning a full kernel matrix in a non parametric way is prohibitive except for very small data. As an alternative, several studies suggested to learn a weighted sum of pre-defined kernels (Lanckriet et al., 2004) where the weights are being learned from data. In some applications this was shown to be inferior to uniform weighting of the kernels (Noble, 2008). The work of Frome et al. (2007) further learns a weighting over local distance function for every image in the training set. Non linear image similarity learning was also studied in the context of dimensionality reduction, as in Hadsell et al. (2006).

Finally, Jain et al. (2008a,b), based on work by Davis et al. (2007), aim to learn metrics in an online setting. This work is one of the closest work with respect to OASIS: it learns a linear model of a [dis-]similarity function between documents in an online way. The main difference is that the work of Jain et al. (2008a) learn a true distance throughout the learning process, imposing positive definiteness constraints, and is slightly less efficient computationally. We argue in this paper that in the large scale regime, such a constraint is not necessary given the amount of available training examples.

Another work closely related to OASIS is that of Rasiwasia and Vasconcelos (2008), which also tries to learn a semantic similarity function between images. In their case, however, semantic similarity is learned by representing each image by the posterior probability distribution over a pre-defined set of semantic tags, and then computing the distance between two images as the distance between the two underlying posterior distributions. The representation size of images in this approach is therefore equal to the number of semantic classes, hence it will not scale when the number of semantic classes is very large as in free text search.

5. Experiments

Evaluating large scale learning algorithms poses special challenges. First, current available benchmarks are limited either in their scale, like 30K images in Caltech256 as described by Griffin et al. (2007), or in their resolution, such as the tiny images data set of Torralba et al. (2007). Large scale methods are not expected to perform particularly well on small data sets, since they are designed to extract limited information from each sample. Second, many images on the web cannot be used without explicit permission, hence they cannot be collected and packed into a single database. Large, proprietary collections of images do exist, but are not available freely for academic research. Finally, except for very few cases, similarity learning approaches in current literature do not scale to handle large data sets effectively, which makes it hard to compare a new large scale method with the existing methods.

To address these issues, this paper takes the approach of conducting experiments at two different scales. First, to demonstrate the scalability of OASIS we applied OASIS to a web-scale data with 2.7 million images. Second, to investigate the properties of OASIS more deeply, we compare OASIS with small-scale methods using the standard Caltech256 benchmark.

5.1 Evaluation Measures

We evaluated the performance of all algorithms using standard ranking precision measures based on nearest neighbors. For each query image in the test set, all other test images were ranked according to their similarity to the query image. The number of same-class images among the top k images (the k nearest neighbors) was computed. When averaged across test images (either within or across classes), this yields a measure known as precision-at-top-k, providing a precision curve as a function of the rank k.

We also calculated the *mean average precision* (mAP), a measure that is widely used in the information retrieval community. To compute average precision, the precision-at-top-k is first calculated for each test image. Then, it is averaged over all positions k that have a positive sample. For example, if all positives are ranked highest, the average-precision is 1. The average-precision measure is then further averaged across all test image queries, yielding the *mean average precision* (mAP).

5.2 Web-Scale Experiment

Our first set of experiments is based on Google proprietary data that is two orders of magnitude larger than current standard benchmarks. We collected a set of $\sim$150K text queries submitted to the Google Image Search system. For each of these queries, we had access to a set of relevant images, each of which is associated with a numerical relevance score. This yielded a total of $\sim$2.7 million images, which we split into a training set of 2.3 million images and a test set of 0.4 million images (see Table 1).

Set	Number of Queries	Number of Images
Training	139944	2292259
Test	41877	402164

Table 1: Statistics of the Web data set.

5.2.1 EXPERIMENTAL SETUP

We used the query-image relevance information to create an image-image relevance as follows. Denote the set of text queries by $\mathcal{Q}$ and the set of images by $\mathcal{P}$. For each $q \in \mathcal{Q}$, let $\mathcal{P}_q^+$ denote the set of images that are relevant to the query q, and let $\mathcal{P}_q^-$ denote the set of irrelevant images. The query-image relevance is defined by the matrix $\mathbf{R}_{QI} : \mathcal{Q} \times \mathcal{P} \to \mathbb{R}^+$, and obeys $\mathbf{R}_{QI}(q, p_q^+) > 0$ and $\mathbf{R}_{QI}(q, p_q^-) = 0$ for all $q \in \mathcal{Q}$, $p_q^+ \in \mathcal{P}_q^+$, $p_q^- \in \mathcal{P}_q^-$. We also computed a normalized version of $\mathbf{R}_{QI}$, which can be interpreted as a joint distribution matrix, or the probability to observe a query q and an image p for that query,

$$Pr(q,p) = \frac{\mathbf{R}_{QI}(q,p)}{\sum_{q',p'} \mathbf{R}_{QI}(q',p')} \quad .$$

In order to compute the image-image relevance matrix $\mathbf{R}_{II} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}^+$, we treated images as being conditionally independent given the queries, $Pr(p_1, p_2|q) = Pr(p_1|q)Pr(p_2|q)$, and computed the joint image-image probability as a relevance measure

$$Pr(p_1,p_2) = \sum_{q \in \mathcal{Q}} Pr(p_1,p_2|q)\, Pr(q) = \sum_{q \in \mathcal{Q}} Pr(p_1 \mid q) Pr(p_2 \mid q) Pr(q) \; .$$

To improve scalability, we used a threshold over this joint distribution, and considered two images to be related only if their joint distribution exceeded a cutoff value θ

$$\mathbf{R}_{II}(p_1,p_2) = [Pr(p_1,p_2)]_\theta \tag{10}$$

where $[x]_\theta = x$ for $x > \theta$ and is zero otherwise. To set the value of θ we have manually inspected a small subset of pairs of related images taken from the training set. We selected the largest θ such that most of those related pairs had scores above the threshold, while minimizing noise in $\mathbf{R}_{II}$.

Equation 10 is written as if one needs to calculate the full joint matrix $\mathbf{R}_{II}$, but this matrix grows quadratically with the number of images. In practice, we can use the fact that $\mathbf{R}_{QI}$ is very sparse, to quickly create a list with images that are relevant to a given image. To do this given an image p_i, we go over all the queries for which it is relevant $\mathbf{R}_{QI}(q, p_i)$, and for each of these queries, collect the list of all images that are relevant to that query. The average number of queries relevant for an image in our data is small (about 100), and so is the number of images relevant for a given query. As a result, $\mathbf{R}_{II}$ can be calculated efficiently even for large image sets.

We trained OASIS over 2.3 million images in the training set using the sampling mechanism based on the relevance of each image, as described in Section 2.3. To select the number of training iterations, we used as a validation set a small subset of the training set to trace the mean average precision of the model at regular intervals during the training process. Training was stopped when the mean average precision had saturated, which happened after 160 million iterations (triplets). Overall, training took a total of $\sim$4000 minutes on a single CPU of a standard modern machine. Finally, we evaluated the trained model on the 400 thousand images of the test set.

5.2.2 RESULTS

We start with specific examples illustrating the behavior of OASIS, and continue with a quantitative analysis of precision and speed. Table 2 shows the top five images as ranked by OASIS on four examples of query-images in the test set. The relevant text queries for each image are shown beneath the image. The first example (top row), shows a query-image that was originally retrieved

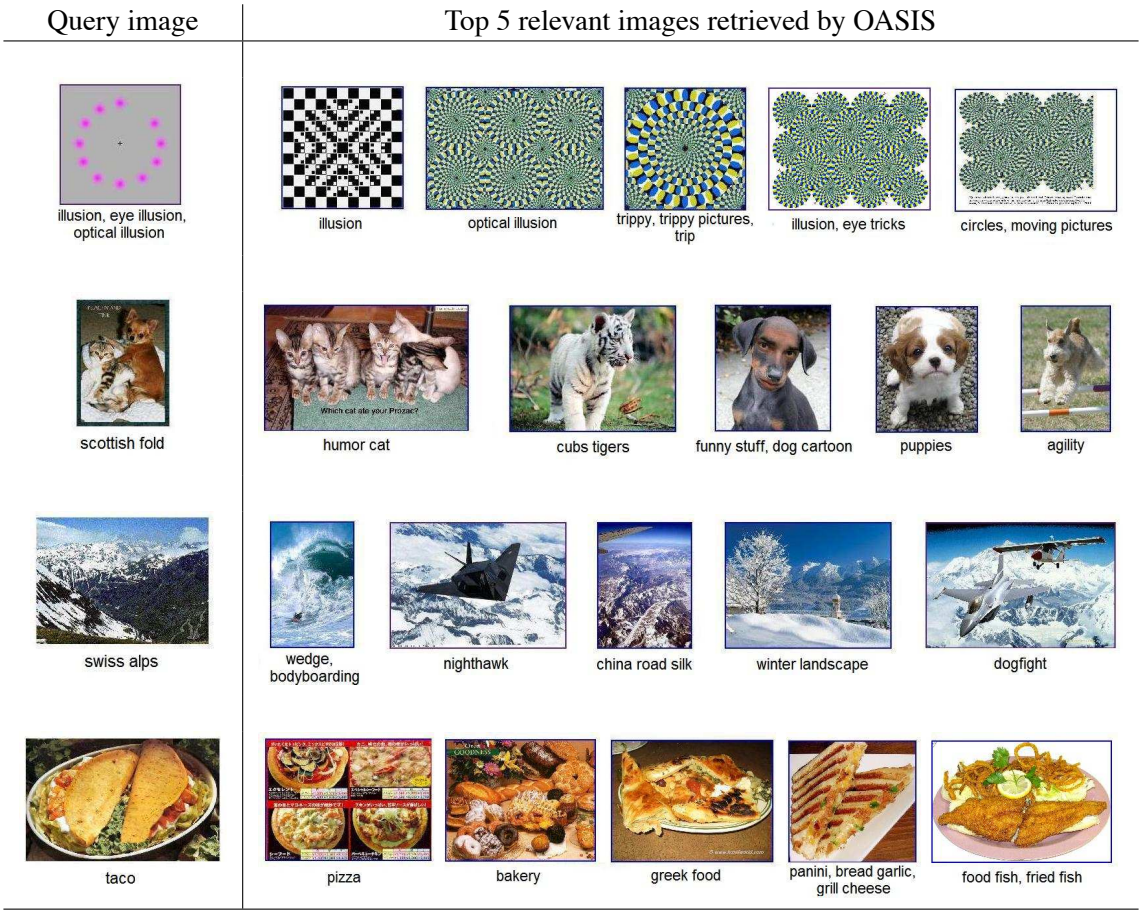


Table 2: OASIS: Successful cases from the Web data set

in response to the text query "illusion". All five images ranked highly by OASIS are semantically related, showing other types of visual illusions. Similar results can be observed for the three remaining examples on this table, where OASIS captures well the semantics of animal photos (cats and dogs), mountains and different food items.

In all these cases, OASIS captures similarity that is both semantic and visual, since the raw visual similarity of these images is not high. A different behavior is demonstrated in Table 3. It shows three cases where OASIS was biased by visual similarity and provided high rankings to images that were semantically non relevant. In the first example, the assortment of flowers is confused with assortments of food items and a thigh section (5th nearest neighbor) which has visually similar shape. The second example presents a query image which in itself has no definite semantic element. The results retrieved are those that merely match texture of the query image and bear no semantic similarity. In the third example, OASIS fails to capture the butterfly in the query image.

To obtain a quantitative evaluation of OASIS we computed the precision at top k, using a threshold $\theta = 0$, which means that an image in the test set is considered relevant to a query image, if there exists at least one text query to which they were both relevant to.

The obtained precision values were quite low, achieving 1.5% precision at the top ranked image. This is drastically lower than the precision described below for Caltech256, and could be the result

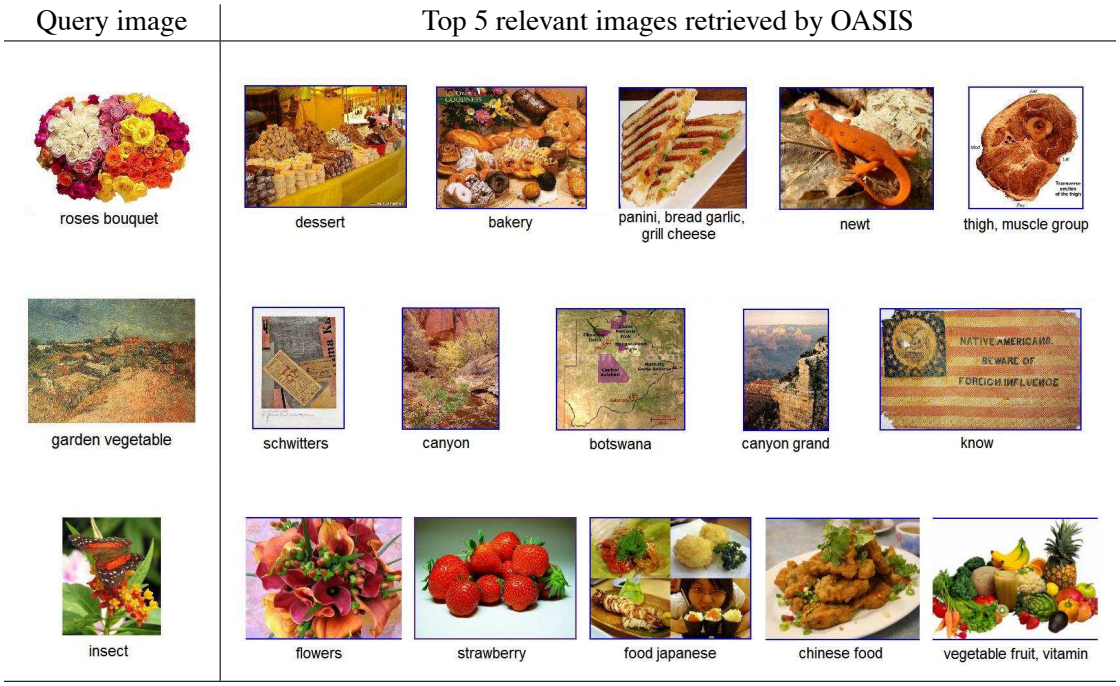


Table 3: OASIS: Failure cases from the Web data set

of multiple reasons. First, the number of unique textual queries in our data is very large (around 150K), hence the images in this data set were significantly more heterogeneous than images in the Caltech256 data.

Second, and most importantly, our labels that measure pairwise relevance are very partial. This means that many pairs of images that are semantically related are not labeled as such. A clear demonstration of this effect is observed in Tables 2 and 3. The query images (like "*scottish fold*") have labels that are usually very different from the labels of the retrieved images (as in "*humor cat*", "*agility*") even if their semantic content is very similar. This is a common problem in content-based analysis, since similar content can be described in many different ways. In the case discussed here, the partial data on the query-image relevance $\mathbf{R}_{QI}$ is further propagated to the image-image relevance measure $\mathbf{R}_{II}$.

5.2.3 HUMAN EVALUATION EXPERIMENTS

In order to obtain a more accurate estimate of the real semantic precision, we performed a rating experiment with human evaluators. We chose the 25 most relevant images[1] from the test set and retrieved their 10 nearest neighbors as determined by OASIS. We excluded query-images which contained porn, racy or duplicates in their 10 nearest neighbors. We also selected randomly a set of 10 negative images p^- that were chosen for each of the query images p such that $\mathbf{R}_{II}(p,p^-)=0$. These negatives were then randomly mixed with the 10 nearest neighbors.

All 25 query images were presented to twenty human evaluators, asking them to mark which of the 20 candidate images are *semantically relevant* to the query image.[2] Evaluators were volunteers

1. The overall relevance of an image was estimated as the sum of relevances of the image with respect to all queries.
2. The description of the task as given to the evaluators is provided in Appendix A.

selected from a pool of friends and colleagues, many of which had experience with search or machine vision problems. We collected the ratings on the positive images and calculated the precision at top k.

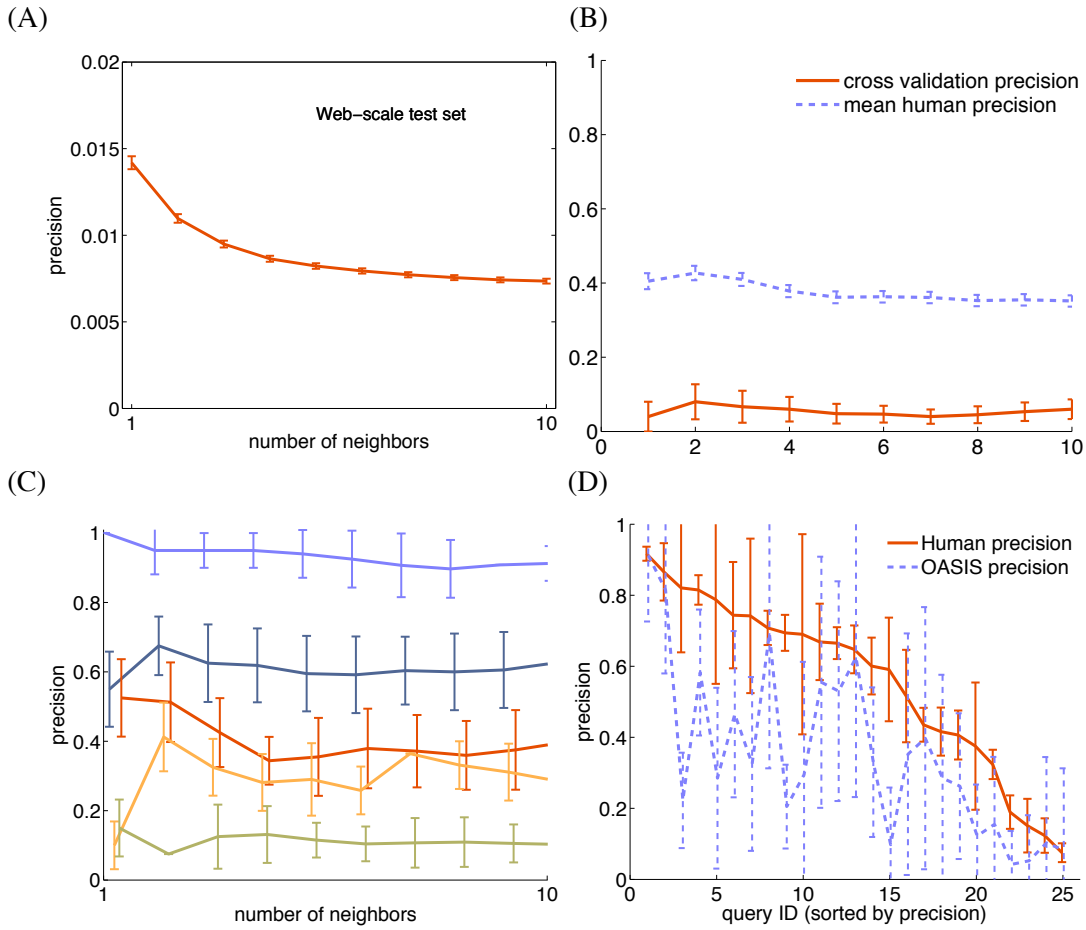


Figure 3: **(A)** Precision at top k as a function of k neighbors computed against $\mathbf{R}_{II}$ ($\theta = 0$) for the web-scale test set. **(B)** Precision at top k as a function of k neighbors for the human evaluation subset. **(C)** Mean precision for 5 selected queries. Error bars denote the standard error of the mean. To select the queries for this plot, we first calculated the mean-average precision per query, sorted the queries by their mAP, and selected the queries ranked at position 1, 6, 11, 16, and 21. **(D)** Precision of OASIS and human evaluators, per query, using rankings of all (remaining) human evaluators as a ground truth.

Figure 3(B) shows the average precision across all queries and evaluators. Precision peaks at 42% and reaches 35% at the top 10 ranked image, being significantly higher than the values calculated automatically using $\mathbf{R}_{II}$.

We observed that the variability across different query images was also very high. Figure 3(C) shows the precision for 5 different queries, selected to span the range of average-precision values. The error bars at each curve show the variability in the responses of different evaluators. The

precision of OASIS varies greatly across different queries. Some query images were "easy" for OASIS, yielding high scores from most evaluators, while other queries retrieved images that were consistently found to be irrelevant by most evaluators.

We also compared the magnitude of variability across human evaluators, with variability across queries. We first calculated the mAP from the precision curves of every query and evaluator, and then calculated the standard deviation in the mAP of every evaluator and of every query. The mean standard deviation over queries was 0.33, suggesting a large variability in the difficulty of image queries, as observed in Figure 3(C) . The mean standard deviation over evaluators was 0.25, suggesting that different evaluators had very different notions of what images should be regarded as "semantically similar" to a query image.

Finally, to estimate an "upper bound" on the difficulty of the task, we also computed the precision of the human evaluators themselves. For every evaluator, we used the rankings of all other evaluators as ground truth, to compute his precision. As with the ranks of OASIS, we computed the fraction of evaluators that marked an image as relevant, and repeated this separately for every query and human evaluator, providing a measure of "coherence" per query. Figure 3(D) shows the mean precision obtained by OASIS and human evaluators for every query in our data. For some queries OASIS achieves precision that is very close to that of the mean human evaluator. In many cases OASIS achieves precision that is as good or better than some evaluators.

5.2.4 SPEED AND SCALABILITY

We further studied how the runtime of OASIS scales with the size of the training set. Figure 4 shows that the runtime of OASIS, as found by early stopping on a separate validation set, grows linearly with the train set size. We compare this to the fastest result we found in the literature, based on a fast implementation of LMNN by Weinberger and Saul (2008). LMNN learns a Mahalanobis distance for k-nearest neighbor classification, aiming to have the nearest neighbors of a sample belong to the same class, and samples from different classes separated by a large margin. The LMNN algorithm is known to scale quadratically with the number of objects, although their experiments with MNIST data show that the active set of constraints grows linearly. This could be because MNIST has 10 classes only. In many real world data however, the number of classes typically grows almost linearly with the number of samples.

5.3 Caltech256 Data Set

To compare OASIS with small-scale methods we used the *Caltech256* data set (Griffin et al., 2007). This data set consists of 30607 images that were obtained from Google image search and from *PicSearch.com*. Images were assigned to 257 categories and evaluated by humans in order to ensure image quality and relevance. After we have pre-processed the images as described in Section 3 and filtered images that were too small, we were left with 29461 images in 256 categories. To allow comparisons with other methods in the literature that were not optimized for sparse representation, we also reduced the block vocabulary size d from 10000 to 1000. This processed data is available online at *http://ai.stanford.edu/∼gal/Research/OASIS*.

Using the Caltech256 data set allows us to compare OASIS with existing similarity learning methods. For OASIS, we treated images that have the same labels as similar. The same labels were used for comparing with methods that learn a metric for classification, as described below.

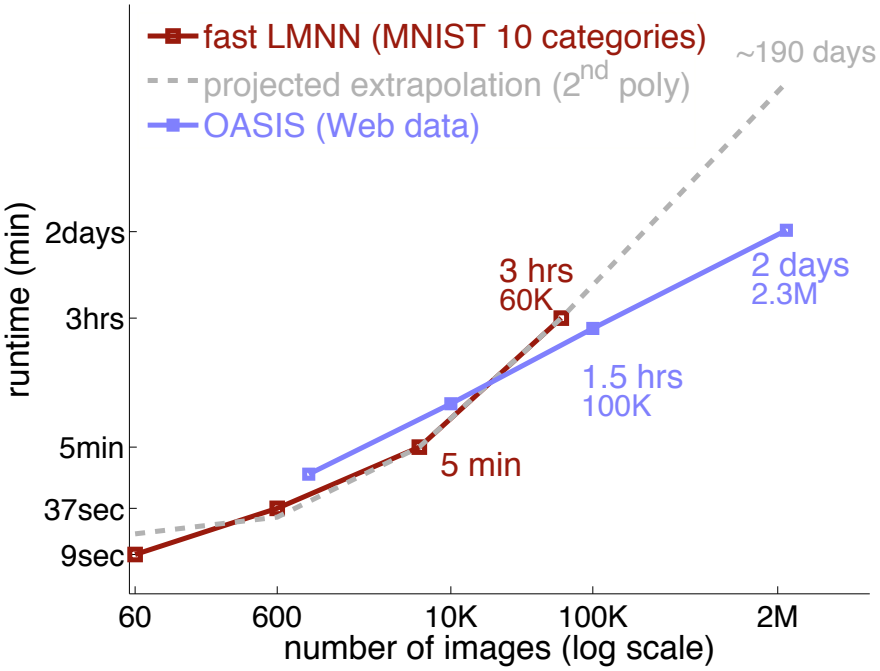


Figure 4: Comparison of the runtime of OASIS and fast-LMNN by Weinberger and Saul (2008), over a wide range of scales. LMNN results (on MNIST data) are faster than OASIS results on subsets of the web data. However LMNN scales quadratically with the number of samples, hence is three times slower on 60K images, and may be infeasible for handling 2.3 million images.

5.3.1 COMPARED METHODS

We compared the following approaches:

1. **OASIS.** - The algorithm described above in Section 2.1.

2. **Euclidean.** - The standard Euclidean distance in feature space. The initialization of OASIS using the identity matrix is equivalent to this distance measure.

3. **MCML** - Metric Learning by Collapsing Classes (Globerson and Roweis, 2006). This approach learns a Mahalanobis distance such that samples from the same class are mapped to the same point. The problem is written as a convex optimization problem, and we have used the gradient-descent implementation provided by the authors.

4. **LMNN** - Large Margin Nearest Neighbor Classification (Weinberger et al., 2006). This approach learns a Mahalanobis distance for k-nearest neighbor classification, aiming to have the k-nearest neighbors of a given sample belong to the same class while examples from different classes are separated by a large margin. As a preprocessing phase, images were projected to a basis of the principal components (PCA) of the data, with no dimensionality reduction, since

this improved the precision results. We also compared with a fast implementation of LMNN, that uses a clever scheme of maintaining a set of active constraints (Weinberger and Saul, 2008). We used the web data discussed above to compare with previously published results obtained with fast-LMNN on MNIST data (see Figure 4).

5. **LEGO** - Online metric learning (Jain et al., 2008a). LEGO learns a Mahalanobis distance in an online fashion using a regularized per instance loss, yielding a positive semidefinite matrix. The main variant of LEGO aims to fit a given set of pairwise distances. We used another variant of LEGO that, like OASIS, learns from relative distances. In our experimental setting, the loss is incurred for same-class examples being more than a certain distance away, and different class examples being less than a certain distance away. LEGO uses the LogDet divergence for regularization, as opposed to the Frobenius norm used in OASIS.

For all these approaches, we used an implementation provided by the authors. Algorithms were implemented in Matlab, with runtime bottlenecks implemented in C for speedup (except LEGO). We test below two variants of OASIS applied to the Caltech256 data set: a pure Matlab implementation, and one that has a *C* components. We used a C^{++} implementation of OASIS for the web-scale experiments described below.

We have also experimented with the methods of Xing et al. (2003) and RCA (Bar-Hillel et al., 2003). We found the method of Xing et al. (2003) to be too slow for the sets in our experiments. RCA is based on a per-class eigen decomposition that is not well defined when the number of samples is smaller than the feature dimensionality. We therefore experimented with a preprocessing phase of dimensionality reduction followed by RCA, but results were inferior to other methods and were not included in the evaluations below. RCA also did not perform well when tested on the full data, where dimensionality was not a problem, possibly because it is not designed to handle well sparse data.

5.3.2 EXPERIMENTAL PROTOCOL

We tested all methods on subsets of classes taken from the Caltech256 repository. Each subset was built such that it included semantically diverse categories, spanning the full range of classification difficulty, as measured by Griffin et al. (2007). We used subsets of sizes 10, 20, 50 and 249 classes (we used 249 classes since classes 251-256 are strongly correlated with other classes, and since class 129 did not contain enough large images). The full lists of categories in each set are given in Appendix B. For each set, images from each class were split into a training set of 40 images and a test set of 25 images, as proposed by Griffin et al. (2007).

We used cross-validation to select the values of hyper parameters for all algorithms except MCML. Models were learned on 80% of the training set (32 images), and evaluated on the remaining 20%. Cross validation was used for setting the following hyper parameters: the early stopping time for OASIS; the ω parameter for LMNN ($\omega \in \{0.125, 0.25, 0.5\}$), and the regularization parameter η for LEGO ($\eta \in \{0.02, 0.08, 0.32\}$). We found that LEGO was usually not sensitive to the choice of η, yielding a variance that was smaller than the variance over different cross-validation splits. Results reported below were obtained by selecting the best value of the hyper parameter and then training again on the full training set (40 images). For MCML, we used the default parameters supplied with the code from the authors, since its very long run time and multiple parameters made it non-feasible to tune hyper parameters on this data.

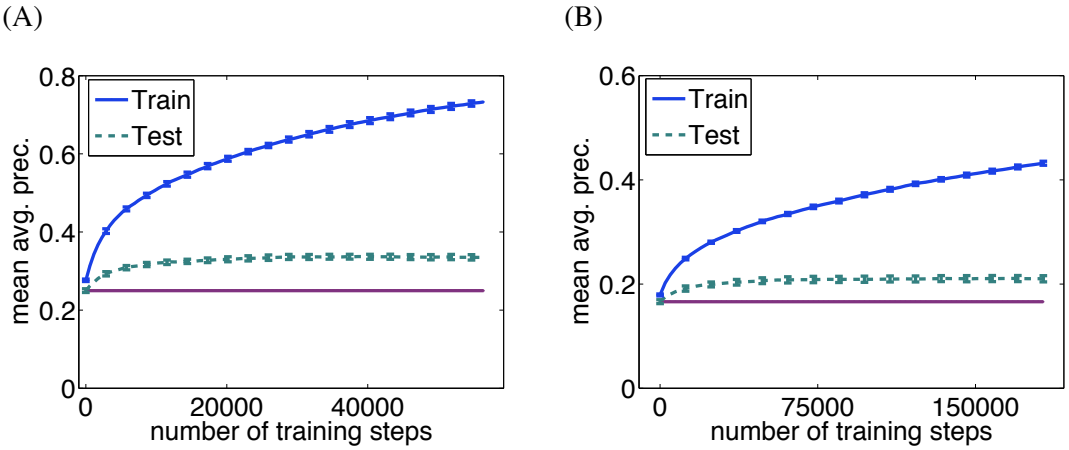


Figure 5: Mean average precision of OASIS as a function of the number of training steps. Error bars represent standard error of the mean over 5 selections of training (40 images) and test (25 images) sets. Performance is compared with a baseline obtained using the naïve Euclidean metric on the feature vector. C=0.1 **(A)** 10 classes. Test performance saturates around 30K training steps, while going over all triplets would require 2.8 million steps. **(B)** 20 classes.

5.3.3 RESULTS

Figure 5 traces the mean average precision over the training and the test sets as it progresses during learning. For the 10 classes task, precision on the test set saturates early (around 35K training steps), and then decreases very slowly.

Figure 6 and Table 4 compare the precision obtained with OASIS, with four competing approaches, as described above (Section 5.3.1). OASIS achieved consistently superior results throughout the full range of k (number of neighbors) tested, and on all four sets studied. Interestingly, we found that LMNN performance on the training set was often high, suggesting that it overfits the training set. This behavior was also noted by Weinberger et al. (2006) in some of their experiments.

OASIS achieves superior or equal performance, with a runtime that is faster by about two orders of magnitudes than MCML, and about one order of magnitude faster than LMNN. The run time of OASIS and LEGO was measured until the point of early stopping.

Table 5 shows the total CPU time in minutes for training each of the algorithms compared (measured on a standard 1.8GHz Intel Xeon CPU). For the purpose of a fair comparison with competing approaches, we tested two implementations of OASIS: The first was fully implemented Matlab. The second had the core of the algorithm implemented in C and called from Matlab.[3] LMNN code and MCML code were supplied by the authors and implemented in Matlab, with core parts implemented in C. LEGO code was supplied by the authors and fully implemented in Matlab.

Importantly, we found that Matlab does not make full use of the speedup that can be gained by sparse image representation. As a result, the C/C^{++} implementation of OASIS that we tested is significantly faster.

3. The OASIS code is available online at *http://ai.stanford.edu/∼gal/Research/OASIS*

10 classes	OASIS Matlab	MCML Matlab+C	LEGO Matlab	LMNN Matlab+C	Euclidean -
Mean avg prec	**33** $\pm$ 1.6	29 $\pm$ 1.7	27 $\pm$ 0.8	24 $\pm$ 1.6	23 $\pm$ 0.9
Top 1 prec.	**43** $\pm$ 4.0	39 $\pm$ 5.1	39 $\pm$ 4.8	38 $\pm$ 5.4	37 $\pm$ 4.1
Top 10 prec.	**38** $\pm$ 1.3	33 $\pm$ 1.8	32 $\pm$ 1.2	29 $\pm$ 2.1	27 $\pm$ 1.5
Top 50 prec.	**23** $\pm$ 1.5	22 $\pm$ 1.3	20 $\pm$ 0.5	18 $\pm$ 1.5	18 $\pm$ 0.7
20 classes	OASIS	MCML	LEGO	LMNN	Euclidean
Mean avg. prec	**21** $\pm$ 1.4	17 $\pm$ 1.2	16 $\pm$ 1.2	14 $\pm$ 0.6	14 $\pm$ 0.7
Top 1 prec.	**29** $\pm$ 2.6	26 $\pm$ 2.3	26 $\pm$ 2.7	26 $\pm$ 3.0	25 $\pm$ 2.6
Top 10 prec.	**24** $\pm$ 1.9	21 $\pm$ 1.5	20 $\pm$ 1.4	19 $\pm$ 1.0	18 $\pm$ 1.0
Top 50 prec.	**15** $\pm$ 0.4	14 $\pm$ 0.5	13 $\pm$ 0.6	11 $\pm$ 0.2	12 $\pm$ 0.2
50 classes	OASIS	MCML	LEGO	LMNN	Euclidean
Mean avg. prec.	**12** $\pm$ 0.4	$*$	9 $\pm$ 0.4	8 $\pm$ 0.4	9 $\pm$ 0.4
Top 1 prec.	**21** $\pm$ 1.6	$*$	18 $\pm$ 0.7	18 $\pm$ 1.3	17 $\pm$ 0.9
Top 10 prec.	**16** $\pm$ 0.4	$*$	13 $\pm$ 0.6	12 $\pm$ 0.5	13 $\pm$ 0.4
Top 50 prec.	**10** $\pm$ 0.3	$*$	8 $\pm$ 0.3	7 $\pm$ 0.2	8 $\pm$ 0.3

Table 4: Mean average precision and precision at top 1, 10, and 50 of all compared methods. Values are averages over 5 cross validation folds; $\pm$ values are the standard deviation across the 5 folds. A '*' denotes cases where a method took more than 5 days to converge.

classes	OASIS Matlab	OASIS Matlab+C	MCML Matlab+C	LEGO Matlab	LMNN (naive) Matlab+C	fast-LMNN Matlab+C
10	42 $\pm$ 15	0.12 $\pm$ .03	1835 $\pm$ 210	143 $\pm$ 44	337 $\pm$ 169	247 $\pm$ 209
20	45 $\pm$ 8	0.15 $\pm$ .02	7425 $\pm$ 106	533 $\pm$ 49	631 $\pm$ 40	365 $\pm$ 62
50	25 $\pm$ 2	1.6 $\pm$ .04	$*$	711 $\pm$ 28	960 $\pm$ 80	2109 $\pm$ 67
249	485 $\pm$ 113	1.13 $\pm$ .15	$*$	$**$	$**$	$**$

Table 5: Runtime (minutes) of all compared methods. Values are averages over 5 cross validation folds, $\pm$ values are the standard deviation across the 5 folds. A '$*$' denotes cases where a method took more than 5 days to converge. A '$**$' denotes cases where performance was worse than the Euclidean baseline.

5.4 Parallel Training

We presented OASIS as optimizing an objective function at each step. Since OASIS is based on the PA framework, it is also known to minimize a global objective of the form

$$\|\mathbf{W}\|_{Fro}^2 + C \sum_i l_i$$

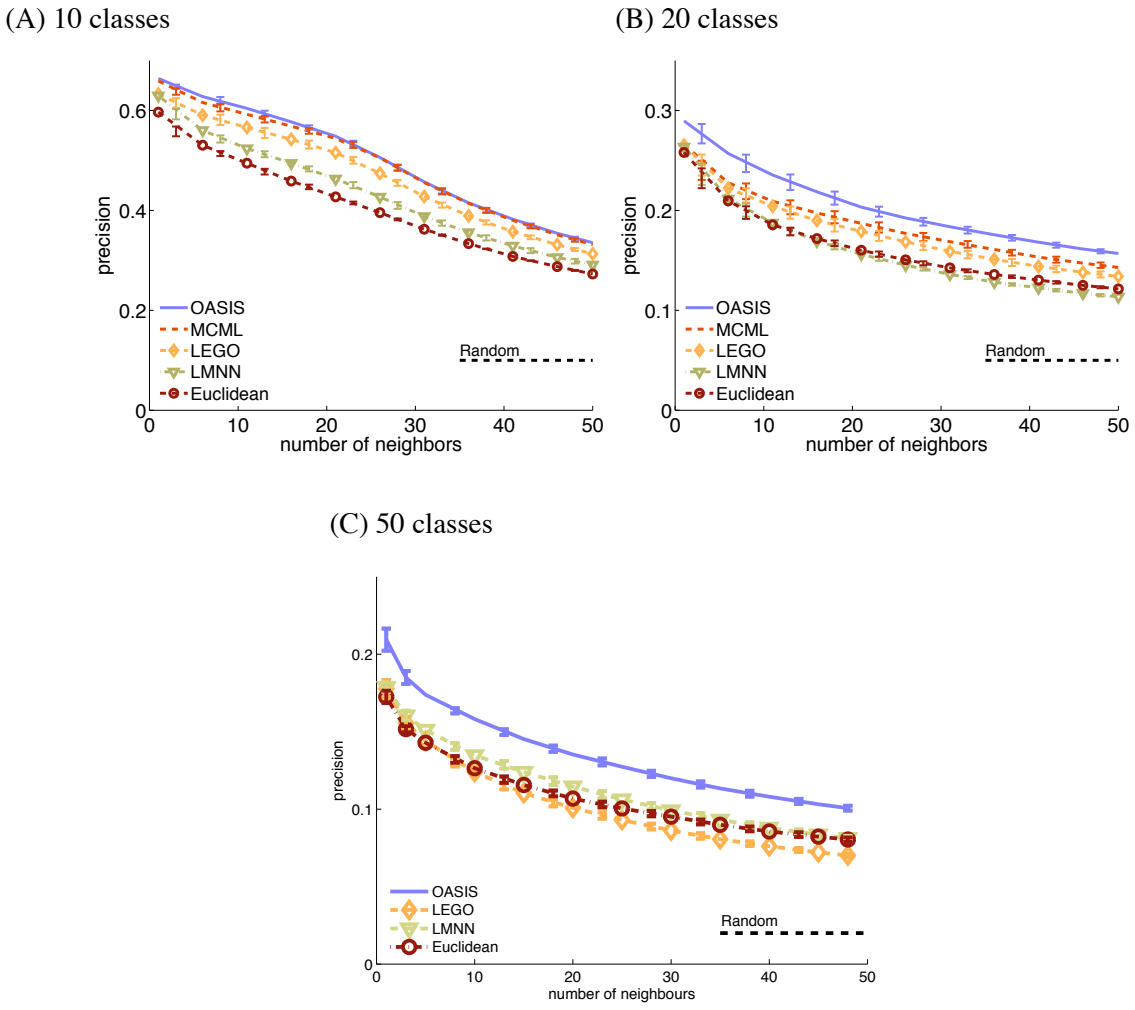


Figure 6: Comparison of the performance of OASIS, LMNN, MCML, LEGO and the Euclidean metric in feature space. Each curve shows the precision at top k as a function of k neighbors. The results are averaged across 5 train/test partitions (40 training images, 25 test images), error bars are standard error of the means (s.e.m.), black dashed line denotes chance performance. **(A)** 10 classes. **(B)** 20 classes. **(C)** 50 classes.

as shown by Crammer et al. (2006) This objective is convex since the losses l_i are linear in $\mathbf{W}$. For such convex functions, it is guaranteed that any linear combination of solutions is superior than each of the individual solutions. This property suggests another way to speed up training, by training multiple rankers in parallel and averaging the resulting models. Each of the individual models can be trained with a smaller number of iterations. Note however that there is no guarantee that the total CPU time is improved.

Figure 7 demonstrates this approach; we trained 5 or 10 rankers in parallel and plot the test set mean average precision as a function of the number of training iterations.

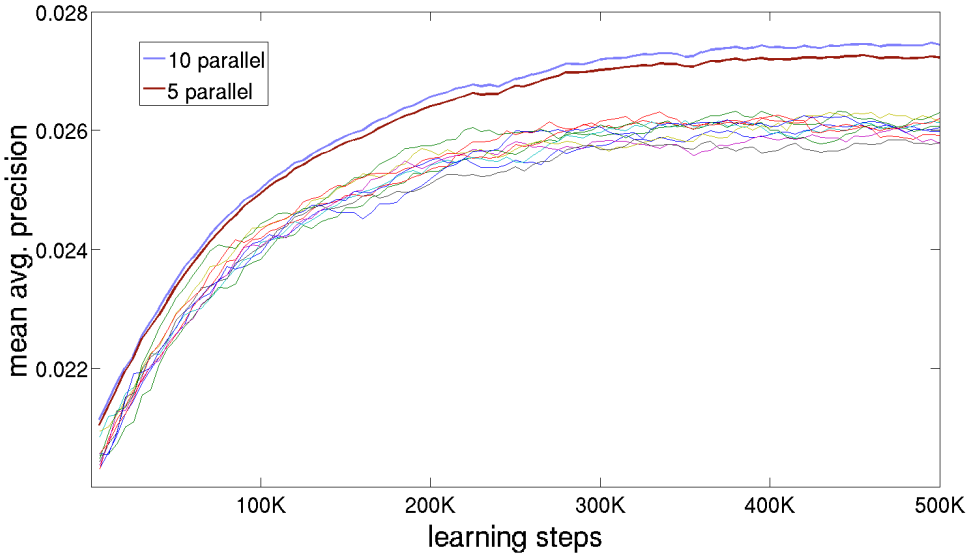


Figure 7: Comparing individual rankers and a linear combination of 5 and 10 rankers. Results are for an experiment with 249 classes of the Caltech256 data set.

.

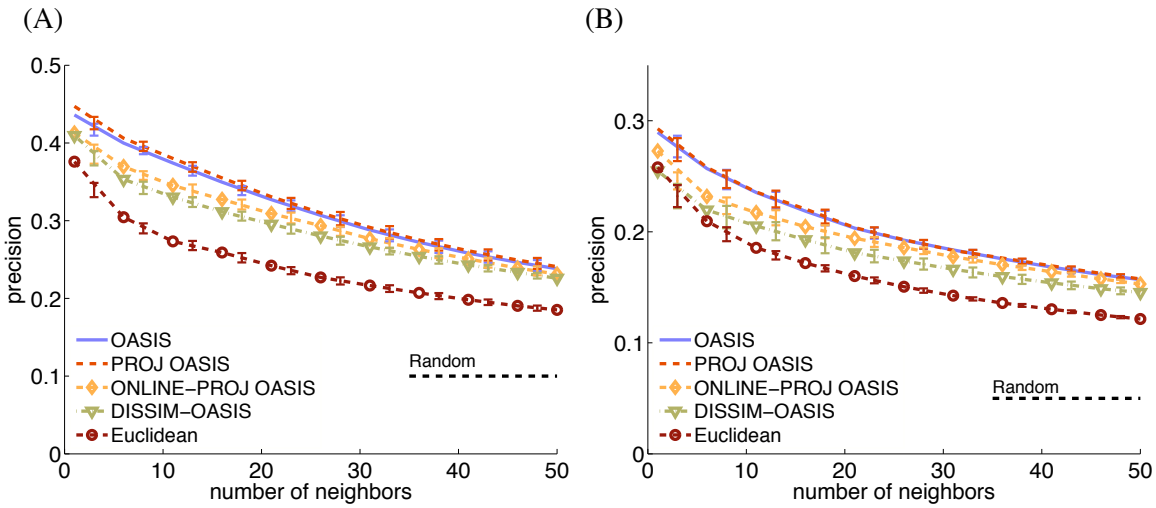


Figure 8: Comparison of Symmetric variants of OASIS. **(A)** 10 classes. **(B)** 20 classes.

6. Symmetry and Positivity

The similarity matrix **W** learned by OASIS is not guaranteed to be positive or even symmetric. Some applications, like ranking images by semantic relevance to a given image query are known to be non-symmetric when based on human judgement (Tversky, 1977). However, in some applications symmetry or positivity constraints reflect a prior knowledge that may help avoiding overfitting.

Furthermore positive $\mathbf{W}$ impose a Mahalanobis metric over the data, that can be further factorized to extract a linear projection of the data into a Euclidean space: $\mathbf{x}^T\mathbf{W}\mathbf{y} = (\mathbf{A}\mathbf{x})^T(\mathbf{A}\mathbf{y})$ such that $\mathbf{A}^T\mathbf{A} = \mathbf{W}$. Such projection $\mathbf{A}$ of the data can be useful for visualization and exploratory analysis of data for example in scientific applications. We now discuss variants of OASIS that learn a symmetric or positive matrices.

6.1 Symmetric Similarities

A simple approach to enforce symmetry is to project the OASIS model $\mathbf{W}$ onto the set of symmetric matrices $\mathbf{W}' = sym(\mathbf{W}) = \frac{1}{2}\left(\mathbf{W}^T + \mathbf{W}\right)$. The update procedure then consists of a series of gradient steps followed by projection to the feasible set (of symmetric matrices). This approach is sometimes called projected gradient, and we denote it here *Online-Proj-Oasis*. Alternatively, projection can also be applied after learning is completed (denoted here *Proj-Oasis*).

Alternatively, the asymmetric score function $S_{\mathbf{W}}(p_i, p_j)$ in the loss l_W can be replaced with a symmetric score

$$S'_{\mathbf{W}}(p_i, p_j) \equiv -(p_i - p_j)^T \mathbf{W} (p_i - p_j) .$$

and derive an OASIS-like algorithm (which we call *Dissim-Oasis*). The optimal update for this loss has a symmetric gradient $\mathbf{V}'^i = (p_i - p_i^+)(p_i - p_i^+)^T - (p_i - p_i^-)(p_i - p_i^-)^T$. Therefore, if $\mathbf{W}^0$ is initialized with a symmetric matrix (for example, the identity matrix) all $\mathbf{W}^i$ are guaranteed to remain symmetric. *Dissim-Oasis* is closely related to LMNN (Weinberger et al., 2006). This can be seen be casting the batch objective of LMNN, into an online setup, which has the form $err(W) = -\omega \cdot S'_{\mathbf{W}}(p_i, p_i^+) + (1-\omega) \cdot l'_{\mathbf{W}}(p_i, p_i^+, p_i^-)$. This online version of LMNN becomes equivalent to *D*issim-Oasis for $\omega = 0$.

Figure 8 compares the precision of the different symmetric methods with the original OASIS. All symmetric variants performed slightly worse, or equal to the original asymmetric OASIS. Asymmetric OASIS is also twice faster than DISSIM-OASIS. The precision of *Proj-Oasis* was equivalent to that of OASIS. This was because the asymmetric OASIS learning rule actually converged to an almost-symmetric model (as measured by a symmetry index $\rho(\mathbf{W}) = \frac{\|sym(\mathbf{W})\|_2}{\|\mathbf{W}\|_2} = 0.94$).

6.2 Positive Similarity

Most similarity learning approaches focus on learning metrics. In the context of OASIS, when $\mathbf{W}$ is positive semi definite (PSD), it defines a Mahalanobis distance over the images. The matrix square-root of $\mathbf{W}$, $\mathbf{A}^T\mathbf{A} = \mathbf{W}$ can then be used to project the data into a new space in which the Euclidean distance is equivalent to the $\mathbf{W}$ distance in the original space.

We experimented with positive variants of OASIS, where we repeatedly projected the learned model onto the set of PSD matrices, once every t iterations. Projection is done by taking the eigen decomposition $\mathbf{W} = \mathbf{V} \cdot \mathbf{D} \cdot \mathbf{V}^T$ where $\mathbf{V}$ is the eigenvector matrix and $\mathbf{D}$ is the diagonal eigenvalues matrix limited to positive eigenvalues. Figure 9 traces precision on the test set throughout learning for various values of t.

The effect of positive projections is complex. First, continuously projecting once every few steps helps to reduce overfitting, as can be observed by the slower decline of the blue curve (upper smooth curve) compared to the orange curve (lowest curve). However, when projection is performed after many steps (instead of continuously), performance of the projected model actually outperforms the continuous-projection model (upper jittery curve). The reason for this effect is likely to be that the

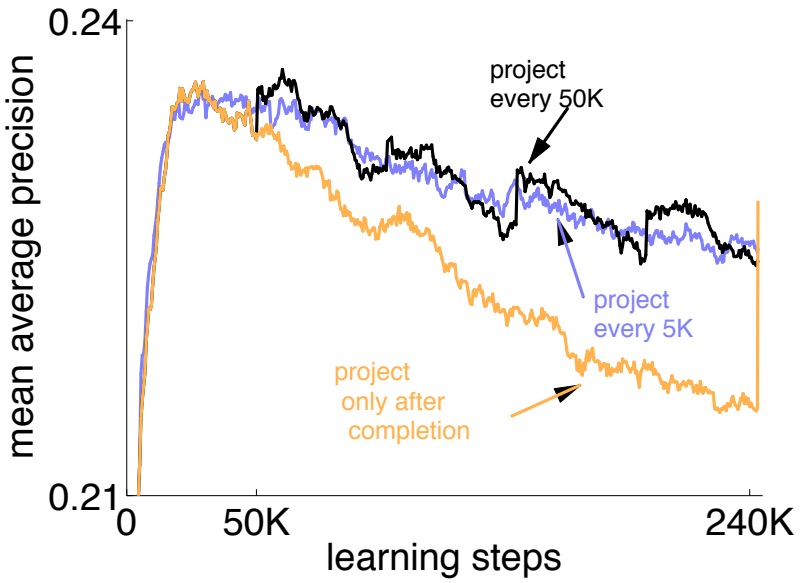


Figure 9: Mean average precision (mAP) during training for three PSD projection schemes, using the set of 20 classes from caltech256.

estimates of the positive sub-space are very noisy when only based on a few samples (see also Chen et al. 2009, Section 2.1). Indeed, accurate estimation of the negative subspace is known to be a hard problem, because small perturbations can turn a negative but small eigenvalue, into a small but positive one. As a result, the set of vectors selected based on having positive eigenvalues, is highly variable. We found that this effect was so strong, that the optimal projection strategy is to avoid projection throughout learning completely. Instead, projecting into PSD after learning (namely, after a model was chosen using early stopping) provided the best performance in our experiments.

An interesting alternative to obtain a PSD matrix was explored by Kulis et al. (2009) and Jain et al. (2008a). Using a LogDet divergence between two matrices $D_{ld}(X,Y) = tr(XY^{-1}) - log(det(XY^{-1}))$ ensures that, given an initial PSD matrix, all subsequent matrices will be PSD as well. It would be interesting to test the effect of using LogDet regularization in the OASIS setup.

7. Discussion

We have presented OASIS, a scalable algorithm for learning image similarity that captures both semantic and visual aspects of image similarity. Three key factors contribute to the scalability of OASIS. First, using a large margin online approach allows training to converge even after seeing a small fraction of potential pairs. Second, the objective function of OASIS does not require the similarity measure to be necessarily a metric during training, although it appears to naturally converge to a symmetric solution. Finally, we use a sparse representation of low level features which allows computing scores very efficiently.

We found that OASIS performs well in a wide range of scales: from problems with thousands of images, where it slightly outperforms existing metric-learning approaches, to large web-scale problems, where it achieves high accuracy, as estimated by human evaluators.

OASIS differs from previous methods in that the similarity measure that it learns is not forced to be a metric, or even symmetric. When the number of available samples is small, it is useful to add constraints that reflect prior knowledge on the type of similarity measure expected to be learned. However, we found that these constraints were not helpful even for problems with a few hundreds of samples. Interestingly, human judgements of pairwise similarity are known to be asymmetric, a property that can be easily captured by an OASIS model.

OASIS learns a class-independent model: it is not aware of which queries or categories were shared by two similar images. As such, it is more limited in its descriptive power and it is likely that class-dependent similarity models could improve precision. On the other hand, class-independent models could generalize to handle classes that were not observed during training, as in transfer learning. Large scale similarity learning, applied to images from a large variety of classes, could therefore be a useful tool to address real-world problems with a large number of classes.

Acknowledgments

This work was supported by the Israeli Science Foundation (ISF 1001/08). We thank Andrea Frome for very helpful discussions and comments on the manuscript. We thank Amir Globerson, Killian Weinberger and Prateek Jain, each providing an implementation of their method for our experiments.

Appendix A. Human Evaluation

The following text was given as instructions to human evaluators when judging the relevance of images to a query image.

```
Scenario:
     A user is searching images to use in a presentation he/she plans to
  give.  The user runs a standard image search, and selects an image,
  the ''query image''.  The user then wishes to refine the search and
  look for images that are SEMANTICALLY similar to the query image.
     The difficulty lies, in the definition of ''SEMANTICALLY''. This can
  have many interpretations, and you should take that into account.
     So for instance, if you see an image of a big red truck, you can
  interpret the user intent (the notion of semantically similar) in
  various ways:
    - any big red truck
    - any red truck
    - any big truck
    - any truck
    - any vehicle
  You should interpret ''SEMANTICALLY'' in a broad sense rather than
  in a strict sense but feel free to draw the line yourself (although
```

```
  be consistent).

Your task:
     You will see a set of query images on the left side of the screen,
  and a set of potential candidate matches, 5 per row, on the
  right. Your job is to decide for each of the candidate images if it
  is a good semantic match to the query image or not. The default is
  that it is NOT a good match. Furthermore, if for some reason you
  cannot make-up your mind, then answer ''can't say''.
```

Appendix B. Caltech256 Class Sets

- **10 classes**: *bear, skyscraper, billiards, yo-yo, minotaur, roulette-wheel, hamburger, laptop-101, hummingbird, blimp*.
- **20 classes***: airplanes-101, mars, homer-simpson, hourglass, waterfall, helicopter-101, mountain-bike starfish-101, teapot, pyramid, refrigerator, cowboy-hat, giraffe, joy-stick, crab-101, bird-bath, fighter-jet tuning-fork, iguana, dog*.
- **50 classes***: car-side-101, tower-pisa, hibiscus, saturn, menorah-101, rainbow, cartman, chandelier-101, backpack, grapes, laptop-101, telephone-box, binoculars, helicopter-101, paper-shredder, eiffel-tower, top-hat, tomato, star-fish-101, hot-air-balloon, tweezer, picnic-table, elk, kangaroo-101, mattress, toaster, electric-guitar-101, bathtub, gorilla, jesus-christ, cormorant, mandolin, light-house, cake, tricycle, speed-boat, computer-mouse, superman, chimp, pram, fried-egg, fighter-jet, unicorn, greyhound, grasshopper, goose, iguana, drinking-straw, snake, hot-dog*.
- **249 classes***: classes 1-250, excluding class 129 (leopards-101), which had less than 65 large enough images*.

References

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proc. of 20th International Conference on Machine Learning (ICML)*, page 11, 2003.

L. Bottou. Large-scale machine learning and stochastic algorithms. In *NIPS 2008 Workshop on Optimization for Machine Learning*, 2008.

Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *The Journal of Machine Learning Research*, 10:747–776, 2009.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research (JMLR)*, 7:551–585, 2006.

J.V. Davis, B. Kulis, P. Jain, S. Sra, and I.S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM Press New York, NY, USA, 2007.

P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *European Conference on Computer Vision (ECCV)*, pages 97–112, 2002.

S.L. Feng, R. Manmatha, and V. Lavrenko. Multiple Bernoulli relevance models for image and video annotation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *International Conference on Computer Vision*, pages 1–8, 2007.

A. Globerson and S. Roweis. Metric learning by collapsing classes. *Advances in Neural Information Processing Systems*, 18:451, 2006.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(8):1371–1384, 2008.

D. Grangier, F. Monay, and S. Bengio. Learning to retrieve images from text queries with a discriminative model. In *International Conference on Adaptive Multimedia Retrieval (AMR)*, 2006.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2006.

P. Jain, B. Kulis, I. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in Neural Information Processing Systems*, volume 22, 2008a.

P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008b.

J. Jeon and R. Manmatha. Using maximum entropy for automatic image annotation. In *International Conference on Image and Video Retrieval*, pages 24–32, 2004.

B. Kulis, M.A. Sustik, and I.S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *Journal of Machine Learning Research*, 10:341–376, 2009.

G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research (JMLR)*, 5:27–72, 2004.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

W.S. Noble. Multi-kernel learning for biology. In *NIPS 2008 workshop on kernel learning*, 2008.

T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(7):971–987, 2002.

P. Quelhas, F. Monay, J. M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. J. Van Gool. Modeling scenes with local descriptors and latent aspects. In *International Conference on Computer Vision*, pages 883–890, 2005.

N. Rasiwasia and N. Vasconcelos. A study of query by semantic example. In *3rd International Workshop on Semantic Learning and Applications in Multimedia*, 2008.

R. Rosales and G. Fung. Learning sparse metrics via linear programming. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 367–373. ACM New York, NY, USA, 2006.

M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*. Bradford Book, 2004.

V. Takala, T. Ahonen, and M. Pietikainen. Block-based methods for image retrieval using local binary patterns. In *Scandinavian Conference on Image Analysis (SCIA)*, 2005.

K. Tieu and P. Viola. Boosting image retrieval. *International Journal of Computer Vision (IJCV)*, 56(1):17 – 36, 2004.

A. Torralba, R. Fergus, and W. T. Freeman. Tiny images. Technical Report MIT-CSAIL-TR-2007-024, Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 2007. URL `http://dspace.mit.edu/handle/1721.1/37291`.

A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Advances in Neural Information Processing Systems*, 18:1473, 2006.

K.Q. Weinberger and L.K. Saul. Fast solvers and efficient implementations for distance metric learning. In *ICML25*, pages 1160–1167, 2008.

E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 521–528, Cambridge, MA, 2003. MIT Press.

L. Yang. Distance metric learning: A comprehensive survey. Technical report, Michigan State University, 2006.

Journal of Machine Learning Research 11 (2010) 1137-1140 Submitted 10/09; Revised 1/10; Published 3/10

Continuous Time Bayesian Network Reasoning and Learning Engine

Christian R. Shelton CSHELTON@CS.UCR.EDU
Yu Fan YFAN@CS.UCR.EDU
William Lam WLAM@CS.UCR.EDU
Joon Lee JLEE133@CS.UCR.EDU
Jing Xu JINGXU@CS.UCR.EDU
Department of Computer Science and Engineering
University of California
Riverside, CA 92521, USA

Editor: Soeren Sonnenburg

Abstract

We present a continuous time Bayesian network reasoning and learning engine (CTBN-RLE). A continuous time Bayesian network (CTBN) provides a compact (factored) description of a continuous-time Markov process. This software provides libraries and programs for most of the algorithms developed for CTBNs. For learning, CTBN-RLE implements structure and parameter learning for both complete and partial data. For inference, it implements exact inference and Gibbs and importance sampling approximate inference for any type of evidence pattern. Additionally, the library supplies visualization methods for graphically displaying CTBNs or trajectories of evidence.

Keywords: continuous time Bayesian networks, C++, open source software

1. Introduction

Continuous time Bayesian networks (CTBNs) represent a continuous-time finite-state Markov process compactly factored according to a graph (Nodelman et al., 2002). The initial distribution of the process is represented as a Bayesian network. The dynamics of the process is also factorized according to a directed graph, but this graph may contain cycles. The edges in this second graph represent causal influence between variables of the system.

CTBNs compactly represent the dynamics differently than models in queueing theory (Bolch et al., 1998), Petri nets (Petri, 1962), or matrix diagrams (Ciardo and Miner, 1999). The representation and algorithms developed so far for CTBNs emphasize reasoning about the transient properties over the steady-state of the system. Unfortunately, previously there were no commonly available software packages implementing CTBN algorithms. Their implementation requires a few critical numerical algorithms, thus making it difficult to quickly try the representation without prior experience.

This software package aims to reduce this barrier to entry by supplying our implementations of these methods in a complete object-oriented design. The software does not require any external libraries. It is implemented in C++ with demonstration programs for common functionality and a documented interface for users to develop their own programs. The class hierarchy was designed with extensions to allow for further innovation.

2. Continuous Time Bayesian Networks

The dynamics of a continuous-time n-state Markov process are often described with an n-by-n intensity (or rate) matrix, Q with elements q_{ij}. The diagonal elements are non-positive and correspond to the parameters of the exponential distributions describing the duration of time the process stays in each state. Therefore the expected duration in state i is $\frac{1}{-q_{ii}}$. All other elements are non-negative and each row sums to zero. The probability of transitioning from state i to state j is proportional to q_{ij}.

A continuous time Bayesian network (CTBN) consists of a set of variables, $\mathcal{X}$, an initial distribution P_0 over $\mathcal{X}$ specified as a Bayesian network, and a graph-factored model of the dynamics of the system which is composed of two parts: (1) a directed (possibly cyclic) graph $\mathcal{G}$ over $\mathcal{X}$ and (2) conditional intensities $\mathbf{Q}_{X|\mathbf{U}}$ for each variable $X \in \mathcal{X}$ with parent set $\mathbf{U}$ in the graph $\mathcal{G}$. $\mathbf{Q}_{X|\mathbf{U}}$ is defined as a set of intensity matrices $\mathbf{Q}_{X|\mathbf{u}}$ for each assignment $\mathbf{u}$ to the variables $\mathbf{U}$. At any instant, the evolution of variable X is governed by the intensity matrix $\mathbf{Q}_{X|\mathbf{u}}$ if $\mathbf{u}$ is the current assignment to the parents of X.

3. Engine Components

In terms of data structures, the library supplies classes for storing and efficiently scanning multivariate trajectories, for representing both the initial distribution (as a Bayesian network) and the dynamics (as a directed graph with associated conditional rate matrices) of a CTBN, for representing the associated sufficient statistics, and for drawing samples from the CTBN process.

3.1 Inference Methods

Inference for CTBNs can take a number of forms. The common three types of queries are all implemented and additional query types can be added (through subclassing) without knowledge of the details of the inference algorithms. In particular, code is supplied for

- querying the marginal distribution of a variable at a particular time (filtering or smoothing),
- querying the expected number of transitions for a variable during an interval of time, and
- querying the expected amount of time a variable stayed in a particular state during an interval.

All are conditioned on a (possibly incomplete) trajectory of events (transitions of the states of the variables of the process). The latter two represent the calculations necessary to compute expected sufficient statistics.

Exact inference (which takes exponential time in terms of the number of variables) is implemented. Additionally, two approximate inference methods based on sampling are also implemented: Gibbs sampling (El-Hay et al., 2008) and importance sampling (Fan and Shelton, 2008).

3.2 Learning

All learning methods estimate both the dynamics graph and the Bayesian network of the initial distribution. For the latter, this is just standard Bayesian network learning which exists in other packages; however, we supply our implementation here for simplicity and to avoid relying on other software packages.

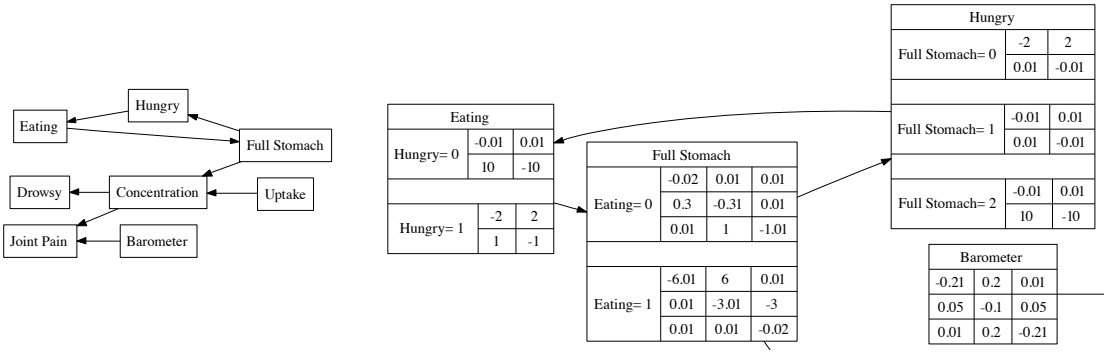


Figure 1: (Left) Automatic layout of the drug effect network without parameters. (Right) Portion of automatic layout with parameters.

Maximum likelihood parameter learning is implemented both for complete data (Nodelman et al., 2003) and for incomplete data (Nodelman et al., 2005) via the expectation-maximization algorithm. Structure learning is also implemented for both complete and incomplete data. The latter represents an implementation of structural expectation maximization. Two structure searches are available: a brute force search that tries all parent sets up to a certain size (not possible for the initial distribution due to acyclic constraints) and a graph edit search that makes local changes to the graph.

3.3 Modularity

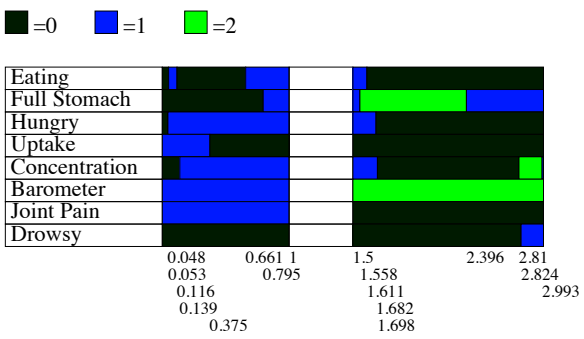
Any supplied (or user-defined) inference method can be used in expectation-maximization and structural expectation maximization. Different proposal distributions for importance sampling can be added through simple subclassing. Similarly, the code allows for the construction of CTBNs from any underlying process type. As an example, we define "toggle" variables (only the *change* of state has meaning) through a very short subclass that implements the necessary parameter tieing. This new process can be mixed freely with other processes to create CTBNs of mixed node types.

3.4 Visualization

Two visualization tools are supplied. The first converts a CTBN into a text file suitable to be read by the open source package `graphviz` which can then layout the CTBN in a variety of formats. Figure 1 shows the output of this automatic visualization on the drug effect network from Nodelman et al. (2002). Additionally, either an postscript or text visualization of a trajectory can be automatically generated. Figure 2 shows these outputs for a partially observed trajectory drawn from the same drug effect network.

Acknowledgments

This project was supported by the Air Force Office of Scientific Research (FA9550-07-1-0076) and by the Defense Advanced Research Project Agency (HR0011-09-1-0030). We thank Uri Nodelman



time \ var	0	1	2	3	4	5	6	7
0.000000	0	0	0	1	0	1	1	0
0.048261			1					
0.053800	1							
0.116566	0							
0.139465					1			
0.375521				0				
0.661087	1							
0.795507		1						
1.000000	-1	-1	-1	-1	-1	-1	-1	-1
1.500000	1	1	1	0	1	2	0	0
1.558340		2						
1.611520	0							
1.682240			0					
1.698080					0			
2.396950		1						
2.810210					2			
2.824950								1
2.993380					1			

Figure 2: Automatic visualizations of a trajectory of the drug effect network (with all variables missing observations from $t = 1$ to 1.5), as a postscript file (left) and in text (right).

for sharing his initial implementation of CTBNs and Tal El-Hay for sharing his implementation of Gibbs sampling for CTBNs. Any faults in our software package are purely our own.

References

Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi. *Queueing Networks and Markov Chains*. John Wiley & Sons, Inc., 1998.

Gianfranco Ciardo and Andrew S. Miner. A data structure for the efficient Kronecker solution of GSPNs. In *Proceedings of the 8th International Workshop on Petri Nets and Performance Models*, pages 22–31, 1999.

Tal El-Hay, Nir Friedman, and Raz Kupferman. Gibbs sampling in factorized continuous-time Markov processes. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 169–178, 2008.

Yu Fan and Christian R. Shelton. Sampling for approximate inference in continuous time Bayesian networks. In *Proceedings of the Tenth International Symposium on Artificial Intelligence and Mathematics*, 2008.

Uri Nodelman, Christian R. Shelton, and Daphne Koller. Continuous time Bayesian networks. In *Proceedings of the Eighteenth International Conference on Uncertainty in Artificial Intelligence*, pages 378–387, 2002.

Uri Nodelman, Christian R. Shelton, and Daphne Koller. Learning continuous time Bayesian networks. In *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence*, pages 451–458, 2003.

Uri Nodelman, Christian R. Shelton, and Daphne Koller. Expectation maximization and complex duration distributions for continuous time Bayesian networks. In *Proceedings of the Twenty-First International Conference on Uncertainty in Artificial Intelligence*, pages 421–430, 2005.

Carl A. Petri. *Kommunikation mit Automaten*. PhD thesis, University of Bonn, 1962.

Journal of Machine Learning Research 11 (2010) 1141-1144 Submitted 6/09; Published 3/10

SFO: A Toolbox for Submodular Function Optimization

Andreas Krause KRAUSEA@CALTECH.EDU
Computer Science
California Institute of Technology
Pasadena, CA 91125 USA

Editor: Soeren Sonnenburg

Abstract

In recent years, a fundamental problem structure has emerged as very useful in a variety of machine learning applications: Submodularity is an intuitive diminishing returns property, stating that adding an element to a smaller set helps more than adding it to a larger set. Similarly to convexity, submodularity allows one to efficiently find provably (near-) optimal solutions for large problems. We present SFO, a toolbox for use in MATLAB or Octave that implements algorithms for minimization and maximization of submodular functions. A tutorial script illustrates the application of submodularity to machine learning and AI problems such as feature selection, clustering, inference and optimized information gathering.

1. Introduction

Convex optimization has become a powerful tool in machine learning: Surprisingly, many problems that intuitively require the optimization of highly multi-modal objectives, such as clustering and non-linear classification, can be reduced to convex programs, allowing efficient and optimal solution. More formally, they require finding a solution $\mathbf{x}^* \in \mathbb{R}^d$:

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} g(\mathbf{x}) \text{ s.t. } \mathbf{x} \in \mathfrak{F},$$

where g is a convex function, and $\mathfrak{F} \subseteq \mathbb{R}^d$ is a (convex) set of feasible solutions.

However, many optimization problems in machine learning, such as feature selection, structure learning and inference in discrete graphical models, require finding solutions to *combinatorial* optimization problems: They can be reduced to the problem

$$\mathcal{A}^* = \operatorname*{argmin}_{\mathcal{A} \subseteq \mathcal{V}} F(\mathcal{A}) \text{ s.t. } \mathcal{A} \in \mathfrak{F},$$

where F is a set function $F : 2^{\mathcal{V}} \to \mathbb{R}$ defined over a finite set $\mathcal{V}$, and $\mathfrak{F} \subseteq 2^{\mathcal{V}}$ is a collection of feasible subsets of $\mathcal{V}$, for example, all sets of size at most k, $\mathfrak{F} = \{\mathcal{A} \subseteq \mathcal{V} : |\mathcal{A}| \leq k\}$.

In many machine learning problems, the function F satisfies *submodularity*, an intuitive diminishing returns property, stating that adding an element to a smaller set helps more than adding it to a larger set. Formally, for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and $s \in \mathcal{V} \setminus \mathcal{B}$ it must hold that $F(\mathcal{A} \cup \{s\}) - F(\mathcal{A}) \geq F(\mathcal{B} \cup \{s\}) - F(B)$. Similarly to convexity, submodularity allows one to efficiently find provably (near-) optimal solutions for large problems. Interestingly, for submodular functions, guarantees can be obtained both for minimization and for maximization problems. This is important, since applications require both minimization (e.g., in clustering, inference and structure learning) and maximization (e.g., in feature selection and optimized information gathering). We present SFO, a

toolbox[1] for use in MATLAB or Octave that implements various algorithms for *minimization and maximization* of submodular functions. Examples illustrate the application of submodularity to machine learning and AI problems such as clustering (Narasimhan et al., 2005), inference in graphical models (Kolmogorov and Zabih, 2004) and optimized information gathering (Krause et al., 2006).

2. Implementation of Submodular Functions

The SFO toolbox includes several examples of submodular functions. It is also easily extendable with additional functions. The ground set $\mathcal{V}$ is implemented as a MATLAB array. Submodular functions are implemented as MATLAB objects, inheriting from sfo_fn. The following shows example code defining the submodular function

$$F(\mathcal{A}) = I(\mathcal{X}_\mathcal{A}; \mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}) = H(\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}) - H(\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}} \mid \mathcal{X}_\mathcal{A}),$$

that is, the mutual information between a set of random variables $\mathcal{X}_\mathcal{A}$ and its complement $\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}$, based on a joint multivariate normal distribution $P(\mathcal{X}_\mathcal{V} = \mathbf{x}_\mathcal{V}) = \mathcal{N}(\mathbf{x}_\mathcal{V}; 0, \Sigma)$ with covariance matrix $\Sigma \in \mathbb{R}^{100\times 100}$:

```
V = 1:100;
F = sfo_fn_mi(Sigma,V);
F(1:3) % evaluate F on set A=[1,2,3]
```

This objective function has been used for experimental design in Gaussian processes (Krause et al., 2008), structure learning (Narasimhan and Bilmes, 2004) and clustering (Narasimhan et al., 2005). Often, algorithms require computing marginal increments

$$\delta_s^+(\mathcal{A}) = F(\mathcal{A}\cup\{s\}) - F(\mathcal{A}) \text{ and } \delta_s^-(\mathcal{A}) = F(\mathcal{A}\setminus\{s\}) - F(\mathcal{A}),$$

that is, computing the change in submodular value by adding (removing) an element s from a set $\mathcal{A}$. Often, computing $F(\mathcal{A}\cup\{s\})$ (or $F(\mathcal{A}\setminus\{s\})$) is more efficient when $F(\mathcal{A})$ has already been computed. E.g., for mutual information, incrementally computing $F(\mathcal{A}\cup\{s\})$ requires up-/downdating of the Cholesky decomposition of covariance matrix $\Sigma_{\mathcal{A}\mathcal{A}}$. To speed up computation, the submodular function objects in SFO support methods inc and dec:

```
F = init(F,1:5); % cache computation of F(1:5)
inc(F,1:5,9) % efficient evaluation of F([1:5 9])
dec(F,1:5,3) % efficient evaluation of F([1:2 4:5])
```

The SFO toolbox implements several other examples of submodular functions, including

sfo_fn_entropy	Entropy of multivariate Gaussians
sfo_fn_infogain	Information gain for multivariate Gaussians
sfo_fn_mi	Mutual information in multivariate Gaussians
sfo_fn_varred	Variance reduction in multivariate Gaussians
sfo_fn_detect	Improvement in detection performance
sfo_fn_cutfun	Cut function in graphs
sfo_fn_ising	Energy in ising models with attractive potentials

1. The toolbox is available at `http://www.submodularity.org`.

Creating submodular functions from other submodular functions is also possible, using sfo_fn_lincomb for nonnegative linear combinations, and sfo_fn_trunc for truncation. Custom submodular functions can be used either by inheriting from sfo_fn, or by using the sfo_fn_wrapper function, which wraps a pointer to an anonymous function in a submodular function object. The following example wraps an anonymous function fn which computes, for any set of integers $\mathcal{A}$, the number of distinct remainders modulo 5:

```
fn = @(A) length(unique(mod(A,5)));
F = sfo_fn_wrapper(fn);
F([1 6]) % returns 1
F([1:10]) % returns 5
```

3. Implemented Algorithms for Submodular Function Optimization

SFO implements various algorithms for (constrained) maximization and minimization of submodular functions. Their use is demonstrated in sfo_tutorial and sfo_tutorial_octave.

Minimization of Submodular Functions

- sfo_min_norm_point: The minimum norm point algorithm of Fujishige (2005) for solving $\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A}\subseteq\mathcal{V}} F(\mathcal{A})$ for general submodular functions.
- sfo_queyranne: Algorithm of Queyranne (1995) solving $\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A}\subseteq\mathcal{V}:0<|\mathcal{A}|<|\mathcal{V}|} F(\mathcal{A})$ for symmetric submodular functions (i.e., $F(\mathcal{A}) = F(\mathcal{V}\setminus\mathcal{A})$ for all sets $\mathcal{A}$).
- sfo_ssp: The submodular-supermodular procedure of Narasimhan and Bilmes (2006) for (heuristically) minimizing the difference between two submodular functions $\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A}\subseteq\mathcal{V}} F_1(\mathcal{A}) - F_2(\mathcal{A})$.
- sfo_s_t_min_cut: Solves $\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A}\subseteq\mathcal{V}} F(\mathcal{A})$ s.t. $s \in \mathcal{A}, t \notin \mathcal{A}$.
- sfo_greedy_splitting: The algorithm of Zhao et al. (2005) for submodular clustering

Maximization of Submodular Functions

- sfo_greedy_lazy: The greedy algorithm of Nemhauser et al. (1978) for constrained maximization / coverage, using the lazy evaluation technique of Minoux (1978).
- sfo_cover: Greedy coverage algorithm using lazy evaluations.
- sfo_celf: The CELF algorithm for approximately solving $\mathcal{A}^* = \operatorname{argmax}_{\mathcal{A}} F(\mathcal{A})$ s.t. $C(\mathcal{A}) \leq B$, for linear cost function C (Leskovec et al., 2007).
- sfo_ls_lazy: The (deterministic) local search algorithm of Feige et al. (2007) for unconstrained maximization of nonnegative submodular functions, using lazy evaluations.
- sfo_pspiel: The PSPIEL algorithm of Krause et al. (2006). PSPIEL approximately solves $\mathcal{A}^* = \operatorname{argmax}_{\mathcal{A}} F(\mathcal{A})$ s.t. $C(\mathcal{A}) \leq B$, where $C(\mathcal{A})$ is the cost of a cheapest path connecting the nodes $\mathcal{A}$ in a graph.
- sfo_saturate: The SATURATE algorithm of Krause et al. (2008) for approximately solving the robust optimization problem $\mathcal{A}^* = \operatorname{argmax}_{|\mathcal{A}|\leq k} \min_i F_i(\mathcal{A})$.
- sfo_balance: The ESPASS algorithm for approximately solving the optimization problem $\max_{|\mathcal{A}_1\cup\cdots\cup\mathcal{A}_k|\leq m} \min_i F(\mathcal{A}_i)$ (Krause et al., 2009).
- sfo_max_dca_lazy: The Data Correcting algorithm for maximizing general (not necessarily nondecreasing) submodular functions (Goldengorin et al., 1999).

Acknowledgments

This research was supported by ONR grant N00014-09-1-1044, NSF CNS-0932392, a gift from Microsoft Corporation and an Okawa Foundation Research Grant.

References

U. Feige, V. Mirrokni, and J. Vondrak. Maximizing non-monotone submodular functions. In *FOCS*, 2007.

S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2nd edition, 2005.

B. Goldengorin, G. Sierksma, G. A. Tijssen, and M. Tso. The data-correcting algorithm for the minimization of supermodular functions. *Mgmt Science*, 45(11):1539–1551, 1999.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans Patt An Mach Int (PAMI)*, 26(2):147–159, February 2004.

A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *IPSN*, 2006.

A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. In *JMLR*, volume 9, 2008.

A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin. Simultaneous placement and scheduling of sensors. In *Information Processing in Sensor Networks*, 2009.

J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD*, 2007.

M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. *Optimization Techniques, LNCS*, pages 234–243, 1978.

M. Narasimhan and J. Bilmes. Pac-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence*, 2004.

M. Narasimhan and J. Bilmes. A submodular-supermodular procedure with applications to discriminative structure learning. In *NIPS 19*, 2006.

M. Narasimhan, N. Jojic, and J. Bilmes. Q-clustering. In *NIPS*, 2005.

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

M. Queyranne. A combinatorial algorithm for minimizing symmetric submodular functions. In *SODA*, 1995.

L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 102(1):167–183, 2005.

Journal of Machine Learning Research 11 (2010) 1145-1200 Submitted 11/08; Revised 11/09; Published 3/10

A Quasi-Newton Approach to Nonsmooth Convex Optimization Problems in Machine Learning

Jin Yu JIN.YU@ADELAIDE.EDU.AU
School of Computer Science
The University of Adelaide
Adelaide SA 5005, Australia

S.V.N. Vishwanathan VISHY@STAT.PURDUE.EDU
Departments of Statistics and Computer Science
Purdue University
West Lafayette, IN 47907-2066 USA

Simon Günter GUENTER_SIMON@HOTMAIL.COM
DV Bern AG
Nussbaumstrasse 21, CH-3000 Bern 22, Switzerland

Nicol N. Schraudolph JMLR@SCHRAUDOLPH.ORG
adaptive tools AG
Canberra ACT 2602, Australia

Editor: Sathiya Keerthi

Abstract

We extend the well-known BFGS quasi-Newton method and its memory-limited variant LBFGS to the optimization of nonsmooth convex objectives. This is done in a rigorous fashion by generalizing three components of BFGS to subdifferentials: the local quadratic model, the identification of a descent direction, and the Wolfe line search conditions. We prove that under some technical conditions, the resulting subBFGS algorithm is globally convergent in objective function value. We apply its memory-limited variant (subLBFGS) to L_2-regularized risk minimization with the binary hinge loss. To extend our algorithm to the multiclass and multilabel settings, we develop a new, efficient, exact line search algorithm. We prove its worst-case time complexity bounds, and show that our line search can also be used to extend a recently developed bundle method to the multiclass and multilabel settings. We also apply the direction-finding component of our algorithm to L_1-regularized risk minimization with logistic loss. In all these contexts our methods perform comparable to or better than specialized state-of-the-art solvers on a number of publicly available data sets. An open source implementation of our algorithms is freely available.

Keywords: BFGS, variable metric methods, Wolfe conditions, subgradient, risk minimization, hinge loss, multiclass, multilabel, bundle methods, BMRM, OCAS, OWL-QN

1. Introduction

The BFGS quasi-Newton method (Nocedal and Wright, 1999) and its memory-limited LBFGS variant are widely regarded as the workhorses of smooth nonlinear optimization due to their combination of computational efficiency and good asymptotic convergence. Given a smooth objective

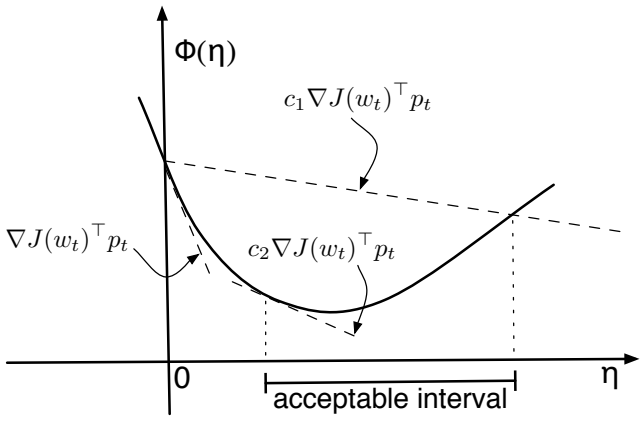


Figure 1: Geometric illustration of the Wolfe conditions (4) and (5).

function $J : \mathbb{R}^d \to \mathbb{R}$ and a current iterate $\boldsymbol{w}_t \in \mathbb{R}^d$, BFGS forms a local quadratic model of J:

$$Q_t(\boldsymbol{p}) \;:=\; J(\boldsymbol{w}_t) + \tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}_t^{-1}\boldsymbol{p} + \nabla J(\boldsymbol{w}_t)^\top \boldsymbol{p}, \tag{1}$$

where $\boldsymbol{B}_t \succ 0$ is a positive-definite estimate of the inverse Hessian of J, and ∇J denotes the gradient. Minimizing $Q_t(\boldsymbol{p})$ gives the quasi-Newton direction

$$\boldsymbol{p}_t := -\boldsymbol{B}_t \nabla J(\boldsymbol{w}_t), \tag{2}$$

which is used for the parameter update:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta_t \boldsymbol{p}_t. \tag{3}$$

The step size $\eta_t > 0$ is normally determined by a line search obeying the Wolfe (1969) conditions:

$$J(\boldsymbol{w}_{t+1}) \;\le\; J(\boldsymbol{w}_t) + c_1\eta_t \nabla J(\boldsymbol{w}_t)^\top \boldsymbol{p}_t \qquad \text{(sufficient decrease)} \tag{4}$$

$$\text{and} \quad \nabla J(\boldsymbol{w}_{t+1})^\top \boldsymbol{p}_t \;\ge\; c_2 \nabla J(\boldsymbol{w}_t)^\top \boldsymbol{p}_t \qquad \text{(curvature)} \tag{5}$$

with $0 < c_1 < c_2 < 1$. Figure 1 illustrates these conditions geometrically. The matrix $\boldsymbol{B}_t$ is then modified via the incremental rank-two update

$$\boldsymbol{B}_{t+1} = (\boldsymbol{I} - \rho_t \boldsymbol{s}_t \boldsymbol{y}_t^\top)\boldsymbol{B}_t(\boldsymbol{I} - \rho_t \boldsymbol{y}_t \boldsymbol{s}_t^\top) + \rho_t \boldsymbol{s}_t \boldsymbol{s}_t^\top, \tag{6}$$

where $\boldsymbol{s}_t := \boldsymbol{w}_{t+1} - \boldsymbol{w}_t$ and $\boldsymbol{y}_t := \nabla J(\boldsymbol{w}_{t+1}) - \nabla J(\boldsymbol{w}_t)$ denote the most recent step along the optimization trajectory in parameter and gradient space, respectively, and $\rho_t := (\boldsymbol{y}_t{}^\top \boldsymbol{s}_t)^{-1}$. The BFGS update (6) enforces the secant equation $\boldsymbol{B}_{t+1}\boldsymbol{y}_t = \boldsymbol{s}_t$. Given a descent direction $\boldsymbol{p}_t$, the Wolfe conditions ensure that $(\forall t)\ \boldsymbol{s}_t^\top \boldsymbol{y}_t > 0$ and hence $\boldsymbol{B}_0 \succ 0 \implies (\forall t)\ \boldsymbol{B}_t \succ 0$.

Limited-memory BFGS (LBFGS, Liu and Nocedal, 1989) is a variant of BFGS designed for high-dimensional optimization problems where the $O(d^2)$ cost of storing and updating $\boldsymbol{B}_t$ would be prohibitive. LBFGS approximates the quasi-Newton direction (2) directly from the last m pairs of

$\boldsymbol{s}_t$ and $\boldsymbol{y}_t$ via a matrix-free approach, reducing the cost to $O(md)$ space and time per iteration, with m freely chosen.

There have been some attempts to apply (L)BFGS directly to nonsmooth optimization problems, in the hope that they would perform well on nonsmooth functions that are convex and differentiable almost everywhere. Indeed, it has been noted that in cases where BFGS (resp., LBFGS) does not encounter any nonsmooth point, it often converges to the optimum (Lemarechal, 1982; Lewis and Overton, 2008a). However, Lukšan and Vlček (1999), Haarala (2004), and Lewis and Overton (2008b) also report catastrophic failures of (L)BFGS on nonsmooth functions. Various fixes can be used to avoid this problem, but only in an ad-hoc manner. Therefore, subgradient-based approaches such as subgradient descent (Nedić and Bertsekas, 2000) or bundle methods (Joachims, 2006; Franc and Sonnenburg, 2008; Teo et al., 2010) have gained considerable attention for minimizing nonsmooth objectives.

Although a convex function might not be differentiable everywhere, a subgradient always exists (Hiriart-Urruty and Lemaréchal, 1993). Let $\boldsymbol{w}$ be a point where a convex function J is finite. Then a subgradient is the normal vector of any tangential supporting hyperplane of J at $\boldsymbol{w}$. Formally, $\boldsymbol{g}$ is called a subgradient of J at $\boldsymbol{w}$ if and only if (Hiriart-Urruty and Lemaréchal, 1993, Definition VI.1.2.1)

$$(\forall \boldsymbol{w}') \;\; J(\boldsymbol{w}') \;\geq\; J(\boldsymbol{w}) + (\boldsymbol{w}' - \boldsymbol{w})^\top \boldsymbol{g}. \tag{7}$$

The set of all subgradients at a point is called the subdifferential, and is denoted $\partial J(\boldsymbol{w})$. If this set is not empty then J is said to be *subdifferentiable at* $\boldsymbol{w}$. If it contains exactly one element, that is, $\partial J(\boldsymbol{w}) = \{\nabla J(\boldsymbol{w})\}$, then J is *differentiable* at $\boldsymbol{w}$. Figure 2 provides the geometric interpretation of (7).

The aim of this paper is to develop principled and robust quasi-Newton methods that are amenable to subgradients. This results in subBFGS and its memory-limited variant subLBFGS, two new subgradient quasi-Newton methods that are applicable to nonsmooth convex optimization problems. In particular, we apply our algorithms to a variety of machine learning problems, exploiting knowledge about the subdifferential of the binary hinge loss and its generalizations to the multiclass and multilabel settings.

In the next section we motivate our work by illustrating the difficulties of LBFGS on nonsmooth functions, and the advantage of incorporating BFGS' curvature estimate into the parameter update. In Section 3 we develop our optimization algorithms generically, before discussing their application to L_2-regularized risk minimization with the hinge loss in Section 4. We describe a new efficient algorithm to identify the nonsmooth points of a one-dimensional pointwise maximum of linear functions in Section 5, then use it to develop an exact line search that extends our optimization algorithms to the multiclass and multilabel settings (Section 6). Section 7 compares and contrasts our work with other recent efforts in this area. We report our experimental results on a number of public data sets in Section 8, and conclude with a discussion and outlook in Section 9.

2. Motivation

The application of standard (L)BFGS to nonsmooth optimization is problematic since the quasi-Newton direction generated at a nonsmooth point is not necessarily a descent direction. Nevertheless, BFGS' inverse Hessian estimate can provide an effective model of the overall shape of a nonsmooth objective; incorporating it into the parameter update can therefore be beneficial. We

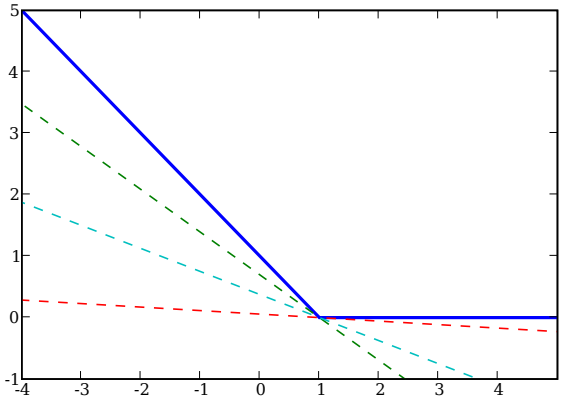


Figure 2: Geometric interpretation of subgradients. The dashed lines are tangential to the hinge function (solid blue line); the slopes of these lines are subgradients.

discuss these two aspects of (L)BFGS to motivate our work on developing new quasi-Newton methods that are amenable to subgradients while preserving the fast convergence properties of standard (L)BFGS.

2.1 Problems of (L)BFGS on Nonsmooth Objectives

Smoothness of the objective function is essential for classical (L)BFGS because both the local quadratic model (1) and the Wolfe conditions (4, 5) require the existence of the gradient ∇J at every point. As pointed out by Hiriart-Urruty and Lemaréchal (1993, Remark VIII.2.1.3), even though nonsmooth convex functions are differentiable everywhere except on a set of Lebesgue measure zero, it is unwise to just use a smooth optimizer on a nonsmooth convex problem under the assumption that "it should work almost surely." Below we illustrate this on both a toy example and real-world machine learning problems.

2.1.1 A TOY EXAMPLE

The following simple example demonstrates the problems faced by BFGS when working with a nonsmooth objective function, and how our subgradient BFGS (subBFGS) method (to be introduced in Section 3) with exact line search overcomes these problems. Consider the task of minimizing

$$f(x,y) = 10\,|x| + |y| \tag{8}$$

with respect to x and y. Clearly, $f(x,y)$ is convex but nonsmooth, with the minimum located at $(0,0)$ (Figure 3, left). It is subdifferentiable whenever x or y is zero:

$$\partial_x f(0,\cdot) = [-10,10] \;\text{ and }\; \partial_y f(\cdot,0) = [-1,1].$$

We call such lines of subdifferentiability in parameter space *hinges*.

We can minimize (8) with the standard BFGS algorithm, employing a backtracking line search (Nocedal and Wright, 1999, Procedure 3.1) that starts with a step size that obeys the curvature

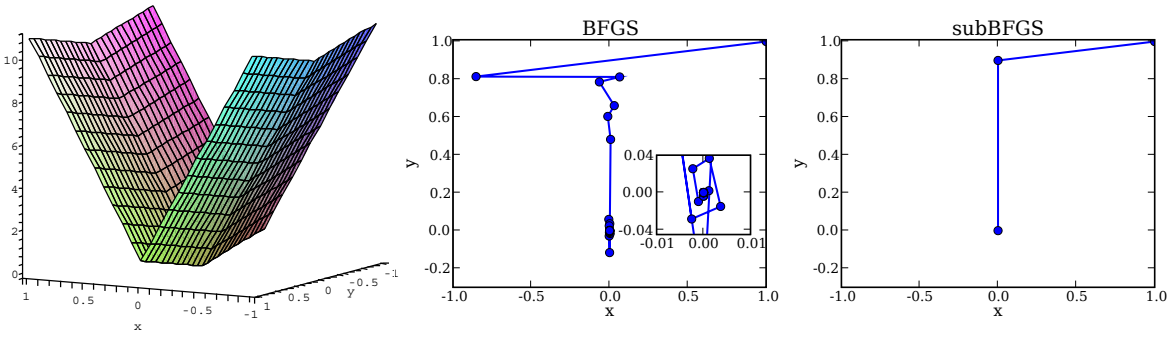


Figure 3: Left: the nonsmooth convex function (8); optimization trajectory of BFGS with inexact line search (center) and subBFGS (right) on this function.

condition (5), then exponentially decays it until both Wolfe conditions (4,5) are satisfied.[1] The curvature condition forces BFGS to jump across at least one hinge, thus ensuring that the gradient displacement vector $\boldsymbol{y}_t$ in (6) is non-zero; this prevents BFGS from diverging. Moreover, with such an *inexact* line search BFGS will generally not step on any hinges directly, thus avoiding (in an ad-hoc manner) the problem of non-differentiability. Although this algorithm quickly decreases the objective from the starting point $(1,1)$, it is then slowed down by heavy oscillations around the optimum (Figure 3, center), caused by the utter mismatch between BFGS' quadratic model and the actual function.

A generally sensible strategy is to use an exact line search that finds the optimum along a given descent direction (cf. Section 4.2.1). However, this line optimum will often lie on a hinge (as it does in our toy example), where the function is not differentiable. If an arbitrary subgradient is supplied instead, the BFGS update (6) can produce a search direction which is not a descent direction, causing the next line search to fail. In our toy example, standard BFGS with exact line search consistently fails after the first step, which takes it to the hinge at $x = 0$.

Unlike standard BFGS, our subBFGS method can handle hinges and thus reap the benefits of an exact line search. As Figure 3 (right) shows, once the first iteration of subBFGS lands it on the hinge at $x = 0$, its direction-finding routine (Algorithm 2) finds a descent direction for the next step. In fact, on this simple example Algorithm 2 yields a vector with zero x component, which takes subBFGS straight to the optimum at the second step.[2]

2.1.2 TYPICAL NONSMOOTH OPTIMIZATION PROBLEMS IN MACHINE LEARNING

The problems faced by smooth quasi-Newton methods on nonsmooth objectives are not only encountered in cleverly constructed toy examples, but also in real-world applications. To show this, we apply LBFGS to L_2-regularized risk minimization problems (30) with binary hinge loss (31), a typical nonsmooth optimization problem encountered in machine learning. For this particular objective function, an exact line search is cheap and easy to compute (see Section 4.2.1 for details). Figure 4 (left & center) shows the behavior of LBFGS with this exact line search (LBFGS-LS)

1. We set $c_1 = 10^{-3}$ in (4) and $c_2 = 0.8$ in (5), and used a decay factor of 0.9.
2. This is achieved for any choice of initial subgradient $\boldsymbol{g}^{(1)}$ (Line 3 of Algorithm 2).

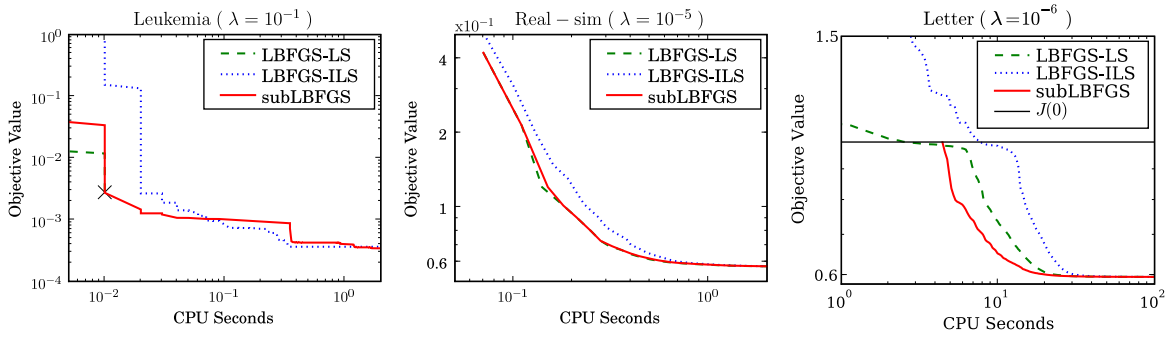


Figure 4: Performance of subLBFGS (solid) and standard LBFGS with exact (dashed) and inexact (dotted) line search methods on sample L_2-regularized risk minimization problems with the binary (left and center) and multiclass hinge losses (right). LBFGS with exact line search (dashed) fails after 3 iterations (marked as $\times$) on the Leukemia data set (left).

on two data sets, namely Leukemia and Real-sim.[3] It can be seen that LBFGS-LS converges on Real-sim but diverges on the Leukemia data set. This is because using an exact line search on a nonsmooth objective function increases the chance of landing on nonsmooth points, a situation that standard BFGS (resp., LBFGS) is not designed to deal with. To prevent (L)BFGS' sudden breakdown, a scheme that actively avoids nonsmooth points must be used. One such possibility is to use an inexact line search that obeys the Wolfe conditions. Here we used an efficient inexact line search that uses a caching scheme specifically designed for L_2-regularized hinge loss (cf. end of Section 4.2). This implementation of LBFGS (LBFGS-ILS) converges on both data sets shown here but may fail on others. It is also slower, due to the inexactness of its line search.

For the multiclass hinge loss (42) we encounter another problem: if we follow the usual practice of initializing $\boldsymbol{w} = \mathbf{0}$, which happens to be a non-differentiable point, then LBFGS stalls. One way to get around this is to force LBFGS to take a unit step along its search direction to escape this nonsmooth point. However, as can be seen on the Letter data set[3] in Figure 4 (right), such an ad-hoc fix increases the value of the objective above $J(\mathbf{0})$ (solid horizontal line), and it takes several CPU seconds for the optimizers to recover from this. In all cases shown in Figure 4, our subgradient LBFGS (subLBFGS) method (as will be introduced later) performs comparable to or better than the best implementation of LBFGS.

2.2 Advantage of Incorporating BFGS' Curvature Estimate

In machine learning one often encounters L_2-regularized risk minimization problems (30) with various hinge losses (31, 42, 55). Since the Hessian of those objective functions at differentiable points equals $\lambda \boldsymbol{I}$ (where λ is the regularization constant), one might be tempted to argue that for such problems, BFGS' approximation $\boldsymbol{B}_t$ to the inverse Hessian should be simply set to $\lambda^{-1}\boldsymbol{I}$. This would reduce the quasi-Newton direction $\boldsymbol{p}_t = -\boldsymbol{B}_t \boldsymbol{g}_t$, $\boldsymbol{g}_t \in \partial J(\boldsymbol{w}_t)$ to simply a scaled subgradient direction.

To check if doing so is beneficial, we compared the performance of our subLBFGS method with two implementations of subgradient descent: a vanilla gradient descent method (denoted GD) that

3. Descriptions of these data sets can be found in Section 8.

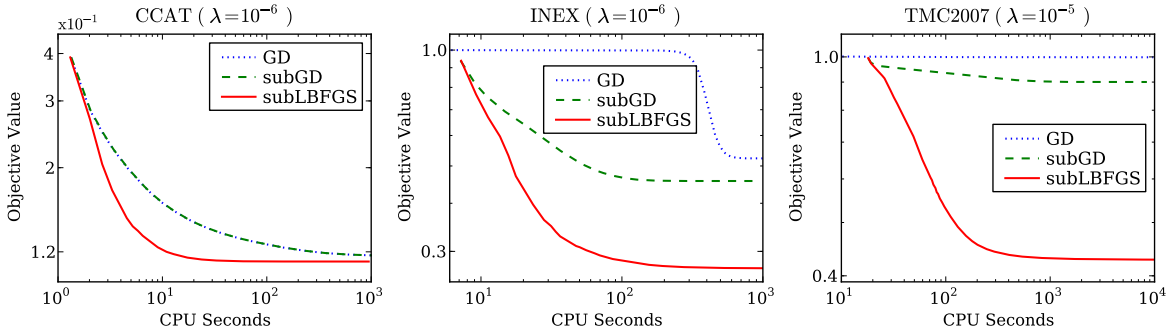


Figure 5: Performance of subLBFGS, GD, and subGD on sample L_2-regularized risk minimization problems with binary (left), multiclass (center), and multilabel (right) hinge losses.

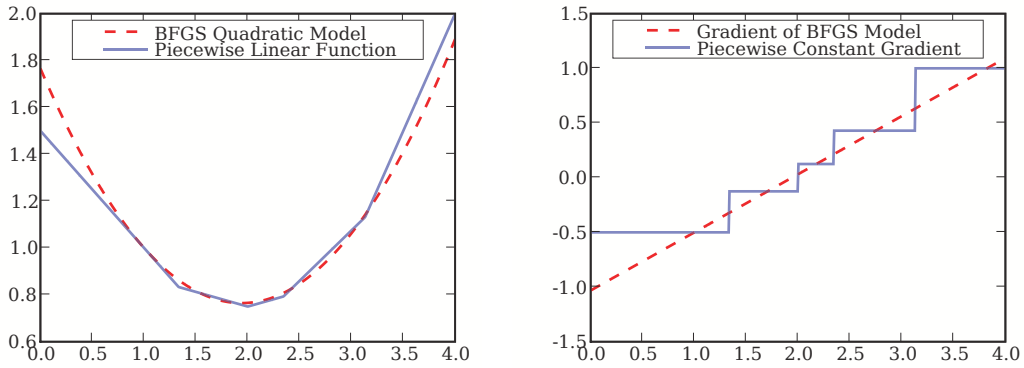


Figure 6: BFGS' quadratic approximation to a piecewise linear function (left), and its estimate of the gradient of this function (right).

uses a random subgradient for its parameter update, and an improved subgradient descent method (denoted subGD) whose parameter is updated in the direction produced by our direction-finding routine (Algorithm 2) with $\boldsymbol{B}_t = \boldsymbol{I}$. All algorithms used exact line search, except that GD took a unit step for the first update in order to avoid the nonsmooth point $\boldsymbol{w}_0 = \boldsymbol{0}$ (cf. the discussion in Section 2.1). As can be seen in Figure 5, on all sample L_2-regularized hinge loss minimization problems, subLBFGS (solid) converges significantly faster than GD (dotted) and subGD (dashed). This indicates that BFGS' $\boldsymbol{B}_t$ matrix is able to model the objective function, including its hinges, better than simply setting $\boldsymbol{B}_t$ to a scaled identity matrix.

We believe that BFGS' curvature update (6) plays an important role in the performance of subLBFGS seen in Figure 5. Recall that (6) satisfies the secant condition $\boldsymbol{B}_{t+1}\boldsymbol{y}_t = \boldsymbol{s}_t$, where $\boldsymbol{s}_t$ and $\boldsymbol{y}_t$ are displacement vectors in parameter and gradient space, respectively. The secant condition in fact implements a *finite differencing* scheme: for a one-dimensional objective function $J : \mathbb{R} \to \mathbb{R}$,

we have

$$\boldsymbol{B}_{t+1} = \frac{(w+p)-w}{\nabla J(w+p) - \nabla J(w)}. \tag{9}$$

Although the original motivation behind the secant condition was to approximate the inverse Hessian, the finite differencing scheme (9) allows BFGS to model the global curvature (i.e., overall shape) of the objective function from first-order information. For instance, Figure 6 (left) shows that the BFGS quadratic model[4] (1) fits a piecewise linear function quite well despite the fact that the actual Hessian in this case is zero almost everywhere, and infinite (in the limit) at nonsmooth points. Figure 6 (right) reveals that BFGS captures the global trend of the gradient rather than its infinitesimal variation, that is, the Hessian. This is beneficial for nonsmooth problems, where Hessian does not fully represent the overall curvature of the objective function.

3. Subgradient BFGS Method

We modify the standard BFGS algorithm to derive our new algorithm (subBFGS, Algorithm 1) for nonsmooth convex optimization, and its memory-limited variant (subLBFGS). Our modifications can be grouped into three areas, which we elaborate on in turn: generalizing the local quadratic model, finding a descent direction, and finding a step size that obeys a subgradient reformulation of the Wolfe conditions. We then show that our algorithm's estimate of the inverse Hessian has a bounded spectrum, which allows us to prove its convergence.

Algorithm 1 Subgradient BFGS (subBFGS)

Initialize: $t := 0, \boldsymbol{w}_0 = \boldsymbol{0}, \boldsymbol{B}_0 = \boldsymbol{I}$
Set: direction-finding tolerance $\varepsilon \geq 0$, iteration limit $k_{\max} > 0$,
    lower bound $h > 0$ on $\frac{\boldsymbol{s}_t^\top \boldsymbol{y}_t}{\boldsymbol{y}_t^\top \boldsymbol{y}_t}$ (cf. discussion in Section 3.4)
Compute subgradient $\boldsymbol{g}_0 \in \partial J(\boldsymbol{w}_0)$
while not converged **do**
  $\boldsymbol{p}_t = \texttt{descentDirection}(\boldsymbol{g}_t, \varepsilon, k_{\max})$ (Algorithm 2)
  if $\boldsymbol{p}_t =$ failure **then**
    Return $\boldsymbol{w}_t$
  end if
  Find η_t that obeys (23) and (24) (e.g., Algorithm 3 or 5)
  $\boldsymbol{s}_t = \eta_t \boldsymbol{p}_t$
  $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \boldsymbol{s}_t$
  Choose subgradient $\boldsymbol{g}_{t+1} \in \partial J(\boldsymbol{w}_{t+1}) : \boldsymbol{s}_t^\top (\boldsymbol{g}_{t+1} - \boldsymbol{g}_t) > 0$
  $\boldsymbol{y}_t := \boldsymbol{g}_{t+1} - \boldsymbol{g}_t$
  $\boldsymbol{s}_t := \boldsymbol{s}_t + \max\left(0,\ h - \frac{\boldsymbol{s}_t^\top \boldsymbol{y}_t}{\boldsymbol{y}_t^\top \boldsymbol{y}_t}\right) \boldsymbol{y}_t$ (ensure $\frac{\boldsymbol{s}_t^\top \boldsymbol{y}_t}{\boldsymbol{y}_t^\top \boldsymbol{y}_t} \geq h$)
  Update $\boldsymbol{B}_{t+1}$ via (6)
  $t := t + 1$
end while

4. For ease of exposition, the model was constructed at a differentiable point.

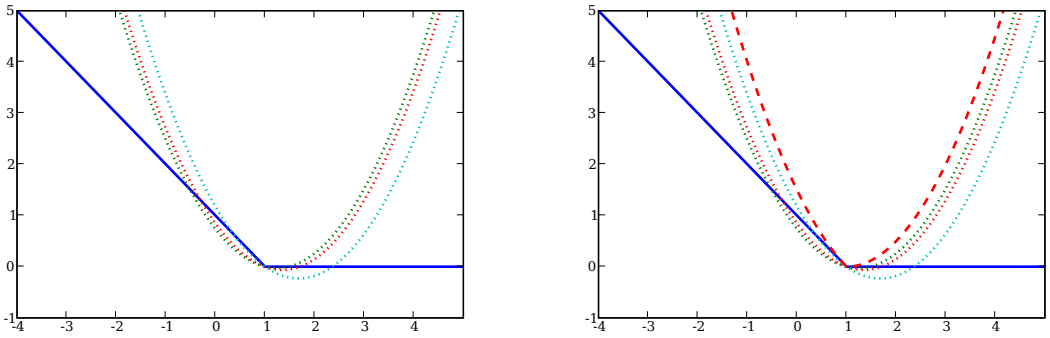


Figure 7: Left: selecting arbitrary subgradients yields many possible quadratic models (dotted lines) for the objective (solid blue line) at a subdifferentiable point. The models were built by keeping $\boldsymbol{B}_t$ fixed, but selecting random subgradients. Right: the tightest pseudo-quadratic fit (10) (bold red dashes); note that it is not a quadratic.

3.1 Generalizing the Local Quadratic Model

Recall that BFGS assumes that the objective function J is differentiable everywhere so that at the current iterate $\boldsymbol{w}_t$ it can construct a local quadratic model (1) of $J(\boldsymbol{w}_t)$. For a nonsmooth objective function, such a model becomes ambiguous at non-differentiable points (Figure 7, left). To resolve the ambiguity, we could simply replace the gradient $\nabla J(\boldsymbol{w}_t)$ in (1) with an arbitrary subgradient $\boldsymbol{g}_t \in \partial J(\boldsymbol{w}_t)$. However, as will be discussed later, the resulting quasi-Newton direction $\boldsymbol{p}_t := -\boldsymbol{B}_t \boldsymbol{g}_t$ is not necessarily a descent direction. To address this fundamental modeling problem, we first generalize the local quadratic model (1) as follows:

$$\begin{aligned} Q_t(\boldsymbol{p}) &:= J(\boldsymbol{w}_t) + M_t(\boldsymbol{p}), \quad \text{where} \\ M_t(\boldsymbol{p}) &:= \tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}_t^{-1}\boldsymbol{p} + \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}. \end{aligned} \tag{10}$$

Note that where J is differentiable, (10) reduces to the familiar BFGS quadratic model (1). At non-differentiable points, however, the model is no longer quadratic, as the supremum may be attained at different elements of $\partial J(\boldsymbol{w}_t)$ for different directions $\boldsymbol{p}$. Instead it can be viewed as the tightest pseudo-quadratic fit to J at $\boldsymbol{w}_t$ (Figure 7, right). Although the local model (10) of subBFGS is nonsmooth, it only incorporates non-differential points present at the current location; all others are smoothly approximated by the quasi-Newton mechanism.

Having constructed the model (10), we can minimize $Q_t(\boldsymbol{p})$, or equivalently $M_t(\boldsymbol{p})$:

$$\min_{\boldsymbol{p}\in\mathbb{R}^d} \left(\tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}_t^{-1}\boldsymbol{p} + \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p} \right) \tag{11}$$

to obtain a search direction. We now show that solving (11) is closely related to the problem of finding a *normalized steepest descent* direction. A normalized steepest descent direction is defined

as the solution to the following problem (Hiriart-Urruty and Lemaréchal, 1993, Chapter VIII):

$$\min_{\boldsymbol{p}\in\mathbb{R}^d} \; J'(\boldsymbol{w}_t,\, \boldsymbol{p}) \;\text{ s.t. }\; |||\boldsymbol{p}||| \leq 1, \tag{12}$$

where

$$J'(\boldsymbol{w}_t,\, \boldsymbol{p}) := \lim_{\eta\downarrow 0} \frac{J(\boldsymbol{w}_t + \eta\boldsymbol{p}) - J(\boldsymbol{w}_t)}{\eta}$$

is the directional derivative of J at $\boldsymbol{w}_t$ in direction $\boldsymbol{p}$, and $|||\cdot|||$ is a norm defined on $\mathbb{R}^d$. In other words, the normalized steepest descent direction is the direction of bounded norm along which the maximum rate of decrease in the objective function value is achieved. Using the property: $J'(\boldsymbol{w}_t,\, \boldsymbol{p}) = \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}$ (Bertsekas, 1999, Proposition B.24.b), we can rewrite (12) as:

$$\min_{\boldsymbol{p}\in\mathbb{R}^d} \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p} \quad \text{s.t. } |||\boldsymbol{p}||| \leq 1. \tag{13}$$

If the matrix $\boldsymbol{B}_t \succ 0$ as in (11) is used to define the norm $|||\cdot|||$ as

$$|||\boldsymbol{p}|||^2 := \boldsymbol{p}^\top \boldsymbol{B}_t^{-1} \boldsymbol{p}, \tag{14}$$

then the solution to (13) points to the same direction as that obtained by minimizing our pseudo-quadratic model (11). To see this, we write the Lagrangian of the constrained minimization problem (13):

$$\begin{aligned} L(\boldsymbol{p},\alpha) \;&:= \alpha\, \boldsymbol{p}^\top \boldsymbol{B}_t^{-1} \boldsymbol{p} \;-\alpha\; + \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p} \\ &= \tfrac{1}{2}\, \boldsymbol{p}^\top (2\alpha \boldsymbol{B}_t^{-1}) \boldsymbol{p} \;-\alpha\; + \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}, \end{aligned} \tag{15}$$

where $\alpha > 0$ is a Lagrangian multiplier. It is easy to see from (15) that minimizing the Lagrangian function L with respect to $\boldsymbol{p}$ is equivalent to solving (11) with $\boldsymbol{B}_t^{-1}$ scaled by a scalar 2α, implying that the steepest descent direction obtained by solving (13) with the weighted norm (14) only differs in length from the search direction obtained by solving (11). Therefore, our search direction is essentially an unnomalized steepest descent direction with respect to the weighted norm (14).

Ideally, we would like to solve (11) to obtain the best search direction. This is generally intractable due to the presence a supremum over the entire subdifferential set $\partial J(\boldsymbol{w}_t)$. In many machine learning problems, however, $\partial J(\boldsymbol{w}_t)$ has some special structure that simplifies the calculation of that supremum. In particular, the subdifferential of all the problems considered in this paper is a convex and compact polyhedron characterised as the convex hull of its extreme points. This dramatically reduces the cost of calculating $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}$ since the supremum can only be attained at an extreme point of the polyhedral set $\partial J(\boldsymbol{w}_t)$ (Bertsekas, 1999, Proposition B.21c). In what follows, we develop an iterative procedure that is guaranteed to find a quasi-Newton descent direction, assuming an oracle that supplies $\arg\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}$ for a given direction $\boldsymbol{p} \in \mathbb{R}^d$. Efficient oracles for this purpose can be derived for many machine learning settings; we provides such oracles for L_2-regularized risk minimization with the binary hinge loss (Section 4.1), multiclass and multilabel hinge losses (Section 6), and L_1-regularized logistic loss (Section 8.4).

Algorithm 2 $\boldsymbol{p}_t = \texttt{descentDirection}(\boldsymbol{g}^{(1)}, \varepsilon, k_{\max})$

input (sub)gradient $\boldsymbol{g}^{(1)} \in \partial J(\boldsymbol{w}_t)$, tolerance $\varepsilon \geq 0$, iteration limit $k_{\max} > 0$,
and an oracle to calculate $\arg\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$ for any given $\boldsymbol{w}$ and $\boldsymbol{p}$
output descent direction $\boldsymbol{p}_t$
Initialize: $i = 1$, $\bar{\boldsymbol{g}}^{(1)} = \boldsymbol{g}^{(1)}$, $\boldsymbol{p}^{(1)} = -\boldsymbol{B}_t \boldsymbol{g}^{(1)}$
$\boldsymbol{g}^{(2)} = \arg\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}^{(1)}$
$\varepsilon^{(1)} := \boldsymbol{p}^{(1)\top} \boldsymbol{g}^{(2)} - \boldsymbol{p}^{(1)\top} \bar{\boldsymbol{g}}^{(1)}$
while $(\boldsymbol{g}^{(i+1)\top} \boldsymbol{p}^{(i)} > 0$ or $\varepsilon^{(i)} > \varepsilon)$ and $\varepsilon^{(i)} > 0$ and $i < k_{\max}$ **do**
  $\mu^* := \min\left[1, \frac{(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}^{(i)}}{(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B}_t (\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})}\right]$; see (97)
  $\bar{\boldsymbol{g}}^{(i+1)} = (1-\mu^*)\bar{\boldsymbol{g}}^{(i)} + \mu^* \boldsymbol{g}^{(i+1)}$
  $\boldsymbol{p}^{(i+1)} = (1-\mu^*)\boldsymbol{p}^{(i)} - \mu^* \boldsymbol{B}_t \boldsymbol{g}^{(i+1)}$; see (76)
  $\boldsymbol{g}^{(i+2)} = \arg\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}^{(i+1)}$
  $\varepsilon^{(i+1)} := \min_{j\leq(i+1)} \left[\boldsymbol{p}^{(j)\top} \boldsymbol{g}^{(j+1)} - \frac{1}{2}(\boldsymbol{p}^{(j)\top} \bar{\boldsymbol{g}}^{(j)} + \boldsymbol{p}^{(i+1)\top} \bar{\boldsymbol{g}}^{(i+1)})\right]$
  $i := i + 1$
end while
$\boldsymbol{p}_t = \operatorname{argmin}_{j\leq i} M_t(\boldsymbol{p}^{(j)})$
if $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t \geq 0$ **then**
  return failure;
else
  return $\boldsymbol{p}_t$.
end if

3.2 Finding a Descent Direction

A direction $\boldsymbol{p}_t$ is a descent direction if and only if $\boldsymbol{g}^\top \boldsymbol{p}_t < 0 \quad \forall \boldsymbol{g} \in \partial J(\boldsymbol{w}_t)$ (Hiriart-Urruty and Lemaréchal, 1993, Theorem VIII.1.1.2), or equivalently

$$\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t \;<\; 0. \tag{16}$$

For a smooth convex function, the quasi-Newton direction (2) is always a descent direction because

$$\nabla J(\boldsymbol{w}_t)^\top \boldsymbol{p}_t \;=\; -\nabla J(\boldsymbol{w}_t)^\top \boldsymbol{B}_t \nabla J(\boldsymbol{w}_t) \;<\; 0$$

holds due to the positivity of $\boldsymbol{B}_t$.

For nonsmooth functions, however, the quasi-Newton direction $\boldsymbol{p}_t := -\boldsymbol{B}_t \boldsymbol{g}_t$ for a given $\boldsymbol{g}_t \in \partial J(\boldsymbol{w}_t)$ may not fulfill the descent condition (16), making it impossible to find a step size $\eta > 0$ that obeys the Wolfe conditions (4, 5), thus causing a failure of the line search. We now present an iterative approach to finding a quasi-Newton *descent* direction.

Our goal is to minimize the pseudo-quadratic model (10), or equivalently minimize $M_t(\boldsymbol{p})$. Inspired by bundle methods (Teo et al., 2010), we achieve this by minimizing convex lower bounds of $M_t(\boldsymbol{p})$ that are designed to progressively approach $M_t(\boldsymbol{p})$ over iterations. At iteration i we build the following convex lower bound on $M_t(\boldsymbol{p})$:

$$M_t^{(i)}(\boldsymbol{p}) \;:=\; \tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}_t^{-1} \boldsymbol{p} + \sup_{j\leq i} \boldsymbol{g}^{(j)\top} \boldsymbol{p}, \tag{17}$$

where $i, j \in \mathbb{N}$ and $\boldsymbol{g}^{(j)} \in \partial J(\boldsymbol{w}_t)\ \forall j \leq i$. Given a $\boldsymbol{p}^{(i)} \in \mathbb{R}^d$ the lower bound (17) is successively tightened by computing

$$\boldsymbol{g}^{(i+1)} := \operatorname*{arg\,sup}_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}^{(i)}, \tag{18}$$

such that $M_t^{(i)}(\boldsymbol{p}) \leq M_t^{(i+1)}(\boldsymbol{p}) \leq M_t(\boldsymbol{p})\ \forall \boldsymbol{p} \in \mathbb{R}^d$. Here we set $\boldsymbol{g}^{(1)} \in \partial J(\boldsymbol{w}_t)$ arbitrarily, and assume that (18) is provided by an oracle (e.g., as described in Section 4.1). To solve $\min_{\boldsymbol{p} \in \mathbb{R}^d} M_t^{(i)}(\boldsymbol{p})$, we rewrite it as a constrained optimization problem:

$$\min_{\boldsymbol{p}, \xi} \left(\tfrac{1}{2} \boldsymbol{p}^\top \boldsymbol{B}_t^{-1} \boldsymbol{p} + \xi \right) \ \text{ s.t. } \ \boldsymbol{g}^{(j)\top} \boldsymbol{p} \leq \xi \ \ \forall j \leq i. \tag{19}$$

This problem can be solved exactly via quadratic programming, but doing so may incur substantial computational expense. Instead we adopt an alternative approach (Algorithm 2) which does not solve (19) to optimality. The key idea is to write the proposed descent direction at iteration $i+1$ as a convex combination of $\boldsymbol{p}^{(i)}$ and $-\boldsymbol{B}_t \boldsymbol{g}^{(i+1)}$ (Line 9 of Algorithm 2); and as will be shown in Appendix B, the returned search direction takes the form

$$\boldsymbol{p}_t = -\boldsymbol{B}_t \bar{\boldsymbol{g}}_t,$$

where $\bar{\boldsymbol{g}}_t$ is a subgradient in $\partial J(\boldsymbol{w}_t)$ that allows $\boldsymbol{p}_t$ to satisfy the descent condition (16). The optimal convex combination coefficient μ^* can be computed exactly (Line 7 of Algorithm 2) using an argument based on maximizing the dual objective of $M_t(\boldsymbol{p})$; see Appendix A for details.

The weak duality theorem (Hiriart-Urruty and Lemaréchal, 1993, Theorem XII.2.1.5) states that the optimal primal value is no less than any dual value, that is, if $D_t(\boldsymbol{\alpha})$ is the dual of $M_t(\boldsymbol{p})$, then $\min_{\boldsymbol{p} \in \mathbb{R}^d} M_t(\boldsymbol{p}) \geq D_t(\boldsymbol{\alpha})$ holds for all feasible dual solutions $\boldsymbol{\alpha}$. Therefore, by iteratively increasing the value of the dual objective we close the gap to optimality in the primal. Based on this argument, we use the following upper bound on the duality gap as our measure of progress:

$$\varepsilon^{(i)} := \min_{j \leq i} \left[\boldsymbol{p}^{(j)\top} \boldsymbol{g}^{(j+1)} - \tfrac{1}{2} (\boldsymbol{p}^{(j)\top} \bar{\boldsymbol{g}}^{(j)} + \boldsymbol{p}^{(i)\top} \bar{\boldsymbol{g}}^{(i)}) \right] \geq \min_{\boldsymbol{p} \in \mathbb{R}^d} M_t(\boldsymbol{p}) - D_t(\boldsymbol{\alpha}^*), \tag{20}$$

where $\bar{\boldsymbol{g}}^{(i)}$ is an aggregated subgradient (Line 8 of Algorithm 2) which lies in the convex hull of $\boldsymbol{g}^{(j)} \in \partial J(\boldsymbol{w}_t)\ \forall j \leq i$, and $\boldsymbol{\alpha}^*$ is the optimal dual solution; Equations 77–79 in Appendix A provide intermediate steps that lead to the inequality in (20). Theorem 7 (Appendix B) shows that $\varepsilon^{(i)}$ is monotonically decreasing, leading us to a practical stopping criterion (Line 6 of Algorithm 2) for our direction-finding procedure.

A detailed derivation of Algorithm 2 is given in Appendix A, where we also prove that at a non-optimal iterate a direction-finding tolerance $\varepsilon \geq 0$ exists such that the search direction produced by Algorithm 2 is a descent direction; in Appendix B we prove that Algorithm 2 converges to a solution with precision ε in $O(1/\varepsilon)$ iterations. Our proofs are based on the assumption that the spectrum (eigenvalues) of BFGS' approximation $\boldsymbol{B}_t$ to the inverse Hessian is bounded from above and below. This is a reasonable assumption if simple safeguards such as those described in Section 3.4 are employed in the practical implementation.

3.3 Subgradient Line Search

Given the current iterate $\boldsymbol{w}_t$ and a search direction $\boldsymbol{p}_t$, the task of a line search is to find a step size $\eta > 0$ which reduces the objective function value along the line $\boldsymbol{w}_t + \eta \boldsymbol{p}_t$:

$$\text{minimize } \Phi(\eta) := J(\boldsymbol{w}_t + \eta \boldsymbol{p}_t). \tag{21}$$

Using the chain rule, we can write

$$\partial \Phi(\eta) := \{\boldsymbol{g}^\top \boldsymbol{p}_t : \boldsymbol{g} \in \partial J(\boldsymbol{w}_t + \eta \boldsymbol{p}_t)\}. \tag{22}$$

Exact line search finds the optimal step size η^* by minimizing $\Phi(\eta)$, such that $0 \in \partial\Phi(\eta^*)$; inexact line searches solve (21) approximately while enforcing conditions designed to ensure convergence. The Wolfe conditions (4) and (5), for instance, achieve this by guaranteeing a sufficient decrease in the value of the objective and excluding pathologically small step sizes, respectively (Wolfe, 1969; Nocedal and Wright, 1999). The original Wolfe conditions, however, require the objective function to be smooth; to extend them to nonsmooth convex problems, we propose the following subgradient reformulation:

$$J(\boldsymbol{w}_{t+1}) \leq J(\boldsymbol{w}_t) + c_1 \eta_t \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t \qquad \text{(sufficient decrease)} \tag{23}$$

$$\text{and} \quad \sup_{\boldsymbol{g}' \in \partial J(\boldsymbol{w}_{t+1})} \boldsymbol{g}'^\top \boldsymbol{p}_t \geq c_2 \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t, \qquad \text{(curvature)} \tag{24}$$

where $0 < c_1 < c_2 < 1$. Figure 8 illustrates how these conditions enforce acceptance of non-trivial step sizes that decrease the objective function value. In Appendix C we formally show that for any given descent direction we can always find a positive step size that satisfies (23) and (24). Moreover, Appendix D shows that the sufficient decrease condition (23) provides a necessary condition for the global convergence of subBFGS.

Employing an exact line search is a common strategy to speed up convergence, but it drastically increases the probability of landing on a non-differentiable point (as in Figure 4, left). In order to leverage the fast convergence provided by an exact line search, one must therefore use an optimizer that can handle subgradients, like our subBFGS.

A natural question to ask is whether the optimal step size η^* obtained by an exact line search satisfies the reformulated Wolfe conditions (resp., the standard Wolfe conditions when J is smooth). The answer is no: depending on the choice of c_1, η^* may violate the sufficient decrease condition (23). For the function shown in Figure 8, for instance, we can increase the value of c_1 such that the acceptable interval for the step size excludes η^*. In practice one can set c_1 to a small value, for example, 10^{-4}, to prevent this from happening.

The curvature condition (24), on the other hand, is always satisfied by η^*, as long as $\boldsymbol{p}_t$ is a descent direction (16):

$$\sup_{\boldsymbol{g}' \in J(\boldsymbol{w}_t + \eta^* \boldsymbol{p}_t)} \boldsymbol{g}'^\top \boldsymbol{p}_t = \sup_{g \in \partial\Phi(\eta^*)} g \geq 0 > \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t$$

because $0 \in \partial\Phi(\eta^*)$.

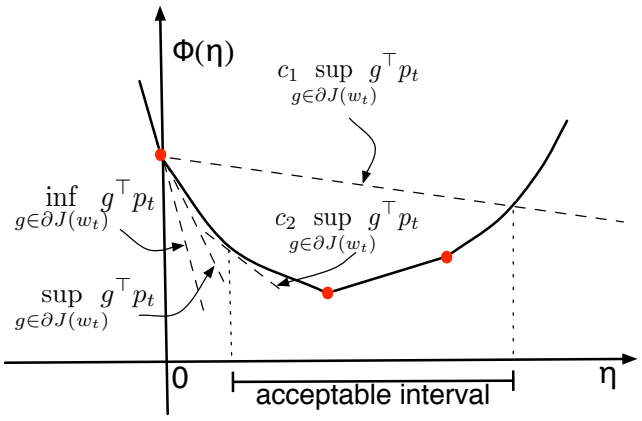


Figure 8: Geometric illustration of the subgradient Wolfe conditions (23) and (24). Solid disks are subdifferentiable points; the slopes of dashed lines are indicated.

3.4 Bounded Spectrum of SubBFGS' Inverse Hessian Estimate

Recall from Section 1 that to ensure positivity of BFGS' estimate $\boldsymbol{B}_t$ of the inverse Hessian, we must have $(\forall t)\ \boldsymbol{s}_t^\top \boldsymbol{y}_t > 0$. Extending this condition to nonsmooth functions, we require

$$(\boldsymbol{w}_{t+1} - \boldsymbol{w}_t)^\top (\boldsymbol{g}_{t+1} - \boldsymbol{g}_t) > 0, \ \text{ where } \ \boldsymbol{g}_{t+1} \in \partial J(\boldsymbol{w}_{t+1}) \ \text{ and } \ \boldsymbol{g}_t \in \partial J(\boldsymbol{w}_t). \tag{25}$$

If J is strongly convex,[5] and $\boldsymbol{w}_{t+1} \neq \boldsymbol{w}_t$, then (25) holds for any choice of $\boldsymbol{g}_{t+1}$ and $\boldsymbol{g}_t$.[6] For general convex functions, $\boldsymbol{g}_{t+1}$ need to be chosen (Line 12 of Algorithm 1) to satisfy (25). The existence of such a subgradient is guaranteed by the convexity of the objective function. To see this, we first use the fact that $\eta_t \boldsymbol{p}_t = \boldsymbol{w}_{t+1} - \boldsymbol{w}_t$ and $\eta_t > 0$ to rewrite (25) as

$$\boldsymbol{p}_t^\top \boldsymbol{g}_{t+1} > \boldsymbol{p}_t^\top \boldsymbol{g}_t, \ \text{ where } \ \boldsymbol{g}_{t+1} \in \partial J(\boldsymbol{w}_{t+1}) \ \text{ and } \ \boldsymbol{g}_t \in \partial J(\boldsymbol{w}_t). \tag{26}$$

It follows from (22) that both sides of inequality (26) are subgradients of $\Phi(\eta)$ at η_t and 0, respectively. The monotonic property of $\partial\Phi(\eta)$ given in Theorem 1 (below) ensures that $\boldsymbol{p}_t^\top \boldsymbol{g}_{t+1}$ is no less than $\boldsymbol{p}_t^\top \boldsymbol{g}_t$ for any choice of $\boldsymbol{g}_{t+1}$ and $\boldsymbol{g}_t$, that is,

$$\inf_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_{t+1})} \boldsymbol{p}_t^\top \boldsymbol{g} \ \geq \ \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{p}_t^\top \boldsymbol{g}. \tag{27}$$

This means that the only case where inequality (26) is violated is when both terms of (27) are equal, and

$$\boldsymbol{g}_{t+1} = \operatorname*{arg\,inf}_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_{t+1})} \boldsymbol{g}^\top \boldsymbol{p}_t \ \text{ and } \ \boldsymbol{g}_t = \operatorname*{arg\,sup}_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t,$$

that is, in this case $\boldsymbol{p}_t^\top \boldsymbol{g}_{t+1} = \boldsymbol{p}_t^\top \boldsymbol{g}_t$. To avoid this, we simply need to set $\boldsymbol{g}_{t+1}$ to a different subgradient in $\partial J(\boldsymbol{w}_{t+1})$.

5. If J is strongly convex, then $(\boldsymbol{g}_2 - \boldsymbol{g}_1)^\top (\boldsymbol{w}_2 - \boldsymbol{w}_1) \geq c \|\boldsymbol{w}_2 - \boldsymbol{w}_1\|^2$, with $c > 0$, $\boldsymbol{g}_i \in \partial J(\boldsymbol{w}_i)$, $i = 1, 2$.
6. We found empirically that no qualitative difference between using random subgradients versus choosing a particular subgradient when updating the $\boldsymbol{B}_t$ matrix.

Theorem 1 (Hiriart-Urruty and Lemaréchal, 1993, Theorem I.4.2.1)
Let Φ be a one-dimensional convex function on its domain, then $\partial\Phi(\eta)$ is increasing in the sense that $g_1 \leq g_2$ whenever $g_1 \in \partial\Phi(\eta_1)$, $g_2 \in \partial\Phi(\eta_2)$, and $\eta_1 < \eta_2$.

Our convergence analysis for the direction-finding procedure (Algorithm 2) as well as the global convergence proof of subBFGS in Appendix D require the spectrum of $\boldsymbol{B}_t$ to be bounded from above and below by a positive scalar:

$$\exists (h, H : 0 < h \leq H < \infty) : (\forall t)\ h \preceq \boldsymbol{B}_t \preceq H. \tag{28}$$

From a theoretical point of view it is difficult to guarantee (28) (Nocedal and Wright, 1999, page 212), but based on the fact that $\boldsymbol{B}_t$ is an approximation to the inverse Hessian $\boldsymbol{H}_t^{-1}$, it is reasonable to expect (28) to be true if

$$(\forall t)\ 1/H \preceq \boldsymbol{H}_t \preceq 1/h.$$

Since BFGS "senses" the Hessian via (6) only through the parameter and gradient displacements $\boldsymbol{s}_t$ and $\boldsymbol{y}_t$, we can translate the bounds on the spectrum of $\boldsymbol{H}_t$ into conditions that only involve $\boldsymbol{s}_t$ and $\boldsymbol{y}_t$:

$$(\forall t)\ \ \frac{\boldsymbol{s}_t^\top \boldsymbol{y}_t}{\boldsymbol{s}_t^\top \boldsymbol{s}_t} \geq \frac{1}{H} \ \text{ and } \ \frac{\boldsymbol{y}_t^\top \boldsymbol{y}_t}{\boldsymbol{s}_t^\top \boldsymbol{y}_t} \leq \frac{1}{h}, \ \text{ with } \ 0 < h \leq H < \infty. \tag{29}$$

This technique is used in Nocedal and Wright (1999, Theorem 8.5). If J is strongly convex[5] and $\boldsymbol{s}_t \neq \boldsymbol{0}$, then there exists an H such that the left inequality in (29) holds. On general convex functions, one can skip BFGS' curvature update if $(\boldsymbol{s}_t^\top \boldsymbol{y}_t / \boldsymbol{s}_t^\top \boldsymbol{s}_t)$ falls below a threshold. To establish the second inequality, we add a fraction of $\boldsymbol{y}_t$ to $\boldsymbol{s}_t$ at Line 14 of Algorithm 1 (though this modification is never actually invoked in our experiments of Section 8, where we set $h = 10^{-8}$).

3.5 Limited-Memory Subgradient BFGS

It is straightforward to implement an LBFGS variant of our subBFGS algorithm: we simply modify Algorithms 1 and 2 to compute all products between $\boldsymbol{B}_t$ and a vector by means of the standard LBFGS matrix-free scheme (Nocedal and Wright, 1999, Algorithm 9.1). We call the resulting algorithm subLBFGS.

3.6 Convergence of Subgradient (L)BFGS

In Section 3.4 we have shown that the spectrum of subBFGS' inverse Hessian estimate is bounded. From this and other technical assumptions, we prove in Appendix D that subBFGS is globally convergent in objective function value, that is, $J(\boldsymbol{w}) \to \inf_{\boldsymbol{w}} J(\boldsymbol{w})$. Moreover, in Appendix E we show that subBFGS converges for all counterexamples we could find in the literature used to illustrate the non-convergence of existing optimization methods on nonsmooth problems.

We have also examined the convergence of subLBFGS empirically. In most of our experiments of Section 8, we observe that after an initial transient, subLBFGS observes a period of linear convergence, until close to the optimum it exhibits superlinear convergence behavior. This is illustrated

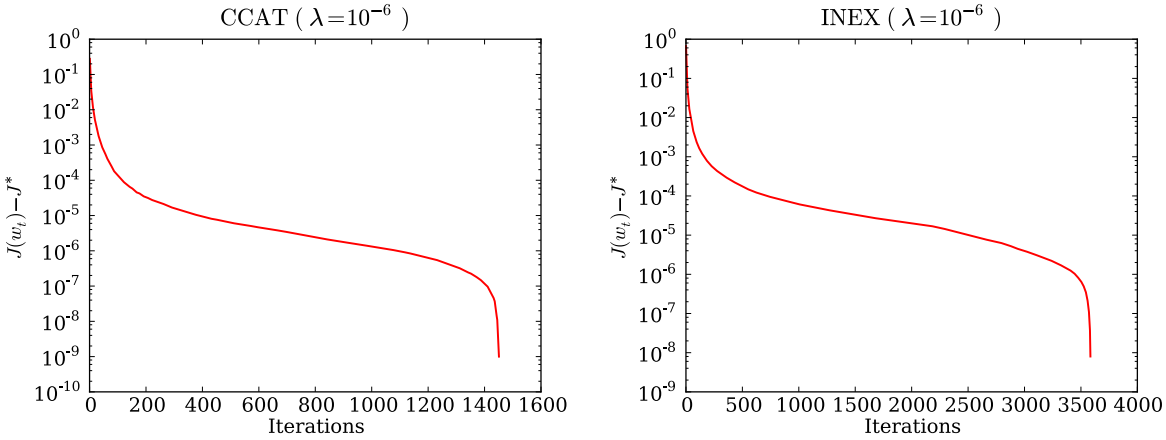


Figure 9: Convergence of subLBFGS in objective function value on sample L_2-regularized risk minimization problems with binary (left) and multiclass (right) hinge losses.

in Figure 9, where we plot (on a log scale) the excess objective function value $J(\boldsymbol{w}_t)$ over its "optimum" J^*[7] against the iteration number in two typical runs. The same kind of convergence behavior was observed by Lewis and Overton (2008a, Figure 5.7), who applied the classical BFGS algorithm with a specially designed line search to nonsmooth functions. They caution that the apparent superlinear convergence may be an artifact caused by the inaccuracy of the estimated optimal value of the objective.

4. SubBFGS for L_2-Regularized Binary Hinge Loss

Many machine learning algorithms can be viewed as minimizing the L_2-regularized risk

$$J(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} l(\boldsymbol{x}_i, z_i, \boldsymbol{w}), \tag{30}$$

where $\lambda > 0$ is a regularization constant, $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ are the input features, $z_i \in \mathcal{Z} \subseteq \mathbb{Z}$ the corresponding labels, and the loss l is a non-negative convex function of $\boldsymbol{w}$ which measures the discrepancy between z_i and the predictions arising from using $\boldsymbol{w}$. A loss function commonly used for binary classification is the binary hinge loss

$$l(\boldsymbol{x}, z, \boldsymbol{w}) := \max(0, 1 - z\boldsymbol{w}^\top \boldsymbol{x}), \tag{31}$$

where $z \in \{\pm 1\}$. L_2-regularized risk minimization with the binary hinge loss is a convex but nonsmooth optimization problem; in this section we show how subBFGS (Algorithm 1) can be applied to this problem.

7. Estimated empirically by running subLBFGS for 10^4 seconds, or until the relative improvement over 5 iterations was less than 10^{-8}.

Let $\mathcal{E}$, $\mathcal{M}$, and $\mathcal{W}$ index the set of points which are in error, on the margin, and well-classified, respectively:

$$\begin{aligned}\mathcal{E} &:= \{i \in \{1,2,\ldots,n\} : 1 - z_i \boldsymbol{w}^\top \boldsymbol{x}_i > 0\},\\ \mathcal{M} &:= \{i \in \{1,2,\ldots,n\} : 1 - z_i \boldsymbol{w}^\top \boldsymbol{x}_i = 0\},\\ \mathcal{W} &:= \{i \in \{1,2,\ldots,n\} : 1 - z_i \boldsymbol{w}^\top \boldsymbol{x}_i < 0\}.\end{aligned}$$

Differentiating (30) after plugging in (31) then yields

$$\partial J(\boldsymbol{w}) \;=\; \lambda \boldsymbol{w} - \frac{1}{n}\sum_{i=1}^{n} \beta_i z_i \boldsymbol{x}_i \;=\; \bar{\boldsymbol{w}} - \frac{1}{n}\sum_{i \in \mathcal{M}} \beta_i z_i \boldsymbol{x}_i, \tag{32}$$

$$\text{where} \quad \bar{\boldsymbol{w}} := \lambda \boldsymbol{w} - \frac{1}{n}\sum_{i \in \mathcal{E}} z_i \boldsymbol{x}_i \quad \text{and} \quad \beta_i := \begin{cases} 1 & \text{if} \quad i \in \mathcal{E}, \\ [0,1] & \text{if} \quad i \in \mathcal{M}, \\ 0 & \text{if} \quad i \in \mathcal{W}. \end{cases}$$

4.1 Efficient Oracle for the Direction-Finding Method

Recall that subBFGS requires an oracle that provides $\arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}$ for a given direction $\boldsymbol{p}$. For L_2-regularized risk minimization with the binary hinge loss we can implement such an oracle at a computational cost of $O(d\,|\,\mathcal{M}_t\,|)$, where d is the dimensionality of $\boldsymbol{p}$ and $|\,\mathcal{M}_t\,|$ the number of current margin points, which is normally much less than n. Towards this end, we use (32) to obtain

$$\begin{aligned}\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p} \;&=\; \sup_{\beta_i, i \in \mathcal{M}_t} \left(\bar{\boldsymbol{w}}_t - \frac{1}{n}\sum_{i \in \mathcal{M}_t} \beta_i z_i \boldsymbol{x}_i \right)^\top \boldsymbol{p} \\ &=\; \bar{\boldsymbol{w}}_t^\top \boldsymbol{p} - \frac{1}{n}\sum_{i \in \mathcal{M}_t} \inf_{\beta_i \in [0,1]} (\beta_i z_i \boldsymbol{x}_i^\top \boldsymbol{p}).\end{aligned} \tag{33}$$

Since for a given $\boldsymbol{p}$ the first term of the right-hand side of (33) is a constant, the supremum is attained when we set $\beta_i \; \forall i \in \mathcal{M}_t$ via the following strategy:

$$\beta_i := \begin{cases} 0 & \text{if } \; z_i \boldsymbol{x}_i^\top \boldsymbol{p}_t \geq 0, \\ 1 & \text{if } \; z_i \boldsymbol{x}_i^\top \boldsymbol{p}_t < 0. \end{cases}$$

4.2 Implementing the Line Search

The one-dimensional convex function $\Phi(\eta) := J(\boldsymbol{w} + \eta \boldsymbol{p})$ (Figure 10, left) obtained by restricting (30) to a line can be evaluated efficiently. To see this, rewrite (30) as

$$J(\boldsymbol{w}) \;:=\; \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\mathbf{1}^\top \max(\mathbf{0},\, \mathbf{1} - \boldsymbol{z} \cdot \boldsymbol{X}\boldsymbol{w}), \tag{34}$$

where $\mathbf{0}$ and $\mathbf{1}$ are column vectors of zeros and ones, respectively, $\cdot$ denotes the Hadamard (component-wise) product, and $\boldsymbol{z} \in \mathbb{R}^n$ collects correct labels corresponding to each row of data in $\boldsymbol{X} := [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$. Given a search direction $\boldsymbol{p}$ at a point $\boldsymbol{w}$, (34) allows us to write

$$\Phi(\eta) = \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \lambda \eta\, \boldsymbol{w}^\top \boldsymbol{p} + \frac{\lambda \eta^2}{2}\|\boldsymbol{p}\|^2 + \frac{1}{n}\mathbf{1}^\top \max\left[0,\, (\mathbf{1} - (\boldsymbol{f} + \eta \Delta \boldsymbol{f}))\right], \tag{35}$$

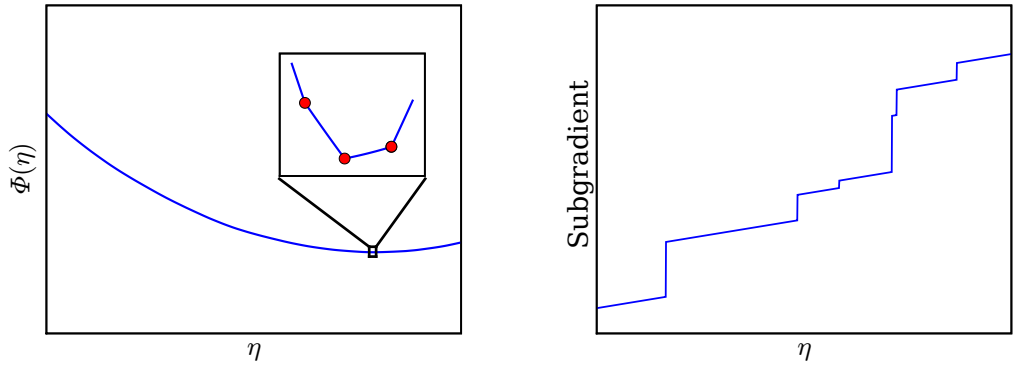


Figure 10: Left: Piecewise quadratic convex function Φ of step size η; solid disks in the zoomed inset are subdifferentiable points. Right: The subgradient of $\Phi(\eta)$ increases monotonically with η, and jumps discontinuously at subdifferentiable points.

where $\boldsymbol{f} := \boldsymbol{z} \cdot \boldsymbol{X}\boldsymbol{w}$ and $\Delta\boldsymbol{f} := \boldsymbol{z} \cdot \boldsymbol{X}\boldsymbol{p}$. Differentiating (35) with respect to η gives the subdifferential of Φ:

$$\partial\Phi(\eta) = \lambda \boldsymbol{w}^\top \boldsymbol{p} + \eta\lambda\|\boldsymbol{p}\|^2 - \frac{1}{n}\boldsymbol{\delta}(\eta)^\top \Delta\boldsymbol{f}, \tag{36}$$

where $\boldsymbol{\delta} : \mathbb{R} \to \mathbb{R}^n$ outputs a column vector $[\delta_1(\eta), \delta_2(\eta), \cdots, \delta_n(\eta)]^\top$ with

$$\delta_i(\eta) := \begin{cases} 1 & \text{if} \quad f_i + \eta\Delta f_i < 1, \\ [0,1] & \text{if} \quad f_i + \eta\Delta f_i = 1, \\ 0 & \text{if} \quad f_i + \eta\Delta f_i > 1. \end{cases} \tag{37}$$

We cache $\boldsymbol{f}$ and $\Delta\boldsymbol{f}$, expending $O(nd)$ computational effort and using $O(n)$ storage. We also cache the scalars $\frac{\lambda}{2}\|\boldsymbol{w}\|^2$, $\lambda\boldsymbol{w}^\top\boldsymbol{p}$, and $\frac{\lambda}{2}\|\boldsymbol{p}\|^2$, each of which requires $O(d)$ work. The evaluation of $\boldsymbol{1} - (\boldsymbol{f} + \eta\Delta\boldsymbol{f})$, $\boldsymbol{\delta}(\eta)$, and the inner products in the final terms of (35) and (36) all take $O(n)$ effort. Given the cached terms, all other terms in (35) can be computed in constant time, thus reducing the cost of evaluating $\Phi(\eta)$ (resp., its subgradient) to $O(n)$. Furthermore, from (37) we see that $\Phi(\eta)$ is differentiable everywhere except at

$$\eta_i := (1 - f_i)/\Delta f_i \;\text{ with }\; \Delta f_i \neq 0, \tag{38}$$

where it becomes subdifferentiable. At these points an element of the indicator vector (37) changes from 0 to 1 or vice versa (causing the subgradient to jump, as shown in Figure 10, right); otherwise $\boldsymbol{\delta}(\eta)$ remains constant. Using this property of $\boldsymbol{\delta}(\eta)$, we can update the last term of (36) in constant time when passing a hinge point (Line 25 of Algorithm 3). We are now in a position to introduce an exact line search which takes advantage of this scheme.

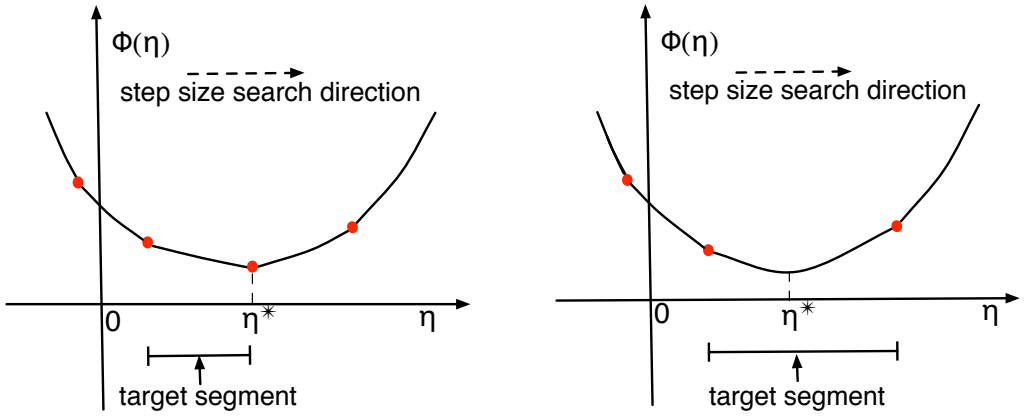


Figure 11: Nonsmooth convex function Φ of step size η. Solid disks are subdifferentiable points; the optimal step η^* either falls on such a point (left), or lies between two such points (right).

4.2.1 EXACT LINE SEARCH

Given a direction $\boldsymbol{p}$, exact line search finds the optimal step size $\eta^* := \operatorname{argmin}_{\eta \geq 0} \Phi(\eta)$ that satisfies $0 \in \partial \Phi(\eta^*)$, or equivalently

$$\inf \partial \Phi(\eta^*) \leq 0 \leq \sup \partial \Phi(\eta^*).$$

By Theorem 1, $\sup \partial \Phi(\eta)$ is monotonically increasing with η. Based on this property, our algorithm first builds a list of all possible subdifferentiable points and $\eta = 0$, sorted by non-descending value of η (Lines 4–5 of Algorithm 3). Then, it starts with $\eta = 0$, and walks through the sorted list until it locates the "target segment", an interval $[\eta_a, \eta_b]$ between two subdifferential points with $\sup \partial \Phi(\eta_a) \leq 0$ and $\sup \partial \Phi(\eta_b) \geq 0$. We now know that the optimal step size either coincides with η_b (Figure 11, left), or lies in (η_a, η_b) (Figure 11, right). If η^* lies in the smooth interval (η_a, η_b), then setting (36) to zero gives

$$\eta^* = \frac{\boldsymbol{\delta}(\eta')^\top \Delta \boldsymbol{f}/n - \lambda \boldsymbol{w}^\top \boldsymbol{p}}{\lambda \|\boldsymbol{p}\|^2}, \quad \forall \eta' \in (\eta_a, \eta_b). \tag{39}$$

Otherwise, $\eta^* = \eta_b$. See Algorithm 3 for the detailed implementation.

5. Segmenting the Pointwise Maximum of 1-D Linear Functions

The line search of Algorithm 3 requires a vector $\boldsymbol{\eta}$ listing the subdifferentiable points along the line $\boldsymbol{w} + \eta \boldsymbol{p}$, and sorts it in non-descending order (Line 5). For an objective function like (30) whose nonsmooth component is just a sum of hinge losses (31), this vector is very easy to compute (cf. (38)). In order to apply our line search approach to multiclass and multilabel losses, however, we must solve a more general problem: we need to efficiently find the subdifferentiable points of a

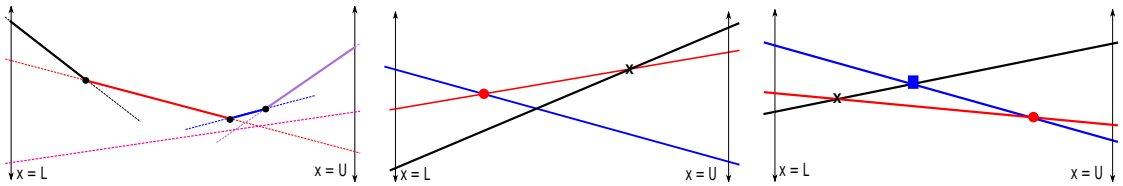
Algorithm 3 Exact Line Search for L_2-Regularized Binary Hinge Loss

input $\boldsymbol{w}, \boldsymbol{p}, \lambda, \boldsymbol{f}$, and $\Delta\boldsymbol{f}$ as in (35)
output optimal step size
$h = \lambda\|\boldsymbol{p}\|^2,\ j := 1$
$\boldsymbol{\eta} := [(\mathbf{1} - \boldsymbol{f})./\Delta\boldsymbol{f}, 0]$ (vector of subdifferentiable points & zero)
$\boldsymbol{\pi} = \texttt{argsort}(\boldsymbol{\eta})$ (indices sorted by non-descending value of $\boldsymbol{\eta}$)
while $\eta_{\pi_j} \leq 0$ **do**
  $j := j + 1$
end while
$\eta := \eta_{\pi_j}/2$
for $i := 1$ to $\boldsymbol{f}.\texttt{size}$ **do**
  $\delta_i := \begin{cases} 1 & \text{if } f_i + \eta\,\Delta f_i < 1 \\ 0 & \text{otherwise} \end{cases}$ (value of $\boldsymbol{\delta}(\eta)$ (37) for any $\eta \in (0, \eta_{\pi_j})$)
end for
$\rho := \boldsymbol{\delta}^\top \Delta\boldsymbol{f}/n - \lambda \boldsymbol{w}^\top \boldsymbol{p}$
$\eta := 0,\ \rho' := 0$
$g := -\rho$ (value of $\sup \partial\Phi(0)$)
while $g < 0$ **do**
  $\rho' := \rho$
  if $j > \boldsymbol{\pi}.\texttt{size}$ **then**
    $\eta := \infty$ (no more subdifferentiable points)
    break
  else
    $\eta := \eta_{\pi_j}$
  end if
  repeat
    $\rho := \begin{cases} \rho - \Delta f_{\pi_j}/n & \text{if } \delta_{\pi_j} = 1 \\ \rho + \Delta f_{\pi_j}/n & \text{otherwise} \end{cases}$ (move to next subdifferentiable point and update ρ accordingly)
    $j := j + 1$
  until $\eta_{\pi_j} \neq \eta_{\pi_{j-1}}$ and $j \leq \boldsymbol{\pi}.\texttt{size}$
  $g := \eta h - \rho$ (value of $\sup \partial\Phi(\eta_{\pi_{j-1}})$)
end while
return $\min(\eta,\ \rho'/h)$ (cf. equation 39)

one-dimensional piecewise linear function $\rho : \mathbb{R} \to \mathbb{R}$ defined to be the pointwise maximum of r lines:

$$\rho(\eta) = \max_{1 \leq p \leq r} (b_p + \eta\, a_p), \tag{40}$$

where a_p and b_p denote the slope and offset of the p^{th} line, respectively. Clearly, ρ is convex since it is the pointwise maximum of linear functions (Boyd and Vandenberghe, 2004, Section 3.2.3), see Figure 12(a). The difficulty here is that although ρ consists of at most r line segments bounded by at most $r-1$ subdifferentiable points, there are $r(r-1)/2$ candidates for these points, namely all intersections between any two of the r lines. A naive algorithm to find the subdifferentiable points of ρ would therefore take $O(r^2)$ time. In what follows, however, we show how this can be done in just $O(r \log r)$ time. In Section 6 we will then use this technique (Algorithm 4) to perform efficient exact line search in the multiclass and multilabel settings.



(a) Pointwise maximum of lines (b) Case 1 (c) Case 2

Figure 12: (a) Convex piecewise linear function defined as the maximum of 5 lines, but comprising only 4 active line segments (bold) separated by 3 subdifferentiable points (black dots). (b, c) Two cases encountered by our algorithm: (b) The new intersection (black cross) lies to the right of the previous one (red dot) and is therefore pushed onto the stack; (c) The new intersection lies to the left of the previous one. In this case the latter is popped from the stack, and a third intersection (blue square) is computed and pushed onto it.

Algorithm 4 Segmenting a Pointwise Maximum of 1-D Linear Functions

input vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ of slopes and offsets
  lower bound L, upper bound U, with $0 \le L < U < \infty$
output sorted stack of subdifferentiable points $\boldsymbol{\eta}$
  and corresponding active line indices $\boldsymbol{\xi}$
$\mathbf{y} := \boldsymbol{b} + L\boldsymbol{a}$
$\boldsymbol{\pi} :=$ `argsort`$(-\mathbf{y})$ (indices sorted by non-ascending value of $\mathbf{y}$)
S.`push` (L, π_1) (initialize stack)
for $q := 2$ **to** $\mathbf{y}$.`size` **do**
  while not S.`empty` **do**
    $(\eta, \xi) := S$.`top`
    $\eta' := \dfrac{b_{\pi_q} - b_\xi}{a_\xi - a_{\pi_q}}$ (intersection of two lines)
    if $L < \eta' \le \eta$ or ($\eta' = L$ and $a_{\pi_q} > a_\xi$) **then**
      S.`pop` (cf. Figure 12(c))
    else
      break
    end if
  end while
  if $L < \eta' \le U$ or ($\eta' = L$ and $a_{\pi_q} > a_\xi$) **then**
    S.`push` (η', π_q) (cf. Figure 12(b))
  end if
end for
return S

We begin by specifying an interval $[L,U]$ $(0 \le L < U < \infty)$ in which to find the subdifferentiable points of ρ, and set $\mathbf{y} := \boldsymbol{b} + L\boldsymbol{a}$, where $\boldsymbol{a} = [a_1, a_2, \cdots, a_r]$ and $\boldsymbol{b} = [b_1, b_2, \cdots, b_r]$. In other words, $\mathbf{y}$ contains the intersections of the r lines defining $\rho(\eta)$ with the vertical line $\eta = L$. Let $\boldsymbol{\pi}$ denote the permutation that sorts $\mathbf{y}$ in non-ascending order, that is, $p < q \implies y_{\pi_p} \ge y_{\pi_q}$, and let $\rho^{(q)}$ be the

function obtained by considering only the top $q \leq r$ lines at $\eta = L$, that is, the first q lines in $\boldsymbol{\pi}$:

$$\rho^{(q)}(\eta) = \max_{1 \leq p \leq q} (b_{\pi_p} + \eta\, a_{\pi_p}). \tag{41}$$

It is clear that $\rho^{(r)} = \rho$. Let $\boldsymbol{\eta}$ contain all $q' \leq q-1$ subdifferentiable points of $\rho^{(q)}$ in $[L,U]$ in ascending order, and $\boldsymbol{\xi}$ the indices of the corresponding *active* lines, that is, the maximum in (41) is attained for line ξ_{j-1} over the interval $[\eta_{j-1}, \eta_j]$: $\xi_{j-1} := \pi_{p^*}$, where $p^* = \operatorname{argmax}_{1 \leq p \leq q}(b_{\pi_p} + \eta a_{\pi_p})$ for $\eta \in [\eta_{j-1}, \eta_j]$, and lines ξ_{j-1} and ξ_j intersect at η_j.

Initially we set $\eta_0 := L$ and $\xi_0 := \pi_1$, the leftmost bold segment in Figure 12(a). Algorithm 4 goes through lines in $\boldsymbol{\pi}$ sequentially, and maintains a Last-In-First-Out stack S which at the end of the q^{th} iteration consists of the tuples

$$(\eta_0, \xi_0), (\eta_1, \xi_1), \ldots, (\eta_{q'}, \xi_{q'})$$

in order of ascending η_i, with $(\eta_{q'}, \xi_{q'})$ at the top. After r iterations S contains a sorted list of all subdifferentiable points (and the corresponding active lines) of $\rho = \rho^{(r)}$ in $[L,U]$, as required by our line searches.

In iteration $q+1$ Algorithm 4 examines the intersection η' between lines $\xi_{q'}$ and π_{q+1}: If $\eta' > U$, line π_{q+1} is irrelevant, and we proceed to the next iteration. If $\eta_{q'} < \eta' \leq U$ as in Figure 12(b), then line π_{q+1} is becoming active at η', and we simply push (η', π_{q+1}) onto the stack. If $\eta' \leq \eta_{q'}$ as in Figure 12(c), on the other hand, then line π_{q+1} dominates line $\xi_{q'}$ over the interval (η', ∞) and hence over $(\eta_{q'}, U] \subset (\eta', \infty)$, so we pop $(\eta_{q'}, \xi_{q'})$ from the stack (deactivating line $\xi_{q'}$), decrement q', and repeat the comparison.

Theorem 2 *The total running time of Algorithm 4 is $O(r \log r)$.*

Proof Computing intersections of lines as well as pushing and popping from the stack require $O(1)$ time. Each of the r lines can be pushed onto and popped from the stack at most once; amortized over r iterations the running time is therefore $O(r)$. The time complexity of Algorithm 4 is thus dominated by the initial sorting of $\mathbf{y}$ (i.e., the computation of $\boldsymbol{\pi}$), which takes $O(r \log r)$ time. ■

6. SubBFGS for Multiclass and Multilabel Hinge Losses

We now use the algorithm developed in Section 5 to generalize the subBFGS method of Section 4 to the multiclass and multilabel settings with finite label set $\mathcal{Z}$. We assume that given a feature vector $\boldsymbol{x}$ our classifier predicts the label

$$z^* = \operatorname*{argmax}_{z \in \mathcal{Z}} f(\boldsymbol{w}, \boldsymbol{x}, z),$$

where f is a linear function of $\boldsymbol{w}$, that is, $f(\boldsymbol{w}, \boldsymbol{x}, z) = \boldsymbol{w}^\top \phi(\boldsymbol{x}, z)$ for some feature map $\phi(\boldsymbol{x}, z)$.

6.1 Multiclass Hinge Loss

A variety of multiclass hinge losses have been proposed in the literature that generalize the binary hinge loss, and enforce a margin of separation between the true label z_i and every other label. We

focus on the following rather general variant (Taskar et al., 2004):[8]

$$l(\boldsymbol{x}_i, z_i, \boldsymbol{w}) \;:=\; \max_{z \in \mathcal{Z}} [\Delta(z, z_i) + f(\boldsymbol{w}, \boldsymbol{x}_i, z) - f(\boldsymbol{w}, \boldsymbol{x}_i, z_i)], \tag{42}$$

where $\Delta(z, z_i) \geq 0$ is the *label loss* specifying the margin required between labels z and z_i. For instance, a uniform margin of separation is achieved by setting $\Delta(z, z') := \tau > 0 \; \forall z \neq z'$ (Crammer and Singer, 2003a). By requiring that $\forall z \in \mathcal{Z} : \Delta(z, z) = 0$ we ensure that (42) always remains non-negative. Adapting (30) to the multiclass hinge loss (42) we obtain

$$J(\boldsymbol{w}) \;:=\; \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{z \in \mathcal{Z}} [\Delta(z, z_i) + f(\boldsymbol{w}, \boldsymbol{x}_i, z) - f(\boldsymbol{w}, \boldsymbol{x}_i, z_i)]. \tag{43}$$

For a given $\boldsymbol{w}$, consider the set

$$\mathcal{Z}_i^* := \operatorname*{argmax}_{z \in \mathcal{Z}} [\Delta(z, z_i) + f(\boldsymbol{w}, \boldsymbol{x}_i, z) - f(\boldsymbol{w}, \boldsymbol{x}_i, z_i)]$$

of maximum-loss labels (possibly more than one) for the i^{th} training instance. Since $f(\boldsymbol{w}, \boldsymbol{x}, z) = \boldsymbol{w}^\top \phi(\boldsymbol{x}, z)$, the subdifferential of (43) can then be written as

$$\partial J(\boldsymbol{w}) \;=\; \lambda \boldsymbol{w} + \frac{1}{n}\sum_{i=1}^{n} \sum_{z \in \mathcal{Z}} \beta_{i,z}\, \phi(\boldsymbol{x}_i, z) \tag{44}$$

$$\text{with} \quad \beta_{i,z} \;=\; \left\{ \begin{array}{cc} [0,1] & \text{if } z \in \mathcal{Z}_i^* \\ 0 & \text{otherwise} \end{array} \right\} - \delta_{z,z_i} \;\; \text{s.t.} \;\; \sum_{z \in \mathcal{Z}} \beta_{i,z} = 0, \tag{45}$$

where δ is the Kronecker delta: $\delta_{a,b} = 1$ if $a = b$, and 0 otherwise.[9]

6.2 Efficient Multiclass Direction-Finding Oracle

For L_2-regularized risk minimization with multiclass hinge loss, we can use a similar scheme as described in Section 4.1 to implement an efficient oracle that provides $\arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$ for the direction-finding procedure (Algorithm 2). Using (44), we can write

$$\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p} \;=\; \lambda \boldsymbol{w}^\top \boldsymbol{p} + \frac{1}{n}\sum_{i=1}^{n} \sum_{z \in \mathcal{Z}} \sup_{\beta_{i,z}} \left(\beta_{i,z}\, \phi(\boldsymbol{x}_i, z)^\top \boldsymbol{p} \right). \tag{46}$$

The supremum in (46) is attained when we pick, from the choices offered by (45),

$$\beta_{i,z} := \delta_{z,z_i^*} - \delta_{z,z_i}, \;\; \text{where} \;\; z_i^* := \operatorname*{argmax}_{z \in \mathcal{Z}_i^*} \phi(\boldsymbol{x}_i, z)^\top \boldsymbol{p}.$$

8. Our algorithm can also deal with the slack-rescaled variant of Tsochantaridis et al. (2005).

9. Let $l_i^* := \max_{z \neq z_i} [\Delta(z, z_i) + f(\boldsymbol{w}, \boldsymbol{x}_i, z) - f(\boldsymbol{w}, \boldsymbol{x}_i, z_i)]$. Definition (45) allows the following values of $\beta_{i,z}$:

$$\left\{ \begin{array}{c|ccc} & z = z_i & z \in \mathcal{Z}_i^* \setminus \{z_i\} & \text{otherwise} \\ \hline l_i^* < 0 & 0 & 0 & 0 \\ l_i^* = 0 & [-1, 0] & [0, 1] & 0 \\ l_i^* > 0 & -1 & [0, 1] & 0 \end{array} \right\} \;\; \text{s.t.} \;\; \sum_{z \in \mathcal{Z}} \beta_{i,z} = 0.$$

6.3 Implementing the Multiclass Line Search

Let $\Phi(\eta) := J(\boldsymbol{w} + \eta \boldsymbol{p})$ be the one-dimensional convex function obtained by restricting (43) to a line along direction $\boldsymbol{p}$. Letting $\rho_i(\eta) := l(\boldsymbol{x}_i, z_i, \boldsymbol{w} + \eta \boldsymbol{p})$, we can write

$$\Phi(\eta) \;=\; \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \lambda\eta \boldsymbol{w}^\top \boldsymbol{p} + \frac{\lambda\eta^2}{2}\|\boldsymbol{p}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \rho_i(\eta). \tag{47}$$

Each $\rho_i(\eta)$ is a piecewise linear convex function. To see this, observe that

$$f(\boldsymbol{w} + \eta \boldsymbol{p}, \boldsymbol{x}, z) := (\boldsymbol{w} + \eta \boldsymbol{p})^\top \phi(\boldsymbol{x}, z) = f(\boldsymbol{w}, \boldsymbol{x}, z) + \eta f(\boldsymbol{p}, \boldsymbol{x}, z)$$

and hence

$$\rho_i(\eta) := \max_{z \in \mathcal{Z}} [\underbrace{\Delta(z, z_i) + f(\boldsymbol{w}, \boldsymbol{x}_i, z) - f(\boldsymbol{w}, \boldsymbol{x}_i, z_i)}_{=: b_z^{(i)}} + \eta \underbrace{(f(\boldsymbol{p}, \boldsymbol{x}_i, z) - f(\boldsymbol{p}, \boldsymbol{x}_i, z_i))}_{=: a_z^{(i)}}], \tag{48}$$

which has the functional form of (40) with $r = |\mathcal{Z}|$. Algorithm 4 can therefore be used to compute a sorted vector $\boldsymbol{\eta}^{(i)}$ of all subdifferentiable points of $\rho_i(\eta)$ and corresponding active lines $\boldsymbol{\xi}^{(i)}$ in the interval $[0, \infty)$ in $O(|\mathcal{Z}| \log |\mathcal{Z}|)$ time. With some abuse of notation, we now have

$$\eta \in [\eta_j^{(i)}, \eta_{j+1}^{(i)}] \implies \rho_i(\eta) = b_{\xi_j^{(i)}} + \eta\, a_{\xi_j^{(i)}}. \tag{49}$$

The first three terms of (47) are constant, linear, and quadratic (with non-negative coefficient) in η, respectively. The remaining sum of piecewise linear convex functions $\rho_i(\eta)$ is also piecewise linear and convex, and so $\Phi(\eta)$ is a piecewise quadratic convex function.

6.3.1 EXACT MULTICLASS LINE SEARCH

Our exact line search employs a similar two-stage strategy as discussed in Section 4.2.1 for locating its minimum $\eta^* := \operatorname{argmin}_{\eta > 0} \Phi(\eta)$: we first find the first *subdifferentiable* point $\check{\eta}$ past the minimum, then locate η^* within the differentiable region to its left. We precompute and cache a vector $\boldsymbol{a}^{(i)}$ of all the slopes $a_z^{(i)}$ (offsets $b_z^{(i)}$ are not needed), the subdifferentiable points $\boldsymbol{\eta}^{(i)}$ (sorted in ascending order via Algorithm 4), and the corresponding indices $\boldsymbol{\xi}^{(i)}$ of active lines of ρ_i for all training instances i, as well as $\|\boldsymbol{w}\|^2$, $\boldsymbol{w}^\top \boldsymbol{p}$, and $\lambda\|\boldsymbol{p}\|^2$.

Since $\Phi(\eta)$ is convex, any point $\eta < \eta^*$ cannot have a non-negative subgradient.[10] The first subdifferentiable point $\check{\eta} \geq \eta^*$ therefore obeys

$$\begin{aligned} \check{\eta} \;&:=\; \min \eta \in \{\boldsymbol{\eta}^{(i)}, i = 1, 2, \ldots, n\} : \eta \geq \eta^* \\ &=\; \min \eta \in \{\boldsymbol{\eta}^{(i)}, i = 1, 2, \ldots, n\} : \sup \partial \Phi(\eta) \geq 0. \end{aligned} \tag{50}$$

We solve (50) via a simple linear search: Starting from $\eta = 0$, we walk from one subdifferentiable point to the next until $\sup \partial \Phi(\eta) \geq 0$. To perform this walk efficiently, define a vector $\boldsymbol{\psi} \in \mathbb{N}^n$ of indices into the sorted vector $\boldsymbol{\eta}^{(i)}$ *resp.* $\boldsymbol{\xi}^{(i)}$; initially $\boldsymbol{\psi} := \mathbf{0}$, indicating that $(\forall i)\ \eta_0^{(i)} = 0$. Given the current index vector $\boldsymbol{\psi}$, the next subdifferentiable point is then

$$\eta' := \eta_{(\psi_{i'}+1)}^{(i')}, \;\; \text{where} \;\; i' = \operatorname*{argmin}_{1 \leq i \leq n} \eta_{(\psi_i+1)}^{(i)}; \tag{51}$$

10. If $\Phi(\eta)$ has a flat optimal region, we define η^* to be the infimum of that region.

Algorithm 5 Exact Line Search for L_2-Regularized Multiclass Hinge Loss

1: **input** base point $\boldsymbol{w}$, descent direction $\boldsymbol{p}$, regularization parameter λ, vector $\boldsymbol{a}$ of all slopes as defined in (48), for each training instance i: sorted stack S_i of subdifferentiable points and active lines, as produced by Algorithm 4
2: **output** optimal step size
3: $\boldsymbol{a} := \boldsymbol{a}/n,\ h := \lambda\|\boldsymbol{p}\|^2$
4: $\rho := \lambda \boldsymbol{w}^\top \boldsymbol{p}$
5: **for** $i := 1$ to n **do**
6: **while** not S_i.`empty` **do**
7: R_i.`push` S_i.`pop` (reverse the stacks)
8: **end while**
9: $(\cdot, \xi_i) := R_i$.`pop`
10: $\rho := \rho + a_{\xi_i}$
11: **end for**
12: $\eta := 0,\ \rho' = 0$
13: $g := \rho$ (value of $\sup \partial\Phi(0)$)
14: **while** $g < 0$ **do**
15: $\rho' := \rho$
16: **if** $\forall i : R_i$.`empty` **then**
17: $\eta := \infty$ (no more subdifferentiable points)
18: **break**
19: **end if**
20: $I := \operatorname{argmin}_{1 \le i \le n}\ \eta' :\ (\eta', \cdot) = R_i$.`top` (find the next subdifferentiable point)
21: $\rho := \rho - \sum_{i \in I} a_{\xi_i}$
22: $\Xi := \{\xi_i : (\eta, \xi_i) := R_i.\texttt{pop},\ \ i \in I\}$
23: $\rho := \rho + \sum_{\xi_i \in \Xi} a_{\xi_i}$
24: $g := \rho + \eta h$ (value of $\sup \partial\Phi(\eta)$)
25: **end while**
26: **return** $\min(\eta, -\rho'/h)$

the step is completed by incrementing $\psi_{i'}$, that is, $\psi_{i'} := \psi_{i'} + 1$ so as to remove $\eta^{(i')}_{\psi_{i'}}$ from future consideration.[11] Note that computing the argmin in (51) takes $O(\log n)$ time (e.g., using a priority queue). Inserting (49) into (47) and differentiating, we find that

$$\sup \partial\Phi(\eta') \ = \ \lambda \boldsymbol{w}^\top \boldsymbol{p} + \lambda \eta' \|\boldsymbol{p}\|^2 + \frac{1}{n} \sum_{i=1}^{n} a_{\xi^{(i)}_{\psi_i}}. \tag{52}$$

The key observation here is that after the initial calculation of $\sup \partial\Phi(0) = \lambda \boldsymbol{w}^\top \boldsymbol{p} + \frac{1}{n}\sum_{i=1}^{n} a_{\xi^{(i)}_0}$ for $\eta = 0$, the sum in (52) can be updated incrementally in constant time through the addition of $a_{\xi^{(i')}_{\psi_{i'}}} - a_{\xi^{(i')}_{(\psi_{i'}-1)}}$ (Lines 20–23 of Algorithm 5).

Suppose we find $\check{\eta} = \eta^{(i')}_{\psi_{i'}}$ for some i'. We then know that the minimum η^* is either equal to $\check{\eta}$ (Figure 11, left), or found within the quadratic segment immediately to its left (Figure 11, right).

11. For ease of exposition, we assume i' in (51) is unique, and deal with multiple choices of i' in Algorithm 5.

We thus decrement $\psi_{i'}$ (i.e., take one step back) so as to index the segment in question, set the right-hand side of (52) to zero, and solve for η' to obtain

$$\eta^* \;=\; \min\left(\check{\eta},\; \frac{\lambda \boldsymbol{w}^\top \boldsymbol{p} + \frac{1}{n}\sum_{i=1}^n a_{\xi^{(i)}_{\psi_i}}}{-\lambda\|\boldsymbol{p}\|^2}\right). \tag{53}$$

This only takes constant time: we have cached $\boldsymbol{w}^\top \boldsymbol{p}$ and $\lambda\|\boldsymbol{p}\|^2$, and the sum in (53) can be obtained incrementally by adding $a_{\xi^{(i')}_{\psi_{i'}}} - a_{\xi^{(i')}_{(\psi_{i'}+1)}}$ to its last value in (52).

To locate $\check{\eta}$ we have to walk at most $O(n|\mathcal{Z}|)$ steps, each requiring $O(\log n)$ computation of argmin as in (51). Given $\check{\eta}$, the exact minimum η^* can be obtained in $O(1)$. Including the preprocessing cost of $O(n|\mathcal{Z}|\log|\mathcal{Z}|)$ (for invoking Algorithm 4), our exact multiclass line search therefore takes $O(n|\mathcal{Z}|(\log n|\mathcal{Z}|))$ time in the worst case. Algorithm 5 provides an implementation which instead of an index vector $\boldsymbol{\psi}$ directly uses the sorted stacks of subdifferentiable points and active lines produced by Algorithm 4. (The cost of reversing those stacks in Lines 6–8 of Algorithm 5 can easily be avoided through the use of double-ended queues.)

6.4 Multilabel Hinge Loss

Recently, there has been interest in extending the concept of the hinge loss to multilabel problems. Multilabel problems generalize the multiclass setting in that each training instance $\boldsymbol{x}_i$ is associated with a set of labels $\mathcal{Z}_i \subseteq \mathcal{Z}$ (Crammer and Singer, 2003b). For a uniform margin of separation τ, a hinge loss can be defined in this setting as follows:

$$l(\boldsymbol{x}_i, \mathcal{Z}_i, \boldsymbol{w}) := \max[0,\; \tau + \max_{z' \notin \mathcal{Z}_i} f(\boldsymbol{w}, \boldsymbol{x}_i, z') - \min_{z \in \mathcal{Z}_i} f(\boldsymbol{w}, \boldsymbol{x}_i, z)]. \tag{54}$$

We can generalize this to a not necessarily uniform label loss $\Delta(z', z) \geq 0$ as follows:

$$l(\boldsymbol{x}_i, \mathcal{Z}_i, \boldsymbol{w}) := \max_{\substack{(z,z'):\, z \in \mathcal{Z}_i \\ z' \notin \mathcal{Z}_i \setminus \{z\}}} [\Delta(z', z) + f(\boldsymbol{w}, \boldsymbol{x}_i, z') - f(\boldsymbol{w}, \boldsymbol{x}_i, z)], \tag{55}$$

where as before we require that $\Delta(z, z) = 0\ \forall z \in \mathcal{Z}$ so that by explicitly allowing $z' = z$ we can ensure that (55) remains non-negative. For a uniform margin $\Delta(z', z) = \tau\ \ \forall z' \neq z$ our multilabel hinge loss (55) reduces to the decoupled version (54), which in turn reduces to the multiclass hinge loss (42) if $\mathcal{Z}_i := \{z_i\}$ for all i.

For a given $\boldsymbol{w}$, let

$$\mathcal{Z}_i^* := \operatorname*{argmax}_{\substack{(z,z'):\, z \in \mathcal{Z}_i \\ z' \notin \mathcal{Z}_i \setminus \{z\}}} [\Delta(z', z) + f(\boldsymbol{w}, \boldsymbol{x}_i, z') - f(\boldsymbol{w}, \boldsymbol{x}_i, z)]$$

be the set of worst label pairs (possibly more than one) for the i^{th} training instance. The subdifferential of the multilabel analogue of L_2-regularized multiclass objective (43) can then be written just as in (44), with coefficients

$$\beta_{i,z} := \sum_{z':\,(z',z) \in \mathcal{Z}_i^*} \gamma^{(i)}_{z',z} - \sum_{z':\,(z,z') \in \mathcal{Z}_i^*} \gamma^{(i)}_{z,z'}, \;\text{ where }\; (\forall i) \sum_{(z,z') \in \mathcal{Z}_i^*} \gamma^{(i)}_{z,z'} = 1 \;\text{ and }\; \gamma^{(i)}_{z,z'} \geq 0. \tag{56}$$

Now let $(z_i, z_i') := \operatorname{argmax}_{(z,z')\in\mathcal{Z}_i^*} [\phi(\boldsymbol{x}_i, z') - \phi(\boldsymbol{x}_i, z)]^\top \boldsymbol{p}$ be a single steepest worst label pair in direction $\boldsymbol{p}$. We obtain $\arg\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$ for our direction-finding procedure by picking, from the choices offered by (56), $\gamma^{(i)}_{z,z'} := \delta_{z,z_i}\delta_{z',z_i'}$.

Finally, the line search we described in Section 6.3 for the multiclass hinge loss can be extended in a straightforward manner to our multilabel setting. The only caveat is that now $\rho_i(\eta) := l(\boldsymbol{x}_i, \mathcal{Z}_i, \boldsymbol{w} + \eta\boldsymbol{p})$ must be written as

$$\rho_i(\eta) := \max_{\substack{(z,z'):\, z\in\mathcal{Z}_i \\ z'\notin\mathcal{Z}_i\setminus\{z\}}} [\underbrace{\Delta(z', z) + f(\boldsymbol{w}, \boldsymbol{x}_i, z') - f(\boldsymbol{w}, \boldsymbol{x}_i, z)}_{=: b^{(i)}_{z,z'}} + \eta \underbrace{(f(\boldsymbol{p}, \boldsymbol{x}_i, z') - f(\boldsymbol{p}, \boldsymbol{x}_i, z))}_{=: a^{(i)}_{z,z'}}]. \tag{57}$$

In the worst case, (57) could be the piecewise maximum of $O(|\mathcal{Z}|^2)$ lines, thus increasing the overall complexity of the line search. In practice, however, the set of true labels $\mathcal{Z}_i$ is usually small, typically of size 2 or 3 (cf. Crammer and Singer, 2003b, Figure 3). As long as $\forall i : |\mathcal{Z}_i| = O(1)$, our complexity estimates of Section 6.3.1 still apply.

7. Related Work

We discuss related work in two areas: nonsmooth convex optimization, and the problem of segmenting the pointwise maximum of a set of one-dimensional linear functions.

7.1 Nonsmooth Convex Optimization

There are four main approaches to nonsmooth convex optimization: quasi-Newton methods, bundle methods, stochastic dual methods, and smooth approximation. We discuss each of these briefly, and compare and contrast our work with the state of the art.

7.1.1 NONSMOOTH QUASI-NEWTON METHODS

These methods try to find a descent quasi-Newton direction at every iteration, and invoke a line search to minimize the one-dimensional convex function along that direction. We note that the line search routines we describe in Sections 4–6 are applicable to all such methods. An example of this class of algorithms is the work of Lukšan and Vlček (1999), who propose an extension of BFGS to nonsmooth convex problems. Their algorithm samples subgradients around non-differentiable points in order to obtain a descent direction. In many machine learning problems evaluating the objective function and its (sub)gradient is very expensive, making such an approach inefficient. In contrast, given a current iterate $\boldsymbol{w}_t$, our direction-finding routine (Algorithm 2) samples subgradients from the set $\partial J(\boldsymbol{w}_t)$ via the oracle. Since this avoids the cost of explicitly evaluating new (sub)gradients, it is computationally more efficient.

Recently, Andrew and Gao (2007) introduced a variant of LBFGS, the Orthant-Wise Limited-memory Quasi-Newton (OWL-QN) algorithm, suitable for optimizing L_1-regularized log-linear models:

$$J(\boldsymbol{w}) := \lambda\|\boldsymbol{w}\|_1 + \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ln(1 + e^{-z_i \boldsymbol{w}^\top \boldsymbol{x}_i})}_{\text{logistic loss}}, \tag{58}$$

where the logistic loss is smooth, but the regularizer is only subdifferentiable at points where $\boldsymbol{w}$ has zero elements. From the optimization viewpoint this objective is very similar to L_2-regularized hinge loss; the direction finding and line search methods that we discussed in Sections 3.2 and 3.3, respectively, can be applied to this problem with slight modifications.

OWL-QN is based on the observation that the L_1 regularizer is linear within any given orthant. Therefore, it maintains an approximation $\boldsymbol{B}^{\text{ow}}$ to the inverse Hessian of the logistic loss, and uses an efficient scheme to select orthants for optimization. In fact, its success greatly depends on its direction-finding subroutine, which demands a specially chosen subgradient $\boldsymbol{g}^{\text{ow}}$ (Andrew and Gao, 2007, Equation 4) to produce the quasi-Newton direction, $\boldsymbol{p}^{\text{ow}} = \pi(\boldsymbol{p}, \boldsymbol{g}^{\text{ow}})$, where $\boldsymbol{p} := -\boldsymbol{B}^{\text{ow}}\boldsymbol{g}^{\text{ow}}$ and the projection π returns a search direction by setting the i^{th} element of $\boldsymbol{p}$ to zero whenever $p_i g_i^{\text{ow}} > 0$. As shown in Section 8.4, the direction-finding subroutine of OWL-QN can be replaced by our Algorithm 2, which makes OWL-QN more robust to the choice of subgradients.

7.1.2 BUNDLE METHODS

Bundle method solvers (Hiriart-Urruty and Lemaréchal, 1993) use past (sub)gradients to build a model of the objective function. The (sub)gradients are used to lower-bound the objective by a piecewise linear function which is minimized to obtain the next iterate. This fundamentally differs from the BFGS approach of using past gradients to approximate the (inverse) Hessian, hence building a quadratic model of the objective function.

Bundle methods have recently been adapted to the machine learning context, where they are known as SVMStruct (Tsochantaridis et al., 2005) *resp*. BMRM (Smola et al., 2007). One notable feature of these variants is that they do not employ a line search. This is justified by noting that a line search involves computing the value of the objective function multiple times, a potentially expensive operation in machine learning applications.

Franc and Sonnenburg (2008) speed up the convergence of SVMStruct for L_2-regularized binary hinge loss. The main idea of their optimized cutting plane algorithm, OCAS, is to perform a line search along the line connecting two successive iterates of a bundle method solver. Recently they have extended OCAS to multiclass classification (Franc and Sonnenburg, 2009). Although developed independently, their line search methods for both settings are very similar to the methods we describe in Sections 4.2.1 and 6.3.1, respectively. In particular, their line search for multiclass classification also involves segmenting the pointwise maximum of r 1-D linear functions (cf. Section 5), though the $O(r^2)$ time complexity of their method is worse than our $O(r \log r)$.

7.1.3 STOCHASTIC DUAL METHODS

Distinct from the above two classes of primal algorithms are methods which work in the dual domain. A prominent member of this class is the LaRank algorithm of Bordes et al. (2007), which achieves state-of-the-art results on multiclass classification problems. While dual algorithms are very competitive on clean data sets, they tend to be slow when given noisy data.

7.1.4 SMOOTH APPROXIMATION

Another possible way to bypass the complications caused by the nonsmoothness of an objective function is to work on a smooth approximation instead—see for instance the recent work of Nesterov (2005) and Nemirovski (2005). Some machine learning applications have also been pursued along these lines (Lee and Mangasarian, 2001; Zhang and Oles, 2001). Although this approach can

be effective, it is unclear how to build a smooth approximation in general. Furthermore, smooth approximations often sacrifice dual sparsity, which often leads to better generalization performance on the test data, and also may be needed to prove generalization bounds.

7.2 Segmenting the Pointwise Maximum of 1-D Linear Functions

The problem of computing the line segments that comprise the pointwise maximum of a given set of line segments has received attention in the area of computational geometry; see Agarwal and Sharir (2000) for a survey. Hershberger (1989) for instance proposed a divide-and-conquer algorithm for this problem with the same time complexity as our Algorithm 4. The Hershberger (1989) algorithm solves a slightly harder problem—his function is the pointwise maximum of line segments, as opposed to our lines—but our algorithm is conceptually simpler and easier to implement.

A similar problem has also been studied under the banner of kinetic data structures by Basch (1999), who proposed a heap-based algorithm for this problem and proved a worst-case $O(r \log^2 r)$ bound, where r is the number of line segments. Basch (1999) also claims that the lower bound is $O(r \log r)$; our Algorithm 4 achieves this bound.

8. Experiments

We evaluated the performance of our subLBFGS algorithm with, and compared it to other state-of-the-art nonsmooth optimization methods on L_2-regularized binary, multiclass, and multilabel hinge loss minimization problems. We also compared OWL-QN with a variant that uses our direction-finding routine on L_1-regularized logistic loss minimization tasks. On strictly convex problems such as these every convergent optimizer will reach the same solution; comparing generalisation performance is therefore pointless. Hence we concentrate on empirically evaluating the convergence behavior (objective function value *vs*. CPU seconds). All experiments were carried out on a Linux machine with dual 2.4 GHz Intel Core 2 processors and 4 GB of RAM.

In all experiments the regularization parameter was chosen from the set $10^{\{-6,-5,\cdots,-1\}}$ so as to achieve the highest prediction accuracy on the test data set, while convergence behavior (objective function value *vs*. CPU seconds) is reported on the training data set. To see the influence of the regularization parameter λ, we also compared the time required by each algorithm to reduce the objective function value to within 2% of the optimal value.[12] For all algorithms the initial iterate $\boldsymbol{w}_0$ was set to $\mathbf{0}$. Open source C++ code implementing our algorithms and experiments is available for download from `http://www.cs.adelaide.edu.au/~jinyu/Code/nonsmoothOpt.tar.gz`.

The subgradient for the construction of the subLBFGS search direction (cf. Line 12 of Algorithm 1) was chosen arbitrarily from the subdifferential. For the binary hinge loss minimization (Section 8.3), for instance, we picked an arbitrary subgradient by randomly setting the coefficient $\beta_i \ \forall i \in \mathcal{M}$ in (32) to either 0 or 1.

8.1 Convergence Tolerance of the Direction-Finding Procedure

The convergence tolerance ε of Algorithm 2 controls the precision of the solution to the direction-finding problem (11): lower tolerance may yield a better search direction. Figure 13 (left) shows

12. For L_1-regularized logistic loss minimization, the "optimal" value was the final objective function value achieved by the OWL-QN* algorithm (cf. Section 8.4). In all other experiments, it was found by running subLBFGS for 10^4 seconds, or until its relative improvement over 5 iterations was less than 10^{-8}.

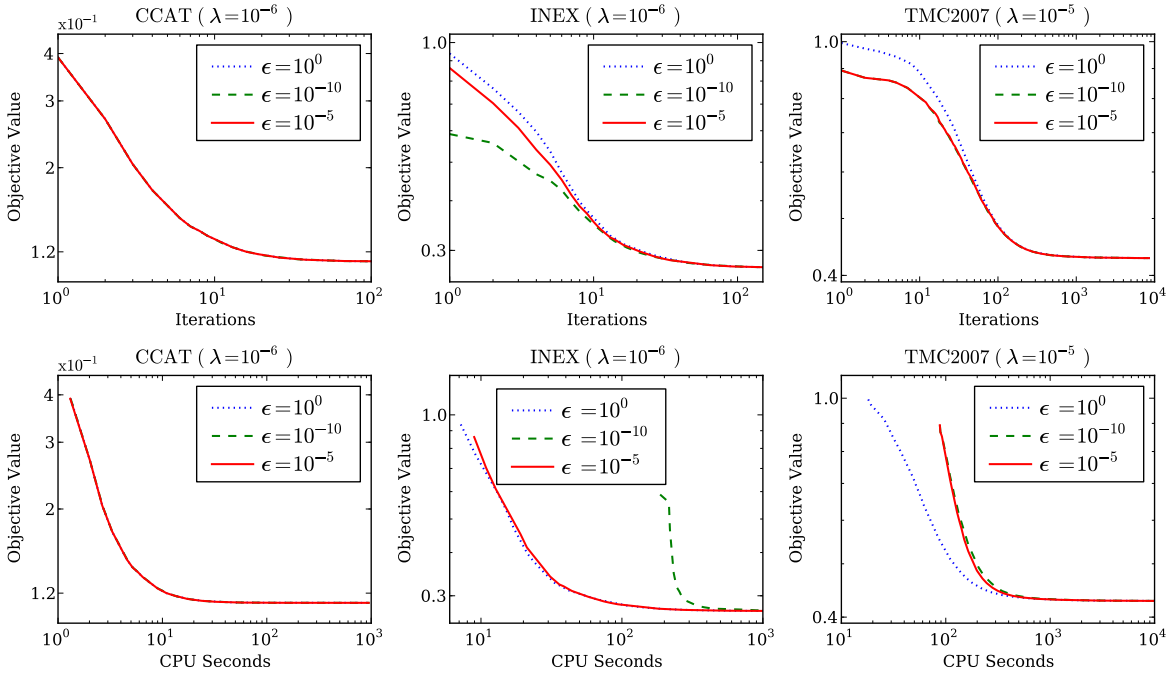


Figure 13: Performance of subLBFGS with varying direction-finding tolerance ε in terms of objective function value *vs*. number of iterations (top row) *resp*. CPU seconds (bottom row) on sample L_2-regularized risk minimization problems with binary (left), multiclass (center), and multilabel (right) hinge losses.

that on binary classification problems, subLBFGS is not sensitive to the choice of ε (i.e., the quality of the search direction). This is due to the fact that $\partial J(\boldsymbol{w})$ as defined in (32) is usually dominated by its constant component $\bar{\boldsymbol{w}}$; search directions that correspond to different choices of ε therefore can not differ too much from each other. In the case of multiclass and multilabel classification, where the structure of $\partial J(\boldsymbol{w})$ is more complicated, we can see from Figure 13 (top center and right) that a better search direction can lead to faster convergence in terms of iteration numbers. However, this is achieved at the cost of more CPU time spent in the direction-finding routine. As shown in Figure 13 (bottom center and right), extensively optimizing the search direction actually slows down convergence in terms of CPU seconds. We therefore used an intermediate value of $\varepsilon = 10^{-5}$ for all our experiments, except that for multiclass and multilabel classification problems we relaxed the tolerance to 1.0 at the initial iterate $\boldsymbol{w} = \boldsymbol{0}$, where the direction-finding oracle $\arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{0})} \boldsymbol{g}^\top \boldsymbol{p}$ is expensive to compute, due to the large number of extreme points in $\partial J(\boldsymbol{0})$.

8.2 Size of SubLBFGS Buffer

The size m of the subLBFGS buffer determines the number of parameter and gradient displacement vectors $\boldsymbol{s}_t$ and $\boldsymbol{y}_t$ used in the construction of the quasi-Newton direction. Figure 14 shows that the performance of subLBFGS is not sensitive to the particular value of m within the range $5 \leq m \leq 25$.

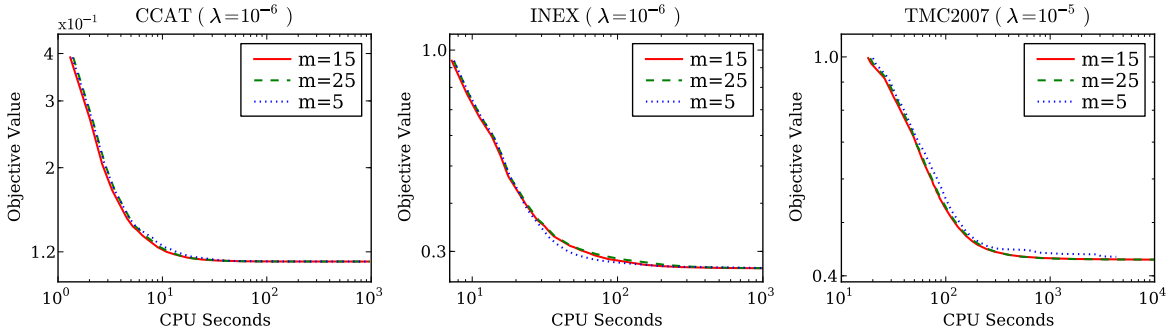


Figure 14: Performance of subLBFGS with varying buffer size on sample L_2-regularized risk minimization problems with binary (left), multiclass (center), and multilabel hinge losses (right).

Data Set	Train/Test Set Size	Dimensionality	Sparsity
Covertype	522911/58101	54	77.8%
CCAT	781265/23149	47236	99.8%
Astro-physics	29882/32487	99757	99.9%
MNIST-binary	60000/10000	780	80.8%
Adult9	32561/16281	123	88.7%
Real-sim	57763/14438	20958	99.8%
Leukemia	38/34	7129	00.0%

Table 1: The binary data sets used in our experiments of Sections 2, 8.3, and 8.4.

We therefore simply set $m = 15$ *a priori* for all subsequent experiments; this is a typical value for LBFGS (Nocedal and Wright, 1999).

8.3 L_2-Regularized Binary Hinge Loss

For our first set of experiments, we applied subLBFGS with exact line search (Algorithm 3) to the task of L_2-regularized binary hinge loss minimization. Our control methods are the bundle method solver BMRM (Teo et al., 2010) and the optimized cutting plane algorithm OCAS (Franc and Sonnenburg, 2008),[13] both of which were shown to perform competitively on this task. SVMStruct (Tsochantaridis et al., 2005) is another well-known bundle method solver that is widely used in the machine learning community. For L_2-regularized optimization problems BMRM is identical to SVMStruct, hence we omit comparisons with SVMStruct.

Table 1 lists the six data sets we used: The Covertype data set of Blackard, Jock & Dean,[14] CCAT from the Reuters RCV1 collection,[15] the Astro-physics data set of abstracts of scientific papers from the Physics ArXiv (Joachims, 2006), the MNIST data set of handwritten digits[16] with

13. The source code of OCAS (version 0.6.0) was obtained from `http://www.shogun-toolbox.org`.
14. Data set can be found at `http://kdd.ics.uci.edu/databases/covertype/covertype.html`.
15. Data set can be found at `http://www.daviddlewis.com/resources/testcollections/rcv1`.
16. Data set can be found at `http://yann.lecun.com/exdb/mnist`.

Data Set	L_1-reg. logistic loss			L_2-reg. binary loss	
	λ_{L_1}	k_{L_1}	$k_{L_1 r}$	λ_{L_2}	k_{L_2}
Covertype	10^{-5}	1	2	10^{-6}	0
CCAT	10^{-6}	284	406	10^{-6}	0
Astro-physics	10^{-5}	1702	1902	10^{-4}	0
MNIST-binary	10^{-4}	55	77	10^{-6}	0
Adult9	10^{-4}	2	6	10^{-5}	1
Real-sim	10^{-6}	1017	1274	10^{-5}	1

Table 2: Regularization parameter λ and overall number k of direction-finding iterations in our experiments of Sections 8.3 and 8.4, respectively.

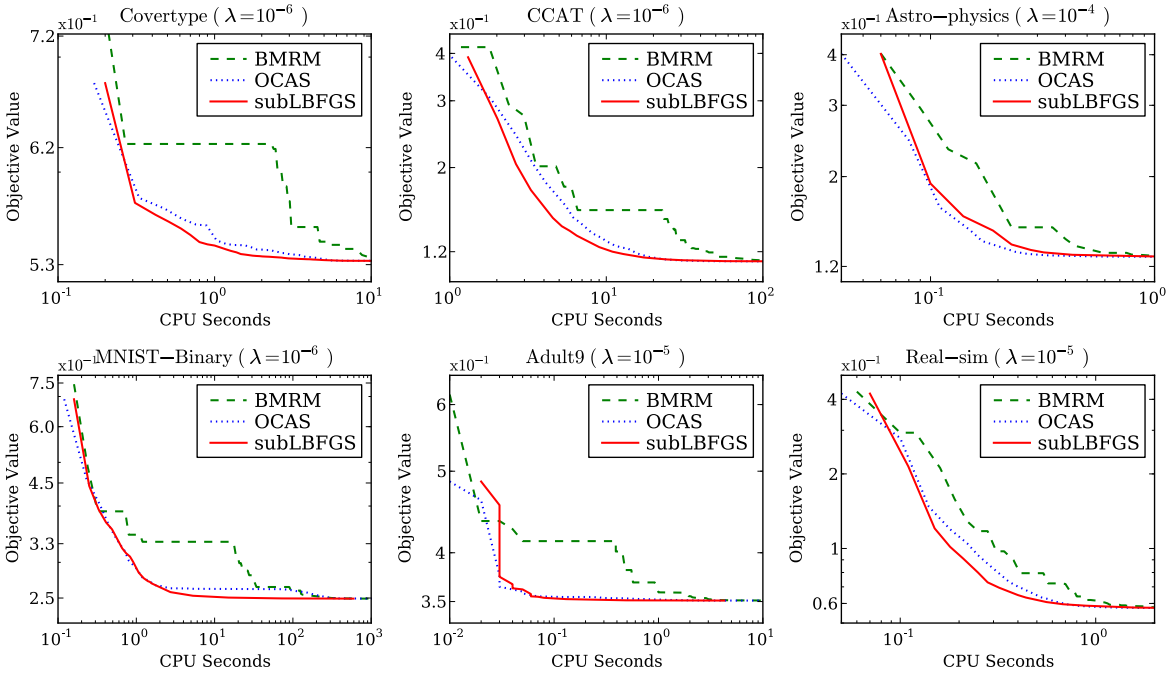


Figure 15: Objective function value *vs*. CPU seconds on L_2-regularized binary hinge loss minimization tasks.

two classes: even and odd digits, the Adult9 data set of census income data,[17] and the Real-sim data set of real *vs*. simulated data.[17] Table 2 lists our parameter settings, and reports the overall number k_{L_2} of iterations through the direction-finding loop (Lines 6–13 of Algorithm 2) for each data set. The very small values of k_{L_2} indicate that on these problems subLBFGS only rarely needs to correct its initial guess of a descent direction.

It can be seen from Figure 15 that subLBFGS (solid) reduces the value of the objective considerably faster than BMRM (dashed). On the binary MNIST data set, for instance, the objective

17. Data set can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.

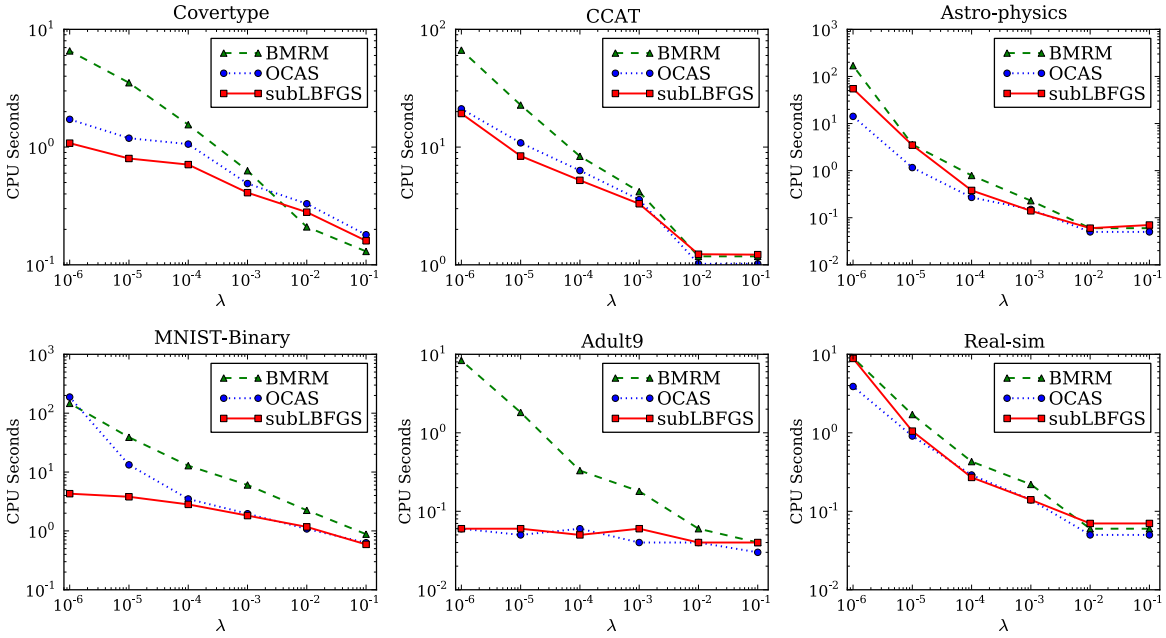


Figure 16: Regularization parameter $\lambda \in \{10^{-6}, \cdots, 10^{-1}\}$ *vs*. CPU seconds taken to reduce the objective function to within 2% of the optimal value on L_2-regularized binary hinge loss minimization tasks.

function value of subLBFGS after 10 CPU seconds is 25% lower than that of BMRM. In this set of experiments the performance of subLBFGS and OCAS (dotted) is very similar.

Figure 16 shows that all algorithms generally converge faster for larger values of the regularization constant λ. However, in most cases subLBFGS converges faster than BMRM across a wide range of λ values, exhibiting a speedup of up to more than two orders of magnitude. SubLBFGS and OCAS show similar performance here: for small values of λ, OCAS converges slightly faster than subLBFGS on the Astro-physics and Real-sim data sets but is outperformed by subLBFGS on the Covertype, CCAT, and binary MNIST data sets.

8.4 L_1-Regularized Logistic Loss

To demonstrate the utility of our direction-finding routine (Algorithm 2) in its own right, we plugged it into the OWL-QN algorithm (Andrew and Gao, 2007)[18] as an alternative direction-finding method such that $\boldsymbol{p}^{\text{ow}} = \texttt{descentDirection}(\boldsymbol{g}^{\text{ow}}, \varepsilon, k_{\max})$, and compared this variant (denoted OWL-QN*) with the original (cf. Section 7.1) on L_1-regularized minimization of the logistic loss (58), on the same data sets as in Section 8.3.

An oracle that supplies $\arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$ for this objective is easily constructed by noting that (58) is nonsmooth whenever at least one component of the parameter vector $\boldsymbol{w}$ is zero. Let $w_i = 0$ be such a component; the corresponding component of the subdifferential $\partial \lambda \|\boldsymbol{w}\|_1$ of the L_1

18. The source code of OWL-QN (original release) was obtained from Microsoft Research through `http://tinyurl.com/p774cx`.

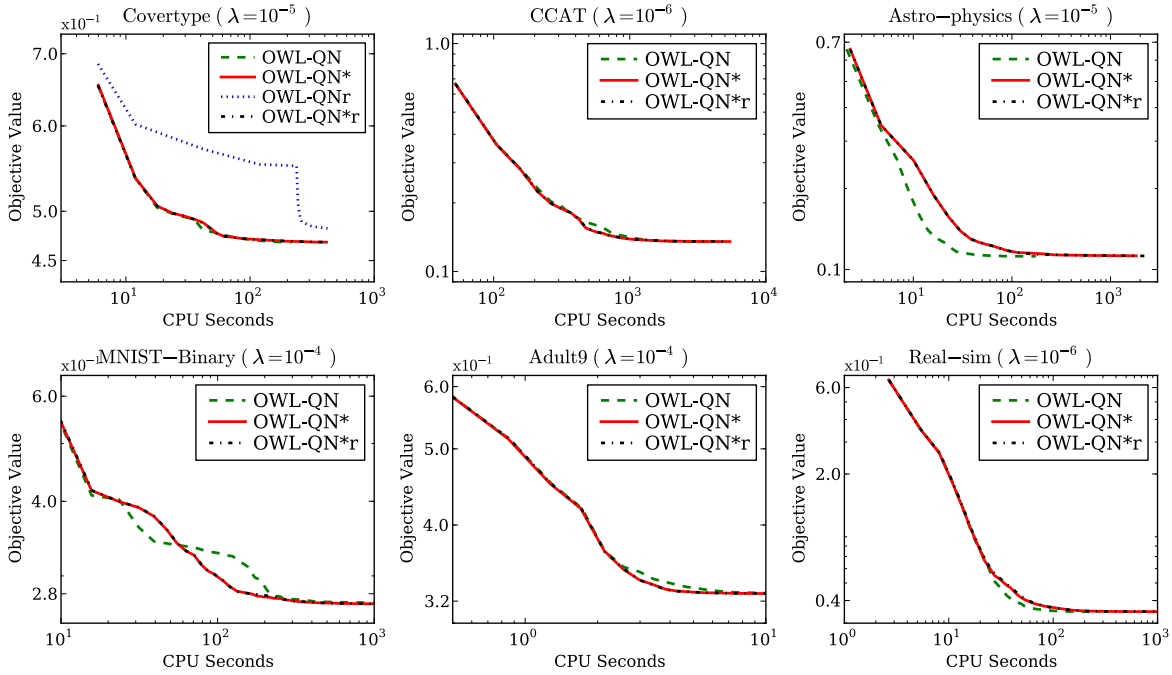


Figure 17: Objective function value *vs*. CPU seconds on L_1-regularized logistic loss minimization tasks.

regularizer is the interval $[-\lambda, \lambda]$. The supremum of $\boldsymbol{g}^\top \boldsymbol{p}$ is attained at the interval boundary whose sign matches that of the corresponding component of the direction vector $\boldsymbol{p}$, that is, at $\lambda \operatorname{sign}(p_i)$.

Using the stopping criterion suggested by Andrew and Gao (2007), we ran experiments until the averaged relative change in objective function value over the previous 5 iterations fell below 10^{-5}. As shown in Figure 17, the only clear difference in convergence between the two algorithms is found on the Astro-physics data set where OWL-QN* is outperformed by the original OWL-QN method. This is because finding a descent direction via Algorithm 2 is particularly difficult on the Astro-physics data set (as indicated by the large inner loop iteration number k_{L_1} in Table 2); the slowdown on this data set can also be found in Figure 18 for other values of λ. Although finding a descent direction can be challenging for the generic direction-finding routine of OWL-QN*, in the following experiment we show that this routine is very robust to the choice of initial subgradients.

To examine the algorithms' sensitivity to the choice of subgradients, we also ran them with subgradients randomly chosen from the set $\partial J(\boldsymbol{w})$ (as opposed to the specially chosen subgradient $\boldsymbol{g}^{\text{ow}}$ used in the previous set of experiments) fed to their corresponding direction-finding routines. OWL-QN relies heavily on its particular choice of subgradients, hence breaks down completely under these conditions: the only data set where we could even plot its (poor) performance was Covertype (dotted "OWL-QNr" line in Figure 17). Our direction-finding routine, by contrast, is self-correcting and thus not affected by this manipulation: the curves for OWL-QN*r lie on top of those for OWL-QN*. Table 2 shows that in this case more direction-finding iterations are needed though: $k_{L_1 r} > k_{L_1}$. This empirically confirms that as long as $\arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$ is given, Algorithm 2 can

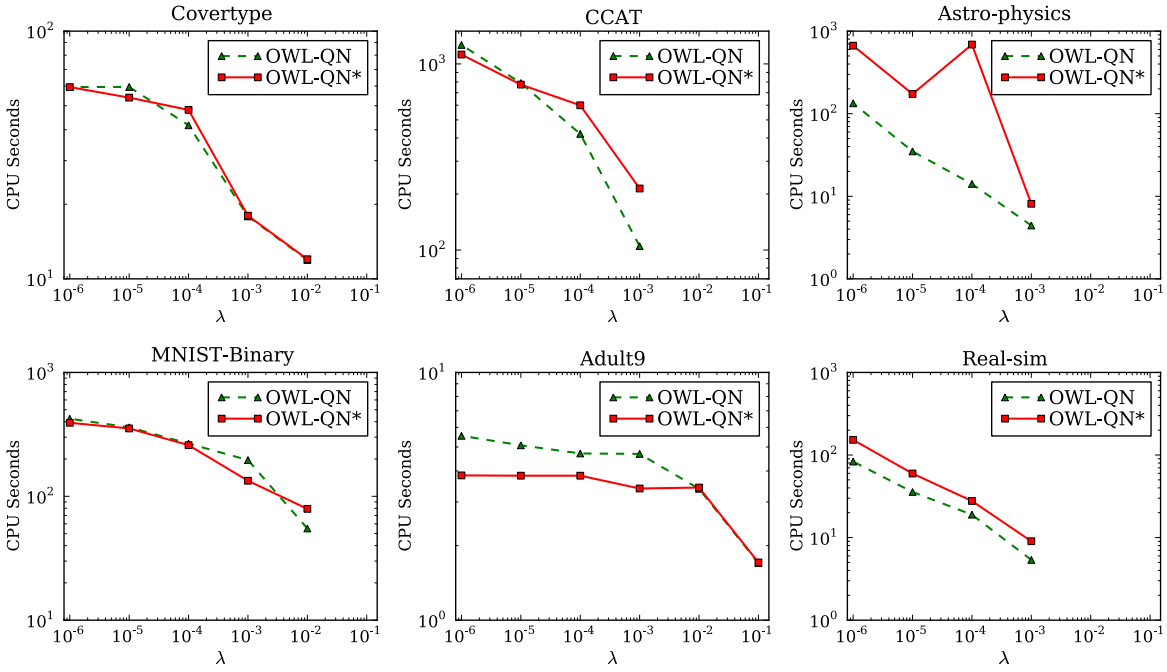


Figure 18: Regularization parameter $\lambda \in \{10^{-6}, \cdots, 10^{-1}\}$ *vs*. CPU seconds taken to reduce the objective function to within 2% of the optimal value on L_1-regularized logistic loss minimization tasks. (No point is plotted if the initial parameter $\boldsymbol{w}_0 = \mathbf{0}$ is already optimal.)

indeed be used as a generic quasi-Newton direction-finding routine that is able to recover from a poor initial choice of subgradients.

8.5 L_2-Regularized Multiclass and Multilabel Hinge Loss

We incorporated our exact line search of Section 6.3.1 into both subLBFGS and OCAS (Franc and Sonnenburg, 2008), thus enabling them to deal with multiclass and multilabel losses. We refer to our generalized version of OCAS as line search BMRM (ls-BMRM). Using the variant of the multiclass and multilabel hinge loss which enforces a uniform margin of separation ($\Delta(z, z') = 1 \; \forall z \neq z'$), we experimentally evaluated both algorithms on a number of publicly available data sets (Table 3). All multiclass data sets except INEX were downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html`, while the multilabel data sets were obtained from `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html`. INEX (Maes et al., 2007) is available from `http://webia.lip6.fr/~bordes/mywiki/doku.php?id=multiclass_data`. The original RCV1 data set consists of 23149 training instances, of which we used 21149 instances for training and the remaining 2000 for testing.

8.5.1 PERFORMANCE ON MULTICLASS PROBLEMS

This set of experiments is designed to demonstrate the convergence properties of multiclass subLBFGS, compared to the BMRM bundle method (Teo et al., 2010) and ls-BMRM. Figure 19 shows

Data Set	Train/Test Set Size	Dimensionality	$\lvert\mathcal{Z}\rvert$	Sparsity	λ	k
Letter	16000/4000	16	26	0.0%	10^{-6}	65
USPS	7291/2007	256	10	3.3%	10^{-3}	14
Protein	14895/6621	357	3	70.7%	10^{-2}	1
MNIST	60000/10000	780	10	80.8%	10^{-3}	1
INEX	6053/6054	167295	18	99.5%	10^{-6}	5
News20	15935/3993	62061	20	99.9%	10^{-2}	12
Scene	1211/1196	294	6	0.0%	10^{-1}	14
TMC2007	21519/7077	30438	22	99.7%	10^{-5}	19
RCV1	21149/2000	47236	103	99.8%	10^{-5}	4

Table 3: The multiclass (top 6 rows) and multilabel (bottom 3 rows) data sets used, values of the regularization parameter, and overall number k of direction-finding iterations in our experiments of Section 8.5.

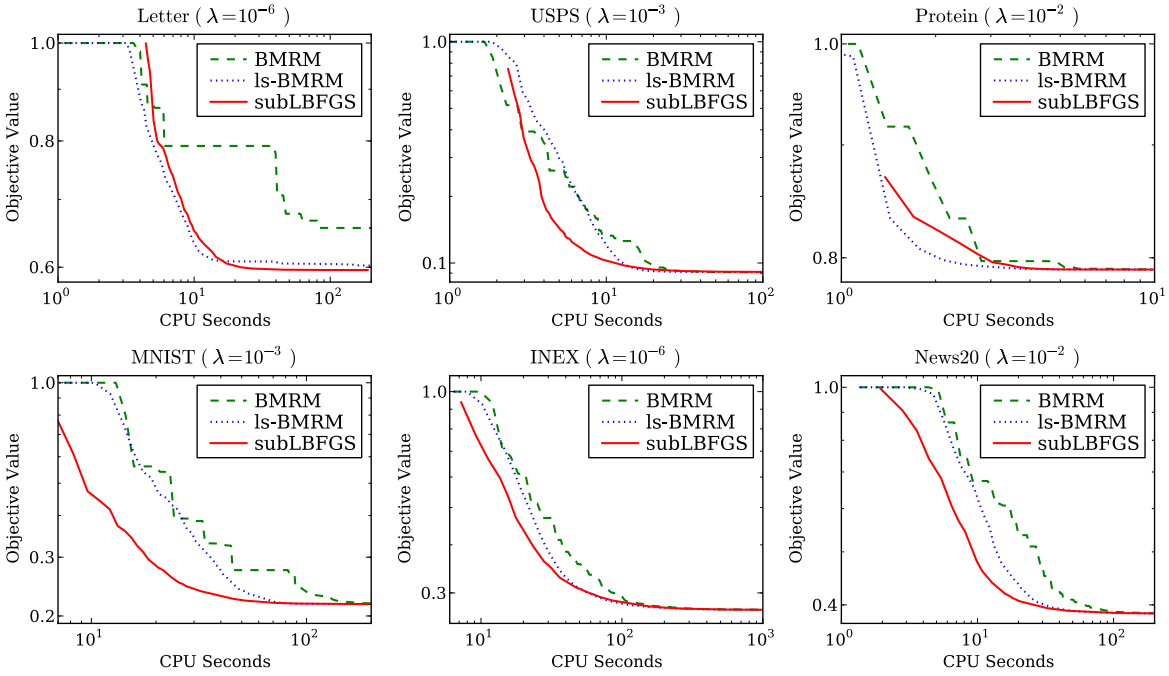


Figure 19: Objective function value *vs.* CPU seconds on L_2-regularized multiclass hinge loss minimization tasks.

that subLBFGS outperforms BMRM on all data sets. On 4 out of 6 data sets, subLBFGS outperforms ls-BMRM as well early on but slows down later, for an overall performance comparable to ls-BMRM. On the MNIST data set, for instance, subLBFGS takes only about half as much CPU time as ls-BMRM to reduce the objective function value to 0.3 (about 50% above the optimal value),

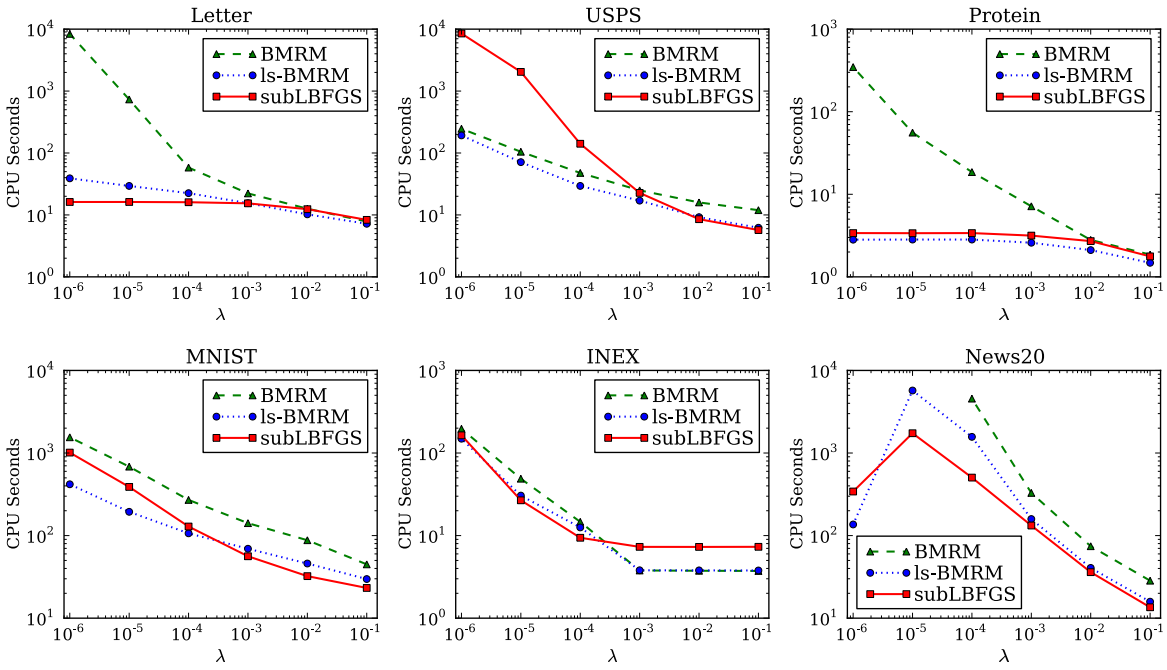


Figure 20: Regularization parameter $\lambda \in \{10^{-6}, \cdots, 10^{-1}\}$ *vs*. CPU seconds taken to reduce the objective function to within 2% of the optimal value. (No point is plotted if an algorithm failed to reach the threshold value within 10^4 seconds.)

yet both algorithms reach within 2% of the optimal value at about the same time (Figure 20, bottom left). We hypothesize that subLBFGS' local model (10) of the objective function facilitates rapid early improvement but is less appropriate for final convergence to the optimum (cf. the discussion in Section 9). Bundle methods, on the other hand, are slower initially because they need to accumulate a sufficient number of gradients to build a faithful piecewise linear model of the objective function. These results suggest that a hybrid approach that first runs subLBFGS then switches to ls-BMRM may be promising.

Similar to what we saw in the binary setting (Figure 16), Figure 20 shows that all algorithms tend to converge faster for large values of λ. Generally, subLBFGS converges faster than BMRM across a wide range of λ values; for small values of λ it can greatly outperform BMRM (as seen on Letter, Protein, and News20). The performance of subLBFGS is worse than that of BMRM in two instances: on USPS for small values of λ, and on INEX for large values of λ. The poor performance on USPS may be caused by a limitation of subLBFGS' local model (10) that causes it to slow down on final convergence. On the INEX data set, the initial point $\boldsymbol{w}_0 = \mathbf{0}$ is nearly optimal for large values of λ; in this situation there is no advantage in using subLBFGS.

Leveraging its exact line search (Algorithm 5), ls-BMRM is competitive on all data sets and across all λ values, exhibiting performance comparable to subLBFGS in many cases. From Figure 20 we find that BMRM never outperforms both subLBFGS and ls-BMRM.

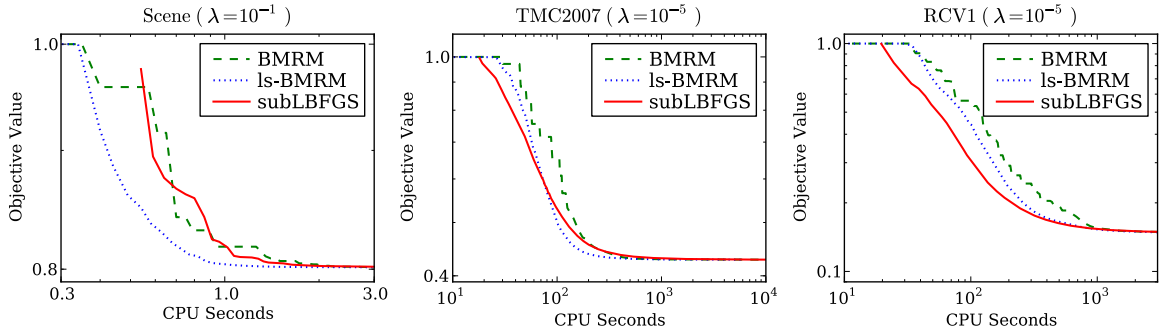


Figure 21: Objective function value *vs*. CPU seconds in L_2-regularized multilabel hinge loss minimization tasks.

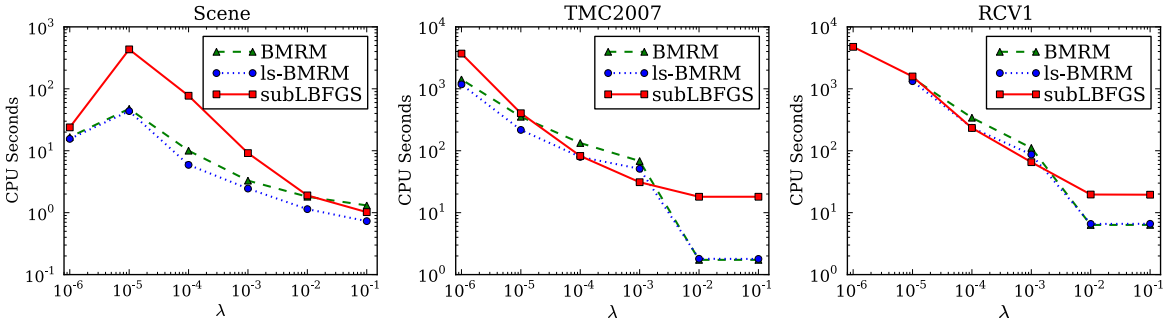


Figure 22: Regularization parameter $\lambda \in \{10^{-6}, \cdots, 10^{-1}\}$ *vs*. CPU seconds taken to reduce the objective function to within 2% of the optimal value. (No point is plotted if an algorithm failed to reach the threshold value within 10^4 seconds.)

8.5.2 PERFORMANCE ON MULTILABEL PROBLEMS

For our final set of experiments we turn to the multilabel setting. Figure 21 shows that on the Scene data set the performance of subLBFGS is similar to that of BMRM, while on the larger TMC2007 and RCV1 sets, subLBFGS outperforms both of its competitors initially but slows down later on, resulting in performance no better than BMRM. Comparing performance across different values of λ (Figure 22), we find that in many cases subLBFGS requires more time than its competitors to reach within 2% of the optimal value, and in contrast to the multiclass setting, here ls-BMRM only performs marginally better than BMRM. The primary reason for this is that the exact line search used by ls-BMRM and subLBFGS requires substantially more computational effort in the multilabel than in the multiclass setting. There is an inherent trade-off here: subLBFGS and ls-BMRM expend computation in an exact line search, while BMRM focuses on improving its local model of the objective function instead. In situations where the line search is very expensive, the latter strategy seems to pay off.

9. Discussion and Outlook

We proposed subBFGS (resp., subLBFGS), an extension of the BFGS quasi-Newton method (resp., its limited-memory variant), for handling nonsmooth convex optimization problems, and proved its global convergence in objective function value. We applied our algorithm to a variety of machine learning problems employing the L_2-regularized binary hinge loss and its multiclass and multilabel generalizations, as well as L_1-regularized risk minimization with logistic loss. Our experiments show that our algorithm is versatile, applicable to many problems, and often outperforms specialized solvers.

Our solver is easy to parallelize: The master node computes the search direction and transmits it to the slaves. The slaves compute the (sub)gradient and loss value on subsets of data, which is aggregated at the master node. This information is used to compute the next search direction, and the process repeats. Similarly, the line search, which is the expensive part of the computation on multiclass and multilabel problems, is easy to parallelize: The slaves run Algorithm 4 on subsets of the data; the results are fed back to the master which can then run Algorithm 5 to compute the step size.

In many of our experiments we observe that subLBFGS decreases the objective function rapidly at the beginning but slows down closer to the optimum. We hypothesize that this is due to an averaging effect: Initially (i.e., when sampled sparsely at a coarse scale) a superposition of many hinges looks sufficiently similar to a smooth function for optimization of a quadratic local model to work well (cf. Figure 6). Later on, when the objective is sampled at finer resolution near the optimum, the few nearest hinges begin to dominate the picture, making a smooth local model less appropriate.

Even though the local model (10) of sub(L)BFGS is nonsmooth, it only explicitly models the hinges at its present location—all others are subject to smooth quadratic approximation. Apparently this strategy works sufficiently well during early iterations to provide for rapid improvement on multiclass problems, which typically comprise a large number of hinges. The exact location of the optimum, however, may depend on individual nearby hinges which are not represented in (10), resulting in the observed slowdown.

Bundle method solvers, by contrast, exhibit slow initial progress but tend to be competitive asymptotically. This is because they build a piecewise linear lower bound of the objective function, which initially is not very good but through successive tightening eventually becomes a faithful model. To take advantage of this we are contemplating hybrid solvers that switch over from sub(L)BFGS to a bundle method as appropriate.

While bundle methods like BMRM have an exact, implementable stopping criterion based on the duality gap, no such stopping criterion exists for BFGS and other quasi-Newton algorithms. Therefore, it is customary to use the relative change in function value as an implementable stopping criterion. Developing a stopping criterion for sub(L)BFGS based on duality arguments remains an important open question.

sub(L)BFGS relies on an efficient exact line search. We proposed such line searches for the multiclass hinge loss and its extension to the multilabel setting, based on a conceptually simple yet optimal algorithm to segment the pointwise maximum of lines. A crucial assumption we had to make is that the number $|\mathcal{Z}|$ of labels is manageable, as it takes $O(|\mathcal{Z}|\log|\mathcal{Z}|)$ time to identify the hinges associated with each training instance. In certain structured prediction problems (Tsochantaridis et al., 2005) which have recently gained prominence in machine learning, the set $\mathcal{Z}$ could

be exponentially large—for instance, predicting binary labels on a chain of length n produces 2^n possible labellings. Clearly our line searches are not efficient in such cases; we are investigating trust region variants of sub(L)BFGS to bridge this gap.

Finally, to put our contributions in perspective, recall that we modified three aspects of the standard BFGS algorithm, namely the quadratic model (Section 3.1), the descent direction finding (Section 3.2), and the Wolfe conditions (Section 3.3). Each of these modifications is versatile enough to be used as a component in other nonsmooth optimization algorithms. This not only offers the promise of improving existing algorithms, but may also help clarify connections between them. We hope that our research will focus attention on the core subroutines that need to be made more efficient in order to handle larger and larger data sets.

Acknowledgments

A short version of this paper was presented at the 2008 ICML conference (Yu et al., 2008). We thank Choon Hui Teo for many useful discussions and help with implementation issues, Xinhua Zhang for proofreading our manuscript, and the anonymous reviewers of both ICML and JMLR for their useful feedback which helped improve this paper. We thank John R. Birge for pointing us to his work (Birge et al., 1998) which led us to the convergence proof in Appendix D.

This publication only reflects the authors' views. All authors were with NICTA and the Australian National University for parts of their work on it. NICTA is funded by the Australian Government's Backing Australia's Ability and Centre of Excellence programs. This work was also supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886.

Appendix A. Bundle Search for a Descent Direction

Recall from Section 3.2 that at a subdifferential point $\boldsymbol{w}$ our goal is to find a descent direction $\boldsymbol{p}^*$ which minimizes the pseudo-quadratic model:[19]

$$M(\boldsymbol{p}) := \tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}^{-1}\boldsymbol{p} + \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}. \tag{59}$$

This is generally intractable due to the presence of a supremum over the entire subdifferential $\partial J(\boldsymbol{w})$. We therefore propose a bundle-based descent direction finding procedure (Algorithm 2) which progressively approaches $M(\boldsymbol{p})$ from below via a series of convex functions $M^{(1)}(\boldsymbol{p}), \cdots, M^{(i)}(\boldsymbol{p})$, each taking the same form as $M(\boldsymbol{p})$ but with the supremum defined over a countable subset of $\partial J(\boldsymbol{w})$. At iteration i our convex lower bound $M^{(i)}(\boldsymbol{p})$ takes the form

$$\begin{aligned} M^{(i)}(\boldsymbol{p}) &:= \tfrac{1}{2}\boldsymbol{p}^\top \boldsymbol{B}^{-1}\boldsymbol{p} + \sup_{\boldsymbol{g}\in\mathcal{V}^{(i)}} \boldsymbol{g}^\top \boldsymbol{p}, \text{ where} \\ \mathcal{V}^{(i)} &:= \{\boldsymbol{g}^{(j)} : j \le i,\ i, j\ \in \mathbb{N}\} \subseteq \partial J(\boldsymbol{w}). \end{aligned} \tag{60}$$

Given an iterate $\boldsymbol{p}^{(j-1)} \in \mathbb{R}^d$ we find a *violating subgradient* $\boldsymbol{g}^{(j)}$ via

$$\boldsymbol{g}^{(j)} := \operatorname*{arg\,sup}_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}^{(j-1)}. \tag{61}$$

19. For ease of exposition we are suppressing the iteration index t here.

Violating subgradients recover the true objective $M(\boldsymbol{p})$ at the iterates $\boldsymbol{p}^{(j-1)}$:

$$M(\boldsymbol{p}^{(j-1)}) = M^{(j)}(\boldsymbol{p}^{(j-1)}) = \tfrac{1}{2}\boldsymbol{p}^{(j-1)\top}\boldsymbol{B}^{-1}\boldsymbol{p}^{(j-1)} + \boldsymbol{g}^{(j)\top}\boldsymbol{p}^{(j-1)}. \tag{62}$$

To produce the iterates $\boldsymbol{p}^{(i)}$, we rewrite $\min_{\boldsymbol{p}\in\mathbb{R}^d} M^{(i)}(\boldsymbol{p})$ as a constrained optimization problem (19), which allows us to write the Lagrangian of (60) as

$$L^{(i)}(\boldsymbol{p},\xi,\boldsymbol{\alpha}) := \tfrac{1}{2}\boldsymbol{p}^\top\boldsymbol{B}^{-1}\boldsymbol{p} + \xi - \boldsymbol{\alpha}^\top(\xi\mathbf{1} - \boldsymbol{G}^{(i)^\top}\boldsymbol{p}), \tag{63}$$

where $\boldsymbol{G}^{(i)} := [\boldsymbol{g}^{(1)}, \boldsymbol{g}^{(2)}, \ldots, \boldsymbol{g}^{(i)}] \in \mathbb{R}^{d\times i}$ collects past violating subgradients, and $\boldsymbol{\alpha}$ is a column vector of non-negative Lagrange multipliers. Setting the derivative of (63) with respect to the primal variables ξ and $\boldsymbol{p}$ to zero yields, respectively,

$$\boldsymbol{\alpha}^\top\mathbf{1} = 1 \quad \text{and} \tag{64}$$

$$\boldsymbol{p} = -\boldsymbol{B}\boldsymbol{G}^{(i)}\boldsymbol{\alpha}. \tag{65}$$

The primal variable $\boldsymbol{p}$ and the dual variable $\boldsymbol{\alpha}$ are related via the dual connection (65). To eliminate the primal variables ξ and $\boldsymbol{p}$, we plug (64) and (65) back into the Lagrangian to obtain the dual of $M^{(i)}(\boldsymbol{p})$:

$$\begin{aligned} D^{(i)}(\boldsymbol{\alpha}) &:= -\tfrac{1}{2}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha})^\top\boldsymbol{B}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}), \\ &\text{s.t. } \boldsymbol{\alpha}\in[0,1]^i,\ \|\boldsymbol{\alpha}\|_1 = 1. \end{aligned} \tag{66}$$

The dual objective $D^{(i)}(\boldsymbol{\alpha})$ (resp., primal objective $M^{(i)}(\boldsymbol{p})$) can be maximized (resp., minimized) exactly via quadratic programming. However, doing so may incur substantial computational expense. Instead we adopt an iterative scheme which is cheap and easy to implement yet guarantees dual improvement.

Let $\boldsymbol{\alpha}^{(i)} \in [0,1]^i$ be a feasible solution for $D^{(i)}(\boldsymbol{\alpha})$.[20] The corresponding primal solution $\boldsymbol{p}^{(i)}$ can be found by using (65). This in turn allows us to compute the next violating subgradient $\boldsymbol{g}^{(i+1)}$ via (61). With the new violating subgradient the dual becomes

$$\begin{aligned} D^{(i+1)}(\boldsymbol{\alpha}) &:= -\tfrac{1}{2}(\boldsymbol{G}^{(i+1)}\boldsymbol{\alpha})^\top\boldsymbol{B}(\boldsymbol{G}^{(i+1)}\boldsymbol{\alpha}), \\ &\text{s.t. } \boldsymbol{\alpha}\in[0,1]^{i+1},\ \|\boldsymbol{\alpha}\|_1 = 1, \end{aligned} \tag{67}$$

where the subgradient matrix is now extended:

$$\boldsymbol{G}^{(i+1)} = [\boldsymbol{G}^{(i)}, \boldsymbol{g}^{(i+1)}]. \tag{68}$$

Our iterative strategy constructs a new feasible solution $\boldsymbol{\alpha}\in[0,1]^{i+1}$ for (67) by constraining it to take the following form:

$$\boldsymbol{\alpha} = \begin{bmatrix} (1-\mu)\boldsymbol{\alpha}^{(i)} \\ \mu \end{bmatrix}, \quad \text{where } \mu\in[0,1]. \tag{69}$$

20. Note that $\boldsymbol{\alpha}^{(1)} = \mathbf{1}$ is a feasible solution for $D^{(1)}(\boldsymbol{\alpha})$.

In other words, we maximize a one-dimensional function $\bar{D}^{(i+1)} : [0,1] \to \mathbb{R}$:

$$\begin{aligned} \bar{D}^{(i+1)}(\mu) &:= -\tfrac{1}{2}\left(\boldsymbol{G}^{(i+1)}\boldsymbol{\alpha}\right)^\top \boldsymbol{B}\left(\boldsymbol{G}^{(i+1)}\boldsymbol{\alpha}\right) \\ &= -\tfrac{1}{2}\left((1-\mu)\bar{\boldsymbol{g}}^{(i)} + \mu \boldsymbol{g}^{(i+1)}\right)^\top \boldsymbol{B}\left((1-\mu)\bar{\boldsymbol{g}}^{(i)} + \mu \boldsymbol{g}^{(i+1)}\right), \end{aligned} \tag{70}$$

where

$$\bar{\boldsymbol{g}}^{(i)} := \boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)} \in \partial J(\boldsymbol{w}) \tag{71}$$

lies in the convex hull of $\boldsymbol{g}^{(j)} \in \partial J(\boldsymbol{w})\ \forall j \leq i$ (and hence in the convex set $\partial J(\boldsymbol{w})$) because $\boldsymbol{\alpha}^{(i)} \in [0,1]^i$ and $\|\boldsymbol{\alpha}^{(i)}\|_1 = 1$. Moreover, $\mu \in [0,1]$ ensures the feasibility of the dual solution. Noting that $\bar{D}^{(i+1)}(\mu)$ is a concave quadratic function, we set

$$\partial \bar{D}^{(i+1)}(\mu) = \left(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)}\right)^\top \boldsymbol{B}\left((1-\eta)\bar{\boldsymbol{g}}^{(i)} + \eta \boldsymbol{g}^{(i+1)}\right) = 0 \tag{72}$$

to obtain the optimum

$$\mu^* := \operatorname*{argmax}_{\mu \in [0,1]} \bar{D}^{(i+1)}(\mu) = \min\left(1, \max\left(0, \frac{(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B}\bar{\boldsymbol{g}}^{(i)}}{(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B}(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})}\right)\right). \tag{73}$$

Our dual solution at step $i+1$ then becomes

$$\boldsymbol{\alpha}^{(i+1)} := \begin{bmatrix} (1-\mu^*)\boldsymbol{\alpha}^{(i)} \\ \mu^* \end{bmatrix}. \tag{74}$$

Furthermore, from (68), (69), and (71) it follows that $\bar{\boldsymbol{g}}^{(i)}$ can be maintained via an incremental update (Line 8 of Algorithm 2):

$$\bar{\boldsymbol{g}}^{(i+1)} := \boldsymbol{G}^{(i+1)}\boldsymbol{\alpha}^{(i+1)} = (1-\mu^*)\bar{\boldsymbol{g}}^{(i)} + \mu^* \boldsymbol{g}^{(i+1)}, \tag{75}$$

which combined with the dual connection (65) yields an incremental update for the primal solution (Line 9 of Algorithm 2):

$$\begin{aligned} \boldsymbol{p}^{(i+1)} := -\boldsymbol{B}\bar{\boldsymbol{g}}^{(i+1)} &= -(1-\mu^*)\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)} - \mu^* \boldsymbol{B}\boldsymbol{g}^{(i+1)} \\ &= (1-\mu^*)\boldsymbol{p}^{(i)} - \mu^* \boldsymbol{B}\boldsymbol{g}^{(i+1)}. \end{aligned} \tag{76}$$

Using (75) and (76), computing a primal solution (Lines 7–9 of Algorithm 2) costs a total of $O(d^2)$ time (resp., $O(md)$ time for LBFGS with buffer size m), where d is the dimensionality of the optimization problem. Note that maximizing $D^{(i+1)}(\boldsymbol{\alpha})$ directly via quadratic programming generally results in a larger progress than that obtained by our approach.

In order to measure the quality of our solution at iteration i, we define the quantity

$$\varepsilon^{(i)} := \min_{j \leq i} M^{(j+1)}(\boldsymbol{p}^{(j)}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}) = \min_{j \leq i} M(\boldsymbol{p}^{(j)}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}), \tag{77}$$

where the second equality follows directly from (62). Let $D(\boldsymbol{\alpha})$ be the corresponding dual problem of $M(\boldsymbol{p})$, with the property $D\left(\begin{bmatrix} \boldsymbol{\alpha}^{(i)} \\ \mathbf{0} \end{bmatrix}\right) = D^{(i)}(\boldsymbol{\alpha}^{(i)})$, and let $\boldsymbol{\alpha}^*$ be the optimal solution to

$\operatorname{argmax}_{\boldsymbol{\alpha}\in\mathcal{A}} D(\boldsymbol{\alpha})$ in some domain $\mathcal{A}$ of interest. As a consequence of the weak duality theorem (Hiriart-Urruty and Lemaréchal, 1993, Theorem XII.2.1.5), $\min_{\boldsymbol{p}\in\mathbb{R}^d} M(\boldsymbol{p}) \geq D(\boldsymbol{\alpha}^*)$. Therefore (77) implies that

$$\varepsilon^{(i)} \;\geq\; \min_{\boldsymbol{p}\in\mathbb{R}^d} M(\boldsymbol{p}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;\geq\; \min_{\boldsymbol{p}\in\mathbb{R}^d} M(\boldsymbol{p}) - D(\boldsymbol{\alpha}^*) \;\geq\; 0. \tag{78}$$

The second inequality essentially says that $\varepsilon^{(i)}$ is an upper bound on the duality gap. In fact, Theorem 7 below shows that $(\varepsilon^{(i)} - \varepsilon^{(i+1)})$ is bounded away from 0, that is, $\varepsilon^{(i)}$ is monotonically decreasing. This guides us to design a practical stopping criterion (Line 6 of Algorithm 2) for our direction-finding procedure. Furthermore, using the dual connection (65), we can derive an implementable formula for $\varepsilon^{(i)}$:

$$\begin{aligned}
\varepsilon^{(i)} &= \min_{j\leq i} \left[\tfrac{1}{2}\boldsymbol{p}^{(j)\top}\boldsymbol{B}^{-1}\boldsymbol{p}^{(j)} + \boldsymbol{p}^{(j)\top}\boldsymbol{g}^{(j+1)} + \tfrac{1}{2}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)})^\top \boldsymbol{B}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)}) \right] \\
&= \min_{j\leq i} \left[-\tfrac{1}{2}\boldsymbol{p}^{(j)\top}\bar{\boldsymbol{g}}^{(j)} + \boldsymbol{p}^{(j)\top}\boldsymbol{g}^{(j+1)} - \tfrac{1}{2}\boldsymbol{p}^{(i)\top}\bar{\boldsymbol{g}}^{(i)} \right] \\
&= \min_{j\leq i} \left[\boldsymbol{p}^{(j)\top}\boldsymbol{g}^{(j+1)} - \tfrac{1}{2}(\boldsymbol{p}^{(j)\top}\bar{\boldsymbol{g}}^{(j)} + \boldsymbol{p}^{(i)^\top}\bar{\boldsymbol{g}}^{(i)}) \right], \\
&\quad \text{where } \boldsymbol{g}^{(j+1)} := \operatorname*{arg\,sup}_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}^{(j)} \text{ and } \bar{\boldsymbol{g}}^{(j)} := \boldsymbol{G}^{(j)}\boldsymbol{\alpha}^{(j)} \;\; \forall j \leq i.
\end{aligned} \tag{79}$$

It is worth noting that continuous progress in the dual objective value does not necessarily prevent an increase in the primal objective value, that is, it is possible that $M(\boldsymbol{p}^{(i+1)}) \geq M(\boldsymbol{p}^{(i)})$. Therefore, we choose the best primal solution so far,

$$\boldsymbol{p} := \operatorname*{argmin}_{j\leq i} M(\boldsymbol{p}^{(j)}), \tag{80}$$

as the search direction (Line 18 of Algorithm 2) for the parameter update (3). This direction is a direction of descent as long as the last iterate $\boldsymbol{p}^{(i)}$ fulfills the descent condition (16). To see this, we use (88–90) below to get $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}^{(i)} = M(\boldsymbol{p}^{(i)}) + D^{(i)}(\boldsymbol{\alpha}^{(i)})$, and since

$$M(\boldsymbol{p}^{(i)}) \geq \min_{j\leq i} M(\boldsymbol{p}^{(j)}) \text{ and } D^{(i)}(\boldsymbol{\alpha}^{(i)}) \geq D^{(j)}(\boldsymbol{\alpha}^{(j)}) \;\; \forall j \leq i,$$

definition (80) immediately gives $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}^{(i)} \;\geq\; \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p}$. Hence if $\boldsymbol{p}^{(i)}$ is a descent direction, then so is $\boldsymbol{p}$.

We now show that if the current parameter vector $\boldsymbol{w}$ is not optimal, then a direction-finding tolerance $\varepsilon \geq 0$ exists for Algorithm 2 such that the returned search direction $\boldsymbol{p}$ is a descent direction, that is, $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p} < 0$.

Lemma 3 *Let $\boldsymbol{B}$ be the current approximation to the inverse Hessian maintained by Algorithm 1, and $h > 0$ a lower bound on the eigenvalues of $\boldsymbol{B}$. If the current iterate $\boldsymbol{w}$ is not optimal: $\mathbf{0} \notin \partial J(\boldsymbol{w})$, and the number of direction-finding iterations is unlimited ($k_{max} = \infty$), then there exists a direction-finding tolerance $\varepsilon \geq 0$ such that the descent direction $\boldsymbol{p} = -\boldsymbol{B}\bar{\boldsymbol{g}}$, $\bar{\boldsymbol{g}} \in \partial J(\boldsymbol{w})$ returned by Algorithm 2 at $\boldsymbol{w}$ satisfies $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top \boldsymbol{p} < 0$.*

Proof Algorithm 2 returns $\boldsymbol{p}$ after i iterations when $\varepsilon^{(i)} \leq \varepsilon$, where $\varepsilon^{(i)} = M(\boldsymbol{p}) - D^{(i)}(\boldsymbol{\alpha}^{(i)})$ by definitions (77) and (80). Using definition (66) of $D^{(i)}(\boldsymbol{\alpha}^{(i)})$, we have

$$-D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;=\; \tfrac{1}{2}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)})^\top \boldsymbol{B}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)}) \;=\; \tfrac{1}{2}\bar{\boldsymbol{g}}^{(i)\top}\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)}, \tag{81}$$

where $\bar{\boldsymbol{g}}^{(i)} = \boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)}$ is a subgradient in $\partial J(\boldsymbol{w})$. On the other hand, using (59) and (76), one can write

$$\begin{aligned} M(\boldsymbol{p}) \;&=\; \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} + \tfrac{1}{2}\boldsymbol{p}^\top\boldsymbol{B}^{-1}\boldsymbol{p} \\ &=\; \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} + \tfrac{1}{2}\bar{\boldsymbol{g}}^\top\boldsymbol{B}\bar{\boldsymbol{g}}, \quad \text{where} \quad \bar{\boldsymbol{g}} \in \partial J(\boldsymbol{w}). \end{aligned} \tag{82}$$

Putting together (81) and (82), and using $\boldsymbol{B} \succ h$, one obtains

$$\varepsilon^{(i)} = \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} + \tfrac{1}{2}\bar{\boldsymbol{g}}^\top\boldsymbol{B}\bar{\boldsymbol{g}} + \tfrac{1}{2}\bar{\boldsymbol{g}}^{(i)\top}\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)} \geq \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} + \frac{h}{2}\|\bar{\boldsymbol{g}}\|^2 + \frac{h}{2}\|\bar{\boldsymbol{g}}^{(i)}\|^2. \tag{83}$$

Since $\mathbf{0} \notin \partial J(\boldsymbol{w})$, the last two terms of (83) are strictly positive; and by (78), $\varepsilon^{(i)} \geq 0$. The claim follows by choosing an ε such that $(\forall i)\;\; \frac{h}{2}(\|\bar{\boldsymbol{g}}\|^2 + \|\bar{\boldsymbol{g}}^{(i)}\|^2) > \varepsilon \geq \varepsilon^{(i)} \geq 0$. ■

Using the notation from Lemma 3, we show in the following corollary that a stricter upper bound on ε allows us to bound $\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p}$ in terms of $\bar{\boldsymbol{g}}^\top\boldsymbol{B}\bar{\boldsymbol{g}}$ and $\|\bar{\boldsymbol{g}}\|$. This will be used in Appendix D to establish the global convergence of the subBFGS algorithm.

Corollary 4 *Under the conditions of Lemma 3, there exists an* $\varepsilon \geq 0$ *for Algorithm 2 such that the search direction* $\boldsymbol{p}$ *generated by Algorithm 2 satisfies*

$$\sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} \leq -\tfrac{1}{2}\bar{\boldsymbol{g}}^\top\boldsymbol{B}\bar{\boldsymbol{g}} \leq -\frac{h}{2}\|\bar{\boldsymbol{g}}\|^2 < 0. \tag{84}$$

Proof Using (83), we have

$$(\forall i)\;\; \varepsilon^{(i)} \geq \sup_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} + \tfrac{1}{2}\bar{\boldsymbol{g}}^\top\boldsymbol{B}\bar{\boldsymbol{g}} + \frac{h}{2}\|\bar{\boldsymbol{g}}^{(i)}\|^2.$$

The first inequality in (84) results from choosing an ε such that

$$(\forall i)\;\; \frac{h}{2}\|\bar{\boldsymbol{g}}^{(i)}\|^2 \geq \varepsilon \geq \varepsilon^{(i)} \geq 0. \tag{85}$$

The lower bound $h > 0$ on the spectrum of $\boldsymbol{B}$ yields the second inequality in (84), and the third follows from the fact that $\|\bar{\boldsymbol{g}}\| > 0$ at non-optimal iterates. ■

Appendix B. Convergence of the Descent Direction Search

Using the notation established in Appendix A, we now prove the convergence of Algorithm 2 via several technical intermediate steps. The proof shares similarities with the proofs found in Smola et al. (2007), Shalev-Shwartz and Singer (2008), and Warmuth et al. (2008). The key idea is that at each iterate Algorithm 2 decreases the upper bound $\varepsilon^{(i)}$ on the distance from the optimality, and the decrease in $\varepsilon^{(i)}$ is characterized by the recurrence $\varepsilon^{(i)} - \varepsilon^{(i+1)} \geq c(\varepsilon^{(i)})^2$ with $c > 0$ (Theorem 7). Analysing this recurrence then gives the convergence rate of the algorithm (Theorem 9).

We first provide two technical lemmas (Lemma 5 and 6) that are needed to prove Theorem 7.

Lemma 5 *Let $\bar{D}^{(i+1)}(\mu)$ be the one-dimensional function defined in* (70)*, and $\varepsilon^{(i)}$ the positive measure defined in* (77)*. Then $\varepsilon^{(i)} \leq \partial\bar{D}^{(i+1)}(0)$.*

Proof Let $\boldsymbol{p}^{(i)}$ be our primal solution at iteration i, derived from the dual solution $\boldsymbol{\alpha}^{(i)}$ using the dual connection (65). We then have

$$\boldsymbol{p}^{(i)} = -\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)}, \quad \text{where} \quad \bar{\boldsymbol{g}}^{(i)} := \boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)}. \tag{86}$$

Definition (59) of $M(\boldsymbol{p})$ implies that

$$M(\boldsymbol{p}^{(i)}) \;=\; \tfrac{1}{2}{\boldsymbol{p}^{(i)}}^{\top}\boldsymbol{B}^{-1}\boldsymbol{p}^{(i)} + {\boldsymbol{p}^{(i)}}^{\top}\boldsymbol{g}^{(i+1)}, \tag{87}$$

where

$$\boldsymbol{g}^{(i+1)} := \operatorname*{arg\,sup}_{\boldsymbol{g}\in\partial J(\boldsymbol{w})} \boldsymbol{g}^{\top}\boldsymbol{p}^{(i)}. \tag{88}$$

Using (86), we have $\boldsymbol{B}^{-1}\boldsymbol{p}^{(i)} = -\boldsymbol{B}^{-1}\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)} = -\bar{\boldsymbol{g}}^{(i)}$, and hence (87) becomes

$$M(\boldsymbol{p}^{(i)}) \;=\; {\boldsymbol{p}^{(i)}}^{\top}\boldsymbol{g}^{(i+1)} - \tfrac{1}{2}{\boldsymbol{p}^{(i)}}^{\top}\bar{\boldsymbol{g}}^{(i)}. \tag{89}$$

Similarly, we have

$$D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;=\; -\tfrac{1}{2}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)})^{\top}\boldsymbol{B}(\boldsymbol{G}^{(i)}\boldsymbol{\alpha}^{(i)}) \;=\; \tfrac{1}{2}{\boldsymbol{p}^{(i)}}^{\top}\bar{\boldsymbol{g}}^{(i)}. \tag{90}$$

From (72) and (86) it follows that

$$\partial\bar{D}^{(i+1)}(0) \;=\; (\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^{\top}\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)} \;=\; (\boldsymbol{g}^{(i+1)} - \bar{\boldsymbol{g}}^{(i)})^{\top}\boldsymbol{p}^{(i)}, \tag{91}$$

where $\boldsymbol{g}^{(i+1)}$ is a violating subgradient chosen via (61), and hence coincides with (88). Using (89)–(91), we obtain

$$M(\boldsymbol{p}^{(i)}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;=\; \left(\boldsymbol{g}^{(i+1)} - \bar{\boldsymbol{g}}^{(i)}\right)^{\top}\boldsymbol{p}^{(i)} \;=\; \partial\bar{D}^{(i+1)}(0). \tag{92}$$

Together with definition (77) of $\varepsilon^{(i)}$, (92) implies that

$$\begin{aligned}\varepsilon^{(i)} \;&=\; \min_{j\leq i} M(\boldsymbol{p}^{(j)}) - D^{(i)}\left(\boldsymbol{\alpha}^{(i)}\right)\\ &\leq\; M(\boldsymbol{p}^{(i)}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;=\; \partial\bar{D}^{(i+1)}(0).\end{aligned}$$

■

Lemma 6 *Let $f : [0,1] \to \mathbb{R}$ be a concave quadratic function with $f(0) = 0$, $\partial f(0) \in [0,a]$, and $\partial f^2(x) \geq -a$ for some $a \geq 0$. Then $\max_{x\in[0,1]} f(x) \geq \frac{(\partial f(0))^2}{2a}$.*

Proof Using a second-order Taylor expansion around 0, we have $f(x) \geq \partial f(0)x - \frac{a}{2}x^2$. $x^* = \partial f(0)/a$ is the unconstrained maximum of the lower bound. Since $\partial f(0) \in [0,a]$, we have $x^* \in [0,1]$. Plugging x^* into the lower bound yields $(\partial f(0))^2/(2a)$. ■

Theorem 7 *Assume that at $\boldsymbol{w}$ the convex objective function $J : \mathbb{R}^d \to \mathbb{R}$ has bounded subgradient: $\|\partial J(\boldsymbol{w})\| \leq G$, and that the approximation $\boldsymbol{B}$ to the inverse Hessian has bounded eigenvalues: $\boldsymbol{B} \preceq H$. Then*

$$\varepsilon^{(i)} - \varepsilon^{(i+1)} \;\geq\; \frac{(\varepsilon^{(i)})^2}{8G^2H}.$$

Proof Recall that we constrain the form of feasible dual solutions for $D^{(i+1)}(\boldsymbol{\alpha})$ as in (69). Instead of $D^{(i+1)}(\boldsymbol{\alpha})$, we thus work with the one-dimensional concave quadratic function $\bar{D}^{(i+1)}(\mu)$ (70). It is obvious that $\begin{bmatrix} \boldsymbol{\alpha}^{(i)} \\ 0 \end{bmatrix}$ is a feasible solution for $D^{(i+1)}(\boldsymbol{\alpha})$. In this case, $\bar{D}^{(i+1)}(0) = D^{(i)}(\boldsymbol{\alpha}^{(i)})$. (74) implies that $\bar{D}^{(i+1)}(\mu^*) = D^{(i+1)}(\boldsymbol{\alpha}^{(i+1)})$. Using the definition (77) of $\varepsilon^{(i)}$, we thus have

$$\varepsilon^{(i)} - \varepsilon^{(i+1)} \;\geq\; D^{(i+1)}(\boldsymbol{\alpha}^{(i+1)}) - D^{(i)}(\boldsymbol{\alpha}^{(i)}) \;=\; \bar{D}^{(i+1)}(\mu^*) - \bar{D}^{(i+1)}(0). \tag{93}$$

It is easy to see from (93) that $\varepsilon^{(i)} - \varepsilon^{(i+1)}$ are upper bounds on the maximal value of the concave quadratic function $f(\mu) := \bar{D}^{(i+1)}(\mu) - \bar{D}^{(i+1)}(0)$ with $\mu \in [0,1]$ and $f(0) = 0$. Furthermore, the definitions of $\bar{D}^{(i+1)}(\mu)$ and $f(\mu)$ imply that

$$\begin{aligned} \partial f(0) \;&=\; \partial \bar{D}^{(i+1)}(0) \;=\; (\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B} \bar{\boldsymbol{g}}^{(i)} \quad \text{and} \\ \partial^2 f(\mu) \;&=\; \partial^2 \bar{D}^{(i+1)}(\mu) \;=\; -(\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B} (\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)}). \end{aligned} \tag{94}$$

Since $\|\partial J(\boldsymbol{w})\| \leq G$ and $\bar{\boldsymbol{g}}^{(i)} \in \partial J(\boldsymbol{w})$ (71), we have $\|\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)}\| \leq 2G$. Our upper bound on the spectrum of $\boldsymbol{B}$ then gives $|\partial f(0)| \leq 2G^2H$ and $\left|\partial^2 f(\mu)\right| \leq 4G^2H$. Additionally, Lemma 5 and the fact that $\boldsymbol{B} \succeq 0$ imply that

$$\partial f(0) \;=\; \partial \bar{D}^{(i+1)}(0) \geq 0 \quad \text{and} \quad \partial^2 f(\mu) \;=\; \partial^2 \bar{D}^{(i+1)}(\mu) \leq 0, \tag{95}$$

which means that

$$\partial f(0) \;\in\; [0, 2G^2H] \subset [0, 4G^2H] \quad \text{and} \quad \partial^2 f(\mu) \;\geq\; -4G^2H.$$

Invoking Lemma 6, we immediately get

$$\varepsilon^{(i)} - \varepsilon^{(i+1)} \;\geq\; \frac{(\partial f(0))^2}{8G^2H} \;=\; \frac{(\partial \bar{D}^{(i+1)}(0))^2}{8G^2H}. \tag{96}$$

Since $\varepsilon^{(i)} \leq \partial \bar{D}^{(i+1)}(0)$ by Lemma 5, the inequality (96) still holds when $\partial \bar{D}^{(i+1)}(0)$ is replaced with $\varepsilon^{(i)}$. ■

(94) and (95) imply that the optimal combination coefficient μ^* (73) has the property

$$\mu^* = \min\left[1, \frac{\partial \bar{D}^{(i+1)}(0)}{-\partial^2 \bar{D}^{(i+1)}(\mu)}\right].$$

Moreover, we can use (65) to reduce the cost of computing μ^* by setting $\boldsymbol{B}\bar{\boldsymbol{g}}^{(i)}$ in (73) to be $-\boldsymbol{p}^{(i)}$ (Line 7 of Algorithm 2), and calculate

$$\mu^* = \min\left[1, \frac{\boldsymbol{g}^{(i+1)\top}\boldsymbol{p}^{(i)} - \bar{\boldsymbol{g}}^{(i)\top}\boldsymbol{p}^{(i)}}{\boldsymbol{g}^{(i+1)\top}\boldsymbol{B}_t\boldsymbol{g}^{(i+1)} + 2\,\boldsymbol{g}^{(i+1)\top}\boldsymbol{p}^{(i)} - \bar{\boldsymbol{g}}^{(i)\top}\boldsymbol{p}^{(i)}}\right], \tag{97}$$

where $\boldsymbol{B}_t\boldsymbol{g}^{(i+1)}$ can be cached for the update of the primal solution at Line 9 of Algorithm 2.

To prove Theorem 9, we use the following lemma proven by induction by Abe et al. (2001, Sublemma 5.4):

Lemma 8 *Let* $\{\varepsilon^{(1)}, \varepsilon^{(2)}, \cdots\}$ *be a sequence of non-negative numbers satisfying* $\forall i \in \mathbb{N}$ *the recurrence*

$$\varepsilon^{(i)} - \varepsilon^{(i+1)} \;\geq\; c\,(\varepsilon^{(i)})^2,$$

where $c \in \mathbb{R}_+$ *is a positive constant. Then* $\forall i \in \mathbb{N}$ *we have*

$$\varepsilon^{(i)} \;\leq\; \frac{1}{c\left(i + \frac{1}{\varepsilon^{(1)}c}\right)}.$$

We now show that Algorithm 2 decreases $\varepsilon^{(i)}$ to a pre-defined tolerance ε in $O(1/\varepsilon)$ steps:

Theorem 9 *Under the assumptions of Theorem 7, Algorithm 2 converges to the desired precision* ε *after*

$$1 \;\leq\; t \;\leq\; \frac{8G^2H}{\varepsilon} - 4$$

steps for any $\varepsilon < 2G^2H$.

Proof Theorem 7 states that

$$\varepsilon^{(i)} - \varepsilon^{(i+1)} \;\geq\; \frac{(\varepsilon^{(i)})^2}{8G^2H},$$

where $\varepsilon^{(i)}$ is non-negative $\forall i \in \mathbb{N}$ by (78). Applying Lemma 8 we thus obtain

$$\varepsilon^{(i)} \;\leq\; \frac{1}{c\left(i + \frac{1}{\varepsilon^{(1)}c}\right)}, \quad \text{where} \quad c := \frac{1}{8G^2H}. \tag{98}$$

Our assumptions on $\|\partial J(\boldsymbol{w})\|$ and the spectrum of $\boldsymbol{B}$ imply that

$$\bar{D}^{(i+1)}(0) \;=\; (\bar{\boldsymbol{g}}^{(i)} - \boldsymbol{g}^{(i+1)})^\top \boldsymbol{B}\bar{\boldsymbol{g}}^{(i)} \;\leq\; 2G^2H.$$

Hence $\varepsilon^{(i)} \leq 2G^2H$ by Lemma 5. This means that (98) holds with $\varepsilon^{(1)} = 2G^2H$. Therefore we can solve

$$\varepsilon \leq \frac{1}{c\left(t + \frac{1}{\varepsilon^{(1)}c}\right)} \quad \text{with} \quad c := \frac{1}{8G^2H} \;\text{ and }\; \varepsilon^{(1)} := 2G^2H \tag{99}$$

to obtain an upper bound on t such that $(\forall i \geq t)\ \varepsilon^{(i)} \leq \varepsilon < 2G^2H$. The solution to (99) is $t \leq \frac{8G^2H}{\varepsilon} - 4$. ■

Appendix C. Satisfiability of the Subgradient Wolfe Conditions

To formally show that there always is a positive step size that satisfies the subgradient Wolfe conditions (23, 24), we restate a result of Hiriart-Urruty and Lemaréchal (1993, Theorem VI.2.3.3) in slightly modified form:

Lemma 10 *Given two points $\boldsymbol{w} \neq \boldsymbol{w}'$ in $\mathbb{R}^d$, define $\boldsymbol{w}_\eta = \eta\boldsymbol{w}' + (1-\eta)\boldsymbol{w}$. Let $J : \mathbb{R}^d \to \mathbb{R}$ be convex. There exists $\eta \in (0,1)$ and $\tilde{\boldsymbol{g}} \in \partial J(\boldsymbol{w}_\eta)$ such that*

$$J(\boldsymbol{w}') - J(\boldsymbol{w}) \;=\; \tilde{\boldsymbol{g}}^\top(\boldsymbol{w}' - \boldsymbol{w}) \;\leq\; \hat{\boldsymbol{g}}^\top(\boldsymbol{w}' - \boldsymbol{w}),$$

where $\hat{\boldsymbol{g}} := \arg\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_\eta)} \boldsymbol{g}^\top(\boldsymbol{w}' - \boldsymbol{w})$.

Theorem 11 *Let $\boldsymbol{p}$ be a descent direction at an iterate $\boldsymbol{w}$. If $\Phi(\eta) := J(\boldsymbol{w} + \eta\boldsymbol{p})$ is bounded below, then there exists a step size $\eta > 0$ which satisfies the subgradient Wolfe conditions (23, 24).*

Proof Since $\boldsymbol{p}$ is a descent direction, the line $J(\boldsymbol{w}) + c_1\eta \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p}$ with $c_1 \in (0,1)$ must intersect $\Phi(\eta)$ at least once at some $\eta > 0$ (see Figure 1 for geometric intuition). Let η' be the smallest such intersection point; then

$$J(\boldsymbol{w} + \eta'\boldsymbol{p}) \;=\; J(\boldsymbol{w}) \;+\; c_1\eta' \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p}. \tag{100}$$

Since $\Phi(\eta)$ is lower bounded, the sufficient decrease condition (23) holds for all $\eta'' \in [0, \eta']$. Setting $\boldsymbol{w}' = \boldsymbol{w} + \eta'\boldsymbol{p}$ in Lemma 10 implies that there exists an $\eta'' \in (0, \eta')$ such that

$$J(\boldsymbol{w} + \eta'\boldsymbol{p}) - J(\boldsymbol{w}) \;\leq\; \eta' \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w} + \eta''\boldsymbol{p})} \boldsymbol{g}^\top\boldsymbol{p}. \tag{101}$$

Plugging (100) into (101) and simplifying it yields

$$c_1 \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} \;\leq \sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w} + \eta''\boldsymbol{p})} \boldsymbol{g}^\top\boldsymbol{p}. \tag{102}$$

Since $\boldsymbol{p}$ is a descent direction, $\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w})} \boldsymbol{g}^\top\boldsymbol{p} < 0$, and thus (102) also holds when c_1 is replaced by $c_2 \in (c_1, 1)$. ■

Algorithm 6 Algorithm 1 of Birge et al. (1998)

1: Initialize: $t := 0$ and $\boldsymbol{w}_0$
2: **while** not converged **do**
3: Find $\boldsymbol{w}_{t+1}$ that obeys

$$J(\boldsymbol{w}_{t+1}) \leq J(\boldsymbol{w}_t) - a_t \|\boldsymbol{g}_{\varepsilon'_t}\|^2 + \varepsilon_t \quad \text{where } \boldsymbol{g}_{\varepsilon'_t} \in \partial_{\varepsilon'_t} J(\boldsymbol{w}_{t+1}),\ a_t > 0,\ \varepsilon_t, \varepsilon'_t \geq 0\,. \tag{104}$$

4: $t := t + 1$
5: **end while**

Appendix D. Global Convergence of SubBFGS

There are technical difficulties in extending the classical BFGS convergence proof to the nonsmooth case. This route was taken by Andrew and Gao (2007), which unfortunately left their proof critically flawed: In a key step (Andrew and Gao, 2007, Equation 7) they seek to establish the non-negativity of the directional derivative $f'(\bar{x};\bar{q})$ of a convex function f at a point $\bar{x}$ in the direction $\bar{q}$, where $\bar{x}$ and $\bar{q}$ are the limit points of convergent sequences $\{x^k\}$ and $\{\hat{q}^k\}_\kappa$, respectively. They do so by taking the limit for $k \in \kappa$ of

$$f'(x^k + \tilde{\alpha}^k \hat{q}^k; \hat{q}^k) \ > \ \gamma f'(x^k; \hat{q}^k), \ \text{ where } \ \{\tilde{\alpha}^k\} \to 0 \ \text{ and } \ \gamma \in (0,1)\,,$$

which leads them to claim that

$$f'(\bar{x};\bar{q}) \ \geq \ \gamma f'(\bar{x};\bar{q})\,, \tag{103}$$

which would imply $f'(\bar{x};\bar{q}) \geq 0$ because $\gamma \in (0,1)$. However, $f'(x^k, \hat{q}^k)$ does not necessarily converge to $f'(\bar{x};\bar{q})$ because the directional derivative of a nonsmooth convex function is not continuous, only *upper semi-continuous* (Bertsekas, 1999, Proposition B.23). Instead of (103) we thus only have

$$f'(\bar{x};\bar{q}) \ \geq \ \gamma \limsup_{k\to\infty, k\in\kappa} f'(x^k;\hat{q}^k)\,,$$

which does not suffice to establish the desired result: $f'(\bar{x};\bar{q}) \geq 0$. A similar mistake is also found in the reasoning of Andrew and Gao (2007) just after Equation 7.

Instead of this flawed approach, we use the technique introduced by Birge et al. (1998) to prove the global convergence of subBFGS (Algorithm 1) in objective function value, that is, $J(\boldsymbol{w}_t) \to \inf_{\boldsymbol{w}} J(\boldsymbol{w})$, provided that the spectrum of BFGS' inverse Hessian approximation $\boldsymbol{B}_t$ is bounded from above and below for all t, and the step size η_t (obtained at Line 9) is not summable: $\sum_{t=0}^{\infty} \eta_t = \infty$.

Birge et al. (1998) provide a unified framework for convergence analysis of optimization algorithms for nonsmooth convex optimization, based on the notion of ε-*subgradients*. Formally, $\boldsymbol{g}$ is called an ε-subgradient of J at $\boldsymbol{w}$ iff (Hiriart-Urruty and Lemaréchal, 1993, Definition XI.1.1.1)

$$(\forall \boldsymbol{w}') \ J(\boldsymbol{w}') \ \geq \ J(\boldsymbol{w}) + (\boldsymbol{w}' - \boldsymbol{w})^\top \boldsymbol{g} - \varepsilon, \ \text{ where } \ \varepsilon \geq 0. \tag{105}$$

The set of all ε-subgradients at a point $\boldsymbol{w}$ is called the ε-subdifferential, and denoted $\partial_\varepsilon J(\boldsymbol{w})$. From the definition of subgradient (7), it is easy to see that $\partial J(\boldsymbol{w}) = \partial_0 J(\boldsymbol{w}) \subseteq \partial_\varepsilon J(\boldsymbol{w})$. Birge et al. (1998) propose an ε-subgradient-based algorithm (Algorithm 6) and provide sufficient conditions for its global convergence:

Theorem 12 (Birge et al., 1998, Theorem 2.1(iv), first sentence)
Let $J : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ be a proper lower semi-continuous[21] *extended-valued convex function, and let $\{(\varepsilon_t, \varepsilon'_t, a_t, \boldsymbol{w}_{t+1}, \boldsymbol{g}_{\varepsilon'_t})\}$ be any sequence generated by Algorithm 6 satisfying*

$$\sum_{t=0}^{\infty} \varepsilon_t < \infty \;\; \text{and} \;\; \sum_{t=0}^{\infty} a_t = \infty. \tag{106}$$

If $\varepsilon'_t \to 0$, and there exists a positive number $\beta > 0$ such that, for all large t,

$$\beta \, \|\boldsymbol{w}_{t+1} - \boldsymbol{w}_t\| \;\leq\; a_t \|\boldsymbol{g}_{\varepsilon'_t}\|, \tag{107}$$

then $J(\boldsymbol{w}_t) \to \inf_{\boldsymbol{w}} J(\boldsymbol{w})$.

We will use this result to establish the global convergence of subBFGS in Theorem 14. Towards this end, we first show that subBFGS is a special case of Algorithm 6:

Lemma 13 *Let $\boldsymbol{p}_t = -\boldsymbol{B}_t \bar{\boldsymbol{g}}_t$ be the descent direction produced by Algorithm 2 at a non-optimal iterate $\boldsymbol{w}_t$, where $\boldsymbol{B}_t \succeq h > 0$ and $\bar{\boldsymbol{g}}_t \in \partial J(\boldsymbol{w}_t)$, and let $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta_t \boldsymbol{p}_t$, where $\eta_t > 0$ satisfies sufficient decrease* (23) *with free parameter $c_1 \in (0,1)$. Then $\boldsymbol{w}_{t+1}$ obeys* (104) *of Algorithm 6 for $a_t := \frac{c_1 \eta_t h}{2}$, $\varepsilon_t = 0$, and $\varepsilon'_t := \eta_t (1 - \frac{c_1}{2}) \, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t$.*

Proof Our sufficient decrease condition (23) and Corollary 4 imply that

$$\begin{aligned} J(\boldsymbol{w}_{t+1}) \;&\leq\; J(\boldsymbol{w}_t) - \frac{c_1 \eta_t}{2} \, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t \\ &\leq\; J(\boldsymbol{w}_t) - a_t \|\bar{\boldsymbol{g}}_t\|^2, \;\; \text{where} \;\; a_t := \frac{c_1 \eta_t h}{2}. \end{aligned} \tag{108}$$

What is left to prove is that $\bar{\boldsymbol{g}}_t \in \partial_{\varepsilon'_t} J(\boldsymbol{w}_{t+1})$ for an $\varepsilon'_t \geq 0$. Using $\bar{\boldsymbol{g}}_t \in \partial J(\boldsymbol{w}_t)$ and the definition (7) of subgradient, we have

$$\begin{aligned} (\forall \boldsymbol{w}) \;\; J(\boldsymbol{w}) \;&\geq\; J(\boldsymbol{w}_t) + (\boldsymbol{w} - \boldsymbol{w}_t)^\top \bar{\boldsymbol{g}}_t \\ &=\; J(\boldsymbol{w}_{t+1}) + (\boldsymbol{w} - \boldsymbol{w}_{t+1})^\top \bar{\boldsymbol{g}}_t + J(\boldsymbol{w}_t) - J(\boldsymbol{w}_{t+1}) + (\boldsymbol{w}_{t+1} - \boldsymbol{w}_t)^\top \bar{\boldsymbol{g}}_t \, . \end{aligned}$$

Using $\boldsymbol{w}_{t+1} - \boldsymbol{w}_t = -\eta_t \boldsymbol{B}_t \bar{\boldsymbol{g}}_t$ and (108) gives

$$\begin{aligned} (\forall \boldsymbol{w}) \;\; J(\boldsymbol{w}) \;&\geq\; J(\boldsymbol{w}_{t+1}) + (\boldsymbol{w} - \boldsymbol{w}_{t+1})^\top \bar{\boldsymbol{g}}_t + \frac{c_1 \eta_t}{2} \, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t - \eta_t \, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t \\ &=\; J(\boldsymbol{w}_{t+1}) + (\boldsymbol{w} - \boldsymbol{w}_{t+1})^\top \bar{\boldsymbol{g}}_t - \varepsilon'_t \, , \end{aligned}$$

where $\varepsilon'_t := \eta_t (1 - \frac{c_1}{2}) \, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t$. Since $\eta_t > 0$, $c_1 < 1$, and $\boldsymbol{B}_t \succeq h > 0$, ε'_t is non-negative. By the definition (105) of ε-subgradient, $\bar{\boldsymbol{g}}_t \in \partial_{\varepsilon'_t} J(\boldsymbol{w}_{t+1})$. ■

21. This means that there exists at least one $\boldsymbol{w} \in \mathbb{R}^d$ such that $J(\boldsymbol{w}) < \infty$, and that for all $\boldsymbol{w} \in \mathbb{R}^d$, $J(\boldsymbol{w}) > -\infty$ and $J(\boldsymbol{w}) \leq \liminf_{t \to \infty} J(\boldsymbol{w}_t)$ for any sequence $\{\boldsymbol{w}_t\}$ converging to $\boldsymbol{w}$. All objective functions considered in this paper fulfill these conditions.

Theorem 14 *Let $J : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ be a proper lower semi-continuous*[21] *extended-valued convex function. Algorithm 1 with a line search that satisfies the sufficient decrease condition* (23) *with $c_1 \in (0,1)$ converges globally to the minimal value of J, provided that:*

1. *the spectrum of its approximation to the inverse Hessian is bounded above and below: $\exists\,(h, H : 0 < h \le H < \infty) : (\forall t)\ h \preceq \boldsymbol{B}_t \preceq H$*
2. *the step size $\eta_t > 0$ satisfies $\sum_{t=0}^{\infty} \eta_t = \infty$, and*
3. *the direction-finding tolerance ε for Algorithm 2 satisfies* (85).

Proof We have already shown in Lemma 13 that subBFGS is a special case of Algorithm 6. Thus if we can show that the technical conditions of Theorem 12 are met, it directly establishes the global convergence of subBFGS.

Recall that for subBFGS $a_t := \frac{c_1 \eta_t h}{2}$, $\varepsilon_t = 0$, $\varepsilon'_t := \eta_t (1 - \frac{c_1}{2})\, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t$, and $\bar{\boldsymbol{g}}_t = \boldsymbol{g}_{\varepsilon'_t}$. Our assumption on η_t implies that $\sum_{t=0}^{\infty} a_t = \frac{c_1 h}{2} \sum_{t=0}^{\infty} \eta_t = \infty$, thus establishing (106). We now show that $\varepsilon'_t \to 0$. Under the third condition of Theorem 14, it follows from the first inequality in (84) in Corollary 4 that

$$\sup_{\boldsymbol{g} \in \partial J(\boldsymbol{w}_t)} \boldsymbol{g}^\top \boldsymbol{p}_t \ \le\ -\tfrac{1}{2}\, \bar{\boldsymbol{g}}_t^\top \boldsymbol{B}_t \bar{\boldsymbol{g}}_t\,, \tag{109}$$

where $\boldsymbol{p}_t = -\boldsymbol{B}_t \bar{\boldsymbol{g}}_t$, $\bar{\boldsymbol{g}}_t \in \partial J(\boldsymbol{w}_t)$ is the search direction returned by Algorithm 2. Together with the sufficient decrease condition (23), (109) implies (108). Now use (108) recursively to obtain

$$J(\boldsymbol{w}_{t+1}) \ \le\ J(\boldsymbol{w}_0) - \frac{c_1}{2} \sum_{i=0}^{t} \eta_i\, \bar{\boldsymbol{g}}_i^\top \boldsymbol{B}_i \bar{\boldsymbol{g}}_i\,.$$

Since J is proper (hence bounded from below), we have

$$\sum_{t=0}^{\infty} \eta_i\, \bar{\boldsymbol{g}}_i^\top \boldsymbol{B}_i \bar{\boldsymbol{g}}_i \ =\ \frac{1}{1 - \frac{c_1}{2}} \sum_{t=0}^{\infty} \varepsilon'_i \ <\ \infty\,. \tag{110}$$

Recall that $\varepsilon'_i \ge 0$. The bounded sum of non-negative terms in (110) implies that the terms in the sum must converge to zero.

Finally, to show (107) we use $\boldsymbol{w}_{t+1} - \boldsymbol{w}_t = -\eta_t \boldsymbol{B}_t \bar{\boldsymbol{g}}_t$, the definition of the matrix norm: $\|\boldsymbol{B}\| := \max_{\boldsymbol{x} \neq 0} \frac{\|\boldsymbol{B}\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$, and the upper bound on the spectrum of $\boldsymbol{B}_t$ to write:

$$\|\boldsymbol{w}_{t+1} - \boldsymbol{w}_t\| \ =\ \eta_t \|\boldsymbol{B}_t \bar{\boldsymbol{g}}_t\| \ \le\ \eta_t \|\boldsymbol{B}_t\| \|\bar{\boldsymbol{g}}_t\| \ \le\ \eta_t H \|\bar{\boldsymbol{g}}_t\|. \tag{111}$$

Recall that $\bar{\boldsymbol{g}}_t = \boldsymbol{g}_{\varepsilon'_t}$ and $a_t = \frac{c_1 \eta_t h}{2}$, and multiply both sides of (111) by $\frac{c_1 h}{2H}$ to obtain (107) with $\beta := \frac{c_1 h}{2H}$. ■

Appendix E. SubBFGS Converges on Various Counterexamples

We demonstrate the global convergence of subBFGS[22] with an exact line search on various counterexamples from the literature, designed to show the failure to converge of other gradient-based algorithms.

22. We run Algorithm 1 with $h = 10^{-8}$ and $\varepsilon = 10^{-5}$.

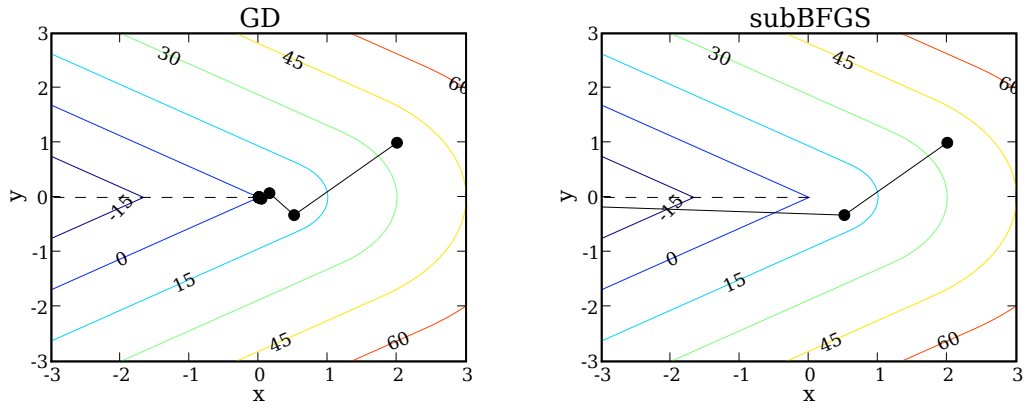


Figure 23: Optimization trajectory of steepest descent (left) and subBFGS (right) on counterexample (112).

E.1 Counterexample for Steepest Descent

The first counterexample (112) is given by Wolfe (1975) to show the non-convergent behaviour of the steepest descent method with an exact line search (denoted GD):

$$f(x,y) := \begin{cases} 5\sqrt{(9x^2+16y^2)} & \text{if} \quad x \geq |y|, \\ 9x+16|y| & \text{otherwise.} \end{cases} \tag{112}$$

This function is subdifferentiable along $x \leq 0,\ y = 0$ (dashed line in Figure 23); its minimal value $(-\infty)$ is attained for $x = -\infty$. As can be seen in Figure 23 (left), starting from a differentiable point $(2,1)$, GD follows successively orthogonal directions, that is, $-\nabla f(x,y)$, and converges to the non-optimal point $(0,0)$. As pointed out by Wolfe (1975), the failure of GD here is due to the fact that GD does not have a global view of f, specifically, it is because the gradient evaluated at each iterate (solid disk) is not informative about $\partial f(0,0)$, which contains subgradients (e.g., $(9,0)$), whose negative directions point toward the minimum. SubBFGS overcomes this "short-sightedness" by incorporating into the parameter update (3) an estimate $\boldsymbol{B}_t$ of the inverse Hessian, whose information about the shape of f prevents subBFGS from zigzagging to a non-optimal point. Figure 23 (right) shows that subBFGS moves to the correct region ($x < 0$) at the second step. In fact, the second step of subBFGS lands exactly on the hinge $x \leq 0, y = 0$, where a subgradient pointing to the optimum is available.

E.2 Counterexample for Steepest Subgradient Descent

The second counterexample (113), due to Hiriart-Urruty and Lemaréchal (1993, Section VIII.2.2), is a piecewise linear function which is subdifferentiable along $0 \leq y = \pm 3x$ and $x = 0$ (dashed lines in Figure 24):

$$f(x,y) := \max\{-100,\ \pm 2x+3y,\ \pm 5x+2y\}. \tag{113}$$

This example shows that steepest subgradient descent with an exact line search (denoted subGD) may not converge to the optimum of a nonsmooth function. Steepest subgradient descent updates

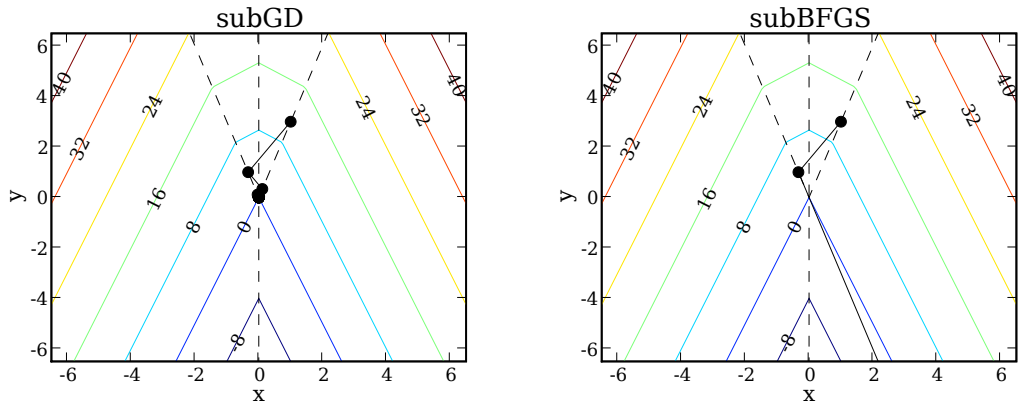


Figure 24: Optimization trajectory of steepest subgradient descent (left) and subBFGS (right) on counterexample (113).

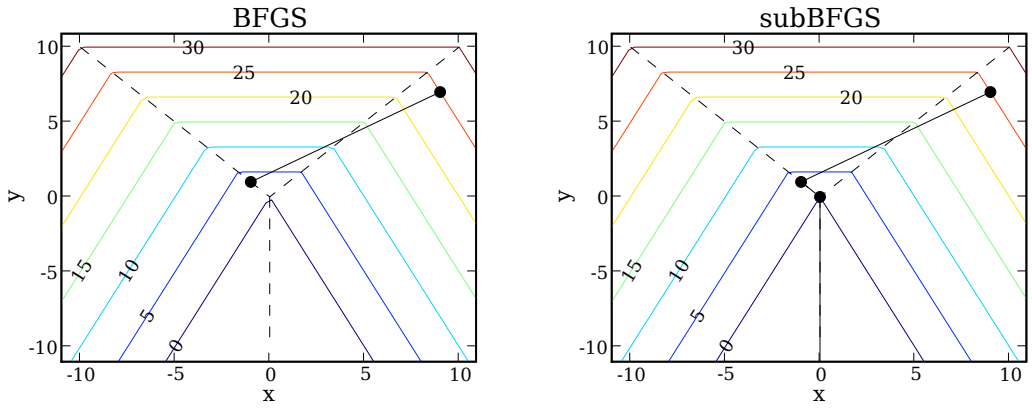


Figure 25: Optimization trajectory of standard BFGS (left) and subBFGS (right) on counterexample (114).

parameters along the *steepest descent* subgradient direction, which is obtained by solving the min-sup problem (13) with respect to the Euclidean norm. Clearly, the minimal value of f (-100) is attained for sufficiently negative values of y. However, subGD oscillates between two hinges $0 \leq y = \pm 3x$, converging to the non-optimal point $(0,0)$, as shown in Figure 24 (left). The zigzagging optimization trajectory of subGD does not allow it to land on any informative position such as the hinge $y = 0$, where the steepest subgradient descent direction points to the desired region $(y < 0)$; Hiriart-Urruty and Lemaréchal (1993, Section VIII.2.2) provide a detailed discussion. By contrast, subBFGS moves to the $y < 0$ region at the second step (Figure 24, right), which ends at the point $(100, -300)$ (not shown in the figure) where the minimal value of f is attained .

E.3 Counterexample for BFGS

The final counterexample (114) is given by Lewis and Overton (2008b) to show that the standard BFGS algorithm with an exact line search can break down when encountering a nonsmooth point:

$$f(x,y) := \max\{2|x| + y,\ 3y\}. \tag{114}$$

This function is subdifferentiable along $x = 0,\ y \leq 0$ and $y = |x|$ (dashed lines in Figure 25). Figure 25 (left) shows that after the first step, BFGS lands on a nonsmooth point, where it fails to find a descent direction. This is not surprising because at a nonsmooth point $\boldsymbol{w}$ the quasi-Newton direction $\boldsymbol{p} := -\boldsymbol{B}\boldsymbol{g}$ for a given subgradient $\boldsymbol{g} \in \partial J(\boldsymbol{w})$ is not necessarily a direction of descent. SubBFGS fixes this problem by using a direction-finding procedure (Algorithm 2), which is guaranteed to generate a descent quasi-Newton direction. Here subBFGS converges to $f = -\infty$ in three iterations (Figure 25, right).

References

N. Abe, J. Takeuchi, and M. K. Warmuth. Polynomial Learnability of Stochastic Rules with Respect to the KL-Divergence and Quadratic Distance. *IEICE Transactions on Information and Systems*, 84(3):299–316, 2001.

P. K. Agarwal and M. Sharir. Davenport-Schinzel sequences and their geometric applications. In J. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 1–47. North-Holland, New York, 2000.

G. Andrew and J. Gao. Scalable training of L_1-regularized log-linear models. In *Proc. Intl. Conf. Machine Learning*, pages 33–40, New York, NY, USA, 2007. ACM.

J. Basch. *Kinetic Data Structures*. PhD thesis, Stanford University, June 1999.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.

J. R. Birge, L. Qi, and Z. Wei. A general approach to convergence properties of some methods for nonsmooth convex optimization. *Applied Mathematics and Optimization*, 38(2):141–158, 1998.

A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with LaRank. In *Proc. Intl. Conf. Machine Learning*, pages 89–96, New York, NY, USA, 2007. ACM.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.

K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991, January 2003a.

K. Crammer and Y. Singer. A family of additive online algorithms for category ranking. *J. Mach. Learn. Res.*, 3:1025–1058, February 2003b.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for support vector machines. In A. McCallum and S. Roweis, editors, *ICML*, pages 320–327. Omnipress, 2008.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for large-scale risk minimization. *Journal of Machine Learning Research*, 10:2157–2192, 2009.

M. Haarala. *Large-Scale Nonsmooth Optimization*. PhD thesis, University of Jyväskylä, 2004.

J. Hershberger. Finding the upper envelope of n line segments in $O(n \log n)$ time. *Information Processing Letters*, 33(4):169–174, December 1989.

J. B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms, I and II*, volume 305 and 306. Springer-Verlag, 1993.

T. Joachims. Training linear SVMs in linear time. In *Proc. ACM Conf. Knowledge Discovery and Data Mining (KDD)*. ACM, 2006.

Y. J. Lee and O. L. Mangasarian. SSVM: A smooth support vector machine for classification. *Computational optimization and Applications*, 20(1):5–22, 2001.

C. Lemarechal. Numerical experiments in nonsmooth optimization. *Progress in Nondifferentiable Optimization*, 82:61–84, 1982.

A. S. Lewis and M. L. Overton. Nonsmooth optimization via BFGS. Technical report, Optimization Online, 2008a. URL `http://www.optimization-online.org/DB_FILE/2008/12/2172.pdf`. Submitted to SIAM J. Optimization.

A. S. Lewis and M. L. Overton. Behavior of BFGS with an exact line search on nonsmooth examples. Technical report, Optimization Online, 2008b. URL `http://www.optimization-online.org/DB_FILE/2008/12/2173.pdf`. Submitted to SIAM J. Optimization.

D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(3):503–528, 1989.

L. Lukšan and J. Vlček. Globally convergent variable metric method for convex nonsmooth unconstrained minimization. *Journal of Optimization Theory and Applications*, 102(3):593–613, 1999.

F. Maes, L. Denoyer, and P. Gallinari. XML structure mapping application to the PASCAL/INEX 2006 XML document mining track. In *Advances in XML Information Retrieval and Evaluation: Fifth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX'06)*, Dagstuhl, Germany, 2007.

A. Nedić and D. P. Bertsekas. Convergence rate of incremental subgradient algorithms. In S. Uryasev and P. M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications*, pages 263–304. Kluwer Academic Publishers, 2000.

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. on Optimization*, 15(1):229–251, 2005. ISSN 1052-6234.

Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

S. Shalev-Shwartz and Y. Singer. On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. In *Proceedings of COLT*, 2008.

A. J. Smola, S. V. N. Vishwanathan, and Q. V. Le. Bundle methods for machine learning. In D. Koller and Y. Singer, editors, *Advances in Neural Information Processing Systems 20*, Cambridge MA, 2007. MIT Press.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 25–32, Cambridge, MA, 2004. MIT Press.

C.-H. Teo, S. V. N. Vishwanthan, A. J. Smola, and Q. V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11:311–365, 2010.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

M. K. Warmuth, K. A. Glocer, and S. V. N. Vishwanathan. Entropy regularized LPBoost. In Y. Freund, Y. Làszlò Györfi, and G. Turàn, editors, *Proc. Intl. Conf. Algorithmic Learning Theory*, number 5254 in Lecture Notes in Artificial Intelligence, pages 256 – 271, Budapest, October 2008. Springer-Verlag.

P. Wolfe. Convergence conditions for ascent methods. *SIAM Review*, 11(2):226–235, 1969.

P. Wolfe. A method of conjugate subgradients for minimizing nondifferentiable functions. *Mathematical Programming Study*, 3:145–173, 1975.

J. Yu, S. V. N. Vishwanathan, S. Günter, and N. N. Schraudolph. A quasi-Newton approach to nonsmooth convex optimization. In A. McCallum and S. Roweis, editors, *ICML*, pages 1216–1223. Omnipress, 2008.

T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, 4:5–31, 2001.

Journal of Machine Learning Research 11 (2010) 1201-1242 Submitted 5/08; Revised 4/09; Published 4/10

Graph Kernels

S.V. N. Vishwanathan VISHY@STAT.PURDUE.EDU
Departments of Statistics and Computer Science
Purdue University
250 N University Street, West Lafayette, IN 47907-2066, USA

Nicol N. Schraudolph JMLR@SCHRAUDOLPH.ORG
adaptive tools AG
Canberra ACT 2602, Australia

Risi Kondor RISI@CALTECH.EDU
Center for the Mathematics of Information
California Institute of Technology
1200 E. California Blvd., MC 305-16, Pasadena, CA 91125, USA

Karsten M. Borgwardt KARSTEN.BORGWARDT@TUEBINGEN.MPG.DE
Interdepartmental Bioinformatics Group
Max Planck Institute for Developmental Biology[*]
Spemannstr. 38, 72076 Tübingen, Germany

Editor: John Lafferty

Abstract

We present a unified framework to study graph kernels, special cases of which include the random walk (Gärtner et al., 2003; Borgwardt et al., 2005) and marginalized (Kashima et al., 2003, 2004; Mahé et al., 2004) graph kernels. Through reduction to a Sylvester equation we improve the time complexity of kernel computation between unlabeled graphs with n vertices from $O(n^6)$ to $O(n^3)$. We find a spectral decomposition approach even more efficient when computing entire kernel matrices. For labeled graphs we develop conjugate gradient and fixed-point methods that take $O(dn^3)$ time per iteration, where d is the size of the label set. By extending the necessary linear algebra to Reproducing Kernel Hilbert Spaces (RKHS) we obtain the same result for d-dimensional edge kernels, and $O(n^4)$ in the infinite-dimensional case; on sparse graphs these algorithms only take $O(n^2)$ time per iteration in all cases. Experiments on graphs from bioinformatics and other application domains show that these techniques can speed up computation of the kernel by an order of magnitude or more. We also show that certain rational kernels (Cortes et al., 2002, 2003, 2004) when specialized to graphs reduce to our random walk graph kernel. Finally, we relate our framework to R-convolution kernels (Haussler, 1999) and provide a kernel that is close to the optimal assignment kernel of Fröhlich et al. (2006) yet provably positive semi-definite.

Keywords: linear algebra in RKHS, Sylvester equations, spectral decomposition, bioinformatics, rational kernels, transducers, semirings, random walks

1. Introduction

Machine learning in domains such as bioinformatics (Sharan and Ideker, 2006), chemoinformatics (Bonchev and Rouvray, 1991), drug discovery (Kubinyi, 2003), web data mining

∗. Also at the Max Planck Institute for Biological Cybernetics.

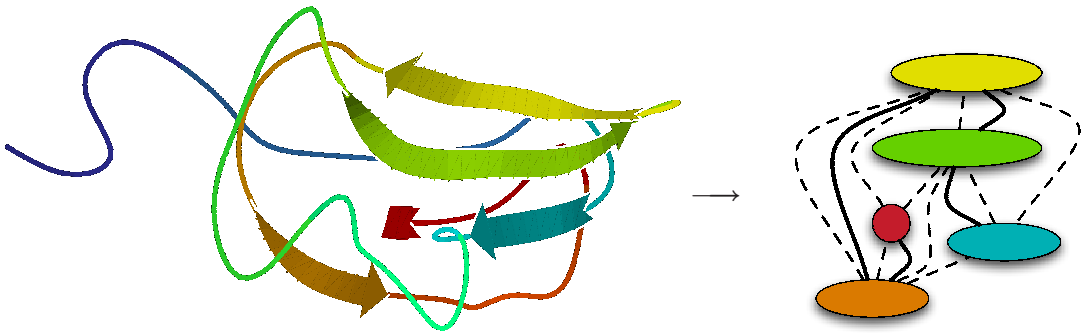


Figure 1: Left: Structure of *E. coli* protein fragment APO-BCCP87 (Yao et al., 1997), ID 1a6x in the Protein Data Bank (Berman et al., 2000). Right: Borgwardt et al.'s (2005) graph representation for this protein fragment. Nodes represent secondary structure elements, and edges encode neighborhood along the amino acid chain (solid) resp. in Euclidean 3D space (dashed).

(Washio and Motoda, 2003), and social networks (Kumar et al., 2006) involves the study of relationships between structured objects. Graphs are natural data structures to model such structures, with nodes representing objects and edges the relations between them. In this context, one often encounters two questions: "How similar are two nodes in a given graph?" and "How similar are two graphs to each other?"

In protein function prediction, for instance, one might want to predict whether a given protein is an enzyme or not. Computational approaches infer protein function by finding proteins with similar sequence, structure, or chemical properties. A very successful recent method is to model the protein as a graph (see Figure 1), and assign similar functions to similar graphs (Borgwardt et al., 2005). In Section 5.2 we compute graph kernels to measure the similarity between proteins and enzymes represented in this fashion.

Another application featured in Section 5.2 involves predicting the toxicity of chemical molecules by comparing their three-dimensional structure. Here the molecular structure is modeled as a graph, and the challenge is to compute the similarity between molecules of known and unknown toxicity.

Finally, consider the task of finding web pages with related content. Since documents on the web link to each other, one can model each web page as the node of a graph, and each link as an edge. Now the problem becomes that of computing similarities between the nodes of a graph. Taking this one step further, detecting mirrored sets of web pages requires computing the similarity between the graphs representing them.

Kernel methods (Schölkopf and Smola, 2002) offer a natural framework to study these questions. Roughly speaking, a kernel $k(x,x')$ is a measure of similarity between objects x and x'. It must satisfy two mathematical requirements: it must be symmetric, that is, $k(x,x') = k(x',x)$, and positive semi-definite (p.s.d.). Comparing nodes in a graph involves constructing a kernel between nodes, while comparing graphs involves constructing a kernel between graphs. In both cases, the challenge is to define a kernel that captures the semantics inherent in the graph structure and is reasonably efficient to evaluate.

The idea of constructing kernels *on* graphs (i.e., between the nodes of a single graph) was first proposed by Kondor and Lafferty (2002), and extended by Smola and Kondor (2003). In con-

trast, in this paper we focus on kernels *between* graphs. The first such kernels were proposed by Gärtner et al. (2003) and later extended by Borgwardt et al. (2005). Much at the same time, the idea of marginalized kernels (Tsuda et al., 2002) was extended to graphs by Kashima et al. (2003, 2004), then further refined by Mahé et al. (2004). Another algebraic approach to graph kernels has appeared recently (Kondor and Borgwardt, 2008). A seemingly independent line of research investigates the so-called rational kernels, which are kernels between finite state automata based on the algebra of abstract semirings (Cortes et al., 2002, 2003, 2004).

The aim of this paper is twofold: on the one hand we present theoretical results showing that all the above graph kernels are in fact closely related, on the other hand we present new algorithms for efficiently computing such kernels. We begin by establishing some notation and reviewing pertinent concepts from linear algebra and graph theory.

1.1 Paper Outline

The first part of this paper (Sections 2–5) elaborates and updates a conference publication of Vishwanathan et al. (2007) to present a unifying framework for graph kernels encompassing many known kernels as special cases, and to discuss connections to yet others. After defining some basic concepts in Section 2, we describe the framework in Section 3, prove that it leads to p.s.d. kernels, and discuss the random walk and marginalized graph kernels as special cases. For ease of exposition we will work with real matrices in the main body of the paper and relegate the RKHS extensions to Appendix A. In Section 4 we present four efficient ways to compute random walk graph kernels, namely: 1. via reduction to a Sylvester equation, 2. with a conjugate gradient solver, 3. using fixed-point iterations, and 4. via spectral decompositions. Experiments on a variety of real and synthetic data sets in Section 5 illustrate the computational advantages of our methods, which generally reduce the time complexity of kernel computation from $O(n^6)$ to $O(n^3)$. The experiments of Section 5.3 were previously presented at a bioinformatics symposium (Borgwardt et al., 2007).

The second part of the paper (Sections 6–7) draws further connections to existing kernels on structured objects. In Section 6 we present a simple proof that rational kernels (Cortes et al., 2002, 2003, 2004) are p.s.d., and show that specializing them to graphs yields random walk graph kernels. In Section 7 we discuss the relation between R-convolution kernels (Haussler, 1999) and various graph kernels, all of which can in fact be shown to be instances of R-convolution kernels. Extending the framework through the use of semirings does not always result in a p.s.d. kernel though; a case in point is the optimal assignment kernel of Fröhlich et al. (2006). We establish sufficient conditions for R-convolution kernels in semirings to be p.s.d., and provide a "mostly optimal assignment kernel" that is provably p.s.d. We conclude in Section 8 with an outlook and discussion.

2. Preliminaries

Here we define the basic concepts and notation from linear algebra and graph theory that will be used in the remainder of the paper.

2.1 Linear Algebra Concepts

We use $\mathbf{e}_i$ to denote the i^{th} standard basis vector (that is, a vector of all zeros with the i^{th} entry set to one), $\mathbf{e}$ to denote a vector with all entries set to one, $\mathbf{0}$ to denote the vector of all zeros, and $\mathbf{I}$ to

denote the identity matrix. When it is clear from the context we will not mention the dimensions of these vectors and matrices.

Definition 1 *Given real matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$, the Kronecker product $A \otimes B \in \mathbb{R}^{np\times mq}$ and column-stacking operator* $\mathrm{vec}(A) \in \mathbb{R}^{nm}$ *are defined as*

$$A \otimes B := \begin{bmatrix} A_{11}B & A_{12}B & \dots & A_{1m}B \\ \vdots & \vdots & \vdots & \vdots \\ A_{n1}B & A_{n2}B & \dots & A_{nm}B \end{bmatrix}, \quad \mathrm{vec}(A) := \begin{bmatrix} A_{*1} \\ \vdots \\ A_{*m} \end{bmatrix},$$

where A_{*j} denotes the j^{th} column of A.

The Kronecker product and vec operator are linked by the well-known property (e.g., Bernstein, 2005, Proposition 7.1.9):

$$\mathrm{vec}(ABC) = (C^\top \otimes A)\,\mathrm{vec}(B). \tag{1}$$

Another well-known property of the Kronecker product which we make use of is (Bernstein, 2005, Proposition 7.1.6):

$$(A \otimes B)(C \otimes D) = AC \otimes BD. \tag{2}$$

Finally, the Hadamard product of two real matrices $A, B \in \mathbb{R}^{n\times m}$, denoted by $A \odot B \in \mathbb{R}^{n\times m}$, is obtained by element-wise multiplication. It interacts with the Kronecker product via

$$(A \otimes B) \odot (C \otimes D) = (A \odot C) \otimes (B \odot D). \tag{3}$$

All the above concepts can be extended to a Reproducing Kernel Hilbert Space (RKHS) (See Appendix A for details).

2.2 Graph Concepts

A graph G consists of an ordered set of n vertices $V = \{v_1, v_2, \dots, v_n\}$, and a set of directed edges $E \subset V \times V$. A vertex v_i is said to be a neighbor of another vertex v_j if they are connected by an edge, that is, if $(v_i, v_j) \in E$; this is also denoted $v_i \sim v_j$. We do not allow self-loops, that is, $(v_i, v_i) \notin E$ for any i. A walk of length k on G is a sequence of indices $i_0, i_1, \dots i_k$ such that $v_{i_{r-1}} \sim v_{i_r}$ for all $1 \le r \le k$. A graph is said to be strongly connected if any two pairs of vertices can be connected by a walk. In this paper we will always work with strongly connected graphs. A graph is said to be undirected if $(v_i, v_j) \in E \iff (v_j, v_i) \in E$.

In much of the following we will be dealing with weighted graphs, which are a slight generalization of the above. In a weighted graph, each edge (v_i, v_j) has an associated weight $w_{ij} > 0$ signifying its "strength". If v_i and v_j are not neighbors, then $w_{ij} = 0$. In an undirected weighted graph $w_{ij} = w_{ji}$.

When G is unweighted, we define its adjacency matrix as the $n \times n$ matrix $\widetilde{\mathrm{A}}$ with $\widetilde{\mathrm{A}}_{ij} = 1$ if $v_j \sim v_i$, and 0 otherwise. For weighted graphs, $\widetilde{\mathrm{A}}_{ij} = w_{ji}$. While some authors would call these matrices the transpose of the adjacency matrix, for our purposes the present definitions will be more convenient. For undirected graphs $\widetilde{\mathrm{A}}$ is symmetric, and the two definitions coincide. The diagonal entries of $\widetilde{\mathrm{A}}$ are always zero.

The adjacency matrix has a normalized cousin, defined $A := \widetilde{\mathrm{A}} D^{-1}$, which has the property that each of its columns sums to one, and it can therefore serve as the transition matrix for a stochastic process. Here, D is a diagonal matrix of node degrees, that is, $D_{ii} = d_i = \sum_j \widetilde{\mathrm{A}}_{ij}$. A random walk on G is a process generating sequences of vertices $v_{i_1}, v_{i_2}, v_{i_3}, \ldots$ according to $\mathbb{P}(i_{k+1}|i_1, \ldots i_k) = A_{i_{k+1},i_k}$, that is, the probability at v_{i_k} of picking $v_{i_{k+1}}$ next is proportional to the weight of the edge $(v_{i_k}, v_{i_{k+1}})$. The t^{th} power of A thus describes t-length walks, that is, $(A^t)_{ij}$ is the probability of a transition from vertex v_j to vertex v_i via a walk of length t. If p_0 is an initial probability distribution over vertices, then the probability distribution p_t describing the location of our random walker at time t is $p_t = A^t p_0$. The j^{th} component of p_t denotes the probability of finishing a t-length walk at vertex v_j.

A random walk need not continue indefinitely; to model this, we associate every node v_{i_k} in the graph with a stopping probability q_{i_k}. Our generalized random walk graph kernels then use the overall probability of stopping after t steps, given by $q^\top p_t$. Like p_0, the vector q of stopping probabilities is a place to embed prior knowledge into the kernel design. Since p_t as a probability distribution sums to one, a *uniform* vector q (as might be chosen in the absence of prior knowledge) would yield the same overall stopping probability for all p_t, thus leading to a kernel that is invariant with respect to the graph structure it is meant to measure. In this case, the unnormalized adjacency matrix $\widetilde{\mathrm{A}}$ (which simply counts random walks instead of measuring their probability) should be used instead.

Let $\mathcal{X}$ be a set of labels which includes the special label ζ. Every edge-labeled graph G is associated with a label matrix $X \in \mathcal{X}^{n \times n}$ in which X_{ij} is the label of the edge (v_j, v_i) and $X_{ij} = \zeta$ if $(v_j, v_i) \notin E$. Let $\mathcal{H}$ be the RKHS induced by a p.s.d. kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, and let $\phi : \mathcal{X} \to \mathcal{H}$ denote the corresponding feature map, which we assume maps ζ to the zero element of $\mathcal{H}$. We use $\Phi(X)$ to denote the feature matrix of G (see Appendix A for details). For ease of exposition we do not consider labels on vertices here, though our results hold for that case as well. Henceforth we use the term labeled graph to denote an edge-labeled graph.

Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic* (denoted by $G \cong G'$) if there exists a bijective mapping $g : V \to V'$ (called the isomorphism function) such that $(v_i, v_j) \in E$ iff $(g(v_i), g(v_j)) \in E'$.

3. Random Walk Graph Kernels

Our generalized random walk graph kernels are based on a simple idea: given a pair of graphs, perform random walks on both, and count the number of matching walks. We show that this simple concept underlies both random walk and marginalized graph kernels. In order to do this, we first need to introduce direct product graphs.

3.1 Direct Product Graphs

Given two graphs $G(V, E)$ and $G'(V', E')$, their direct product $G_\times$ is a graph with vertex set

$$V_\times = \{(v_i, v'_r) : v_i \in V,\ v'_r \in V'\}, \tag{4}$$

and edge set

$$E_\times = \{((v_i, v'_r), (v_j, v'_s)) : (v_i, v_j) \in E \wedge (v'_r, v'_s) \in E'\}. \tag{5}$$

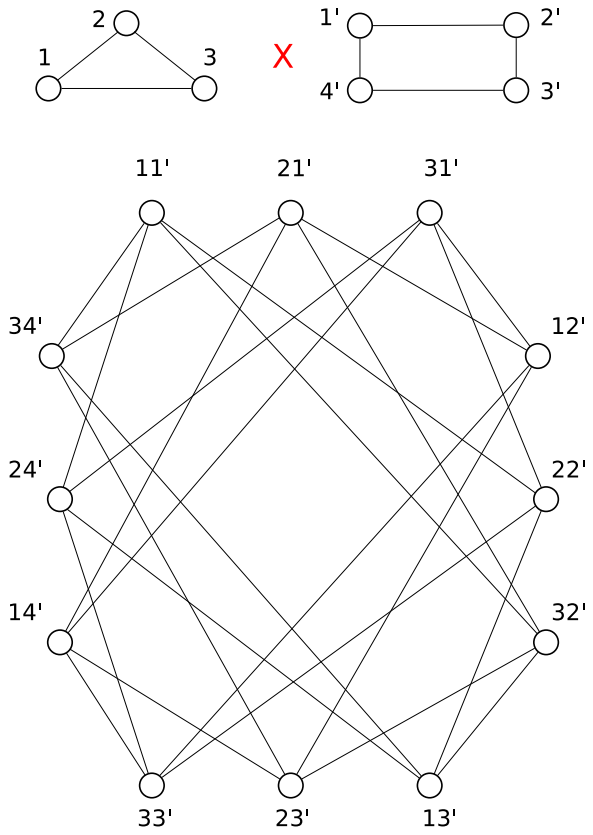


Figure 2: Two graphs (top left & right) and their direct product (bottom). Each node of the direct product graph is labeled with a pair of nodes (4); an edge exists in the direct product if and only if the corresponding nodes are adjacent in both original graphs (5). For instance, nodes $11'$ and $32'$ are adjacent because there is an edge between nodes 1 and 3 in the first, and $1'$ and $2'$ in the second graph.

In other words, $G_\times$ is a graph over pairs of vertices from G and G', and two vertices in $G_\times$ are neighbors if and only if the corresponding vertices in G and G' are both neighbors; see Figure 2 for an illustration. If $\widetilde{\mathrm{A}}$ and $\widetilde{\mathrm{A}}'$ are the respective adjacency matrices of G and G', then the adjacency matrix of $G_\times$ is $\widetilde{\mathrm{A}}_\times = \widetilde{\mathrm{A}} \otimes \widetilde{\mathrm{A}}'$. Similarly, $A_\times = A \otimes A'$.

Performing a random walk on the direct product graph is equivalent to performing a simultaneous random walk on G and G' (Imrich and Klavžar, 2000). If p and p' denote initial probability distributions over the vertices of G and G', then the corresponding initial probability distribution on the direct product graph is $p_\times := p \otimes p'$. Likewise, if q and q' are stopping probabilities (that is, the probability that a random walk ends at a given vertex), then the stopping probability on the direct product graph is $q_\times := q \otimes q'$.

Let $|V| =: n$ and $|V'| =: n'$. If G and G' are edge-labeled, we can associate a weight matrix $W_\times \in \mathbb{R}^{nn' \times nn'}$ with $G_\times$ using our extension of the Kronecker product (Definition 1) into RKHS (Definition 11 in Appendix A):

$$W_\times = \Phi(X) \otimes \Phi(X'). \tag{6}$$

As a consequence of the definition of $\Phi(X)$ and $\Phi(X')$, the entries of $W_\times$ are non-zero only if the corresponding edge exists in the direct product graph. If we simply let $\mathcal{H} = \mathbb{R}$, $\Phi(X) = \widetilde{\mathrm{A}}$, and $\Phi(X') = \widetilde{\mathrm{A}}'$ then (6) reduces to $\widetilde{\mathrm{A}}_\times$, the adjacency matrix of the direct product graph. Normalization can be incorporated by letting $\phi(X_{ij}) = 1/d_i$ if $(v_j, v_i) \in E$, and zero otherwise.[1] Then $\Phi(X) = A$ and $\Phi(X') = A'$, and consequently $W_\times = A_\times$.

If the edges of our graphs take on labels from a finite set, without loss of generality $\{1, 2, \ldots, d\}$, we can let $\mathcal{H}$ be $\mathbb{R}^d$ endowed with the usual inner product. For each edge $(v_j, v_i) \in E$ we set $\phi(X_{ij}) = \mathbf{e}_l / d_i$ if the edge (v_j, v_i) is labeled l; all other entries of $\Phi(X)$ are $\mathbf{0}$. Thus the weight matrix (6) has a non-zero entry iff an edge exists in the direct product graph and the corresponding edges in G and G' have the same label. Let ${}^lA$ denote the normalized adjacency matrix of the graph filtered by the label l, that is, ${}^lA_{ij} = A_{ij}$ if $X_{ij} = l$, and zero otherwise. Some simple algebra (omitted for the sake of brevity) shows that the weight matrix of the direct product graph can then be written as

$$W_\times = \sum_{l=1}^{d} {}^lA \otimes {}^lA'. \tag{7}$$

In Section 4 we will develop efficient methods to compute kernels defined using the weight matrix of the direct product graph. The applicability and time complexity of a particular method will depend on whether the graphs are unlabeled though possibly edge-weighted ($W_\times = A_\times$), have discrete edge labels (7), or—in the most general case—employ an arbitrary edge kernel (6); see Table 1 for a summary.

3.2 Kernel Definition

As stated above, performing a random walk on the direct product graph $G_\times$ is equivalent to performing a simultaneous random walk on the graphs G and G' (Imrich and Klavžar, 2000). Therefore, the $((i-1)n' + r, (j-1)n' + s)^{\text{th}}$ entry of $A_\times^k$ represents the probability of simultaneous length k random walks on G (starting from vertex v_j and ending in vertex v_i) and G' (starting from vertex v'_s and ending in vertex v'_r). The entries of $W_\times$ (6) represent similarity between edges: The $((i-1)n' + r, (j-1)n' + s)$ entry of $W_\times^k$ represents the similarity between simultaneous length k random walks on G and G', measured via the kernel function κ. Given initial and stopping probability distributions $p_\times$ and $q_\times$ one can compute $q_\times^\top W_\times^k p_\times$, which is the expected similarity between simultaneous length k random walks on G and G'.

To define a kernel which computes the similarity between G and G', one natural idea is to simply sum up $q_\times^\top W_\times^k p_\times$ for all values of k. However, this sum might not converge, leaving the kernel value undefined. To overcome this problem, we introduce appropriately chosen non-negative coefficients $\mu(k)$, and define the kernel between G and G' as

$$k(G, G') := \sum_{k=0}^{\infty} \mu(k)\, q_\times^\top W_\times^k p_\times. \tag{8}$$

This definition is very flexible and offers the kernel designer many parameters to adjust in an application-specific manner: Appropriately choosing $\mu(k)$ allows one to (de-)emphasize walks of different lengths; if initial and stopping probabilities are known for a particular application, then

1. The technical problem that now $\phi(X_{ij})$ depends on d_i can be addressed by making d_i a feature of all edges $(v_j, v_i) \in E$.

this knowledge can be incorporated into the kernel; and finally, appropriate kernels or similarity measures between edges can be incorporated via the weight matrix $W_\times$. Despite its flexibility, this kernel is guaranteed to be p.s.d. and—as we will see in Section 4—can be computed efficiently by exploiting the special structure of $W_\times$. To show that (8) is a valid p.s.d. kernel we need the following technical lemma:

Lemma 2 $\forall\, k \in \mathbb{N}:\ W_\times^k p_\times = \mathrm{vec}[\Phi(X')^k p' (\Phi(X)^k p)^\top]$.

Proof By induction over k. Base case: $k = 0$. Using (1) we find

$$W_\times^0 p_\times = p_\times = (p \otimes p')\,\mathrm{vec}(1) = \mathrm{vec}(p'\,1\,p^\top) = \mathrm{vec}[\Phi(X')^0 p' (\Phi(X)^0 p)^\top]. \tag{9}$$

Induction from k to $k+1$: Using the induction assumption $W_\times^k p_\times = \mathrm{vec}[\Phi(X')^k p' (\Phi(X)^k p)^\top]$ and Lemma 12 we obtain

$$\begin{aligned} W_\times^{k+1} p_\times = W_\times W_\times^k p_\times &= (\Phi(X) \otimes \Phi(X'))\,\mathrm{vec}[\Phi(X')^k p' (\Phi(X)^k p)^\top] \\ &= \mathrm{vec}[\Phi(X')\Phi(X')^k p' (\Phi(X)^k p)^\top \Phi(X)^\top] \\ &= \mathrm{vec}[\Phi(X')^{k+1} p' (\Phi(X)^{k+1} p)^\top]. \end{aligned} \tag{10}$$

Base case (9) and induction (10) together imply Lemma 2 $\forall\, k \in \mathbb{N}_0$. ■

Theorem 3 *If the coefficients $\mu(k)$ are such that* (8) *converges, then* (8) *defines a valid p.s.d. kernel.*

Proof Using Lemmas 12 and 2 we can write

$$\begin{aligned} q_\times^\top W_\times^k p_\times &= (q \otimes q')^\top\,\mathrm{vec}[\Phi(X')^k p' (\Phi(X)^k p)^\top] \\ &= \mathrm{vec}[q'^\top \Phi(X')^k p' (\Phi(X)^k p)^\top q] \\ &= \underbrace{(q^\top \Phi(X)^k p)^\top}_{\rho_k(G)^\top} \underbrace{(q'^\top \Phi(X')^k p')}_{\rho_k(G')}. \end{aligned} \tag{11}$$

Each individual term of (11) equals $\rho_k(G)^\top \rho_k(G')$ for some function ρ_k, and is therefore a valid p.s.d. kernel. The theorem follows because the class of p.s.d. kernels is closed under non-negative linear combinations and pointwise limits (Berg et al., 1984). ■

3.3 Special Cases

Kashima et al. (2004) define a kernel between labeled graphs via walks and their label sequences. Recall that a walk of length t on G is a sequence of indices $i_1, i_2, \ldots i_{t+1}$ such that $v_{i_k} \sim v_{i_{k+1}}$ for all $1 \le k \le t$. In our setting (where we do not consider node labels), the label sequence $h = h_1, \ldots, h_t$ associated with a walk is simply the sequence of edge labels encountered during the walk. Let P denote a transition probability matrix, where P_{ij} denotes the probability of transition from node v_i to node v_j. For instance, P might be the normalized adjacency matrix of G. Furthermore, let p

and q denote starting and stopping probabilities. Then one can compute the probability of a walk $i_1, i_2, \ldots i_{t+1}$ and hence the label sequence h associated with it as

$$p(h|G) := q_{i_{t+1}} \prod_{j=1}^{t} P_{i_j,i_{j+1}} p_{i_1}. \tag{12}$$

Now let $\hat{\phi}$ denote a feature map on edge labels, and define a kernel between label sequences of length t by

$$\kappa(h,h') := \prod_{i=1}^{t} \kappa(h_i,h'_i) = \prod_{i=1}^{t} \left\langle \hat{\phi}(h_i), \hat{\phi}(h'_i) \right\rangle \tag{13}$$

if h and h' have the same length t, and zero otherwise. Using (12) and (13) we can define a kernel between graphs via marginalization:

$$k(G,G') := \sum_{h} \sum_{h'} \kappa(h,h')\, p(h|G)\, p(h|G'). \tag{14}$$

Kashima et al. (2004, Eq. 1.19) show that (14) can be written as

$$k(G,G') = q_\times^\top (\mathbf{I} - T_\times)^{-1} p_\times, \tag{15}$$

where $T_\times = [\mathrm{vec}(P)\,\mathrm{vec}(P')^\top] \odot [\hat{\Phi}(X) \otimes \hat{\Phi}(X')]$. (As usual, X and X' denote the edge label matrices of G and G', respectively, and $\hat{\Phi}$ the corresponding feature matrices.)

Although this kernel is differently motivated, it can be obtained as a special case of our framework. Towards this end, assume $\mu(k) = \lambda^k$ for some $\lambda > 0$. We can then write

$$k(G,G') \;=\; \sum_{k=0}^{\infty} \lambda^k q_\times^\top W_\times^k p_\times \;=\; q_\times^\top (\mathbf{I} - \lambda W_\times)^{-1} p_\times. \tag{16}$$

To recover the marginalized graph kernels let $\lambda = 1$, and define $\Phi(X_{ij}) = P_{ij} \hat{\Phi}(X_{ij})$, in which case $W_\times = T_\times$, thus recovering (15).

Given a pair of graphs, Gärtner et al. (2003) also perform random walks on both, but then *count* the number of matching walks. Their kernel is defined as (Gärtner et al., 2003, Definition 6):

$$k(G,G') = \sum_{i=1}^{n} \sum_{j=1}^{n'} \sum_{k=0}^{\infty} \lambda_k \left[\widetilde{\mathrm{A}}_\times^k \right]_{ij}. \tag{17}$$

To obtain (17) in our framework, set $\mu(k) := \lambda_k$, assume uniform distributions for the starting and stopping probabilities over the vertices of G and G' (i.e., $p_i = q_i = 1/n$ and $p'_i = q'_i = 1/n'$), and let $\Phi(X) := \widetilde{\mathrm{A}}$ and $\Phi(X') = \widetilde{\mathrm{A}}'$. Consequently, $p_\times = q_\times = \mathbf{e}/(nn')$, and $W_\times = \widetilde{\mathrm{A}}_\times$, the unnormalized adjacency matrix of the direct product graph. This allows us to rewrite (8) to obtain

$$k(G,G') \;=\; \sum_{k=0}^{\infty} \mu(k)\, q_\times^\top W_\times^k p_\times \;=\; \frac{1}{n^2 n'^2} \sum_{i=1}^{n} \sum_{j=1}^{n'} \sum_{k=0}^{\infty} \lambda_k \left[\widetilde{\mathrm{A}}_\times^k \right]_{ij}, \tag{18}$$

which recovers (17) to within a constant factor. Gärtner et al. (2003) also extend their kernels to graphs with labels from a finite set by replacing $\widetilde{\mathrm{A}}_\times$ in (17) with a sum $\widetilde{\mathrm{W}}_\times$ of label-filtered (but

sparsity	dense				sparse
edge labels	none/scalar	finite set	finite-dim.	∞-dim.	any
Method (Section) $W_\times =$	$A \otimes A'$	(7)	kernel (6)		
Sylvester Equation (4.1)	m^2n^3	unknown	—		—
Conjugate Gradient (4.2)	m^2rn^3	m^2rdn^3		m^2rn^4	m^2rn^2
Fixed-Point Iterations (4.3)	m^2kn^3	m^2kdn^3		m^2kn^4	m^2kn^2
Spectral Decomposition (4.4)	$(m+n)mn^2$	m^2n^6			—
Nearest Kron. Product (4.5)	1	$m^2k'dn^2$	$m^2k'n^4$		$m^2k'n^2$

Table 1: Worst-case time complexity (in $O(\cdot)$ notation) of our methods for an $m \times m$ graph kernel matrix, where $n =$ size of the graphs (number of nodes), $d =$ size of label set *resp*. dimensionality of feature map, $r =$ effective rank of $W_\times$, $k =$ number of fixed-point iterations (31), and $k' =$ number of power iterations (37).

unnormalized) adjacency matrices, analogous to our (7). The reduction to our framework extends to this setting in a straightforward manner.

Gärtner et al. (2003) discuss two cases of special interest: First, their *geometric* kernel employs a fixed decay factor λ to down-weight the contribution of long walks to the kernel, setting $\lambda_k := \lambda^k$ as in our (16). The choice of λ is critical here: It must be small enough for the sum in (17) to converge, depending on the spectrum of $W_\times$ (Vishwanathan, 2002, Chapter 6). Second, their *exponential* kernel is defined as

$$k(G,G') = \sum_{i=1}^{n}\sum_{j=1}^{n'} [e^{\lambda\widetilde{A}_\times}]_{ij} = \mathbf{e}^\top e^{\lambda\widetilde{A}_\times}\mathbf{e}, \tag{19}$$

using the matrix exponential. This is obtained in our framework by setting $\lambda_k := \lambda^k/k!$, so that the right-most sum in (18) becomes the series expansion of $e^{\lambda\widetilde{A}_\times}$.

The kernels of Gärtner et al. (2003) differ from our definition (8) in that they do not explicitly model starting or stopping probabilities, and employ unnormalized adjacency matrices instead of our more general weight matrix (6) which allows for normalization and arbitrary kernels on edges.

4. Efficient Computation

Computing a geometric random walk graph kernel with $\mu(k) = \lambda^k$ amounts to inverting $(\mathbf{I} - \lambda W_\times)$, an $n^2 \times n^2$ matrix if G and G' have n vertices each. Since the complexity of inverting a matrix is essentially cubic in its dimensions, direct computation of (16) would require $O(n^6)$ time. Below we develop methods based on Sylvester equations (Section 4.1), conjugate gradients (Section 4.2), fixed-point iterations (Section 4.3), and spectral decompositions (Section 4.4) that greatly accelerate this computation. Section 4.5 introduces an approximation that can further speed up the kernel computation for labeled graphs.

Table 1 summarizes our results, listing the worst-case time complexity of our methods as a function of graph density and labeling. Exact computation of the full kernel matrix between m dense, unlabeled (but possibly edge-weighted) graphs of n nodes each (leftmost column) is generally quadratic in the number of graphs and cubic in their size; for the iterative methods this must be multiplied by the number of iterations, which is given by the effective rank r of the weight matrix $W_\times$

for conjugate gradient, and by (31) for fixed-point iterations. The spectral decomposition approach (Section 4.4) is exceptional here in that it can be linear in m *resp*. quadratic in n (but not both) if precomputation of the m spectral graph decompositions dominates (*resp*. is dominated by) the actual kernel computations.

The cost of the iterative algorithms increases by another factor of d for graphs with edge labels from a finite set of d symbols or an edge kernel with d-dimensional feature map; for an arbitrary edge kernel (whose feature map may be infinite-dimensional) this factor becomes n. On labeled graphs our spectral decomposition approach offers no savings, and the Sylvester equation method applies only if the labels come from a finite set of symbols, and then with unknown time complexity. A nearest Kronecker product approximation can be used, however, to approximate the direct product of labeled graphs with a weight matrix that can be handled by any of our methods for unlabeled graphs. This approximation requires k' (37) iterations, each costing $O(dn^2)$ time when the labels come from a finite set of d symbols, and $O(n^4)$ in general.

Finally, when the graphs are sparse (i.e., only have $O(n)$ edges each; rightmost column in Table 1) our iterative methods (conjugate gradient, fixed-point, and nearest Kronecker product) take only $O(n^2)$ time per iteration, regardless of how the graphs are labeled. We cannot authoritatively state the time complexity for sparse graphs of solving Sylvester equations or performing spectral decompositions. Spielman and Teng (2008) have shown that graphs can be sparsified (i.e., approximated by sparse graphs) in nearly linear time, although the constants involved are quite large.

4.1 Sylvester Equation Methods

Consider the following equation, commonly known as the Sylvester or Lyapunov equation:

$$M = SMT + M_0. \tag{20}$$

Here, $S, T, M_0 \in \mathbb{R}^{n\times n}$ are given and we need to solve for $M \in \mathbb{R}^{n\times n}$. These equations can be readily solved in $O(n^3)$ time with freely available code (Gardiner et al., 1992), such as Matlab's `dlyap` method. Solving the generalized Sylvester equation

$$M = \sum_{i=1}^{d} S_i M T_i + M_0 \tag{21}$$

involves computing generalized simultaneous Schur factorizations of d symmetric matrices (Lathauwer et al., 2004). Although technically involved, (21) can also be solved efficiently, albeit at a higher computational cost. The computational complexity of this generalized factorization is at present unknown.

We now show that for graphs with discrete edge labels, whose weight matrix $W_\times$ can be written as (7), the problem of computing the graph kernel (16) can be reduced to solving the following generalized Sylvester equation:

$$M = \sum_{i=1}^{d} \lambda {}^iA' M {}^iA^\top + M_0, \tag{22}$$

where $\mathrm{vec}(M_0) = p_\times$. We begin by *flattening* (22):

$$\mathrm{vec}(M) = \lambda \sum_{i=1}^{d} \mathrm{vec}({}^iA' M {}^iA^\top) + p_\times. \tag{23}$$

Using Lemma 12 (which extends (1) into an RKHS) we can rewrite (23) as

$$(\mathbf{I} - \lambda \sum_{i=1}^{d} {}^{i}A \otimes {}^{i}A') \operatorname{vec}(M) = p_\times, \tag{24}$$

use (7), and solve (24) for $\operatorname{vec}(M)$:

$$\operatorname{vec}(M) = (\mathbf{I} - \lambda W_\times)^{-1} p_\times. \tag{25}$$

Multiplying both sides of (25) by $q_\times^\top$ yields

$$q_\times^\top \operatorname{vec}(M) = q_\times^\top (\mathbf{I} - \lambda W_\times)^{-1} p_\times. \tag{26}$$

The right-hand side of (26) is the graph kernel (16). Given the solution M of the Sylvester equation (22), the graph kernel can be obtained as $q_\times^\top \operatorname{vec}(M)$ in $O(n^2)$ time. The same argument applies for unlabeled graphs by simply setting $d = 1$, which turns (22) into a simple Sylvester equation. Since solving that only takes $O(n^3)$ time, computing the random walk graph kernel in this fashion is much faster than the $O(n^6)$ time required by the direct approach.

One drawback of this strategy is that Sylvester equation solvers are quite sophisticated and typically available only as black-box library routines, which limits their applicability. Matlab's `dlyap` solver, for instance, does not exploit sparsity, and only handles the cases $d = 1$ and $d = 2$. A solver for the simple Sylvester equation (20) can still be used to efficiently compute kernels between labeled graphs though by employing the nearest Kronecker product approximation (Section 4.5).

4.2 Conjugate Gradient Methods

Given a matrix M and a vector b, conjugate gradient (CG) methods solve the system of equations $Mx = b$ efficiently (Nocedal and Wright, 1999). While they are designed for symmetric p.s.d. matrices, CG solvers can also be used to solve other linear systems efficiently. They are particularly efficient if the matrix is rank deficient, or has a small *effective rank*, that is, number of distinct eigenvalues. Furthermore, if computing matrix-vector products is cheap—because M is sparse, for instance—the CG solver can be sped up significantly (Nocedal and Wright, 1999). Specifically, if computing Mv for an arbitrary vector v requires $O(m)$ time, and the effective rank of M is r, then a CG solver takes $O(r)$ iterations, and hence only $O(rm)$ time, to solve $Mx = b$.

The graph kernel (16) can be computed by a two-step procedure: First we solve the linear system

$$(\mathbf{I} - \lambda W_\times) x = p_\times, \tag{27}$$

for x, then we compute $q_\times^\top x$. We now focus on efficient ways to solve (27) with a CG solver. Recall that if G and G' contain n vertices each then $W_\times$ is an $n^2 \times n^2$ matrix. Naively, multiplying W by some vector y inside the CG algorithm requires $O(n^4)$ operations. However, by our extension of the vec-ABC formula (1) into RKHS (Lemma 12), introducing the matrix $Y \in \mathbb{R}^{n \times n}$ with $y = \operatorname{vec}(Y)$, and recalling that $W_\times = \Phi(X) \otimes \Phi(X')$, by Lemma 12 we can write

$$W_\times y = (\Phi(X) \otimes \Phi(X')) \operatorname{vec}(Y) = \operatorname{vec}(\Phi(X') Y \Phi(X)^\top). \tag{28}$$

If $\phi(\cdot) \in \mathbb{R}^d$ then the above matrix-vector product can be computed in $O(dn^3)$ time. If $\Phi(X)$ and $\Phi(X')$ are sparse, then $\Phi(X') Y \Phi(X)^\top$ can be computed yet more efficiently: If there are $O(n)$ non-ζ entries in $\Phi(X)$ and $\Phi(X')$, then computing (28) takes only $O(n^2)$ time.

4.3 Fixed-Point Iterations

Fixed-point methods begin by rewriting (27) as

$$x = p_\times + \lambda W_\times x. \tag{29}$$

Now, solving for x is equivalent to finding a fixed point of (29) taken as an iteration (Nocedal and Wright, 1999). Letting x_t denote the value of x at iteration t, we set $x_0 := p_\times$, then compute

$$x_{t+1} = p_\times + \lambda W_\times x_t \tag{30}$$

repeatedly until $\|x_{t+1} - x_t\| < \varepsilon$, where $\|\cdot\|$ denotes the Euclidean norm and ε some pre-defined tolerance. This is guaranteed to converge if all eigenvalues of $\lambda W_\times$ lie inside the unit disk; this can be ensured by setting $\lambda < |\xi_1|^{-1}$, where ξ_1 is the largest-magnitude eigenvalue of $W_\times$. Assuming that each iteration of (30) contracts x to the fixpoint by a factor of $\lambda\xi_1$, we converge to within ε of the fixpoint in k iterations, where

$$k = O\left(\frac{\ln \varepsilon}{\ln \lambda + \ln |\xi_1|}\right). \tag{31}$$

The above is closely related to the power method used to compute the largest eigenvalue of a matrix (Golub and Van Loan, 1996); efficient preconditioners can also be used to speed up convergence (Golub and Van Loan, 1996). Since each iteration of (30) involves computation of the matrix-vector product $W_\times x_t$, all speed-ups for computing the matrix-vector product discussed in Section 4.2 are applicable here. In particular, we exploit the fact that $W_\times$ is a sum of Kronecker products to reduce the worst-case time complexity to $O(dn^3)$ per iteration in our experiments, in contrast to Kashima et al. (2004) who computed the matrix-vector product explicitly.

4.4 Spectral Decomposition Method

In the previous two sections we have introduced methods that are efficient for both unlabeled and labeled graphs, but specifically computed the geometric kernel (16), that is, assumed that $\mu(k) = \lambda^k$. We now turn to a method based on spectral decompositions that can compute the general random walk kernel (8) for any convergent choice of $\mu(k)$, but is only efficient for unlabeled graphs. (In fact, it will turn out to be our *most* efficient method for computing an entire kernel matrix between unlabeled graphs.)

Let $W_\times = P_\times D_\times P_\times^{-1}$ denote the spectral decomposition of $W_\times$, that is, the columns of $P_\times$ are its eigenvectors, and $D_\times$ is a diagonal matrix of corresponding eigenvalues. The random walk graph kernel (8) can then be written as

$$k(G,G') := \sum_{k=0}^{\infty} \mu(k)\, q_\times^\top (P_\times D_\times P_\times^{-1})^k p_\times = q_\times^\top P_\times \left(\sum_{k=0}^{\infty} \mu(k) D_\times^k\right) P_\times^{-1} p_\times. \tag{32}$$

This simplifies matters in that (32) only takes weighted powers of a diagonal matrix, which decouple into scalar powers of its entries. An implementable graph kernel can then be obtained by employing a power series that is known to converge to a given nonlinear function. The geometric kernel (16), for instance, uses the fact that $\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$; setting $\mu(k) := \lambda^k$ in (32) we thus obtain

$$k(G,G') := q_\times^\top P_\times (\mathbf{I} - \lambda D_\times)^{-1} P_\times^{-1} p_\times. \tag{33}$$

The crucial difference to (16) is that the inverse in (33) is of a diagonal matrix, hence trivial to compute: just take the reciprocal of each entry. To give another example, setting $\mu(k) := \lambda^k/k!$ in (32) yields the exponential kernel (19) by way of spectral decomposition:

$$k(G,G') := q_\times^\top P_\times e^{\lambda D_\times} P_\times^{-1} p_\times, \tag{34}$$

because $e^x = \sum_{k=0}^\infty x^k/k!$. Again, unlike in (19) the matrix exponential here is trivial to compute because $\lambda D_\times$ is diagonal: simply exponentiate each entry.

Thus by diagonalizing the nonlinearity central to a random walk graph kernel, spectral decomposition can greatly expedite its computation. As described above, however, it is computationally unattractive: Computing the spectral decomposition of a dense matrix takes time cubic in its size (Golub and Van Loan, 1996); since $W_\times$ is an $n^2 \times n^2$ matrix this would result in $O(n^6)$ time complexity per kernel computation.[2] By leveraging the properties of the Kronecker product, however, we can obtain a far better result for unlabeled (though possibly edge-weighted) graphs:

Theorem 4 *The kernel matrix for any random walk kernel* (8) *between m unlabeled, possibly edge-weighted graphs with n nodes can be computed in $O((mp+n)mn^2)$ time via spectral decompositions, where computing the corresponding scalar power series takes $O(p)$ time.*

Proof Because the graphs are unlabeled, we have $W_\times := A_\times = A_i \otimes A_j$, where A_i and A_j ($i, j \in \{1,2,\ldots,m\}$) are the adjacency matrices (normalized or not) of individual graphs. Begin by precomputing the spectral decomposition of each graph: $(\forall i)\, A_i = P_i D_i P_i^{-1}$. Using Propositions 7.1.6, 7.1.7 of Bernstein (2005) we have

$$A_i \otimes A_j \;=\; (P_i D_i P_i^{-1}) \otimes (P_j D_j P_j^{-1}) \;=\; (P_i \otimes P_j)(D_i \otimes D_j)(P_i \otimes P_j)^{-1}. \tag{35}$$

Proposition 7.1.10 of Bernstein (2005) tells us that in fact $D_i \otimes D_j = D_\times$, which implies that also $P_i \otimes P_j = P_\times$ and that indeed the spectral decomposition of a Kronecker product decomposes into those of its constituents, as seen in (35). We can therefore use Propositions 7.1.6, 7.1.7 of Bernstein (2005) again to rewrite (32) as

$$k(G_i, G_j) \;=\; (q_i^\top P_i \otimes q_j^\top P_j) \left(\sum_{k=0}^\infty \mu(k) (D_i \otimes D_j)^k \right) (P_i^{-1} p_i \otimes P_j^{-1} p_j). \tag{36}$$

Computing the central power series here takes $O(n^2 p)$ time just as in (32), but the cost of calculating the two flanking factors has been reduced from $O(n^4)$ to $O(n^2)$ in (36). The entire $m \times m$ kernel matrix can thus be obtained in $O(m^2 n^2 p)$ time, plus the $O(mn^3)$ time it takes to precompute spectral decompositions of the m individual adjacency matrices. ■

Note that in practice we will always pick a power series with known limit that is trivial to evaluate (i.e., $p = 1$), as exemplified by the geometric (33) and exponential (34) kernels. Theorem 4 then gives us a very efficient method to compute entire kernel matrices, albeit only between unlabeled graphs. (It is tempting to try to extend the spectral approach for the exponential kernel to labeled graphs, but this runs into a key technical difficulty: a sum of (label-filtered adjacency) matrices in the exponent cannot be separated unless those matrices commute, that is, generally $e^{A+B} \neq e^A e^B$ unless $AB = BA$.)

2. Thus Gärtner et al. (2003) give a time complexity cubic in the $O(n^2)$ size of the product graph.

4.5 Nearest Kronecker Product Approximation

As we have seen above, some of our fast methods for computing random walk graph kernels may become computationally expensive, or not even be available, for labeled graphs, in particular when the number d of distinct labels is large or a general edge kernel is employed. In such cases we can find the *nearest Kronecker product* to $W_\times$, that is, compute matrices S and T such that $W_\times \approx S \otimes T$, then use any of our methods on $S \otimes T$ as if it were the adjacency matrix of a direct product of *unlabeled* graphs.

Finding the nearest Kronecker product approximating a matrix such as $W_\times$ is a well-studied problem in numerical linear algebra, and efficient algorithms which can exploit the sparsity of $W_\times$ are available (Pitsianis, 1992; Van Loan, 2000). Formally, these methods minimize the Frobenius norm $\|W_\times - S \otimes T\|_F$ by computing the largest singular value of $\hat{W}_\times$, a permuted version of $W_\times$. We employ the power method[3] for this purpose, each iteration of which entails computing the matrix-vector product $\hat{W}_\times \mathrm{vec}(T')$, where $T' \in \mathbb{R}^{n \times n}$ is the current approximation of T. The result of the matrix-vector product is then reshaped into an $n \times n$ matrix to form T' for the next iteration (Pitsianis, 1992). It is easy to see that computing $\hat{W}_\times \mathrm{vec}(T')$ requires $O(n^4)$ time.

If $W_\times$ can be written as a sum of d Kronecker products (7), then so can $\hat{W}_\times$ (Pitsianis, 1992; Van Loan, 2000), and the cost per iteration hence drops to $O(dn^2)$. Furthermore, if the two graphs are sparse with $O(n)$ edges each, then $W_\times$ will have $O(n^2)$ non-zero entries, and each iteration only takes $O(n^2)$ time. The number k' of iterations required is

$$k' = O\left(\frac{\ln n}{\ln|\xi_1| - \ln|\xi_2|}\right), \tag{37}$$

where ξ_1 and ξ_2 are the eigenvalues of $W_\times$ with largest *resp*. second-largest magnitude.

As described above, the nearest Kronecker product approximation is calculated separately for each entry of an $m \times m$ kernel matrix. This causes two problems: First, the spectral decomposition method will now take $O(m^2 n^3)$ time, as it is no longer possible to precompute the m graph spectra. Second, like the optimal assignment kernel (Section 7.2) the resulting kernel matrix may have negative eigenvalues, and hence fail to be p.s.d. In future work, it may be possible to address these shortcomings by computing a *simultaneous* nearest Kronecker product approximation for the entire kernel matrix. For now, we verified empirically on the MUTAG and PTC data sets (cf. Section 5.2) that the most negative eigenvalue is relatively small: its magnitude was 4.4% *resp*. 0.1% of ξ_2. We also found the nearest Kronecker product to provide a better approximation than simply ignoring the graph labels: the angle between $\mathrm{vec}(W_\times)$ and its unlabeled variant was 2.2 *resp*. 4.7 times greater than that between $\mathrm{vec}(W_\times)$ and $\mathrm{vec}(S \otimes T)$.

5. Experiments

Numerous studies have applied random walk graph kernels to problems such as protein function prediction (Borgwardt et al., 2005) and chemoinformatics (Kashima et al., 2004). In our experiments we therefore focus on the runtime of computing the kernels, rather than their utility in any given application. We present three sets of experiments: First, we study the scaling behaviour of our algorithms on unlabeled random graphs. Second, we assess the practical impact of our algorithmic improvement on four real-world data sets whose size mandates fast kernel computation. Third, we

3. Lanczos iterations are typically faster but more difficult to handle numerically.

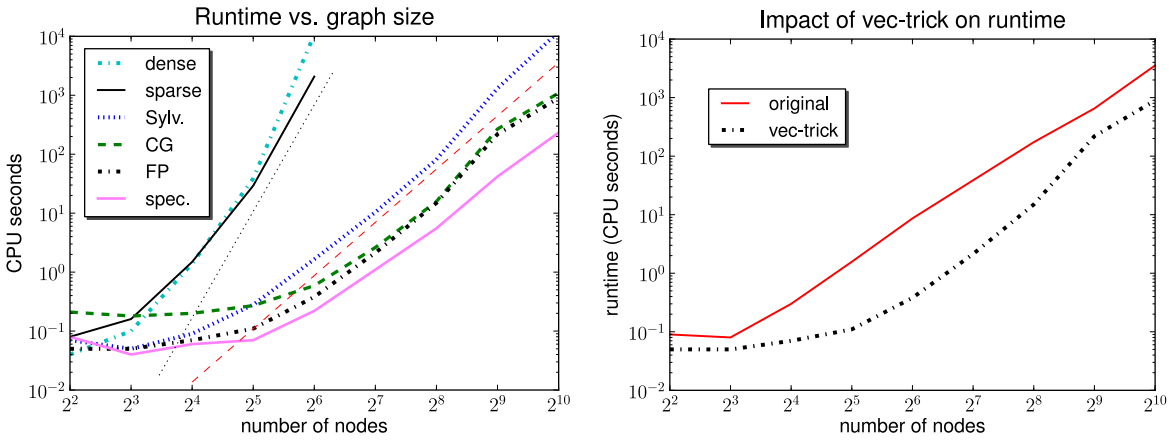


Figure 3: Time to compute a 10×10 kernel matrix on random graphs with n nodes and $3n$ edges as a function of the graph size n. Left: The Sylvester equation (Sylv.), conjugate gradient (CG), fixed-point iteration (FP), and spectral decomposition (spec.) approaches, compared to the dense and sparse direct method. Thin straight lines indicate $O(n^6)$ (black dots) *resp*. $O(n^3)$ (red dashes) scaling. Right: Kashima et al.'s (2004) fixed-point iteration (original) compared to our version, which exploits Lemma 12 (vec-trick).

devise novel methods for protein-protein interaction (PPI) network comparison using graph kernels. The algorithmic challenge here is to efficiently compute kernels on large sparse graphs.

The baseline for comparison in all our experiments is the direct approach of Gärtner et al. (2003), implemented via a sparse LU factorization; this already runs orders of magnitude faster on our data sets than a dense (i.e., non-sparse) implementation. Our code was written in Matlab Release 2008a, and all experiments were run under Mac OS X 10.5.5 on an Apple Mac Pro with a 3.0 GHz Intel 8-Core processor and 16 GB of main memory. We employed Lemma 12 to speed up matrix-vector multiplication for both CG and fixed-point methods (cf. Section 4.2), and used the function `dlyap` from Matlab's control toolbox to solve the Sylvester equation. By default, we used a value of $\lambda = 10^{-4}$, and set the convergence tolerance for both CG solver and fixed-point iteration to 10^{-6}. For the real-world data sets, the value of λ was chosen to ensure that the random walk graph kernel converges. Since our methods are exact and produce the same kernel values (to numerical precision), we only report the CPU time of each algorithm.

5.1 Unlabeled Random Graphs

The aim here is to study the scaling behaviour of our algorithms as a function of graph size and sparsity. We generated several sets of unlabeled random graphs. For the first set we began with an empty graph of $n = 2^k$ nodes, where $k = 2, 3, \ldots, 10$, randomly added $3n$ edges, then checked the graph's connectivity. For each k we repeated this process until we had collected 10 strongly connected random graphs.

The time required to compute the 10×10 kernel matrix between these graphs for each value of n is shown in Figure 3 (left). We see that the direct approach scales asymptotically as $O(n^6)$ in both the dense and the sparse implementation. For a graph of 64 nodes the direct approach already takes over half an hour (sparse) *resp*. 3 hours (dense) of CPU time. Our Sylvester equation (Sylv.),

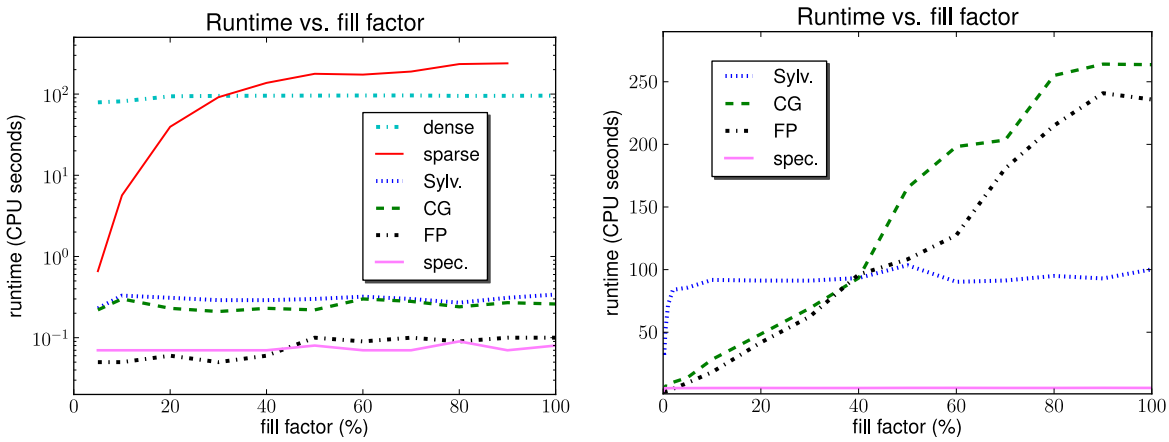


Figure 4: Time to compute a 10×10 kernel matrix on random graphs as a function of their fill factor. Left: The dense and sparse direct method on 32-node graphs, compared to our Sylvester equation (Sylv.), conjugate gradient (CG), fixed point iteration (FP), and spectral decomposition (spec.) approaches. Right: Our approaches on larger graphs with 256 nodes, where the direct method is infeasible.

conjugate gradient (CG) and fixed-point iteration (FP) methods, by contrast, all scale as $O(n^3)$, and can thus be applied to far larger graphs. Our spectral decomposition approach (spec.) is the fastest method here; it too scales as $O(n^3)$ as n asymptotically dominates over the fixed kernel matrix size $m = 10$.

We also examined the impact of Lemma 12 on enhancing the runtime performance of the fixed-point iteration approach as originally proposed by Kashima et al. (2004). For this experiment, we again computed the 10×10 kernel matrix on the above random graphs, once using the original fixed-point iteration, and once using fixed-point iteration enhanced by Lemma 12. As Figure 3 (right) shows, our approach consistently outperforms the original version, sometimes by over an order of magnitude.

For the next set of experiments we fixed the graph size at 32 nodes (the largest size that the direct method could handle comfortably), and randomly added edges until the fill factor (i.e., the number of non-zero entries in the adjacency matrix) reached $x\%$, where $x = 5, 10, 20, 30, \ldots, 100$. For each x, we generated 10 such graphs and computed the 10×10 kernel matrix between them. Figure 4 (left) shows that as expected, the sparse direct method is faster than its dense counterpart for small fill factors but slower for larger ones. Both however are consistently outperformed by our four methods, which are up to three orders of magnitude faster, with fixed-point iterations (FP) and spectral decompositions (spec.) the most efficient.

To better understand how our algorithms take advantage of sparsity, we generated a set of larger random graphs (with 256 nodes) by the same procedure as before, but with a geometric progression of fill factors: $x = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100$. The direct methods are infeasible here. The CPU times taken by our algorithms to compute a 10×10 kernel matrix is shown in Figure 4 (right). Both conjugate gradient and fixed point iteration methods have runtimes roughly proportional to the fill factor. The runtime of the Sylvester equation solver, by contrast, is largely independent of the fill factor because our black-box `dlyap` solver does not aggressively exploit sparsity in the adja-

cency matrices. The same holds for our spectral decomposition approach, which however exhibits impressive performance here: although it does not exploit sparsity at all, it is the fastest method by far on all but the sparsest ($\leq 2\%$ filled) graphs. Clearly, well-designed *sparse* implementations of Sylvester equation solvers, and in particular spectral decomposition, could facilitate substantial further gains in efficiency here.

5.2 Real-World Data Sets

Our next set of experiments used four real-world data sets: Two sets of molecular compounds (MUTAG and PTC), and two data sets describing protein tertiary structure (Protein and Enzyme). Graph kernels provide useful measures of similarity for all of these.

5.2.1 THE DATA SETS

We now briefly describe each data set, and discuss how graph kernels are applicable.

Chemical Molecules. Toxicity of chemical molecules can be predicted to some degree by comparing their three-dimensional structure. We employed graph kernels to measure similarity between molecules from the MUTAG and PTC data sets (Toivonen et al., 2003). The average number of nodes per graph in these data sets is 17.72 *resp*. 26.70; the average number of edges is 38.76 *resp*. 52.06.

Protein Graphs. A standard approach to protein function prediction involves classifying proteins into enzymes and non-enzymes, then further assigning enzymes to one of the six top-level classes of the Enzyme Commission (EC) hierarchy. Towards this end, Borgwardt et al. (2005) modeled a data set of 1128 proteins as graphs in which vertices represent secondary structure elements, and edges represent neighborhood within the 3-D structure or along the amino acid chain, as illustrated in Figure 1.

Comparing these graphs via a modified random walk graph kernel and classifying them with a Support Vector Machine (SVM) led to function prediction accuracies competitive with state-of-the-art approaches (Borgwardt et al., 2005). We used Borgwardt et al.'s (2005) data to test the efficacy of our methods on a large data set. The average number of nodes and edges per graph in this data is 38.57 *resp*. 143.75. We used a single label on the edges, and the delta kernel to define similarity between edges.

Enzyme Graphs. We repeated the above experiment on an enzyme graph data set, also due to Borgwardt et al. (2005). This data set contains 600 graphs, with 32.63 nodes and 124.27 edges on average. Graphs in this data set represent enzymes from the BRENDA enzyme database (Schomburg et al., 2004). The biological challenge on this data is to correctly assign the enzymes to one of the EC top-level classes.

5.2.2 UNLABELED GRAPHS

For this experiment, we computed kernels taking into account only the topology of the graph, that is, we did not consider node or edge labels. Table 2 lists the CPU time required to compute the full kernel matrix for each data set, as well as—for comparison purposes—a 100×100 submatrix. The latter is also shown graphically in Figure 5 (left).

data set	MUTAG		PTC		Enzyme		Protein	
nodes/graph	17.7		26.7		32.6		38.6	
edges/node	2.2		1.9		3.8		3.7	
#graphs	100	230	100	417	100	600	100	1128
Sparse	31”	1’45”	45”	7’23”	1’52”	1h21’	23’23”	2.1d*
Sylvester	10”	54”	28”	7’33”	31”	23’28”	5’25”	11h29’
Conj. Grad.	23”	1’29”	26”	4’29”	14”	10’00”	45”	39’39”
Fixed-Point	8”	43”	15”	2’38”	5”	5’44”	43”	22’09”
Spectral	5”	27”	7”	1’54”	7”	4’32”	27”	23’52”

*extrapolated number of days; run did not finish in time available.

Table 2: Time to compute kernel matrix for unlabeled graphs from various data sets.

On these unlabeled graphs, conjugate gradient, fixed-point iterations, and spectral decompositions—sped up via Lemma 12—are consistently faster than the sparse direct method. The Sylvester equation approach is very competitive on smaller graphs (outperforming CG on MUTAG) but slows down with increasing number of nodes per graph. Even so, it still outperforms the sparse direct method. Overall, spectral decomposition is the most efficient approach, followed by fixed-point iterations.

5.2.3 LABELED GRAPHS

For this experiment, we compared graphs with edge labels. Note that node labels can be dealt with by concatenating them to the edge labels of adjacent edges. On the two protein data sets we employed a linear kernel to measure similarity between edge weights representing distances (in Ångströms) between secondary structure elements; since $d = 1$ we can use all our methods for unlabeled graphs here. On the two chemical data sets we used a delta kernel to compare edge labels reflecting types of bonds in molecules; for the Sylvester equation and spectral decomposition approaches we then employed the nearest Kronecker product approximation. We report CPU times for the full kernel matrix as well as a 100×100 submatrix in Table 3; the latter is also shown graphically in Figure 5 (right).

On labeled graphs, the conjugate gradient and the fixed-point iteration always outperform the sparse direct approach, more so on the larger graphs and with the linear kernel. As expected, spectral decompositions are inefficient in combination with the nearest Kronecker product approximation,

kernel	delta, $d=7$		delta, $d=22$		linear, $d=1$			
data set	MUTAG		PTC		Enzyme		Protein	
#graphs	100	230	100	417	100	600	100	1128
Sparse	42”	2’44”	1’07”	14’22”	1’25”	57’43”	12’38”	1.1d*
Sylvester	1’08”	6’05”	1’06”	18’20”	2’13”	76’43”	19’20”	11h19’
Conj. Grad.	39”	3’16”	53”	14’19”	20”	13’20”	41”	57’35”
Fixed-Point	25”	2’17”	37”	7’55”	10”	6’46”	25”	31’09”
Spectral	1’20”	7’08”	1’40”	26’54”	8”	4’22”	26”	21’23”

*extrapolated number of days; run did not finish in time available.

Table 3: Time to compute kernel matrix for labeled graphs from various data sets.

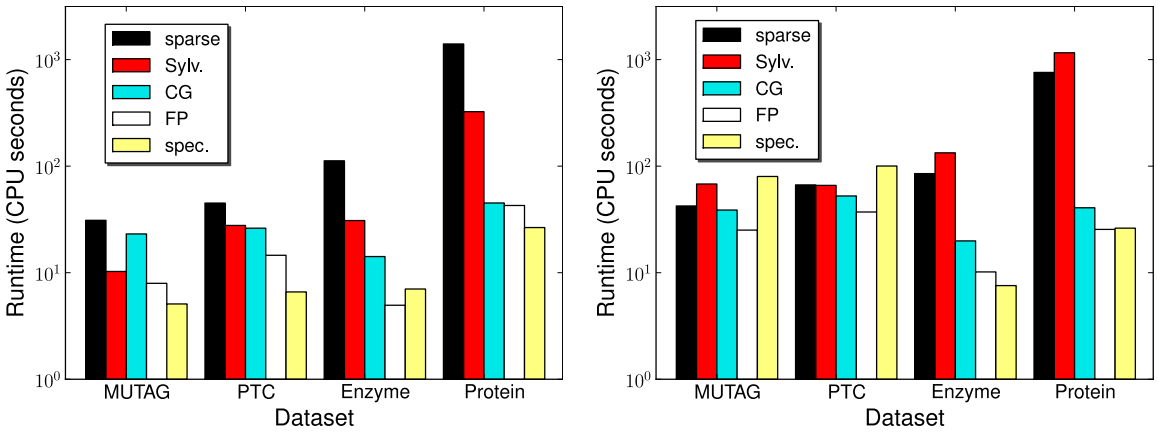


Figure 5: Time (in seconds on a log-scale) to compute 100×100 kernel matrix for unlabeled (left) *resp*. labeled (right) graphs from several data sets, comparing the conventional sparse method to our fast Sylvester equation, conjugate gradient (CG), fixed-point iteration (FP), and spectral approaches.

but with the linear kernel they perform as well as fixed-point iterations for $m = 100$, and better yet on the large kernel matrices. The Sylvester equation approach (at least with the Sylvester solver we used) cannot take advantage of sparsity, but still manages to perform almost as well as the sparse direct method.

5.3 Protein-Protein Interaction Networks

In our third experiment, we used random walk graph kernels to tackle a large-scale problem in bioinformatics involving the comparison of fairly large protein-protein interaction (PPI) networks. Using a combination of human PPI and clinical microarray gene expression data, the task is to predict the disease outcome (dead or alive, relapse or no relapse) of cancer patients. As before, we set $\lambda = 0.001$ and the convergence tolerance to 10^{-6} for all our experiments reported below.

5.3.1 CO-INTEGRATION OF GENE EXPRESSION AND PPI DATA

We co-integrated clinical microarray gene expression data for cancer patients with known human PPI from Rual et al. (2005). Specifically, a patient's gene expression profile was transformed into a graph as follows: A node was created for every protein which—according to Rual et al. (2005)—participates in an interaction, and whose corresponding gene expression level was measured on this patient's microarray. We connect two proteins in this graph by an edge if Rual et al. (2005) list these proteins as interacting, and both genes are up- *resp*. downregulated with respect to a reference measurement. Each node bears the name of the corresponding protein as its label.

This approach of co-integrating PPI and gene expression data is built on the assumption that genes with similar gene expression levels are translated into proteins that are more likely to interact. Recent studies confirm that this assumption holds significantly more often for co-expressed than for random pairs of proteins (Fraser et al., 2004; Bhardwaj and Lu, 2005). To measure similarity between these networks in a biologically meaningful manner, we compare which groups of proteins interact and are co-regulated in each patient. For this purpose, a random walk graph kernel is the

data set	Leukemia		Breast Cancer	
kernel	vanilla	composite	vanilla	composite
Sparse	24”	52”	39”	1’19”
Conj. Grad.	6”	13”	12”	26”
Fixed-Point	4”	7”	7”	13”

Table 4: Average time to compute kernel matrix on protein-protein interaction networks.

natural choice, as a random walk in this graph represents a group of proteins in which consecutive proteins along the walk are co-expressed and interact. As each node bears the name of its corresponding protein as its node label, the size of the product graph is at most that of the smaller of the two input graphs.

5.3.2 COMPOSITE GRAPH KERNEL

The presence of an edge in a graph signifies an interaction between the corresponding nodes. In chemoinformatics, for instance, edges indicate chemical bonds between two atoms; in PPI networks, edges indicate interactions between proteins. When studying protein interactions in disease, however, the *absence* of a given interaction can be as significant as its presence. Since existing graph kernels cannot take this into account, we propose to modify them appropriately. Key to our approach is the notion of a complement graph:

Definition 5 *Let $G = (V, E)$ be a graph with vertex set V and edge set E. Its complement $\bar{G} = (V, \bar{E})$ is a graph over the same vertices but with complementary edges $\bar{E} := (V \times V) \setminus E$.*

In other words, the complement graph consists of exactly those edges *not* present in the original graph. Using this notion we define the *composite* graph kernel

$$k_{comp}(G, G') := k(G, G') + k(\bar{G}, \bar{G}'). \tag{38}$$

This deceptively simple kernel leads to substantial gains in performance in our experiments comparing co-integrated gene expression/protein-protein interaction networks.

5.3.3 DATA SETS

Leukemia. Bullinger et al. (2004) provide a data set of microarrays of 119 leukemia patients. Since 50 patients survived after a median follow-up time of 334 days, always predicting a lethal outcome here would result in a baseline prediction accuracy of 1 - 50/119 = 58.0%. Co-integrating this data with human PPI, we found 2,167 proteins from Rual et al. (2005) for which Bullinger et al. (2004) report expression levels among the 26,260 genes they examined.
Breast Cancer. This data set consists of microarrays of 78 breast cancer patients, of which 44 had shown no relapse of metastases within 5 years after initial treatment (van’t Veer et al., 2002). Always predicting survival thus gives a baseline prediction accuracy of 44/78 = 56.4% on this data. When generating co-integrated graphs, we found 2,429 proteins from Rual et al. (2005) for which van’t Veer et al. (2002) measure gene expression out of the 24,479 genes they studied.

5.3.4 RESULTS

In Table 4 we contrast the CPU runtimes of our conjugate gradient and fixed-point approaches to graph kernel computation on the cancer patients modeled as labeled graphs with that of the direct

sparse method. On both data sets, our fast graph kernel computation methods yield an impressive gain in speed.

Using either the "vanilla" graph kernel (16) or our composite graph kernel (38), we predict the survivors by means of a support vector machine (SVM) in 10-fold cross-validation. The vanilla random walk graph kernel offers slightly higher prediction accuracy than the baseline classifier on one task (Leukemia: 59.2 % vs 58.0 %), and gives identical results on the other (Breast Cancer: both 56.4 %). Our composite graph kernel attains 5 percentage points above baseline in both experiments (Leukemia: 63.3 %; Breast cancer: 61.5 %).

The vanilla kernel suffers from its inability to measure network discrepancies, the paucity of the graph model employed, and the fact that only a small minority of genes could be mapped to interacting proteins; due to these problems, its accuracy remains close to the baseline. The composite kernel, by contrast, also models missing interactions. With it, even our simple graph model, which only considers 10% of the genes examined in both studies, is able to capture some relevant biological information, which in turn leads to better classification accuracy on these challenging data sets (Warnat et al., 2005).

6. Rational Kernels

Rational kernels (Cortes et al., 2004) were conceived to compute similarity between variable-length sequences and, more generally, weighted automata. For instance, the output of a large-vocabulary speech recognizer for a particular input speech utterance is typically a weighted automaton compactly representing a large set of alternative sequences. The weights assigned by the system to each sequence are used to rank different alternatives according to the models the system is based on. It is therefore natural to compare two weighted automata by defining a kernel.

As discussed in Section 3, random walk graph kernels have a very different basis: They compute the similarity between two random graphs by matching random walks. Here the graph itself is the object to be compared, and we want to find a semantically meaningful kernel. Contrast this with a weighted automaton, whose graph is merely a compact representation of the set of variable-length sequences which we wish to compare. Despite these differences we find rational kernels and random walk graph kernels to be closely related.

To understand the connection recall that every random walk on a labeled graph produces a sequence of edge labels encountered during the walk. Viewing the set of all label sequences generated by random walks on a graph as a language, one can design a weighted transducer which accepts this language, with the weight assigned to each label sequence being the probability of a random walk generating this sequence. (This transducer can be represented by a graph whose adjacency matrix is the normalized weight matrix of the original graph.)

In this section we formalize this observation and thus establish connections between rational kernels on transducers (Cortes et al., 2004) and random walk graph kernels. In particular, we show that composition of transducers is analogous to computing product graphs, and that rational kernels on weighted transducers may be viewed as generalizations of random walk graph kernels to weighted automata. In order to make these connections explicit we adopt notation commonly used for describing *algebraic path problems*, wherein disparate problems related to graphs, automata, and transducers are described in a common framework using matrices and tensors (Eilenberg, 1974; Lehmann, 1977; Berstel, 1979; Kuich and Salomaa, 1986).

6.1 Semirings

At the most general level, weighted transducers are defined over semirings. In a semiring addition and multiplication are generalized to abstract operations $\bar{\oplus}$ and $\bar{\odot}$ with the same distributive properties:

Definition 6 (Mohri, 2002) *A semiring is a system* $(\mathbb{K}, \bar{\oplus}, \bar{\odot}, \bar{0}, \bar{1})$ *such that*

1. $(\mathbb{K}, \bar{\oplus}, \bar{0})$ *is a commutative monoid in which* $\bar{0} \in \mathbb{K}$ *is the identity element for* $\bar{\oplus}$ *(i.e., for any* $x, y, z \in \mathbb{K}$*, we have* $x \bar{\oplus} y \in \mathbb{K}$*,* $(x \bar{\oplus} y) \bar{\oplus} z = x \bar{\oplus} (y \bar{\oplus} z)$*,* $x \bar{\oplus} \bar{0} = \bar{0} \bar{\oplus} x = x$ *and* $x \bar{\oplus} y = y \bar{\oplus} x$*);*

2. $(\mathbb{K}, \bar{\odot}, \bar{1})$ *is a monoid in which* $\bar{1}$ *is the identity operator for* $\bar{\odot}$ *(i.e., for any* $x, y, z \in \mathbb{K}$*, we have* $x \bar{\odot} y \in \mathbb{K}$*,* $(x \bar{\odot} y) \bar{\odot} z = x \bar{\odot} (y \bar{\odot} z)$*, and* $x \bar{\odot} \bar{1} = \bar{1} \bar{\odot} x = x$*);*

3. $\bar{\odot}$ *distributes over* $\bar{\oplus}$*, that is, for any* $x, y, z \in \mathbb{K}$*,*

$$\begin{aligned} (x \bar{\oplus} y) \bar{\odot} z &= (x \bar{\odot} z) \bar{\oplus} (y \bar{\odot} z) \\ \textit{and}\ \ z \bar{\odot} (x \bar{\oplus} y) &= (z \bar{\odot} x) \bar{\oplus} (z \bar{\odot} y); \end{aligned}$$

4. $\bar{0}$ *is an annihilator for* $\bar{\odot}$*:* $\forall x \in \mathbb{K}$*,* $x \bar{\odot} \bar{0} = \bar{0} \bar{\odot} x = \bar{0}$.

Thus, a semiring is a ring that may lack negation. $(\mathbb{R}, +, \cdot, 0, 1)$ is the familiar semiring of real numbers. Other examples include

Boolean: $(\{\text{FALSE}, \text{TRUE}\}, \vee, \wedge, \text{FALSE}, \text{TRUE})$;

Logarithmic: $(\mathbb{R} \cup \{-\infty\}, \bar{\oplus}_{\ln}, +, -\infty, 0)$, where $\forall x, y \in \mathbb{K}: \ x \bar{\oplus}_{\ln} y := \ln(e^x + e^y)$;

Tropical: $(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$.

Linear algebra operations such as matrix addition and multiplication as well as Kronecker products can be carried over to a semiring in a straightforward manner. For instance, for $M, M' \in \mathbb{K}^{n \times n}$ we have

$$[M \bar{\odot} M']_{i,j} = \bigoplus_{k=1}^{n} M_{ik} \bar{\odot} M'_{kj}. \tag{39}$$

The $(\bar{\oplus}, \bar{\odot})$ operations in some semirings can be mapped into ordinary $(+, \cdot)$ operations by applying an appropriate *morphism*:

Definition 7 *Let* $(\mathbb{K}, \bar{\oplus}, \bar{\odot}, \bar{0}, \bar{1})$ *be a semiring. A function* $\psi : \mathbb{K} \to \mathbb{R}$ *is a morphism if*

$$\begin{aligned} \psi(x \bar{\oplus} y) &= \psi(x) + \psi(y); \\ \psi(x \bar{\odot} y) &= \psi(x) \cdot \psi(y); \\ \psi(\bar{0}) = 0 \ \ &\textit{and} \ \ \psi(\bar{1}) = 1. \end{aligned}$$

In the following, by 'morphism' we will always mean a morphism from a semiring to the real numbers. Not all semirings have such morphisms: For instance, the logarithmic semiring has a morphism—namely, the exponential function—but the tropical semiring does not have one. If the semiring has a morphism ψ, applying it to the matrix product (39), for instance, yields

$$\begin{aligned}\psi([M \bar{\odot} M']_{i,j}) &= \psi\left(\overline{\bigoplus_{k=1}^{n}} M_{ik} \bar{\odot} M'_{kj}\right) \\ &= \sum_{k=1}^{n} \psi(M_{ik} \bar{\odot} M'_{kj}) = \sum_{k=1}^{n} \psi(M_{ik}) \cdot \psi(M'_{kj}). \end{aligned} \tag{40}$$

As in Appendix A, we can extend the morphism ψ to matrices (and analogously to vectors) by defining $[\Psi(M)]_{ij} := \psi(M_{ij})$. We can then write (40) concisely as

$$\Psi(M \bar{\odot} M') = \Psi(M)\Psi(M'). \tag{41}$$

6.2 Weighted Transducers

Loosely speaking, a transducer is a weighted automaton with an input and an output alphabet. We will work with the following slightly specialized definition:[4]

Definition 8 *A weighted finite-state transducer T over a semiring $(\mathbb{K}, \bar{\oplus}, \bar{\odot}, \bar{0}, \bar{1})$ is a 5-tuple $T = (\Sigma, Q, H, p, q)$, where Σ is a finite input-output alphabet, Q is a finite set of n states, $p \in \mathbb{K}^n$ is a vector of initial weights, $q \in \mathbb{K}^n$ is a vector of final weights, and H is a four-dimensional tensor in $\mathbb{K}^{n \times |\Sigma| \times |\Sigma| \times n}$ which encodes transitions and their corresponding weights.*

For $a, b \in \Sigma$ we will use the shorthand H_{ab} to denote the $n \times n$ slice H_{*ab*} of the transition tensor, which represents all valid transitions on input symbol a emitting the output symbol b. The output weight assigned by T to a pair of strings $\alpha = a_1 a_2 \ldots a_l$ and $\beta = b_1 b_2 \ldots b_l$ is

$$[\![T]\!](\alpha, \beta) = q^\top \bar{\odot} H_{a_1 b_1} \bar{\odot} H_{a_2 b_2} \bar{\odot} \ldots \bar{\odot} H_{a_l b_l} \bar{\odot} p. \tag{42}$$

A transducer is said to accept a pair of strings (α, β) if it assigns non-zero output weight to them, that is, $[\![T]\!](\alpha, \beta) \neq \bar{0}$. A transducer is said to be regulated if the output weight it assigns to any pair of strings is well-defined in $\mathbb{K}$. Since we disallow ε transitions, our transducers are always regulated.

The inverse of $T = (\Sigma, Q, H, p, q)$, denoted by T^{-1}, is obtained by transposing the input and output labels of each transition. Formally, $T^{-1} = (\Sigma, Q, H^\top, p, q)$ where $H^\top_{ab} := H_{ba}$. The composition of two transducers $T = (\Sigma, Q, H, p, q)$ and $T' = (\Sigma, Q', H', p', q')$ is a transducer $T_\times = T \circ T' = (\Sigma, Q_\times, H_\times, p_\times, q_\times)$, where $Q_\times = Q \times Q'$, $p_\times = p \bar{\otimes} p'$,[5] $q_\times := q \bar{\otimes} q'$, and $(H_\times)_{ab} = \bar{\bigoplus}_{c \in \Sigma} H_{ac} \bar{\otimes} H'_{cb}$. It can be shown that

$$[\![T_\times]\!](\alpha, \beta) = [\![T \circ T']\!](\alpha, \beta) = \overline{\bigoplus_{\gamma}} [\![T]\!](\alpha, \gamma) \bar{\odot} [\![T']\!](\gamma, \beta). \tag{43}$$

4. We disallow ε transitions, and use the same alphabet for both input and output. Furthermore, in a departure from tradition, we represent the transition function as a four-dimensional tensor.
5. We use $\bar{\otimes}$ to denote the Kronecker product using the semiring operation $\bar{\odot}$, in order to distinguish it from the regular Kronecker product $\otimes$.

Composing T with its inverse yields $T \circ T^{-1} = (\Sigma, Q \times Q, H^*, p \bar{\otimes} p, q \bar{\otimes} q)$, where $H^*_{ab} = \bar{\bigoplus}_{c \in \Sigma} H_{ac} \bar{\otimes} H_{bc}$. There exists a general and efficient algorithm for composing transducers as in (43) which takes advantage of the sparseness of the input transducers (Mohri et al., 1996; Pereira and Riley, 1997).

6.3 Weighted Automata

A weighted automaton is a transducer with identical input and output symbols. The transition matrix of a weighted automaton is therefore a three-dimensional tensor in $\mathbb{K}^{n \times |\Sigma| \times n}$. As before, we will use the shorthand H_a to denote the $n \times n$ slice H_{*a*} of the transition tensor, which represents all valid transitions on the input symbol a emitting output symbolx a. If Σ contains d symbols, then by specializing (42) it is easy to see that a weighted automaton accepts a string $\alpha = a_1 a_2 \ldots a_l$ with weight

$$[\![T]\!](\alpha) = q^\top \bar{\odot} H_{a_1} \bar{\odot} H_{a_2} \bar{\odot} \ldots \bar{\odot} H_{a_l} \bar{\odot} p. \tag{44}$$

The composition of two weighted automata $T = (\Sigma, Q, H, p, q)$ and $T' = (\Sigma, Q', H', p', q')$ is an automaton $T_\times = T \circ T' = (\Sigma, Q_\times, H_\times, p_\times, q_\times)$, where $Q_\times = Q \times Q'$, $p_\times = p \bar{\otimes} p'$, $q_\times := q \bar{\otimes} q'$, and $(H_\times)_a = H_a \bar{\otimes} H'_a$. The composition operation is also defined for a weighted automaton W and a transducer T:

$$[\![W \circ T]\!](\alpha, \beta) = [\![W]\!](\alpha) \bar{\odot} [\![T]\!](\alpha, \beta). \tag{45}$$

Every random walk on a labeled graph results in a sequence of edge labels encountered during the walk. The set of all label sequences generated by random walks on a given graph is a language. One can construct a weighted automaton which accepts this language as follows: Use the standard semiring $(\mathbb{R}, +, \cdot, 0, 1)$, let the alphabet Σ consist of the labels $\{1, \ldots, d\}$ of the graph, and identify the nodes of the graph with the states of the weighted automaton. Let the starting and stopping probabilities p and q on the graph equal those of the weighted automaton, and complete the construction by identifying for each $l \in \Sigma$ the label-filtered adjacency matrix ${}^l\!A$ of the graph with H_l, the transition tensor of the weighted automaton for that symbol.

Under the above mapping (44) has a natural interpretation: The weight assigned by the automaton to a string of symbols is the probability of encountering the corresponding labels while performing a random walk on the corresponding labeled graph. The composition of weighted automata, when specialized to labeled graphs, is equivalent to computing a direct product graph.

An unlabeled graph corresponds to a weighted automaton whose input-output alphabet contains exactly one symbol, and which therefore only accepts strings of the form $a^k = aa \ldots a$. The transition matrix of such a graph (equivalently, its adjacency matrix) is a 2-dimensional tensor in $\mathbb{K}^{n \times n}$. If A denotes the adjacency matrix of a graph G, then the output weight assigned by G to a^k is $[\![G]\!](a^k) = q^\top AA \ldots Ap = q^\top A^k p$.

6.4 The Rational Kernel for Strings

Given a weighted transducer T and a function $\psi : \mathbb{K} \to \mathbb{R}$, the rational kernel between two strings $\alpha = a_1 a_2 \ldots a_l$ and $\beta = b_1 b_2 \ldots b_l$ is defined as (Cortes et al., 2004):

$$k(\alpha, \beta) := \psi\left([\![T]\!](\alpha, \beta)\right). \tag{46}$$

Cortes et al. (2004) show that a generic way to obtain p.s.d. rational kernels is to replace T in (46) by $T \circ T^{-1}$, and let ψ be a semiring morphism. We now present an alternate proof which uses properties of the Kronecker product. Since ψ is a semiring morphism, by specializing (42) to $T \circ T^{-1}$, we can write $k(\alpha,\beta) = \psi\left([\![T \circ T^{-1}]\!](\alpha,\beta)\right)$ as

$$\Psi(q \bar{\otimes} q)^\top \Psi\left(\overline{\bigoplus_{c_1}} H_{a_1 c_1} \bar{\otimes} H_{b_1 c_1}\right) \ldots \Psi\left(\overline{\bigoplus_{c_l}} H_{a_l c_l} \bar{\otimes} H_{b_l c_l}\right) \Psi(p \bar{\otimes} p). \tag{47}$$

Rules analogous to (41) give us

$$\Psi\left(\overline{\bigoplus_{c \in \Sigma}} H_{ac} \bar{\otimes} H_{bc}\right) = \sum_{c \in \Sigma} \Psi(H_{ac}) \otimes \Psi(H_{bc}). \tag{48}$$

Using (48) we can rewrite (47) as

$$\sum_{c_1 c_2 \ldots c_l} \Psi(q)^\top \otimes \Psi(q)^\top \left(\Psi(H_{a_1 c_1}) \otimes \Psi(H_{b_1 c_1})\right) \ldots \left(\Psi(H_{a_l c_l}) \otimes \Psi(H_{b_l c_l})\right) \Psi(p) \otimes \Psi(p). \tag{49}$$

Finally, successively applying (2) to (49) yields

$$k(\alpha,\beta) = \sum_{c_1 c_2 \ldots c_l} \underbrace{\left(\Psi(q)^\top \Psi(H_{a_1 c_1}) \ldots \Psi(H_{a_l c_l}) \Psi(p)\right)}_{\rho(\alpha)} \underbrace{\left(\Psi(q)^\top \Psi(H_{b_1 c_1}) \ldots \Psi(H_{b_l c_l}) \Psi(p)\right)}_{\rho(\beta)}, \tag{50}$$

Each term of (50) equals $\rho(\alpha)\rho(\beta)$ for some scalar function ρ, and is therefore a valid p.s.d. kernel. Since p.s.d. kernels are closed under addition and pointwise limits (Berg et al., 1984), $k(\alpha,\beta)$ is a valid p.s.d. kernel.

6.5 The Rational Kernel for Weighted Automata

Rational kernels on strings can be naturally extended to weighted automata S and U via (Cortes et al., 2004):

$$\begin{aligned} k(S,U) &= \psi\left(\overline{\bigoplus_{\alpha,\beta}} [\![S]\!](\alpha) \bar{\odot} [\![T]\!](\alpha,\beta) \bar{\odot} [\![U]\!](\beta)\right) \\ &= \psi\left(\overline{\bigoplus_{\alpha,\beta}} [\![S \circ T \circ U]\!](\alpha,\beta)\right), \end{aligned} \tag{51}$$

where we obtained (51) by using (45) twice. If ψ is a semiring morphism, then we can use Definition 7 to rewrite (51) as

$$k(S,U) = \sum_{\alpha,\beta} \psi\left([\![S \circ T \circ U]\!](\alpha,\beta)\right). \tag{52}$$

Since p.s.d. kernels are closed under addition and pointwise limits, if $\psi\left([\![S \circ T \circ U]\!](\alpha,\beta)\right)$ is a p.s.d. kernel for any given α and β, then so is (52).

6.6 Recovering Random Walk Graph Kernels

In order to recover random walk graph kernels we use the standard $(\mathbb{R},+,\cdot,0,1)$ ring as our semiring, and hence set ψ to be the identity function. Next we set the transducer T to simply transform any input string of length k into an identical output string with weight $\mu(k) \geq 0$. With these restrictions (52) can be written as

$$k(S,U) = \sum_{\alpha} \mu(|\alpha|)[\![S \circ U]\!](\alpha), \tag{53}$$

where $|\alpha|$ denotes the length of α. Let us rearrange (53) to

$$k(S,U) = \sum_{k} \mu(k) \left(\sum_{a_1,a_2,\ldots,a_k} [\![S \circ U]\!](a_1,a_2,\ldots,a_k) \right). \tag{54}$$

Specializing the definition of $\circ$ to weighted automata, and letting H_a (resp. H'_a) denote the transition tensor of S (resp. U), we can rewrite (54) as

$$\begin{aligned}
k(S,U) &= \sum_{k} \mu(k) \left(\sum_{a_1,a_2,\ldots,a_k \in \Sigma^k} (q \otimes q')^\top (H_{a_1} \otimes H'_{a_1}) \ldots (H_{a_k} \otimes H'_{a_k})(p \otimes p') \right) \\
&= \sum_{k} \mu(k) (q \otimes q')^\top \left(\sum_{a_1,a_2,\ldots,a_k \in \Sigma^k} (H_{a_1} \otimes H'_{a_1}) \ldots (H_{a_k} \otimes H'_{a_k}) \right) (p \otimes p') \\
&= \sum_{k} \mu(k) (q \otimes q')^\top \left(\sum_{a \in \Sigma} H_a \otimes H'_a \right) \ldots \left(\sum_{a \in \Sigma} H_a \otimes H'_a \right) (p \otimes p') \\
&= \sum_{k} \mu(k) (q \otimes q')^\top \left(\sum_{a} H_a \otimes H'_a \right)^k (p \otimes p').
\end{aligned} \tag{55}$$

Next, we identify H_a (resp. H'_a) with the label-filtered adjacency matrix ${}^aA$ (resp. ${}^aA'$) of a graph G (resp. G') with discrete edge labels. It easy to see that $H_\times := \sum_a H_a \otimes H'_a$ is the weight matrix (7) of the direct product of G and G'. Letting $p_\times = p \otimes p'$ and $q_\times = q \otimes q'$, (55) reduces to

$$k(G,G') = \sum_{k} \mu(k)\, q_\times^\top H_\times^k\, p_\times, \tag{56}$$

which recovers the random walk graph kernel (8) with $W_\times = H_\times$.

The generality of the rational kernel comes at a computational cost: Even when restricted as in (53), it requires the composition $S \circ U$ of two transducers, which takes up to $O((|Q_S|+|E_S|)(|Q_U|+|E_U|))$ time, where $|Q|$ is the number of states and $|E|$ the number of transitions (Cortes et al., 2004, Section 4.1). In our setting $|Q| = n$, the number of nodes in the graph, and $|E|$ is the number of its edges, which can be of $O(n^2)$; the worst-case time complexity of the composition operation is therefore $O(n^4)$. Cortes et al. (2004) showed that the sum in (53) can be computed via a single-source shortest distance algorithm over a semiring. If $S \circ U$ is acyclic, then this is linear in the size of $S \circ U$, which in the worst case is $O(n^2)$. In general, however, an all-pairs shortest-distance algorithm must be employed, such as the generalization of the Floyd-Warshall algorithm due to Lehmann (1977). This algorithm is cubic in the size of $S \circ U$, thus leading to an $O(n^6)$ worst-case time complexity. Since computing $S \circ U$ is the same as computing $W_\times$, and the Lehmann

(1977) algorithm is a way to compute $(I - W_\times)^{-1}$ (the *transitive closure* of $W_\times$), our linear algebra techniques to speed up computation of random walk graph kernels can also be applied to rational kernels.[6] The key insight here is that we never explicitly construct the composition (i.e., direct product graph) in order to compute the kernel.

For ease of exposition we derived (56) by setting T to be an identity transducer. Instead, one can use a weighted transducer which allows for more flexible matching between strings in the alphabet. Basically, the transducer now plays the role of the kernel function κ, and this in turn leads to a more flexible similarity matrix $W_\times$.

There is one important difference between graph kernels and rational kernels. Graph kernels can handle arbitrary edge kernels, including continuous edge labels via the weight matrix $W_\times$. In contrast, rational kernels, which were designed to work with strings and automata, assume that the alphabet (set of labels) is finite. As we saw above, they can incorporate flexible similarity matrices $W_\times$ in this setting, but cannot handle continuous edge labels. Furthermore, to date rational kernels have not been extended to deal with labels mapped to an RKHS.

7. R-convolution Kernels

Haussler's (1999) R-convolution kernels provide a generic way to construct kernels for discrete compound objects. Let $x \in \mathcal{X}$ be such an object, and $\vec{x} := (x_1, x_2, \ldots, x_D)$ denote a decomposition of x, with each $x_i \in \mathcal{X}_i$. We can define a boolean predicate

$$R : \mathcal{X} \times \mathcal{X} \to \{\text{TRUE}, \text{FALSE}\}, \tag{57}$$

where $\mathcal{X} := \mathcal{X}_1 \times \ldots \times \mathcal{X}_D$ and $R(x, \vec{x})$ is TRUE whenever $\vec{x}$ is a valid decomposition of x. Now consider the inverse of (57), the set of all valid decompositions of an object:

$$R^{-1}(x) := \{\vec{x} | R(x, \vec{x}) = \text{TRUE}\}. \tag{58}$$

Like Haussler (1999) we assume that (58) is countable. We define the R-convolution $\star$ of the kernels $\kappa_1, \kappa_2, \ldots, \kappa_D$ with $\kappa_i : \mathcal{X}_i \times \mathcal{X}_i \to \mathbb{R}$ to be

$$k(x, x') = \kappa_1 \star \kappa_2 \star \ldots \star \kappa_D(x, x') := \sum_{\substack{\vec{x} \in R^{-1}(x) \\ \vec{x}' \in R^{-1}(x')}} \mu(\vec{x}, \vec{x}') \prod_{i=1}^{D} \kappa_i(x_i, x_i'), \tag{59}$$

where μ denotes a set of non-negative coefficients on $\mathcal{X} \times \mathcal{X}$, which ensures that the sum in (59) converges.[7] Haussler (1999) showed that $k(x, x')$ is p.s.d. and hence admissible as a kernel (Schölkopf and Smola, 2002), provided that all the individual κ_i are. The deliberate vagueness of this setup in regard to the nature of the underlying decomposition leads to a rich framework: Many different kernels can be obtained by simply changing the decomposition.

7.1 Graph Kernels as R-Convolutions

To apply R-convolution kernels to graphs, one decomposes the graph into smaller substructures, and builds the kernel based on similarities between those components. Most graph kernels are—knowingly or not—based on R-convolutions; they mainly differ in the way they decompose the graph for comparison and the similarity measure they use to compare the components.

6. We thank an anonymous reviewer for pointing this out.
7. Haussler (1999) implicitly assumed this sum to be well-defined, hence did not use μ in his definition.

Gärtner et al. (2003) observed that any graph kernel whose feature map is injective could be used to determine whether two graphs G and G' are isomorphic: Simply compute $d(G,G') := k(G,G) - 2k(G,G') + k(G',G')$; since by definition *any* structural difference between the graphs would yield a non-zero $d(G,G')$, they are isomorphic iff $d(G,G') = 0$. The graph isomorphism problem, however, is widely believed to be not solvable in polynomial time (Garey and Johnson, 1979). Gärtner et al. (2003) also showed that computing inner products in a feature space constructed over all subgraphs of a graph is NP-hard. One must therefore choose which substructures to distinguish in defining a practical graph kernel, and this choice is generally motivated by runtime considerations.

Random walks provide a straightforward graph decomposition that—as we have seen in Section 4—leads to kernels that can be computed efficiently. To see that our random walk graph kernel (8) is indeed an R-convolution kernel, note that the definition of our weight matrix (6) and the RKHS Kronecker product (Definition 11) imply

$$\begin{aligned}[W_\times]_{(i-1)n'+r,\,(j-1)n'+s} &= [\Phi(X) \otimes \Phi(X')]_{(i-1)n'+r,\,(j-1)n'+s} \\ &= \left\langle \phi(v_i, v_j), \phi(v'_r, v'_s) \right\rangle_{\mathcal{H}} =: \kappa((v_i, v_j), (v'_r, v'_s)),\end{aligned}$$

where κ is our edge kernel. We can thus expand (8) by explicitly taking all paths through the repeated matrix products, giving

$$\begin{aligned}k(G,G') &:= \sum_{k=1}^{\infty} \mu(k)\, q_\times^\top W_\times^k p_\times = \sum_{k=1}^{\infty} \mu(k)\, q_\times^\top \left(\prod_{i=1}^{k} W_\times \right) p_\times \\ &= \sum_{k=1}^{\infty} \mu(k) \sum_{\substack{v_0, v_1, \ldots v_k \in V \\ v'_0, v'_1, \ldots v'_k \in V'}} q_{v_k} q'_{v'_k} \left(\prod_{i=1}^{k} \kappa((v_{i-1}, v_i), (v'_{i-1}, v'_i)) \right) p_{v_0}\, p'_{v'_0}. \end{aligned} \tag{60}$$

This is easily identified as an instance of the R-convolution kernel (59), where the decomposition is into all equal-length sequences $\vec{v}, \vec{v}'$ of nodes from V and V', respectively, and

$$\mu(\vec{v}, \vec{v}') := \mu(|\vec{v}|)\, q_{v_{|\vec{v}|}} q'_{v'_{|\vec{v}|}} p_{v_0}\, p'_{v'_0}, \tag{61}$$

where $|\cdot|$ in (61) denotes the length of a sequence. Finally, note that by definition of our edge kernel κ, only pairs of sequences that are both actual walks on their respective graphs will make a non-zero contribution to (60).

Random walk graph kernels as proposed by Gärtner et al. (2003) likewise decompose a graph into random walks, but then employ a delta kernel between nodes. Borgwardt et al. (2005), on the other hand, use a kernel defined on both nodes and edges. The marginalized graph kernels of Kashima et al. (2004) are closely related but subtly different in that they decompose the graph into all possible *label* sequences generated by a walk. Mahé et al. (2004) extend this approach in two ways: They enrich the labels via the so-called Morgan index, and modify the kernel definition to prevent *tottering*, that is, the generation of high similarity scores by multiple, similar, small substructures. Both these extensions are particularly relevant for chemoinformatics applications.

Further afield, Horváth et al. (2004) decompose a graph into cyclic patterns, then count the number of common cyclic patterns which occur in both graphs. Their kernel is plagued by computational issues; in fact they show that computing the cyclic pattern kernel of a general graph is NP-hard. They consequently restrict their attention to practical problem classes where the number of simple cycles is bounded.

Ramon and Gärtner (2003) consider subtree patterns to define graph kernels. Starting from a given node v, a tree is created by adding all the nodes that can be reached from v in $1,\ldots,h$ steps, where h is the height of the tree. If more than one walk connects two nodes, then each one of these is used to define a distinct subtree. This means that the same node is counted several times, thus leading to tottering. Furthermore, the number of candidate trees grows exponentially with the height of the subtree under consideration, thus severely limiting the depth of graph structure one can probe at reasonable computational cost.

Borgwardt and Kriegel (2005) define a kernel (62) based on shortest paths. They represent a graph $G = (V,E)$ by a complete graph $S = (V,\bar{E})$ over the same vertices, wherein the weight of each edge in $\bar{E}$ equals the length of the shortest path between the corresponding nodes in G. Their shortest path kernel is then defined as

$$k_{\mathrm{sp}}(G,G') = \sum_{e\in\bar{E}} \sum_{e'\in\bar{E}'} \kappa(e,e'), \tag{62}$$

where κ is any kernel defined on the edges of S and S'.

Shervashidze et al. (2009) use subgraphs of fixed size to define kernels. Their key idea is to represent the graph by a normalized frequency vector which counts the frequency of occurrence of various fixed-size subgraphs. The kernel is then simply computed as the dot product between these vectors.

Other decompositions of graphs which are well suited for particular application domains include molecular fingerprints based on various types of depth-first searches (Ralaivola et al., 2005) and structural elements such as rings or functional groups (Fröhlich et al., 2006).

7.2 R-Convolutions in Abstract Semirings

There have been a few attempts to extend the R-convolution kernel (59) to abstract semirings, by defining:

$$k(x,x') := \bigoplus_{\substack{\vec{x}\in R^{-1}(x)\\ \vec{x}'\in R^{-1}(x')}} \mu(\vec{x},\vec{x}') \,\bar{\odot} \bigodot_{i=1}^{D} \kappa_i(x_i,x_i'). \tag{63}$$

The optimal assignment graph kernel of Fröhlich et al. (2006) is motivated along these lines, using the tropical semiring. It can be defined as

$$k(x,x') = \max_{\substack{\vec{x}\in R^{-1}(x)\\ \vec{x}'\in R^{-1}(x')}} \left(\mu(\vec{x},\vec{x}') + \sum_{i=1}^{D} \kappa_i(x_i,x_i') \right). \tag{64}$$

Unfortunately (64) is not always p.s.d. (Vert, 2008). The problem is that the class of p.s.d. kernels is not closed under the max operation (Berg et al., 1984).

For semirings that have a morphism ψ to the reals, however, we can rewrite (63) as

$$\psi(k(x,x')) = \sum_{\substack{\vec{x}\in R^{-1}(x)\\ \vec{x}'\in R^{-1}(x')}} \mu(\vec{x},\vec{x}') \prod_{i=1}^{D} \psi(\kappa_i(x_i,x_i')). \tag{65}$$

Comparing (65) with (59) makes it clear that $\psi \circ k$ is p.s.d. and hence admissible if all $\psi \circ \kappa_i$ are. This can be used to construct p.s.d. R-convolution kernels in such semirings.

For instance, take the logarithmic semiring $(\mathbb{R} \cup \{-\infty\}, \bar{\oplus}_{\ln}, +, -\infty, 0)$ augmented with an inverse temperature parameter $\beta > 0$, so that $x \bar{\oplus}_{\ln} y := \ln(e^{\beta x} + e^{\beta y})/\beta$. This has the morphism $\psi(x) = e^{\beta x}$. We can thus specialize (65) to define

$$k(x,x') := \sum_{\substack{\vec{x} \in R^{-1}(x) \\ \vec{x}' \in R^{-1}(x')}} e^{\beta \kappa(\vec{x},\vec{x}')}, \quad \text{where} \quad \kappa(\vec{x},\vec{x}') := \mu(\vec{x},\vec{x}') + \sum_{i=1}^{D} \kappa_i(x_i,x_i'), \tag{66}$$

which is a valid p.s.d. kernel if all $e^{\beta \kappa_i}$ are. Note that if κ_i is a p.s.d. kernel, then since $\beta > 0$ so is $\beta \kappa_i$, and since p.s.d. kernels are closed under exponentiation (Genton, 2001, Equation 5) so is $e^{\beta \kappa_i}$.

What makes (66) interesting is that when the temperature approaches zero ($\beta \to \infty$), the augmented logarithmic semiring approaches the tropical semiring, as $x \bar{\oplus}_{\ln} y \to \max(x,y)$. We thus obtain a kernel that approximates (an exponentiated version of) the optimal assignment kernel (64) yet is provably p.s.d. Since at low temperatures the value of (66) is dominated by the optimal assignment, one might call it the "mostly optimal assignment kernel."

The finite range of floating-point computer arithmetic unfortunately limits how low a temperature (66) can be used with in practice, though this can be greatly extended via suitable software, such as the `extnum` C++ class.[8]

8. Discussion and Outlook

As evidenced by the large number of recent papers, random walk and marginalized graph kernels have received considerable research attention. Although the connections between these two kernels were hinted at by Kashima et al. (2004), no effort was made to pursue this further. Our aim in presenting a unified framework for random walk and marginalized graph kernels that combines the best features of previous formulations is to highlight the similarities as well as the differences between these approaches. Furthermore, it allows us to use extended linear algebra in an RKHS to efficiently compute all these kernels by exploiting common structure inherent in these problems.

As more and more graph-structured data (e.g., molecular structures and protein interaction networks) becomes available in fields such as biology, web data mining, *etc*., graph classification will gain importance over the coming years. Hence there is a pressing need to speed up the computation of similarity metrics on graphs. We have shown that sparsity, low effective rank, and Kronecker product structure can be exploited to greatly reduce the computational cost of graph kernels; taking advantage of other forms of structure in $W_\times$ remains a computational challenge. Now that the computation of random walk graph kernels is viable for practical problem sizes, it will open the doors for their application in hitherto unexplored domains.

A major deficiency of geometric random walk graph kernels is that the admissible range of values of the decay parameter λ in (16) depends on the spectrum of the weight matrix $W_\times$. Since this is typically unknown, in practice one often resorts to very low values of λ—but this makes the contributions of higher-order terms (corresponding to long walks) to the kernel negligible. In fact in many applications a naive kernel which simply computes the average kernel between all pairs of edges in the two graphs has performance comparable to the random walk graph kernel.

8. `extnum` can be found at `http://darwin.nmsu.edu/molb_resources/bioinformatics/extnum/extnum.html`.

Trying to remedy this situation by normalizing the matrices involved leads to another phenomenon called *tottering* (Mahé et al., 2004). Roughly speaking tottering occurs when short self-repeating walks make a disproportionately large contribution to the kernel value. Consider two adjacent vertices v and v' in a graph. Because of tottering, contributions due to walks of the form $v \to v' \to v \to \ldots$ dominate the kernel value. Unfortunately a kernel using self-avoiding walks (walks which do not visit the same vertex twice) cannot be computed in polynomial time.

A natural question to ask is the following: Since diffusion can be viewed as a continuous time limit of random walks, can the ideas behind the random walk kernel be extended to diffusion? Unfortunately, the Laplacian of the product graph does not decompose into the Kronecker product of the Laplacian matrices of the constituent graphs; this rules out a straightforward extension.

Although rational kernels have always been viewed as distinct from graph kernels, we have shown that in fact these two research areas are closely related. It is our hope that this will facilitate cross-pollination of ideas such as the use of semirings and transducers in defining graph kernels. A return to the tensor and matrix notation which was commonly used to describe algebraic path problems would help make these connections explicit.

It is fair to say that R-convolution kernels are the mother of all kernels on structured data. It is enlightening to view various graph kernels as instances of R-convolution kernels since this brings into focus the relevant decomposition used to define a given kernel, and the similarities and differences between various kernels. Extending R-convolutions to abstract semirings, however, does not always result in a valid p.s.d. kernel. We have shown that a morphism to the reals is sufficient to successfully transport an R-convolution kernel into a semiring; whether it is necessary remains an open problem.

We do not believe that the last word on graph comparison has been said yet. Thus far, simple decompositions like random walks have been used to compare graphs. This is mainly driven by computational considerations and not by the application domain at hand. The algorithmic challenge of the future is to integrate higher-order structures such as spanning trees in graph comparisons, and to compute such kernels efficiently.

Acknowledgments

We thank Markus Hegland and Tim Sears for enlightening discussions, Alex Smola for pointing out to us that the optimal assignment kernel may fail to be p.s.d., and the anonymous reviewers for their detailed comments and suggestions which greatly helped improve this paper.

This publication only reflects the authors' views. It was supported by NICTA, funded by the Australian Government through the Backing Australia's Ability and Centre of Excellence programs, by the IST Programme of the European Community under the PASCAL2 Network of Excellence, IST-2007-216886, by the German Ministry for Education, Science, Research and Technology (BMBF) under grant No. 031U112F within the BFAM (Bioinformatics for the Functional Analysis of Mammalian Genomes) project, part of the German Genome Analysis Network (NGFN), by NIH grant GM063208-05 "Tools and Data Resources in Support of Structural Genomics", and NSF grant IIS-0916686.

Appendix A. Extending Linear Algebra to RKHS

It is well known that any continuous, symmetric, positive definite kernel $\kappa\colon \mathcal{X}\times\mathcal{X}\to\mathbb{R}$ has a corresponding Hilbert space $\mathcal{H}$, called the Reproducing Kernel Hilbert Space or RKHS, which induces a feature map $\phi : \mathcal{X}\to\mathcal{H}$ satisfying $\kappa(x,x') = \langle\phi(x),\phi(x')\rangle_{\mathcal{H}}$. The natural extension of this so-called feature map to matrices is $\Phi\colon \mathcal{X}^{n\times m}\to\mathcal{H}^{n\times m}$ defined $[\Phi(A)]_{ij} := \phi(A_{ij})$. In what follows, we use Φ to lift tensor algebra from $\mathcal{X}$ to $\mathcal{H}$, extending various matrix products to the RKHS, and proving some of their their useful properties. Straightforward extensions via the commutativity properties of the operators have been omitted for the sake of brevity.

A.1 Matrix Product

Definition 9 *Let $A\in\mathcal{X}^{n\times m}$, $B\in\mathcal{X}^{m\times p}$, and $C\in\mathbb{R}^{m\times p}$. The matrix products $\Phi(A)\Phi(B)\in\mathbb{R}^{n\times p}$ and $\Phi(A)C\in\mathcal{H}^{n\times p}$ are given by*

$$[\Phi(A)\Phi(B)]_{ik} := \sum_j \big\langle\phi(A_{ij}),\phi(B_{jk})\big\rangle_{\mathcal{H}} \quad \textit{and} \quad [\Phi(A)C]_{ik} := \sum_j \phi(A_{ij})C_{jk}.$$

It is straightforward to show that the usual properties of matrix multiplication—namely associativity, transpose-commutativity, and distributivity with addition—hold for Definition 9 above, with one exception: associativity does *not* hold if the elements of all three matrices involved belong to the RKHS. In other words, given $A\in\mathcal{X}^{n\times m}$, $B\in\mathcal{X}^{m\times p}$, and $C\in\mathcal{X}^{p\times q}$, generally $[\Phi(A)\Phi(B)]\Phi(C)\neq\Phi(A)[\Phi(B)\Phi(C)]$. The technical difficulty is that in general

$$\big\langle\phi(A_{ij}),\phi(B_{jk})\big\rangle_{\mathcal{H}}\phi(C_{kl}) \neq \phi(A_{ij})\big\langle\phi(B_{jk}),\phi(C_{kl})\big\rangle_{\mathcal{H}}. \tag{67}$$

Further examples of statements like (67), involving properties which do not hold when extended to an RKHS, can be found for the other matrix products at (69) and (76) below.

Definition 9 allows us to state a first RKHS extension of the vec(ABC) formula (1):

Lemma 10 *If $A\in\mathbb{R}^{n\times m}$, $B\in\mathcal{X}^{m\times p}$, and $C\in\mathbb{R}^{p\times q}$, then*

$$\operatorname{vec}(A\Phi(B)C)) = (C^\top\otimes A)\operatorname{vec}(\Phi(B)) \ \in\mathcal{X}^{nq\times 1}.$$

Proof Analogous to Lemma 12 below. ■

A.2 Kronecker Product

Definition 11 *Let $A\in\mathcal{X}^{n\times m}$ and $B\in\mathcal{X}^{p\times q}$. The Kronecker product $\Phi(A)\otimes\Phi(B)\in\mathbb{R}^{np\times mq}$ is defined as*

$$[\Phi(A)\otimes\Phi(B)]_{(i-1)p+k,(j-1)q+l} := \big\langle\phi(A_{ij}),\phi(B_{kl})\big\rangle_{\mathcal{H}}.$$

Similarly to (67) above, for matrices in an RKHS

$$* \qquad (\Phi(A)\otimes\Phi(B))(\Phi(C)\otimes\Phi(D)) = (\Phi(A)\,\Phi(C))\otimes(\Phi(B)\,\Phi(D)) \tag{68}$$

does *not* necessarily hold. The technical problem with (68) is that generally

$$\langle \phi(A_{ir}), \phi(B_{ks}) \rangle_{\mathcal{H}} \langle \phi(C_{rj}), \phi(D_{sl}) \rangle_{\mathcal{H}} \neq \langle \phi(A_{ir}), \phi(C_{rj}) \rangle_{\mathcal{H}} \langle \phi(B_{ks}), \phi(D_{sl}) \rangle_{\mathcal{H}}. \tag{69}$$

In Section A.3 we show that analogous properties (Lemmas 14 and 15) do hold for the *heterogeneous* Kronecker product between RKHS and real matrices.

Definition 11 gives us a second extension of the vec(ABC) formula (1) to RKHS:

Lemma 12 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathbb{R}^{m\times p}$, and $C \in \mathcal{X}^{p\times q}$, then*

$$\operatorname{vec}(\Phi(A)\, B\, \Phi(C)) = (\Phi(C)^\top \otimes \Phi(A)) \operatorname{vec}(B) \ \in \mathbb{R}^{nq\times 1}.$$

Proof We begin by rewriting the k^{th} column of $\Phi(A)B\Phi(C)$ as

$$\begin{aligned}
[\Phi(A)B\Phi(C)]_{*k} = \Phi(A) \sum_j B_{*j}\, \phi(C_{jk}) &= \sum_j \phi(C_{jk}) \Phi(A) B_{*j} \\
&= [\phi(C_{1k})\Phi(A), \phi(C_{2k})\Phi(A), \ldots \phi(C_{nk})\Phi(A)] \underbrace{\begin{bmatrix} B_{*1} \\ B_{*2} \\ \vdots \\ B_{*n} \end{bmatrix}}_{\operatorname{vec}(B)} \\
&= ([\phi(C_{1k}), \phi(C_{2k}), \ldots \phi(C_{nk})] \otimes \Phi(A)) \operatorname{vec}(B). \qquad (70)
\end{aligned}$$

To obtain Lemma 12 we stack up the columns of (70):

$$\begin{aligned}
\operatorname{vec}(\Phi(A)\, B\, \Phi(C)) &= \left(\begin{bmatrix} \phi(C_{11}) & \phi(C_{21}) & \ldots & \phi(C_{n1}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(C_{1n}) & \phi(C_{2n}) & \ldots & \phi(C_{nn}) \end{bmatrix} \otimes \Phi(A) \right) \operatorname{vec}(B) \\
&= (\Phi(C)^\top \otimes \Phi(A)) \operatorname{vec}(B).
\end{aligned}$$

■

Direct computation of the right-hand side of Lemma 12 requires $nmpq$ kernel evaluations; when m, p, and q are all $O(n)$ this is $O(n^4)$. If $\mathcal{H}$ is finite-dimensional, however—in other words, if the feature map can be taken to be $\phi\colon \mathcal{X} \to \mathbb{R}^d$ with $d < \infty$—then the left-hand side of Lemma 12 can be obtained in $O(n^3 d)$ operations. Our efficient computation schemes in Section 4 exploit this observation.

A.3 Heterogeneous Kronecker Product

Definition 13 *Let $A \in \mathcal{X}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$. The heterogeneous Kronecker product $\Phi(A) \otimes B \in \mathcal{X}^{np\times mq}$ is given by*

$$[\Phi(A) \otimes B]_{(i-1)p+k,(j-1)q+l} = \phi(A_{ij})\, B_{kl}.$$

Recall that the standard Kronecker product obeys (2); here we prove two extensions:

Lemma 14 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathcal{X}^{p\times q}$, $C \in \mathbb{R}^{m\times o}$, and $D \in \mathbb{R}^{q\times r}$, then*

$$(\Phi(A)\otimes\Phi(B))(C\otimes D) = (\Phi(A)C)\otimes(\Phi(B)D).$$

Proof Using the linearity of the inner product we directly verify

$$\begin{aligned}
[(\Phi(A)\otimes\Phi(B))(C\otimes D)]_{(i-1)p+k,(j-1)q+l} &= \sum_{r,s}\langle\phi(A_{ir}),\phi(B_{ks})\rangle_{\mathcal{H}} C_{rj}D_{sl} \\
&= \left\langle \sum_r \phi(A_{ir})C_{rj}, \sum_s \phi(B_{ks})D_{sl} \right\rangle_{\mathcal{H}} \\
&= \left\langle [\Phi(A)C]_{ij}, [\Phi(B)D]_{kl} \right\rangle_{\mathcal{H}} \\
&= [(\Phi(A)C)\otimes(\Phi(B)D)]_{(i-1)p+k,(j-1)q+l}
\end{aligned}$$

■

Lemma 15 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathbb{R}^{p\times q}$, $C \in \mathcal{X}^{m\times o}$, and $D \in \mathbb{R}^{q\times r}$, then*

$$(\Phi(A)\otimes B)(\Phi(C)\otimes D) = (\Phi(A)\,\Phi(C))\otimes(BD).$$

Proof Using the linearity of the inner product we directly verify

$$\begin{aligned}
[(\Phi(A)\otimes B)(\Phi(C)\otimes D)]_{(i-1)p+k,(j-1)q+l} &= \sum_{r,s}\left\langle \phi(A_{ir})B_{ks}, \phi(C_{rj})D_{sl}\right\rangle_{\mathcal{H}} \\
&= \sum_r \left\langle \phi(A_{ir}),\phi(C_{rj})\right\rangle_{\mathcal{H}} \sum_s B_{ks}D_{sl} \\
&= [\Phi(A)\,\Phi(C)]_{ij}\,[BD]_{kl} \\
&= [(\Phi(A)\,\Phi(C))\otimes(BD)]_{(i-1)p+k,(j-1)q+l}
\end{aligned}$$

■

Using the heterogeneous Kronecker product, we can state four more RKHS extensions of the vec-ABC formula (1):

Lemma 16 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathbb{R}^{m\times p}$, and $C \in \mathbb{R}^{p\times q}$, then*

$$\operatorname{vec}(\Phi(A)\,BC) = (C^\top\otimes\Phi(A))\operatorname{vec}(B) \ \in \mathcal{X}^{nq\times 1}.$$

Proof Analogous to Lemma 12. ■

Lemma 17 *If $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{m\times p}$, and $C \in \mathcal{X}^{p\times q}$, then*

$$\operatorname{vec}(AB\,\Phi(C)) = (\Phi(C)^\top\otimes A)\operatorname{vec}(B) \ \in \mathcal{X}^{nq\times 1}.$$

Proof Analogous to Lemma 12. ■

Lemma 18 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathcal{X}^{m\times p}$, and $C \in \mathbb{R}^{p\times q}$, then*

$$\operatorname{vec}(\Phi(A)\,\Phi(B)C) = (C^\top \otimes \Phi(A))\operatorname{vec}(\Phi(B)) \;\in \mathbb{R}^{nq\times 1}.$$

Proof Analogous to Lemma 12. ■

Lemma 19 *If $A \in \mathbb{R}^{n\times m}$, $B \in \mathcal{X}^{m\times p}$, and $C \in \mathcal{X}^{p\times q}$, then*

$$\operatorname{vec}(A\,\Phi(B)\,\Phi(C)) = (\Phi(C)^\top \otimes A)\operatorname{vec}(\Phi(B)) \;\in \mathbb{R}^{nq\times 1}.$$

Proof Analogous to Lemma 12. ■

Note that there is no analogous lemma for $\operatorname{vec}(\Phi(A)\,\Phi(B)\,\Phi(C))$ since this term is not well-defined due to non-associativity (67).

A.4 Kronecker Sum

A concept closely related to the Kronecker product is that of the Kronecker sum, which is defined for real matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$ as

$$A \oplus B := A \otimes \mathbf{I}_{pq} + \mathbf{I}_{nm} \otimes B, \tag{71}$$

with $\mathbf{I}_{nm}$ (resp. $\mathbf{I}_{pq}$) denoting the $n \times m$ (resp. $p \times q$) identity matrix. Many of its properties can be derived from those of the Kronecker product. Unlike the Kronecker product, however, the Kronecker sum of two matrices in an RKHS is a matrix in the RKHS. From Definition 1 and (71) we find that

$$[A \oplus B]_{(i-1)p+k,(j-1)q+l} := A_{ij}\delta_{kl} + \delta_{ij}B_{kl}. \tag{72}$$

We can extend (72) to RKHS, defining analogously:

Definition 20 *Let $A \in \mathcal{X}^{n\times m}$ and $B \in \mathcal{X}^{p\times q}$. The Kronecker sum $\Phi(A) \oplus \Phi(B) \in \mathcal{X}^{np\times mq}$ is defined as*

$$[\Phi(A) \oplus \Phi(B)]_{(i-1)p+k,(j-1)q+l} := \phi(A_{ij})\delta_{kl} + \delta_{ij}\phi(B_{kl}).$$

In other words, in an RKHS the Kronecker sum is defined just as in (71):

$$\Phi(A) \oplus \Phi(B) \;=\; \Phi(A) \otimes \mathbf{I}_B + \mathbf{I}_A \otimes \Phi(B), \tag{73}$$

where $\mathbf{I}_M$ denotes the real-valued identity matrix of the same dimensions (not necessarily square) as matrix M. In accordance with Definition 13, the result of (73) is an RKHS matrix.

The equivalent of the vec-ABC formula (1) for Kronecker sums is:

$$\begin{aligned}(A \oplus B)\operatorname{vec}(C) &= (A \otimes \mathbf{I}_B + \mathbf{I}_A \otimes B)\operatorname{vec}(C)\\ &= (A \otimes \mathbf{I}_B)\operatorname{vec}(C) + (\mathbf{I}_A \otimes B)\operatorname{vec}(C)\\ &= \operatorname{vec}(\mathbf{I}_B C A^\top) + \operatorname{vec}(BC\mathbf{I}_A^\top)\\ &= \operatorname{vec}(\mathbf{I}_B C A^\top + BC\mathbf{I}_A^\top).\end{aligned} \tag{74}$$

This also works for matrices in an RKHS:

Lemma 21 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathcal{X}^{p\times q}$, and $C \in \mathcal{X}^{q\times m}$, then*

$$(\Phi(A) \oplus \Phi(B))\,\mathrm{vec}(\Phi(C)) = \mathrm{vec}(\mathbf{I}_B\,\Phi(C)\,\Phi(A)^\top + \Phi(B)\,\Phi(C)\,\mathbf{I}_A^\top) \in \mathbb{R}^{np\times 1}.$$

Proof Analogous to (74), using Lemmas 18 and 19. ■

Furthermore, we have two valid heterogeneous forms that map into the RKHS:

Lemma 22 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathcal{X}^{p\times q}$, and $C \in \mathbb{R}^{q\times m}$, then*

$$(\Phi(A) \oplus \Phi(B))\,\mathrm{vec}(C) = \mathrm{vec}(\mathbf{I}_B\,C\,\Phi(A)^\top + \Phi(B)\,C\,\mathbf{I}_A^\top) \in \mathcal{X}^{np\times 1}.$$

Proof Analogous to (74), using Lemmas 16 and 17. ■

Lemma 23 *If $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{p\times q}$, and $C \in \mathcal{X}^{q\times m}$, then*

$$(A \oplus B)\,\mathrm{vec}(\Phi(C)) = \mathrm{vec}(\mathbf{I}_B\,\Phi(C)\,A^\top + B\,\Phi(C)\,\mathbf{I}_A^\top) \in \mathcal{X}^{np\times 1}.$$

Proof Analogous to (74), using Lemma 10. ■

A.5 Hadamard Product

While the extension of the Hadamard (element-wise) product to an RKHS is not required to implement our fast graph kernels, the reader may find it interesting in its own right.

Definition 24 *Let $A, B \in \mathcal{X}^{n\times m}$ and $C \in \mathbb{R}^{n\times m}$. The Hadamard products $\Phi(A) \odot \Phi(B) \in \mathbb{R}^{n\times m}$ and $\Phi(A) \odot C \in \mathcal{H}^{n\times m}$ are given by*

$$[\Phi(A) \odot \Phi(B)]_{ij} = \big\langle \phi(A_{ij}), \phi(B_{ij}) \big\rangle_{\mathcal{H}} \quad \text{and} \quad [\Phi(A) \odot C]_{ij} = \phi(A_{ij})\,C_{ij}.$$

We prove two extensions of (3):

Lemma 25 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathcal{X}^{p\times q}$, $C \in \mathbb{R}^{n\times m}$, and $D \in \mathbb{R}^{p\times q}$, then*

$$(\Phi(A) \otimes \Phi(B)) \odot (C \otimes D) = (\Phi(A) \odot C) \otimes (\Phi(B) \odot D).$$

Proof Using the linearity of the inner product we directly verify

$$\begin{aligned}
[(\Phi(A) \otimes \Phi(B)) \odot (C \otimes D)]_{(i-1)p+k,(j-1)q+l} &= \big\langle \phi(A_{ij}), \phi(B_{kl}) \big\rangle_{\mathcal{H}} C_{ij} D_{kl} \\
&= \big\langle \phi(A_{ij})C_{ij}, \phi(B_{kl})D_{kl} \big\rangle_{\mathcal{H}} \\
&= \big\langle [\Phi(A) \odot C]_{ij}, [\Phi(B) \odot D]_{kl} \big\rangle_{\mathcal{H}} \\
&= [(\Phi(A) \odot C) \otimes (\Phi(B) \odot D)]_{(i-1)p+k,(j-1)q+l}
\end{aligned}$$

■

Lemma 26 *If $A \in \mathcal{X}^{n\times m}$, $B \in \mathbb{R}^{p\times q}$, $C \in \mathcal{X}^{n\times m}$, and $D \in \mathbb{R}^{p\times q}$, then*

$$(\Phi(A) \otimes B) \odot (\Phi(C) \otimes D) = (\Phi(A) \odot \Phi(C)) \otimes (B \odot D).$$

Proof Using the linearity of the inner product we directly verify

$$\begin{aligned}
[(\Phi(A) \otimes B) \odot (\Phi(C) \otimes D)]_{(i-1)p+k,(j-1)q+l} &= \left\langle \phi(A_{ij})B_{kl}, \phi(C_{ij})D_{kl} \right\rangle_{\mathcal{H}} \\
&= \left\langle \phi(A_{ij}), \phi(C_{ij}) \right\rangle_{\mathcal{H}} B_{kl}D_{kl} \\
&= [\Phi(A) \odot \Phi(C)]_{ij}[B \odot D]_{kl} \\
&= [(\Phi(A) \odot \Phi(C)) \otimes (B \odot D)]_{(i-1)p+k,(j-1)q+l}
\end{aligned}$$

■

As before,

$$* \qquad (\Phi(A) \otimes \Phi(B)) \odot (\Phi(C) \otimes \Phi(D)) = (\Phi(A) \odot \Phi(C)) \otimes (\Phi(B) \odot \Phi(D)) \tag{75}$$

does *not* necessarily hold, the difficulty with (75) being that in general,

$$\left\langle \phi(A_{ij}), \phi(B_{kl}) \right\rangle_{\mathcal{H}} \left\langle \phi(C_{ij}), \phi(D_{kl}) \right\rangle_{\mathcal{H}} \neq \left\langle \phi(A_{ij}), \phi(C_{ij}) \right\rangle_{\mathcal{H}} \left\langle \phi(B_{kl}), \phi(D_{kl}) \right\rangle_{\mathcal{H}}. \tag{76}$$

References

Christian Berg, Jens P. R. Christensen, and Paul Ressel. *Harmonic Analysis on Semigroups*. Springer, New York, 1984.

Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

Dennis S. Bernstein. *Matrix Mathematics*. Princeton University Press, 2005.

Jean Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.

Nitin Bhardwaj and Hui Lu. Correlation between gene expression profiles and protein-protein interactions within and across genomes. *Bioinformatics*, 21(11):2730–2738, June 2005.

Danail Bonchev and Dennis H. Rouvray, editors. *Chemical Graph Theory: Introduction and Fundamentals*, volume 1. Gordon and Breach Science Publishers, London, UK, 1991.

Karsten M. Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *Proceedings of the International Conference on Data Mining*, pages 74–81, 2005.

Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alexander J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. In *Proceedings of Intelligent Systems in Molecular Biology (ISMB)*, Detroit, USA, 2005. `http://www.stat.purdue.edu/~vishy/papers/BorOngSchVisetal05.pdf`.

Karsten M. Borgwardt, Hans-Peter Kriegel, S. V. N. Vishwanathan, and Nicol N. Schraudolph. Graph kernels for disease outcome prediction from protein-protein interaction networks. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Tiffany Murray, and Teri E. Klein, editors, *Proceedings of the Pacific Symposium of Biocomputing 2007*, Maui Hawaii, January 2007. World Scientific.

Lars Bullinger, Konstanze Döhner, Eric Bair, Stefan Fröhling, Richard F. Schlenk, Robert Tibshirani, Hartmut Döhner, and Jonathan R. Pollack. Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia. *New England Journal of Medicine*, 350 (16):1605–1616, Apr 2004.

Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Rational kernels. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, volume 14, Cambridge, MA, 2002. MIT Press.

Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Positive definite rational kernels. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Procedings of the Annual Conference on Computational Learning Theory*, pages 41–56, 2003.

Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Rational kernels: Theory and algorithms. *Journal of Machine Learning Research*, 5:1035–1062, 2004.

Samuel Eilenberg. *Automata, Languages and Machines*, volume A. Academic Press, 1974.

Hunter B. Fraser, Aaron E. Hirsh, Dennis P. Wall, and Michael B. Eisen. Coevolution of gene expression among interacting proteins. *Proceedings of the National Academy of Science USA*, 101(24):9033–9038, Jun 2004.

Holger Fröhlich, Jörg K Wegner, Florian Siker, and andreas Zell. Kernel functions for attributed molecular graphs — a new similarity based approach to ADME prediction in classification and regression. *QSAR and Combinatorial Science*, 25(4):317–326, 2006.

Judith D. Gardiner, Alan J. Laub, James J. Amato, and Cleve B. Moler. Solution of the Sylvester matrix equation $AXB^\top + CXD^\top = E$. *ACM Transactions on Mathematical Software*, 18(2):223–231, 1992.

Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Series of Books in Mathematical Sciences. W. H. Freeman, 1979.

Thomas Gärtner, Peter A. Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Proceedings of the Annual Conference on Computational Learning Theory*, pages 129–143. Springer, 2003.

Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2:299–312, 2001.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

David Haussler. Convolutional kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, UC Santa Cruz, 1999.

Tamás Horváth, Thomas Gärtner, and Stefan Wrobel. Cyclic pattern kernels for predictive graph mining. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 158–167, 2004.

Wilfried Imrich and Sandi Klavžar. *Product Graphs, Structure and Recognition*. Wiley, 2000.

Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the International Conference on Machine Learning*, pages 321–328, San Francisco, CA, 2003. Morgan Kaufmann.

Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Kernels for graphs. In Koji Tsuda, Bernhard Schölkopf, and Jean-Philippe Vert, editors, *Kernels and Bioinformatics*, pages 155–170, Cambridge, MA, 2004. MIT Press.

Risi Kondor and Karsten Borgwardt. The skew spectrum of graphs. In *Proceedings of the International Conference on Machine Learning*, pages 496–503. ACM, 2008.

Risi Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the International Conference on Machine Learning*, pages 315–322, San Francisco, CA, 2002. Morgan Kaufmann.

Hugo Kubinyi. Drug research: Myths, hype and reality. *Nature Reviews: Drug Discovery*, 2(8): 665–668, August 2003.

Werner Kuich and Arto Salomaa. *Semirings, Automata, Languages*. Number 5 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1986.

Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 611–617. ACM, 2006. ISBN 1-59593-339-5.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. Computation of the canonical decomposition by means of a simultaneous generalized Schur decomposition. *SIAM Journal on Matrix Analysis and Applications*, 26(2):295–327, 2004.

Daniel J. Lehmann. Algebraic structures for transitive closure. *Theoretical Computer Science*, 4(1): 59–76, February 1977.

Pierre Mahé, Nobuhisa Ueda, Tatsuya Akutsu, Jean-Luc Perret, and Jean-Philippe Vert. Extensions of marginalized graph kernels. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 552–559, 2004.

Mehryar Mohri. Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350, 2002.

Mehryar Mohri, Fernando C. N. Pereira, and Michael D. Riley. Weighted automata in text and speech processing. In András Kornai, editor, *Extended Finite State Models of Language: Proceedings of the ECAI'96 Workshop*, pages 46–50, 1996.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

Fernando C. N. Pereira and Michael D. Riley. Speech recognition by composition of weighted finite automata. In *Finite-State Language Processing*, pages 431–453. MIT Press, 1997.

Nikos P. Pitsianis. *The Kronecker Product in Approximation and Fast Transform Generation*. PhD thesis, Department of Computer Science, Cornell University, 1992.

Liva Ralaivola, Sanjay J. Swamidass, Hiroto Saigo, and Pierre Baldi. Graph kernels for chemical informatics. *Neural Networks*, 18(8):1093–1110, October 2005.

Jan Ramon and Thomas Gärtner. Expressivity versus efficiency of graph kernels. Technical report, First International Workshop on Mining Graphs, Trees and Sequences (held with ECML/PKDD'03), 2003.

Jean-François Rual, Kavitha Venkatesan, Tong Hao, Tomoko Hirozane-Kishikawa, Amélie Dricot, Ning Li, Gabriel F. Berriz, Francis D. Gibbons, Matija Dreze, Nono Ayivi-Guedehoussou, Niels Klitgord, Christophe Simon, Mike Boxem, Stuart Milstein, Jennifer Rosenberg, Debra S. Goldberg, Lan V. Zhang, Sharyl L. Wong, Giovanni Franklin, Siming Li, Joanna S. Albala, Janghoo Lim, Carlene Fraughton, Estelle Llamosas, Sebiha Cevik, Camille Bex, Philippe Lamesch, Robert S. Sikorski, Jean Vandenhaute, Huda Y. Zoghbi, Alex Smolyar, Stephanie Bosak, Reynaldo Sequerra, Lynn Doucette-Stamm, Michael E. Cusick, David E. Hill, Frederick P. Roth, and Marc Vidal. Towards a proteome-scale map of the human protein-protein interaction network. *Nature*, 437(7062):1173–1178, Oct 2005.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. BRENDA, the enzyme database: Updates and major new developments. *Nucleic Acids Research*, 32D:431–433, Jan 2004.

Roded Sharan and Trey Ideker. Modeling cellular machinery through biological network comparison. *Nature Biotechnology*, 24(4):427–433, Apr 2006.

Nino Shervashidze, S. V. N. Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In Max Welling and David van Dyk, editors, *Proceedings of the International Workshop on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, 2009.

Alexander J. Smola and Risi Kondor. Kernels and regularization on graphs. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Proceedings of the Annual Conference on Computational Learning Theory*, Lecture Notes in Computer Science, pages 144–158, Heidelberg, Germany, 2003. Springer-Verlag.

Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. Technical Report 0808.4134, arXiv, 2008. http://arxiv.org/abs/0808.4134.

Hannu Toivonen, Ashwin Srinivasan, Ross D. King, Stefan Kramer, and Christoph Helma. Statistical evaluation of the predictive toxicology challenge 2000-2001. *Bioinformatics*, 19(10): 1183–1193, July 2003.

Koji Tsuda, Taishin Kin, and Kiyoshi Asai. Marginalized kernels for biological sequences. *Bioinformatics*, 18 (Suppl. 2):S268–S275, 2002.

Charles F. Van Loan. The ubiquitous Kronecker product. *Journal of Computational and Applied Mathematics*, 123(1–2):85–100, 2000.

Laura J. van't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A. M. Hart, Mao Mao, Hans L. Peterse, Karin van der Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, René Bernards, and Stephen H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.

Jean-Philippe Vert. The optimal assignment kernel is not positive definite. Technical Report 0801.4061, arXiv, May 2008. http://aps.arxiv.org/abs/0801.4061.

S. V. N. Vishwanathan. *Kernel Methods: Fast Algorithms and Real Life Applications*. PhD thesis, Indian Institute of Science, Bangalore, India, November 2002. http://www.stat.purdue.edu/~vishy/papers/Vishwanathan02.pdf.

S. V. N. Vishwanathan, Karsten Borgwardt, and Nicol N. Schraudolph. Fast computation of graph kernels. In B. Schölkopf, J. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge MA, 2007. MIT Press.

Patrick Warnat, Roland Eils, and Benedikt Brors. Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes. *BMC Bioinformatics*, 6:265, Nov 2005.

Takashi Washio and Hiroshi Motoda. State of the art of graph-based data mining. *SIGKDD Explorations*, 5(1):59–68, 2003.

Xiang Yao, Dong Wei, Cylburn Soden Jr., Michael F. Summers, and Dorothy Beckett. Structure of the carboxy-terminal fragment of the apo-biotin carboxyl carrier subunit of Escherichia coli acetyl-CoA carboxylase. *Biochemistry*, 36:15089–15100, 1997.

Journal of Machine Learning Research 11 (2010) 1243-1272 Submitted 5/08; Revised 12/09; Published 4/10

Stochastic Complexity and Generalization Error of a Restricted Boltzmann Machine in Bayesian Estimation

Miki Aoyagi AOYAGI.MIKI@NIHON-U.AC.JP
Department of Mathematics, College of Science & Technology
Nihon University
1-8-14, Surugadai, Kanda, Chiyoda-ku
Tokyo 101-8308, Japan

Editor: Tommi Jaakola

Abstract

In this paper, we consider the asymptotic form of the generalization error for the restricted Boltzmann machine in Bayesian estimation. It has been shown that obtaining the maximum pole of zeta functions is related to the asymptotic form of the generalization error for hierarchical learning models (Watanabe, 2001a,b). The zeta function is defined by using a Kullback function. We use two methods to obtain the maximum pole: a new eigenvalue analysis method and a recursive blowing up process. We show that these methods are effective for obtaining the asymptotic form of the generalization error of hierarchical learning models.

Keywords: Boltzmann machine, non-regular learning machine, resolution of singularities, zeta function

1. Introduction

A learning system consists of data, a learning model and a learning algorithm. The purpose of such a system is to estimate an unknown true density function from data distributed by the true density function. The data associated with image or speech recognition, artificial intelligence, the control of a robot, genetic analysis, data mining, time series prediction, and so on, are very complicated and usually not generated by a simple normal distribution, as they are influenced by many factors. Learning models for analyzing such data should likewise have complicated structures. Hierarchical learning models such as the Boltzmann machine, layered neural network, reduced rank regression and the normal mixture model are known to be effective learning models. They are, however, non-regular statistical models, which cannot be analyzed using the classic theories of regular statistical models (Hartigan, 1985; Sussmann, 1992; Hagiwara, Toda, and Usui, 1993; Fukumizu, 1996).

For example, consider a simple restricted Boltzmann machine that has two observable units and one hidden unit with binary variables (Fig. 1). The model is expressed by the probability form of two observable units $x=(x_1,x_2)\in\{1,-1\}^2$ with a parameter $a=(a_1,a_2)\in\mathbb{R}^2$:

$$p(x|a)=\sum_{y=\pm 1}p(x,y|a)=\frac{\exp(a_1x_1+a_2x_2)+\exp(-a_1x_1-a_2x_2)}{Z(a)},$$

where $y\in\{1,-1\}$ is the hidden variable,

$$p(x,y|a)=\frac{\exp(a_1x_1y+a_2x_2y)}{Z(a)},\ \text{and}\ Z(a)=\sum_{x_i=\pm 1,y=\pm 1,}\exp(a_1x_1y+a_2x_2y).$$

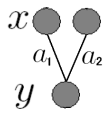


Figure 1: Simple restricted Boltzmann machine model: Two observable units and one hidden unit. The learning model is $p(x|a) \approx \exp(a_1x_1 + a_2x_2) + \exp(-a_1x_1 - a_2x_2)$.

We have

$$p(x|a) = \{(\prod_{i=1}^{2}(1 + x_i \tanh(a_i)) + \prod_{i=1}^{2}(1 - x_i \tanh(a_i)))\} \frac{\prod_{i=1}^{2} \cosh(a_i)}{Z(a)}$$

$$= \frac{\prod_{i=1}^{2} \cosh(a_i)}{Z(a)}(2 + 2x_1x_2 \tanh(a_1)\tanh(a_2)) = \frac{1 + x_1x_2 \tanh(a_1)\tanh(a_2)}{4}.$$

Assume that the true density function is $p(x|a^*)$ with $a^* = 0$. Then the true parameter set is $\{a = (a_1, a_2) \in \mathbb{R}^2 | p(x|a^*) = p(x|a)\} = \{a_1 = 0\} \cup \{a_2 = 0\}$. This set does not consist of only one point, resulting in a non-positive definite Fisher matrix function. On the other hand, the true parameter set of regular models should be one point and its Fisher matrix function is positive definite. Usually, the true parameter set of non-regular models is an analytic set with complicated singularities. Consequently, the many theoretical problems, such as clarifying generalization errors in learning theory, have remained unsolved.

The generalization error measures the difference between the true density function $q(x)$ and the predictive density function $p(x|x^n)$ obtained using n distributed training samples $x^n = (x_1, \ldots, x_n)$ of x from the true density function $q(x)$. We define it as the average Kullback distance between $q(x)$ and $p(x|x^n)$:

$$G(n) = E_n\{\sum_x q(x) \log \frac{q(x)}{p(x|x^n)}\},$$

where E_n is the expectation value over n training samples. This function clarifies precisely how $p(x|x^n)$ can approximate $q(x)$. Thus, $G(n)$ is also called a learning curve or a learning efficiency. For an arbitrary fixed parameter w^* in a parameter space W, we have

$$G(n) = \sum_x q(x) \log \frac{q(x)}{p(x|w^*)} + E_n\{\sum_x q(x) \log \frac{p(x|w^*)}{p(x|x^n)}\}.$$

The first and second terms are called the function approximation error and the statistical estimation error, respectively. The asymptotic form of the generalization error is important for model selection methods. The optimal model balances the function approximation error with the statistical estimation error. Since the Fisher matrix function is singular, non-regular models cannot be analyzed using the classic model selection methods of regular statistical models such as AIC (Akaike, 1974), TIC (Takeuchi, 1976), HQ (Hannan and Quinn, 1979), NIC (Murata, Yoshizawa, and Amari, 1994), BIC (Schwarz, 1978), and MDL (Rissanen, 1984). Therefore, it is important to construct a mathematical foundation for clarifying the generalization error of non-regular models.

In this paper, we clarify the generalization error of certain restricted Boltzmann machines, explicitly (Theorem 2 and Theorem 3), and give new bounds for the generalization error of the other types (Theorem 4), using both a new method of eigenvalue analysis and a recursive blowing up process. The restricted Boltzmann machine is one of the non-regular models and a complete bipartite graph type model that does not allow connections between hidden units (Hinton, 2004; Salakhutdinov, Mnih, and Hinton, 2007). It has been applied efficiently in recognizing hand-written digits and faces.

Several papers (Yamazaki and Watanabe, 2005; Nishiyama and Watanabe, 2006) have reported upper bounds for the asymptotic form of the generalization error for the Boltzmann machine model, but not the exact main terms.

We usually consider the generalization error in terms of a direct and an inverse problem. The direct problem involves solving the generalization error with a known true density function. The inverse problem is finding proper learning models and learning algorithms to minimize the generalization error under the condition of an unknown true density function. The inverse problem is important for practical usage, but in order to solve it, we first need to solve the direct problem. In this paper, we consider the direct problem of the restricted Boltzmann machine model.

We have already obtained the exact asymptotic forms of the generalization errors for the three layered neural network (Aoyagi and Watanabe, 2005a; Aoyagi, 2006), and for the reduced rank regression (Aoyagi and Watanabe, 2005b). In addition, Rusakov and Geiger (2005) obtained the same for Naive Bayesian networks (cf. Remark 1).

This paper consists of four sections. In Section 2, we summarize the framework of Bayesian learning models. In Section 3, we explain the restricted Boltzmann machine and show our main results, and we give our conclusions in Section 4.

2. Stochastic Complexity and Generalization Error in Bayesian Estimation

It is well known that Bayesian estimation is more appropriate than the maximum likelihood method when a learning machine is non-regular (Akaike, 1980; Mackay, 1992). In this paper, we consider the stochastic complexity and the generalization error in Bayesian estimation.

Let $q(x)$ be a true probability density function and $x^n := \{x_i\}_{i=1}^n$ be n training samples randomly selected from $q(x)$. Consider a learning model which is written by a probability form $p(x|w)$, where w is a parameter. The purpose of the learning system is to estimate $q(x)$ from x^n by using $p(x|w)$.

Let $p(w|x^n)$ be the *a posteriori* probability density function:

$$p(w|x^n) = \frac{1}{Z_n}\psi(w)\prod_{i=1}^{n} p(x_i|w),$$

where $\psi(w)$ is an *a priori* probability density function on the parameter set W and

$$Z_n = \int_W \psi(w)\prod_{i=1}^{n} p(x_i|w)\mathrm{d}w.$$

So the average inference $p(x|x^n)$ of the Bayesian density function is given by

$$p(x|x^n) = \int p(x|w)p(w|x^n)\mathrm{d}w,$$

which is the predictive density function.

Set

$$K(q||p) = \sum_x q(x) \log \frac{q(x)}{p(x|x^n)}.$$

This function always has a positive value and satisfies $K(q||p) = 0$ if and only if $q(x) = p(x|x^n)$.

The generalization error $G(n)$ is its expectation value E_n over n training samples:

$$G(n) = E_n\{\sum_x q(x) \log \frac{q(x)}{p(x|x^n)}\}.$$

Let

$$K_n(w) = \frac{1}{n} \sum_{i=1}^{n} \log \frac{q(x)}{p(x^n|w)}.$$

The average stochastic complexity or the free energy is defined by

$$F(n) = -E_n\{\log \int \exp(-nK_n(w))\psi(w)dw\}.$$

Then we have $G(n) = F(n+1) - F(n)$ for an arbitrary natural number n (Levin, Tishby, and Solla, 1990; Amari, Fujita, and Shinomoto, 1992; Amari and Murata, 1993). $F(n)$ is known as the Bayesian criterion in Bayesian model selection (Schwarz, 1978), stochastic complexity in universal coding (Rissanen, 1986; Yamanishi, 1998), Akaike's Bayesian criterion in optimization of hyper-parameters (Akaike, 1980) and evidence in neural network learning (Mackay, 1992). In addition, $F(n)$ is an important function for analyzing the generalization error.

It has recently been proved that the maximum pole of a zeta function gives the generalization error of hierarchical learning models asymptotically, assuming that the function approximation error is negligible compared to the statistical estimation error (Watanabe, 2001a,b). This assumption is natural for the model selection problem. To compare various models of different parameter's dimension, we assume that the true distribution is a certain dimensional model. If the parameter's dimension of the true distribution is larger than that of the learning model, clarifying the behavior of the generalization error is rather easy. We assume, therefore, that the true density distribution $q(x)$ is included in the learning model, that is, $q(x) = p(x|w^*)$ for $w^* \in W$, where W is the parameter space.

Define the zeta function $J(z)$ of a complex variable z for the learning model by

$$J(z) = \int K(w)^z \psi(w) \mathrm{d}w,$$

where $K(w)$ is the Kullback function:

$$K(w) = \sum_x p(x|w^*) \log \frac{p(x|w^*)}{p(x|w)}.$$

Then, for the maximum pole $-\lambda$ of $J(z)$ and its order θ, we have

$$F(n) = \lambda \log n - (\theta - 1) \log \log n + O(1), \tag{1}$$

where $O(1)$ is a bounded function of n, and if $G(n)$ has an asymptotic expansion,

$$G(n) \cong \lambda/n - (\theta - 1)/(n \log n) \text{ as } n \to \infty. \tag{2}$$

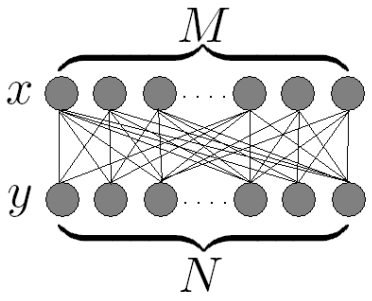


Figure 2: A restricted Boltzmann machine: M is the number of binary observable units x and N is the number of binary hidden units y. The learning model is $p(x,y|a) \propto \exp(\sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}x_iy_j)$, where a_{ij} is a parameter between x_i and y_j.

Therefore, our aim in this paper is to obtain λ and θ.

To assist in achieving this aim, we use the desingularization in algebraic geometry (Watanabe, 2009). It is, however, a new problem, even in mathematics, to obtain the desingularization of Kullback functions, since the singularities of these functions are very complicated and as such most of them have not yet been investigated (Appendix A). We, therefore, need a new method of eigenvalue analysis and a recursive blowing up process.

3. Restricted Boltzmann Machine

From now on, for simplicity, we denote

$$\{\{n\}\} = \begin{cases} 0, & \text{if } n = 0 \mod 2, \\ 1, & \text{if } n = 1 \mod 2, \end{cases} \quad \{\{(n_1,\cdots,n_m)\}\} = (\{\{n_1\}\},\cdots,\{\{n_m\}\}),$$

and we use the notation $\mathrm{d}a$ instead of $\prod_{i=1}^{H}\prod_{j=1}^{H'} \mathrm{d}a_{ij}$ for $a = (a_{ij})$.

Let $2 \leq M \in \mathbb{N}$ and $N \in \mathbb{N}$. Set

$$p(x,y|a) = \frac{\exp(\sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}x_iy_j)}{Z(a)},$$

where

$$Z(a) = \sum_{x_i=\pm 1, y_j=\pm 1,} \exp(\sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}x_iy_j),$$

$x = (x_i) \in \{1,-1\}^M$ and $y = (y_j) \in \{1,-1\}^N$ (Fig. 2).

Consider a restricted Boltzmann machine

$$\begin{aligned} p(x|a) &= \sum_{y_j=\pm 1} p(x,y|a) = \frac{\prod_{j=1}^N (\prod_{i=1}^M \exp(a_{ij}x_i) + \prod_{i=1}^M \exp(-a_{ij}x_i))}{Z(a)} \\ &= \{\prod_{j=1}^N (\prod_{i=1}^M (1+x_i \tanh(a_{ij})) + \prod_{i=1}^M (1-x_i \tanh(a_{ij})))\} \frac{\prod_{j=1}^N \prod_{i=1}^M \cosh(a_{ij})}{Z(a)} \\ &= \frac{\prod_{j=1}^N \prod_{i=1}^M \cosh(a_{ij})}{Z(a)} \\ &\times \prod_{j=1}^N (2 \sum_{0 \le p \le M/2} \sum_{i_1 < \cdots < i_{2p}} x_{i_1} x_{i_2} \cdots x_{i_{2p}} \tanh(a_{i_1 j}) \tanh(a_{i_2 j}) \cdots \tanh(a_{i_{2p} j})). \end{aligned}$$

Let $B = (b_{ij}) = (\tanh(a_{ij}))$. Denote $B^J = \prod_{i=1}^M \prod_{j=1}^N b_{ij}^{J_{ij}}$ and $x^J = \prod_{i=1}^M x_i^{\sum_{j=1}^N J_{ij}}$, where $J = (J_{ij})$ is an $M \times N$ matrix with $J_{ij} \in \{0,1\}$. Then we have

$$p(x|a) = \frac{2^N \prod_{j=1}^N \prod_{i=1}^M \cosh(a_{ij})}{Z(a)} \sum_{J:\{\{\sum_{i=1}^M J_{ij}\}\}=0 \text{ for all } j} B^J x^J.$$

Let

$$Z(b) = \frac{Z(a)}{2^N \prod_{j=1}^N \prod_{i=1}^M \cosh(a_{ij})}.$$

Set $I = \{I = (I_i) \in \{0,1\}^M | \{\{\sum_{i=1}^M I_i\}\} = 0\}$, and $B^I = \sum_{\substack{J:\{\{\sum_{i=1}^M J_{ij}\}\}=0 \\ \{\{\sum_{j=1}^N J_{ij}\}\}=I_i}} B^J$ for $I \in I$. Then we have

$$p(x|a) = \frac{1}{Z(b)} \sum_{I \in I} B^I x^I$$

and $Z(b) = 2^N B^0$. Since $\sum_{0 \le i \le M/2} \begin{pmatrix} M \\ 2i \end{pmatrix} = ((1+1)^M + (1-1)^M)/2 = 2^{M-1}$, the number of elements in I is 2^{M-1}.

Remark 1 Rusakov and Geiger (2005) obtained λ and θ for the following class of Naive Bayesian networks with two hidden states and binary features:

$$p(x|c,d,t) = t \prod_{i=1}^M c_i^{(1+x_i)/2} (1-c_i)^{(1-x_i)/2} + (1-t) \prod_{i=1}^N d_i^{(1+x_i)/2} (1-d_i)^{(1-x_i)/2}.$$

where $x \in \{1,-1\}^M$, $c = \{c_i\}_{i=1}^M \in \mathbb{R}^M$, $d = \{d_i\}_{i=1}^M \in \mathbb{R}^M$ and $0 \le t \le 1$. Our models with one hidden unit ($N = 1$) are obtained by setting $t = 1/2$, $\tanh(a_i) = 2c_i - 1$ and $d_i = -c_i$. The relation $d_i = -c_i$ creates a parameter space different from that of our models.

Assume that the true distribution is $p(x|a^*)$ with $a^* = (a_{ij}^*)$ and set $B^* = b^* = (b_{ij}^*) = (\tanh(a_{ij}^*))$. Then the Kullback function $K(a)$ is

$$\begin{aligned} &\sum_{x_i=\pm 1} p(x|a^*)(\log p(x|a^*) - \log p(x|a)) = \sum_{x_i=\pm 1} p(x|a^*) \sum_{k=2}^\infty \frac{(-1)^k}{k} (\frac{p(x|a)}{p(x|a^*)} - 1)^k \\ = &\sum_{x_i=\pm 1} \frac{(p(x|a) - p(x|a^*))^2}{p(x|a^*)} (1 + \sum_{k=1}^\infty \frac{(-1)^k}{k+2} (\frac{p(x|a)}{p(x|a^*)} - 1)^k). \end{aligned}$$

Lemma 1 Watanabe, 2001c *If analytic functions K_1, K_2 satisfy $\gamma_1|K_2| \leq |K_1| \leq \gamma_2|K_2|$ for some positive constants γ_1 and γ_2, then the maximum pole and its order of $\int |K_1|^z dw$ are those of $\int |K_2|^z dw$.*

By Lemma 1, since we consider a neighborhood of $\frac{p(x|a)}{p(x|a^*)} = 1$, we only need to obtain the maximum pole of $J(z) = \int \Psi_0^z \mathrm{d}b$, where

$$\Psi_0 = \sum_{x_i=\pm 1} (p(x|a) - p(x|a^*))^2 = \sum_{x_i=\pm 1} (\frac{\sum_{I\in I} B^I x^I}{Z(b)} - \frac{\sum_{I\in I} B^{*I} x^I}{Z(b^*)})^2$$

$$= \sum_{x_i=\pm 1} (\sum_{I\in I} \left(\frac{B^I}{Z(b)} - \frac{B^{*I}}{Z(b^*)} \right) x^I)^2 = 2^M \sum_{I\in I} \left(\frac{B^I}{Z(b)} - \frac{B^{*I}}{Z(b^*)} \right)^2.$$

By Lemma 1 again, we can replace Ψ_0 by

$$\Psi = \sum_{I\in\{0,1\}^M} 2^{2N} (\frac{B^I}{Z(b)} - \frac{B^{*I}}{Z(b^*)})^2 = \sum_{I\in\{0,1\}^M} (\frac{B^I}{B^0} - \frac{B^{*I}}{B^{*0}})^2. \tag{3}$$

4. Main Results

Consider the zeta function $J(z) = \int_V \Psi^z \mathrm{d}b$, where V is a sufficiently small neighborhood of a^*.

From the eigenvalue analysis method, we obtain the following theorem.

Theorem 2 *The average stochastic complexity $F(n)$ in Eq. (1) and the generalization error $G(n)$ in Eq. (2) are given by using the following maximum pole $-\lambda$ of $J(z)$ and its order θ.*

(Case 1)*: If $M = 2$ then $\lambda = 1/2$ and* $\theta = \begin{cases} 2, & \text{if } N = 1, b^* = 0 \\ 1, & \text{otherwise.} \end{cases}$

(Case 2)*: If $M = 3$ then* $\lambda = \begin{cases} 3/4, & \text{if } N = 1, b^* = 0 \\ 1/2, & \text{if } N = 1, b^* \neq 0, \prod_{i=1}^3 b_{i1}^* = 0 \\ 3/2, & \text{if } N = 1, \prod_{i=1}^3 b_{i1}^* \neq 0 \\ 3/2, & \text{if } N \geq 2, \end{cases}$

and $\theta = \begin{cases} 3, & \text{if } N = 2, b^* = 0, \\ 2, & \text{if } N = 2, b^* \neq 0, b_{i_0 j}^* = b_{i_1 j}^* = 0 \text{ for } 1 \leq j \leq N, \\ 2, & \text{if } N = 2, b_{i_0 j_0}^* b_{i_1 j_0}^* \neq 0, b_{i_2 j_0} = b_{ij}^* = 0 \text{ for } 1 \leq i \leq 3, 1 \leq j \leq N, j \neq j_0, \\ 1, & \text{otherwise}, \end{cases}$ *where* $i_0, i_1, i_2 \in \{1,2,3\}$ *are different from each other and* $1 \leq j_0 \leq N$.

For its proof, we use the eigenvalues and the eigenvectors of the matrix $C_j = (c_j^{I,I'})$ where $b_j^I = \prod_{i=1}^M b_{ij}^{I_i}$, and $c_j^{I,I'} = b_j^{I''}$ with $\{\{I' + I''\}\} = I$, for $I, I', I'' \in I$. Its proof appears in Appendix B.

We obtain λ and θ in Eqs. (1) and (2) for $M > N$ using a recursive blowing up.

Theorem 3 *Assume that $M > N$ and $a^* = 0$. The average stochastic complexity $F(n)$ in Eq. (1) and the generalization error $G(n)$ in Eq. (2) are given by using the maximum pole $-\lambda = -\frac{MN}{4}$ of $J(z)$ and its order* $\theta = \begin{cases} 1, & \text{if } M > N+1, \\ M, & \text{if } M = N+1. \end{cases}$

We also bound values of λ for other cases.

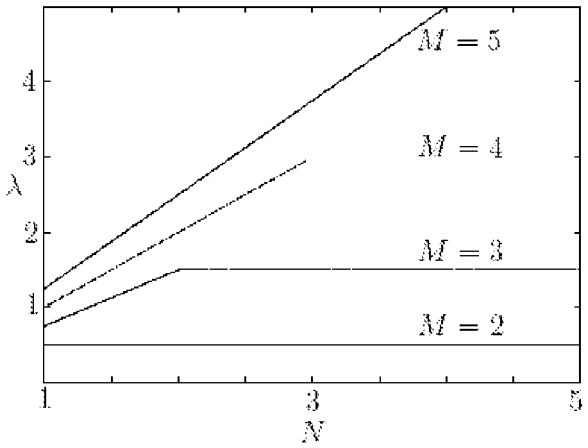


Figure 3: The curve of λ along the y-axis and N along the x-axis, when $M = 2,3,4,5$ and $a^* = 0$.

Theorem 4 *Let* $(a_{1j}, a_{2j}, \cdots, a_{Mj}) \neq 0$ *for* $j = 1, \ldots, N_0$ *and* $(a_{1j}, a_{2j}, \cdots, a_{Mj}) = 0$ *for* $j = N_0 + 1, \ldots, N$ *in* V*, where* V *is a sufficiently small neighborhood of* a^*.

Then we have

$$\begin{aligned} &\frac{M(N-N_0)}{4} \leq \lambda \leq \frac{M(N-N_0)}{4} + \frac{MN_0}{2}, && \text{if } M > N - N_0 \\ \frac{M(M-1)}{4} + \frac{MN_0}{2} \leq \lambda &\leq \frac{2N_0+(M-1)(M-2)}{4} + \frac{MN_0}{2} \left(< \frac{MN_0}{2} + \frac{M(N-N_0)}{4} \right), && \text{if } M \leq N - N_0. \end{aligned}$$

The proofs for these two theorems appear in Appendix C.

5. Conclusion

In this paper, we obtain the generalization error of restricted Boltzmann machines asymptotically (Fig. 3).

We use a new method of eigenvalue analysis and a recursive blowing up in algebraic geometry and show that these are effective for solving the problem in learning theory.

We have not used the eigenvalue analysis method where $M > N$, which is usually the case in applications. Eigenvalue analysis seems to be necessary for solving the behavior of the restricted Boltzmann machine model's generalization error for $M \leq N$.

In this paper, we clarify the generalization error for (i) $M = 3$ (Theorem 2) and (ii) $M > N$, $a^* = 0$ (Theorem 3) explicitly and give new bounds for the generalization error of the other types (Theorem 4). The case (i) shows that λ is independent of a^* for $M - 1 = 2 \leq N$, and so implies that we need more careful consideration for obtaining the exact values λ for the case of Theorem 4.

Our future research aims to improve our methods, and to apply them to the case of Theorem 4 and to obtain the generalization error of the general Boltzmann machine, which is also known as the Bayesian network, the graphical model and the spin model, as such models are widely used in many fields. We believe that extending our results would provide a mathematical foundation for the analysis of various graphical models.

This study involves applying techniques of algebraic geometry to learning theory and it seems that we can contribute to the development of both these fields in the future.

The application of our results is as follows. The results of this paper introduce a mathematical measure of preciseness for numerical calculations such as the Markov Chain Monte Carlo. Using the Markov Chain Monte Carlo (MCMC) method, estimated values for marginal likelihoods had previously been calculated for hyper-parameter estimations and model selection methods of complex learning models, but the theoretical values were not known. The theoretical values of marginal likelihoods have been given in this paper. This enables us to construct a mathematical foundation for analyzing and developing the precision of the MCMC method (Nagata and Watanabe, 2005). Moreover, Nagata and Watanabe (2007) studied the setting of temperatures for the exchange MCMC method and proved the mathematical relation between the symmetrized Kullback function and the exchange ratio, from which an optimal setting of temperatures could be devised. Our theoretical results will be helpful in these numerical experiments. Furthermore, these values have been compared with those of the generalization error of a localized Bayes estimation (Takamatsu, Nakajima, and Watanabe, 2005).

Acknowledgments

This research was supported by the Ministry of Education, Science, Sports and Culture in Japan, Grant-in-Aid for Scientific Research 18079007.

Appendix A. Hironaka's Theorem

We introduce Hironaka's Theorem about the desingularization.

Theorem 5 [Desingularization (Fig. 4)] (Hironaka, 1964)
Let f be a real analytic function in a neighborhood of $w = (w_1, \cdots, w_d) \in \mathbb{R}^d$ with $f(w) = 0$. There exist an open set $V \ni w$, a real analytic manifold U, and a proper analytic map μ from U to V such that
(1) $\mu : U - \mathcal{E} \to V - f^{-1}(0)$ is an isomorphism, where $\mathcal{E} = \mu^{-1}(f^{-1}(0))$,
(2) for each $u \in U$, there is a local analytic coordinate system $(u_1, \cdots, u_n)$ such that $f(\mu(u)) = \pm u_1^{s_1} u_2^{s_2} \cdots u_n^{s_n}$, where $s_1, \cdots, s_n$ are non-negative integers.

Applying Hironaka's theorem to the Kullback function $K(w)$, for each $w \in K^{-1}(0) \cap W$, we have a proper analytic map μ_w from an analytic manifold U_w to a neighborhood V_w of w satisfying Hironaka's Theorem (1) and (2). Then the local integration on V_w of the zeta function $J(z)$ of the learning model is

$$\begin{aligned} J_w(z) &= \int_{V_w} K(w)^z \psi(w) dw \\ &= \int_{U_w} \sum_u (u_1^{2s_1} u_2^{2s_2} \cdots u_d^{2s_d})^z \psi(\mu_w(u)) |\mu_w'(u)| du. \end{aligned} \tag{4}$$

Therefore, the poles of $J_w(z)$ can be obtained. For example, the function

$$\int_{U_0} (u_1^{2s_1} u_2^{2s_2} \cdots u_d^{2s_d})^z u_1^{t_1} u_1^{t_2} \cdots u_1^{t_d} du$$

has the poles $-(t_1+1)/(2s_1), \cdots, -(t_d+1)/(2s_d)$, where U_0 is a small neighborhood of 0. For each $w \in W \setminus K^{-1}(0)$, there exists a neighborhood V_w such that $K(w') \neq 0$, for all $w' \in V_w$. So

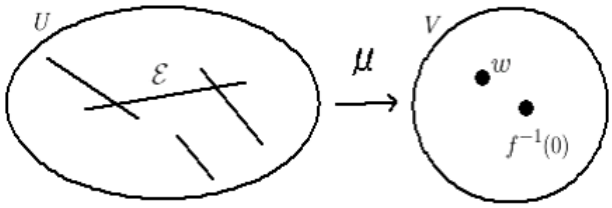


Figure 4: Hironaka's Theorem: This is the picture of a desingularization μ of $f : \mathcal{E}$ maps to $f^{-1}(0)$. $U - \mathcal{E}$ is isomorphic to $V - f^{-1}(0)$ by μ, where V is a small neighborhood of w with $f(w) = 0$.

$J_w(z) = \int_{V_w} K(w)^z \psi(w) dw$ has no poles. It is known that μ of an arbitrary polynomial in Hironaka's Theorem can be obtained by using a blowing up process. Note that the exponents in the integral are $2s_i$ instead of s_i as shown in Eq. (4), since the Kullback function is positive.

In spite of such results, it is still difficult to obtain the generalization error mainly for the following two reasons. (a) The desingularization of any polynomial is in general very difficult, although it is known to be a finite process. Furthermore, most of the Kullback functions of non-regular statistical models are degenerate (over $\mathbb{R}$) with respect to their Newton polyhedrons, which is the condition for using a toric resolution (Fulton, 1993; Watanabe, Hagiwara, Akaho, Motomura, Fukumizu, Okada, and Aoyagi, 2005). Also, points in the singularity set $\{K = \partial K/\partial w = 0\}$ of Kullback functions $K(w)$ are not isolated, and Kullback functions are not simple polynomials, as their number of variables and number of terms grow with parameters, for example, M and N in Eq. (3). It is therefore, a new problem, even in mathematics, to obtain desingularizations of such Kullback functions, since their singularities are very complicated and as such most of them have not yet been investigated. (b) Since our main purpose is to obtain the maximum pole, obtaining a desingularization is not enough. We need techniques for choosing the maximum one from all poles. However, to the best of our knowledge, no theorems for such a purpose have been developed.

We give below Lemmas 2 and 3 in (Aoyagi and Watanabe, 2005b), as they are frequently used in this paper. Define the norm of a matrix $C = (c_{ij})$ by $\|C\| = \sqrt{\sum_{i,j} |c_{ij}|^2}$.

Lemma 6 (Aoyagi and Watanabe, 2005b) *Let U be a neighborhood of $w_0 \in \mathbb{R}^d$, $C(w)$ be an analytic $H \times H'$ matrix function from U, $\psi(w)$ be a C^∞ function from U with compact support, and P and Q be any regular $H \times H$ and $H' \times H'$ matrices, respectively. Then the maximum pole of $\int_U \|C(w)\|^{2z} \psi(w) dw$ and its order are those of $\int_U \|PC(w)Q\|^{2z} \psi(w) dw$.*

Lemma 7 *Assume that $p(x|a) = \frac{\prod_{j=1}^N W_j(x,a)}{\sum_x \prod_{j=1}^N W_j(x,a)}$ for $x \in X$. Then the maximum pole of $\int_U \{\sum_{x \in X} (p(x|a) - p(x|a^*))^2\}^z \psi(w) da$ and its order are those of*

$$\int_U \{ \sum_{x,x' \in X} (\sum_j^N (\log W_j(x,a) - \log W_j(x,a^*) - \log W_j(x',a) + \log W_j(x',a^*))\}^2 \psi(w) dw.$$

(Proof)

Consider the ideal I generated by $p(x|a)-p(x|a^*)$ for $x\in X$.

Then I is generated by $\frac{\prod_{j=1}^N W_j(x,a)}{\prod_{j=1}^N W_j(x,a^*)}-\frac{\sum_x\prod_{j=1}^N W_j(x,a)}{\sum_x\prod_{j=1}^N W_j(x,a^*)}$, and so by $\frac{\prod_{j=1}^N W_j(x,a)}{\prod_{j=1}^N W_j(x,a^*)}-\frac{\prod_{j=1}^N W_j(x',a)}{\prod_{j=1}^N W_j(x',a^*)}$ for $x,x'\in X$.

Since $|x-1|/2\le|\log x|\le 2|x-1|$ for $|x-1|<1/2$, we have

$$\sum_{x,x'\in X}(\frac{\prod_{j=1}^N W_j(x,a)}{\prod_{j=1}^N W_j(x,a^*)}\frac{\prod_{j=1}^N W_j(x',a^*)}{\prod_{j=1}^N W_j(x',a)}-1)^2/4$$

$$\le\sum_{x,x'\in X}(\sum_j^N(\log(W_j(x,a))-\log(W_j(x,a^*))+\log(W_j(x',a^*))-\log(W_j(x',a))))^2$$

$$\le\sum_{x,x'\in X}(\frac{\prod_{j=1}^N W_i(x,a)}{\prod_{j=1}^N W_j(x,a^*)}\frac{\prod_{j=1}^N W_j(x',a^*)}{\prod_{j=1}^N W_j(x',a)}-1)^2 4$$

Q.E.D.

Appendix B. Eigenvalue Analysis

The purpose of eigenvalue analysis is to simplify the blowing up process.

Hierarchical learning machines often have Kullback functions involving a matrix product such as $K(w)=\|D_1D_2\cdots D_N\|^2$, where D_i is a parameter matrix. Therefore, analyzing the eigenvalues of these matrices and applying Lemma 6 sometimes results in an easier function to handle. For example, the restricted Boltzmann machine has a Kullback function $\|\tilde{B}_N\|^2=\|\begin{pmatrix}\mathbf{0} & E\end{pmatrix}C_N\cdots C_2C_1(1,0,\ldots,0)^t\|^2$, where E is the identity matrix (t denotes the transpose). Theorem 9 (4) below shows that analyzing the eigenvalues of C_N makes an easier function $\|R\tilde{B}_N\|^2$ to blow up, where R is a certain regular matrix. This is the main point of this method.

Let $I,I',I''\in I$. We set $B_N^I=B^I$, $b_j^I=\prod_{i=1}^M b_{ij}^{I_i}$, and

$$B_N=(B_N^I)=(B_N^{(0,\ldots,0)},B_N^{(1,1,0,\ldots,0)},B_N^{(1,0,1,0,\ldots,0)},\ldots).$$

We now have $B_N^I=\sum_{\{\{I'+I''\}\}=I}b_N^{I''}B_{N-1}^{I'}$.

For convenience, we denote the "(I,I')th" element of a $2^{M-1}\times 2^{M-1}$ matrix C by $c^{I,I'}$.

Now consider the eigenvalues of the matrix $C_N=(c_N^{I,I'})$ where $c_N^{I,I'}=b_N^{I''}$ with $\{\{I'+I''\}\}=I$.

Note that $B_N=C_NB_{N-1}$.

Let $\ell=(\ell_1,\ldots,\ell_{2^{M-1}})=(\ell_I)\in\{-1,1\}^{2^{M-1}}$ with $\ell_{(0,\ldots,0)}=1$. ℓ is an eigenvector, if and only if

$$\sum_{I'\in I}c_N^{I,I'}\ell_{I'}=\ell_I\sum_{I'\in I}c_N^{(0,\ldots,0),I'}\ell_{I'}=\ell_I\sum_{I'\in I}b_N^{I'}\ell_{I'},\ \text{for all } I\in I.$$

That is,

$$\ell\text{ is an eigenvector}\iff\begin{array}{l}\text{if }\{\{I+I'\}\}=I''\ (\{\{I+I'+I''\}\}=0)\\ \text{then }\ell_{I''}=\ell_I\ell_{I'}\ (\ \ell_I\ell_{I'}\ell_{I''}=1).\end{array}$$

Denote the number of all elements in a set K by $\#K$.

Theorem 8 *Let* $K_1 \subset \{2,\ldots,M\}$. *Set* $\ell_I = \begin{cases} -1, & \text{if } \#\{i \in K_1 : I_i = 1\} \text{ is odd,} \\ 1, & \text{otherwise.} \end{cases}$ *Then* $\ell = (\ell_I)$ *is an eigenvector of* C_N *and its eigenvalue is*

$$\sum_{I \in I} \ell_I b_N^I = \frac{\prod_{i=1}^M (1+x_i b_i) + \prod_{i=1}^M (1-x_i b_i)}{2}, \text{ where } x_i = -1 \text{ if } i \in K_1, \text{ and } x_i = 1 \text{ if } i \notin K_1.$$

Note that $\sum_{I \in I} \ell_I b_N^I > 0$ *since* $b_i = \tanh(a_i)$.

(Proof)

Assume that $\{\{I' + I'' + I'''\}\} = 0$. If all $\#\{i \in K_1 : I'_i = 1\}$, $\#\{i \in K_1 : I''_i = 1\}$ and $\#\{i \in K_1 : I'''_i = 1\}$ are even, then $\ell_{I'}\ell_{I''}\ell_{I'''} = 1$.

If $\#\{i \in K_1 : I'_i = 1\}$ is odd, then $\#\{i \in K_1 : I''_i = 1\}$ or $\#\{i \in K_1 : I'''_i = 1\}$ is odd, since $\{\{I' + I'' + I'''\}\} = 0$.

If $\#\{i \in K_1 : I'_i = 1\}$ and $\#\{i \in K_1 : I''_i = 1\}$ are odd, then $\#\{i \in K_1 : I'''_i = 1\}$ is even and $\ell_{I'}\ell_{I''}\ell_{I'''} = 1$ since $\{\{I' + I'' + I'''\}\} = 0$.

Q.E.D.

We have 2^{M-1} eigenvectors ℓ. Moreover, they are orthogonal to each other, since the eigenvectors of a symmetric matrix are orthogonal. These eigenvectors ℓ's, therefore, span the whole space $\mathbb{R}^{2^{M-1}}$.

Set $\mathbf{1} = (1,\ldots,1)^t \in \mathbb{Z}^{2^{M-1}-1}$ (t denotes the transpose). Let $D = (D^{I,I'})$ be a symmetric matrix formed by arranging the eigenvectors ℓ's such that $D = \begin{pmatrix} 1 & \mathbf{1}^t \\ \mathbf{1} & D' \end{pmatrix}$ and $DD = 2^{M-1}E$, where E is the identity matrix and $D^{I,I'}$ is "(I,I')th" element of D.

Since $DD = \begin{pmatrix} 2^{M-1} & \mathbf{1}^t D' \\ \mathbf{1} + D'\mathbf{1} & \mathbf{1}\mathbf{1}^t + D'D' \end{pmatrix} = 2^{M-1}E$, we have $D'\mathbf{1} = -\mathbf{1}$.

Let $C'_N = DC_N D/2^{M-1} = DC_N D^{-1} = \begin{pmatrix} s_N^0 & 0 & 0 & \cdots & 0 \\ 0 & s_N^1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & s_N^{2^{M-1}-1} \end{pmatrix}$.

We use s_N^i or s_N^I ($I \in I$), depending on the situation.

Since $C_N = D^{-1}C'_N D$, we have $b_N^{\{\{I+K\}\}} = \sum_{J \in I} D^{I,J} s_N^J D^{J,K}/2^{M-1}$.

B.1 Example

Let $M = 4$.

We have the matrix by arranging the eigenvectors of C_N,

$$D = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 \\ 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \end{pmatrix}$$ and the eigenvalues

$$\begin{aligned}
s_N^0 &= 1 + b_{1N}b_{2N} + b_{1N}b_{3N} + b_{1N}b_{4N} + b_{2N}b_{3N} + b_{2N}b_{4N} + b_{3N}b_{4N} + b_{1N}b_{2N}b_{3N}b_{4N}, \\
s_N^1 &= 1 + b_{2N}b_{3N} + b_{2N}b_{4N} + b_{3N}b_{4N} - b_{1N}(b_{2N} + b_{3N} + b_{4N} + b_{2N}b_{3N}b_{4N}), \\
s_N^2 &= 1 + b_{1N}b_{3N} + b_{1N}b_{4N} + b_{3N}b_{4N} - b_{2N}(b_{1N} + b_{3N} + b_{4N} + b_{1N}b_{3N}b_{4N}), \\
s_N^3 &= 1 + b_{1N}b_{3N} + b_{2N}b_{4N} + b_{1N}b_{2N}b_{3N}b_{4N} - (b_{1N} + b_{3N})(b_{2N} + b_{4N}), \\
s_N^4 &= 1 + b_{1N}b_{2N} + b_{3N}b_{4N} + b_{1N}b_{2N}b_{3N}b_{4N} - (b_{1N} + b_{2N})(b_{3N} + b_{4N}), \\
s_N^5 &= 1 + b_{1N}b_{2N} + b_{1N}b_{4N} + b_{2N}b_{4N} - b_{3N}(b_{1N} + b_{2N} + b_{4N} + b_{1N}b_{2N}b_{4N}), \\
s_N^6 &= 1 + b_{1N}b_{2N} + b_{1N}b_{3N} + b_{2N}b_{3N} - b_{4N}(b_{1N} + b_{2N} + b_{3N} + b_{1N}b_{2N}b_{3N}), \\
s_N^7 &= 1 + b_{1N}b_{4N} + b_{2N}b_{3N} + b_{1N}b_{2N}b_{3N}b_{4N} - (b_{1N} + b_{4N})(b_{2N} + b_{3N}).
\end{aligned}$$

Theorem 9 *Let $H = 2^{M-1} - 1$.*

(1) *Let* $d_{ij} = \begin{cases} 1, & \text{if } i = 1 \text{ or } j = 1, \\ D^{I,J}, & \text{if } I = (1, 0, \ldots, 0, \overset{i}{1}, 0, \ldots, 0) \\ & \text{and } J = (1, 0, \ldots, 0, \overset{j}{1}, 0, \ldots, 0). \end{cases}$

Then $D^{I,J} = \prod_{i,j:I_i=1,J_j=1} d_{ij}$ for all $I, J \in \mathcal{I}$.

(2) $B_N = C_N B_{N-1} = C_N \cdots C_2 B_1 = DC'_N \cdots C'_2 D^{-1} B_1 = \dfrac{DC'_N \cdots C'_1 \mathbf{1}}{2^{M-1}}$.

(3) *We have* $2^{M-1} D'^{-1} = D' - \mathbf{1}\mathbf{1}^t$.

(4) *Let* $\tilde{B}_1 = (B_1^I)_{I \neq 0}$, $\tilde{B}_N = (B_N^I)_{I \neq 0}$ *and*

$$S = -\frac{1}{H+1} \begin{pmatrix} \prod_{j=1}^N s^{*1}_j - \prod_{j=1}^N s^{*0}_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j - \prod_{j=1}^N s^{*0}_j \end{pmatrix} \begin{pmatrix} \prod_{j=2}^N s^1_j - \prod_{j=2}^N s^0_j & \cdots & \prod_{j=2}^N s^H_j - \prod_{j=2}^N s^0_j \end{pmatrix}$$
$$+ B^{*0}_N \prod_{j=2}^N s^0_j \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \vdots & \vdots \\ 1 & \cdots & 1 \end{pmatrix} + B^{*0}_N \begin{pmatrix} \prod_{j=2}^N s^1_j & 0 & 0 & \cdots & 0 \\ 0 & \prod_{j=2}^N s^2_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix}.$$

We have

$$(\det S) D'^{-1} S^{-1} D'^{-1} 2^{M-1} (\tilde{B}_N B^{*0}_N - \tilde{B}^*_N B^0_N)$$

$$= (\det S)\tilde{B}_1 - (B^{*0}_N)^{H-1}(\mathbf{1}\ D')\begin{pmatrix} \prod_{j=1}^N s^{*0}_j \prod_{i\neq 0}\prod_{j=2}^N s^i_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \prod_{i\neq H}\prod_{j=2}^N s^i_j \end{pmatrix}.$$

(5) *The corresponding element to* I *of* $(\mathbf{1}\ D')\begin{pmatrix} \prod_{i\neq 0}\prod_{j=2}^N s^i_j \\ \vdots \\ \prod_{i\neq H}\prod_{j=2}^N s^i_j \end{pmatrix}$ *consists of monomials* $c_J \prod_{i=1}^M \prod_{j=2}^N b_{ij}^{J_{ij}}$, *where* $c_J \in \mathbb{R}, 0 \leq J_{ij} \in \mathbb{Z}$ *and* $\{\{\sum_{j=1}^N J_{ij}\}\} = I_i$.

(Proof)
(1) Fix $K_1 \subset \{2,\ldots,M\}$.
Consider the eigenvector ℓ defined by using K_1.
Set $d'_1 = 1$ and $d'_i = \ell_I$ for $I = (1,\ 0,\ \ldots,\ 0,\ \overset{i}{1},\ 0,\ \ldots,\ 0)$, $i \geq 2$.
Since $\ell_I = \prod_{i\in K_1 : I_i = 1}(-1) = \prod_{i:I_i=1} d'_i$ and D is symmetric, we have statement (1).
(2) is obvious.
(3) Since $DD = \begin{pmatrix} 2^{M-1} & \mathbf{1}^t D' \\ \mathbf{1}+D'\mathbf{1} & \mathbf{1}\mathbf{1}^t + D'D' \end{pmatrix} = 2^{M-1}E$, we have $D'D' = 2^{M-1}E' - \mathbf{1}\mathbf{1}^t$ and $D'(D' - \mathbf{1}\mathbf{1}^t) = 2^{M-1}E' - \mathbf{1}\mathbf{1}^t - D'\mathbf{1}\mathbf{1}^t = 2^{M-1}E' - \mathbf{1}\mathbf{1}^t + \mathbf{1}\mathbf{1}^t = 2^{M-1}E'$, where E' is the identity matrix.
(4)

$$2^{M-1}(\tilde{B}_N B^{*0}_N - \tilde{B}^*_N B^0_N) = 2^{M-1}\begin{pmatrix} -\tilde{B}^*_N & B^{*0}_N E \end{pmatrix} B_N$$
$$= \begin{pmatrix} -\tilde{B}^*_N & B^{*0}_N E \end{pmatrix} D \begin{pmatrix} \prod_{j=2}^N s^0_j & 0 & 0 & \cdots & 0 \\ 0 & \prod_{j=2}^N s^1_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix} DB_1$$

$$= (-\tilde{B}^*_N \begin{pmatrix} 1 & \mathbf{1}^t \end{pmatrix} + B^{*0}_N \begin{pmatrix} \mathbf{1} & D' \end{pmatrix})$$
$$\begin{pmatrix} \prod_{j=2}^N s^0_j & 0 & 0 & \cdots & 0 \\ 0 & \prod_{j=2}^N s^1_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix} DB_1$$

$$= (\frac{-\begin{pmatrix} \mathbf{1} & D' \end{pmatrix}}{H+1} \begin{pmatrix} \prod_{j=1}^N s^{*0}_j \\ \prod_{j=1}^N s^{*1}_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \end{pmatrix} \begin{pmatrix} 1 & \mathbf{1}^t \end{pmatrix} + B^{*0}_N \begin{pmatrix} \mathbf{1} & D' \end{pmatrix})$$
$$\begin{pmatrix} \prod_{j=2}^N s^0_j & 0 & 0 & \cdots & 0 \\ 0 & \prod_{j=2}^N s^1_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix} DB_1$$

$$= \begin{pmatrix} \mathbf{1} & D' \end{pmatrix} (-\frac{1}{H+1} \begin{pmatrix} \prod_{j=1}^N s^{*0}_j \\ \prod_{j=1}^N s^{*1}_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \end{pmatrix} \begin{pmatrix} \prod_{j=2}^N s^0_j & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix}$$
$$+ B^{*0}_N \begin{pmatrix} \prod_{j=2}^N s^0_j & 0 & 0 & \cdots & 0 \\ 0 & \prod_{j=2}^N s^1_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{j=2}^N s^H_j \end{pmatrix}) (\begin{pmatrix} 1 \\ \mathbf{1} \end{pmatrix} + \begin{pmatrix} \mathbf{1}^t \\ D' \end{pmatrix} \tilde{B}_1)$$

$$= D' (-T^0 \begin{pmatrix} \prod_{j=1}^N s^{*1}_j - \prod_{j=1}^N s^{*0}_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j - \prod_{j=1}^N s^{*0}_j \end{pmatrix} + B^{*0}_N \begin{pmatrix} \prod_{j=2}^N s^1_j - \prod_{j=2}^N s^0_j \\ \vdots \\ \prod_{j=2}^N s^H_j - \prod_{j=2}^N s^0_j \end{pmatrix}) + D' S D' \tilde{B}_1,$$

where $T^0 = \frac{\prod_{j=2}^N s^0_j + \cdots + \prod_{j=2}^N s^H_j}{H+1}$.

Also we have $S^{-1}_{i_1 j_1} = (\det S)^{-1} \times$

$$\begin{cases} \frac{(B^{*0}_N)^{H-2}}{H+1} \sum_{i_2=0, i_2 \neq i_1}^H (\prod_{j=1}^N s^{*i_1}_j + H \prod_{j=1}^N s^{*i_2}_j) \prod_{0 \leq i \leq H, i \neq i_1, i_2} \prod_{j=2}^N s^i_j, & \text{if } i_1 = j_1, \\ \frac{(B^{*0}_N)^{H-2}}{H+1} \sum_{0 \leq i_2 \leq H, i_2 \neq i_1, j_1} (\prod_{j=1}^N s^{*i_1}_j - \prod_{j=1}^N s^{*i_2}_j) \prod_{0 \leq i \leq H, i \neq i_1, i_2} \prod_{j=2}^N s^i_j & \\ -\frac{(B^{*0}_N)^{H-2}}{H+1} (H \prod_{j=1}^N s^{*i_1}_j + \prod_{j=1}^N s^{*j_1}_j) \prod_{0 \leq i \leq H, i \neq i_1, j_1} \prod_{j=2}^N s^i_j, & \text{if } i_1 \neq j_1 \end{cases}$$

and $\det S = (B^{*0}_N)^{H-1} \sum_{i_2=0}^H \prod_{j=1}^N s^{*i_2}_j \prod_{i \neq i_2} \prod_{j=2}^N s^i_j$.

Let $\mathbf{s} = \begin{pmatrix} \prod_{j=1}^N s^{*0}_j \prod_{i \neq 0} \prod_{j=2}^N s^i_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \prod_{i \neq H} \prod_{j=2}^N s^i_j \end{pmatrix}$ and $\tilde{\mathbf{s}} = \begin{pmatrix} \prod_{j=1}^N s^{*1}_j \prod_{i \neq 1} \prod_{j=2}^N s^i_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \prod_{i \neq H} \prod_{j=2}^N s^i_j \end{pmatrix}$.

Since $(\det S) S^{-1} (-T^0 \begin{pmatrix} \prod_{j=1}^N s^{*1}_j - \prod_{j=1}^N s^{*0}_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j - \prod_{j=1}^N s^{*0}_j \end{pmatrix} + B^{*0}_N \begin{pmatrix} \prod_{j=2}^N s^1_j - \prod_{j=2}^N s^0_j \\ \vdots \\ \prod_{j=2}^N s^H_j - \prod_{j=2}^N s^0_j \end{pmatrix})$

$$= (B^{*0}_N)^{H-1} \{ \sum_{i_2=0}^H \prod_{j=1}^N s^{*i_2}_j \prod_{i \neq i_2} \prod_{j=2}^N s^i_j \mathbf{1} - (H+1) \begin{pmatrix} \prod_{j=1}^N s^{*1}_j \prod_{i \neq 1} \prod_{j=2}^N s^i_j \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \prod_{i \neq H} \prod_{j=2}^N s^i_j \end{pmatrix} \},$$

we have

$$\begin{aligned} & D'^{-1} S^{-1} D'^{-1} 2^{M-1} (\tilde{B}_N B^{*0}_N - \tilde{B}^*_N B^0_N) \\ &= (\det S) \tilde{B}_1 - (B^{*0}_N)^{H-1} \sum_{i_2=0}^H \prod_{j=1}^N s^{*i_2}_j \prod_{i \neq i_2} \prod_{j=2}^N s^i_j \mathbf{1} - (H+1)(B^{*0}_N)^{H-1} D'^{-1} \tilde{\mathbf{s}} \\ &= (\det S) \tilde{B}_1 - (B^{*0}_N)^{H-1} \sum_{i_2=0}^H \prod_{j=1}^N s^{*i_2}_j \prod_{i \neq i_2} \prod_{j=2}^N s^i_j \mathbf{1} - (B^{*0}_N)^{H-1} (D' - \mathbf{1}\mathbf{1}^t) \tilde{\mathbf{s}} \\ &= (\det S) \tilde{B}_1 - (B^{*0}_N)^{H-1} \prod_{j=1}^N s^{*0}_j \prod_{i \neq 0} \prod_{j=2}^N s^i_j \mathbf{1} - (B^{*0}_N)^{H-1} D' \tilde{\mathbf{s}} \\ &= (\det S) \tilde{B}_1 - (B^{*0}_N)^{H-1} (\mathbf{1}, D') \mathbf{s}, \end{aligned}$$

by using (3).
(5) Since $b_j^{\{\{I+K\}\}} = \sum_{J\in I} D^{I,J} s_j^J D^{J,K}/2^{M-1}$, we have for $I' \in I$,

$$\begin{aligned}
&\sum_{J\in I} D^{I,J} s_j^{\{\{J+I'\}\}} D^{J,K} = D^{I,I'} D^{I',K} \sum_{J\in I} D^{I,\{\{J+I'\}\}} s_j^{\{\{J+I'\}\}} D^{\{\{J+I'\}\},K} \\
= \;& 2^{M-1} D^{I,I'} D^{I',K} b_j^{\{\{I+K\}\}},
\end{aligned}$$

by using (1).

Let $I_0 = (0,\ldots,0), I_1 = (1,1,0,\ldots,0), I_2 = (1,0,1,0,\ldots,0),\ldots$.

The fact that

$$\begin{aligned}
& D \begin{pmatrix} \prod_{i\neq 0}\prod_{j=2}^N s_j^i \\ \prod_{i\neq 1}\prod_{j=2}^N s_j^i \\ \vdots \\ \prod_{i\neq H}\prod_{j=2}^N s_j^i \end{pmatrix} \\
= \;& -D \begin{pmatrix} \prod_{i\neq 0}\prod_{j=2}^N s_j^i & 0 & 0 & \cdots & 0 \\ 0 & \prod_{i\neq 1}\prod_{j=2}^N s_j^i & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \prod_{i\neq H}\prod_{j=2}^N s_j^i \end{pmatrix} D^{-1} 2^{M-1} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\
= \;& -\left\{ \prod_{j=2}^N \prod_{0\neq I'\in I} D \begin{pmatrix} s_j^{\{\{I_0+I'\}\}} & 0 & 0 & \cdots & 0 \\ 0 & s_j^{\{\{I_1+I'\}\}} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & s_j^{\{\{I_H+I'\}\}} \end{pmatrix} D^{-1} \right\} 2^{M-1} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix},
\end{aligned}$$

and $\sum_{J\in I} D^{I,J} s_j^{\{\{J+I'\}\}} D^{J,K} = 2^{M-1} D^{I,I'} D^{I',K} b_j^{\{\{I+K\}\}}$ yields statement (5).

Q.E.D.

Proof of Theorem 2

By Theorem 9 (4) and Lemma 6, we only need to consider the maximum pole of $J(z) = \int \|\Psi'\|^{2z} \mathrm{d}b$, where $\Psi' = (\det S)\tilde{B}_1 - (B^{*0}_N)^{H-1}(\mathbf{1}\ D') \begin{pmatrix} \prod_{j=1}^N s^{*0}_j \prod_{i\neq 0}\prod_{j=2}^N s_j^i \\ \vdots \\ \prod_{j=1}^N s^{*H}_j \prod_{i\neq H}\prod_{j=2}^N s_j^i \end{pmatrix}$.

(Case 1): The fact that $B^{11} = \sum_{k=1}^N b_{1k}b_{2k} + \cdots$ provides Case 1.

(Case 2): Assume that $M = 3$.

We have $D' = \begin{pmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}$, $\begin{cases} s_j^0 = 1 + b_{1j}b_{2j} + b_{1j}b_{3j} + b_{2j}b_{3j}, \\ s_j^1 = 1 + b_{1j}b_{2j} - b_{1j}b_{3j} - b_{2j}b_{3j}, \\ s_j^2 = 1 - b_{1j}b_{2j} + b_{1j}b_{3j} - b_{2j}b_{3j}, \\ s_j^3 = 1 - b_{1j}b_{2j} - b_{1j}b_{3j} + b_{2j}b_{3j}, \end{cases}$

and $\Psi' = (\det S) \begin{pmatrix} b_{11}b_{21} \\ b_{11}b_{31} \\ b_{21}b_{31} \end{pmatrix} - \prod_{i=0}^3 \prod_{j=2}^N s_j^i (B^{*0}_N)^2 (1, D') \begin{pmatrix} \prod_{j=1}^N s^{*0}_j / \prod_{j=2}^N s_j^0 \\ \prod_{j=1}^N s^{*1}_j / \prod_{j=2}^N s_j^1 \\ \prod_{j=1}^N s^{*2}_j / \prod_{j=2}^N s_j^2 \\ \prod_{j=1}^N s^{*3}_j / \prod_{j=2}^N s_j^3 \end{pmatrix}$.

Let $N=1$. The fact that $\Psi' = 4(B^{*0}_N)^2\tilde{B}_1 - 4(B^{*0}_N)^2 \begin{pmatrix} b^*_{11}b^*_{21} \\ b^*_{11}b^*_{31} \\ b^*_{21}b^*_{31} \end{pmatrix}$ yields the statement for $N=1$.

Assume that $N \geq 2$, $b^* \neq 0$ and $b^*_{11} \neq 0, b^*_{21} \neq 0, b^*_{31} \neq 0$. Set $b'_{21} = b_{11}b_{21}$, $b'_{31} = b_{11}b_{31}$ and $b'_{11} = b_{21}b_{31} = b'_{21}b'_{31}/b^2_{11}$. Then

$$\Psi' = (\det S)\begin{pmatrix} b'_{21} \\ b'_{31} \\ b'_{11} \end{pmatrix} - \prod_{i=0}^{3}\prod_{j=2}^{N} s^i_j (B^{*0}_N)^2 (\mathbf{1}, D') \begin{pmatrix} \prod_{j=1}^N s^{*0}_j / \prod_{j=2}^N s^0_j \\ \prod_{j=1}^N s^{*1}_j / \prod_{j=2}^N s^1_j \\ \prod_{j=1}^N s^{*2}_j / \prod_{j=2}^N s^2_j \\ \prod_{j=1}^N s^{*3}_j / \prod_{j=2}^N s^3_j \end{pmatrix}$$

and its maximum pole is $3/2$ and its order is 1.

Assume that $N \geq 2$, $b^* \neq 0$, $b^*_{11} \neq 0$ and $\prod_{i=1}^3 b^*_{ij} = 0$ for all j. Let $\psi = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = (\mathbf{1}, D') \begin{pmatrix} \prod_{j=1}^N s^{*0}_j / \prod_{j=2}^N s^0_j \\ \prod_{j=1}^N s^{*1}_j / \prod_{j=2}^N s^1_j \\ \prod_{j=1}^N s^{*2}_j / \prod_{j=2}^N s^2_j \\ \prod_{j=1}^N s^{*3}_j / \prod_{j=2}^N s^3_j \end{pmatrix}$. By setting $\begin{pmatrix} b'_{21} \\ b'_{31} \end{pmatrix} = (\det S)\begin{pmatrix} b_{11}b_{21} \\ b_{11}b_{31} \end{pmatrix}$

$$-\prod_{i=0}^3 \prod_{j=2}^N s^i_j (B^{*0}_N)^2 \begin{pmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \begin{pmatrix} \prod_{j=1}^N s^{*0}_j / \prod_{j=2}^N s^0_j \\ \prod_{j=1}^N s^{*1}_j / \prod_{j=2}^N s^1_j \\ \prod_{j=1}^N s^{*2}_j / \prod_{j=2}^N s^2_j \\ \prod_{j=1}^N s^{*3}_j / \prod_{j=2}^N s^3_j \end{pmatrix} \text{ and}$$

$$\Psi'' = \begin{pmatrix} \Psi''_1 \\ \Psi''_2 \\ \Psi''_3 \end{pmatrix} = \begin{pmatrix} b'_{21} \\ b'_{31} \\ \frac{(\prod_{i=0}^3 \prod_{j=2}^N s^i_j (B^{*0}_N)^2)^2 \psi_1 \psi_2}{b^2_{11} \det S} - (B^{*0}_N)^2 \prod_{i=0}^3 \prod_{j=2}^N s^i_j \psi_3 \end{pmatrix},$$

and by using Lemma 6, we need the maximum pole of $\int \|\Psi''\|^{2z} db$. Ψ'' is singular in the following cases: (i) $b^*_{11}b^*_{21} = b^*_{2j} = b^*_{3j} = 0$ for all j, (ii) $b^*_{11}b^*_{21} \neq 0$, $b^*_{1j} = b^*_{2j} = b^*_{3j} = 0$ for all j, since we have $\frac{\partial \psi}{\partial b_j}|_{b^*}$

$$= -(\mathbf{1}, D') \begin{pmatrix} s^{*0}_1/s^{*0}_j & 0 & 0 & 0 \\ 0 & s^{*1}_1/s^{*1}_j & 0 & 0 \\ 0 & 0 & s^{*2}_1/s^{*2}_j & 0 \\ 0 & 0 & 0 & s^{*3}_1/s^{*3}_j \end{pmatrix} \begin{pmatrix} b^*_{2j}+b^*_{3j} & b^*_{1j}+b^*_{3j} & b^*_{1j}+b^*_{2j} \\ b^*_{2j}-b^*_{3j} & b^*_{1j}-b^*_{3j} & -b^*_{1j}-b^*_{2j} \\ -b^*_{2j}+b^*_{3j} & -b^*_{1j}-b^*_{3j} & b^*_{1j}-b^*_{2j} \\ -b^*_{2j}-b^*_{3j} & -b^*_{1j}+b^*_{3j} & -b^*_{1j}+b^*_{2j} \end{pmatrix}.$$ If

Ψ'' is not singular, its maximum pole is $3/2$ and its order is 1. Assume that Ψ'' is singular that is, (i) $b^*_{11}b^*_{21} = b^*_{2j} = b^*_{3j} = 0$ for all j and (ii) $b^*_{11}b^*_{21} \neq 0, b^*_{1j} = b^*_{2j} = b^*_{3j} = 0$ for all j. Construct the blow-up of Ψ'' along the submanifold $\{b_{3j} = 0, 2 \leq j \leq N\}$. Let $b_{32} = u$ and $b_{3j} = ub'_{3j}$ for $j \geq 2$. In the case (i), the coefficient of b_{2j_0} is around $4ub_{1j_0}\sum_{j=2}^N b_{1j}b'_{3j}(1/b^2_{11} - 1) + 4ub'_{3j_0}(1 - b^2_{1j_0})$, since $\prod_{i\neq 0}\prod_{j=2}^N s^i_j \cong \prod_{j=2}^N (1 - b_{1j}b_{2j} - ub_{1j}b'_{3j} - ub_{2j}b'_{3j} + 2ub^2_{1j}b_{2j}b'_{3j}) \cong 1 + \sum_{j=2}^N (-b_{1j}b_{2j} - ub_{1j}b'_{3j} - ub_{2j}b'_{3j} + 2ub^2_{1j}b_{2j}b'_{3j}) + \sum_{j\neq j'} ub_{1j}b_{1j'}b_{2j}b'_{3j'}$, $\prod_{i=0}\prod_{j=2}^N s^i_j \psi_1 \cong -4\sum_{j=2}^N b_{1j}b_{2j}$, $\prod_{i=0}\prod_{j=2}^N s^i_j \psi_2 \cong -4u\sum_{j=2}^N b_{1j}b'_{3j}$, and $\prod_{i=0}\prod_{j=2}^N s^i_j \psi_3 \cong 4\sum_{j=2}^N (-ub_{2j}b'_{3j} + 2ub^2_{1j}b_{2j}b'_{3j}) + 4\sum_{j\neq j'} ub_{1j}b_{1j'}b_{2j}b'_{3j'}$. If $4ub_{1j_0}\sum_{j=2}^N b_{1j}b'_{3j}(1/b^2_{11} - 1) +$

$4ub'_{3j_0}(1-b_{1j_0}^2)=0$ for all j_0 then $b'_{3j_0}=0$ for all j_0 since $|b_{1j}|<1$. It contradicts $b'_{32}=1$. So $(\frac{(\prod_{i=0}^{3}\prod_{j=2}^{N}s_j^i(B^{*0}_N)^2)^2\psi_1\psi_2}{b_{11}^2\det S}-(B^{*0}_N)^2\prod_{i=0}^{3}\prod_{j=2}^{N}s_j^i\psi_3)/u$ is smooth.

In the case (ii), the coefficient b_{2j_0} is around $4u(1-{b^*_{21}}^2)b'_{3j_0}$ since ${s^*}^0_1\prod_{i\neq 0}\prod_{j=2}^{N}s_j^i\cong 4(1+b^*_{11}b^*_{21})\prod_{j=2}^{N}(1-ub_{2j}b'_{3j})\cong 4(1+b^*_{11}b^*_{21})(1-u\sum_{j=2}^{N}b_{2j}b'_{3j})$, $(1+b^*_{11}b^*_{21})\prod_{i=0}\prod_{j=2}^{N}s_j^i\psi_1\cong 4b^*_{11}b^*_{21}$, $\prod_{i=0}\prod_{j=2}^{N}s_j^i\psi_2\cong -4ub^*_{11}b^*_{21}\sum_{j=2}^{N}b_{2j}b'_{3j}$, and $\prod_{i=0}\prod_{j=2}^{N}s_j^i\psi_3\cong -4u\sum_{j=2}^{N}b_{2j}b'_{3j}$. So $(\frac{(\prod_{i=0}^{3}\prod_{j=2}^{N}s_j^i(B^{*0}_N)^2)^2\psi_1\psi_2}{b_{11}^2\det S}-(B^{*0}_N)^2\prod_{i=0}^{3}\prod_{j=2}^{N}s_j^i\psi_3)/u$ is smooth.

We have $\Psi''=\begin{pmatrix} b'_{21}\\ b'_{31}\\ ub'_{22}\end{pmatrix}$, for a variable b'_{22} for both cases (i) and (ii) and we have the statement for $N\geq 2$, $b^*\neq 0$, $b^*_{11}\neq 0$ and $\prod_{i=1}^{3}b^*_{ij}=0$ for all j.

Let $N\geq 2$ and $b^*=0$.

Construct the blow-up of Ψ' along the submanifold $\{b_{ij}=0, 1\leq i\leq M, 1\leq j\leq N\}$.

Let $b_{11}=u$ and $b_{ij}=ub'_{ij}$ for $(i,j)\neq(1,1)$.

We have $\Psi''=u^2(\det S)\begin{pmatrix} b'_{21}\\ b'_{31}\\ b'_{21}b'_{31}\end{pmatrix}+4u^2\begin{pmatrix} \sum_{k=2}^{N}b'_{1k}b'_{2k}+u^2f_1\\ \sum_{k=2}^{N}b'_{1k}b'_{3k}+u^2f_2\\ \sum_{k=2}^{N}b'_{2k}b'_{3k}+u^2f_3\end{pmatrix}$, where f_1, f_2 and f_3 are polynomials of b'_{ij} of at least degree two.

By setting $\begin{pmatrix} b''_{21}\\ b''_{31}\end{pmatrix}=\begin{pmatrix} b'_{21}\\ b'_{31}\end{pmatrix}+4\begin{pmatrix} \sum_{k=2}^{N}b'_{1k}b'_{2k}+u^2f_1\\ \sum_{k=2}^{N}b'_{1k}b'_{3k}+u^2f_2\end{pmatrix}/(\det S)$, we have

$$\begin{aligned}\Psi''=&\frac{u^2}{\det S}\\ &\times\begin{pmatrix} (\det S)^2b''_{21}\\ (\det S)^2b''_{31}\\ (b''_{21}\det S-4\sum_{k=2}^{N}b'_{1k}b'_{2k}-4u^2f_1)(b''_{31}\det S-4\sum_{k=2}^{N}b'_{1k}b'_{3k}-4u^2f_2)\end{pmatrix}\\ &+u^2\begin{pmatrix} 0\\ 0\\ 4\sum_{k=2}^{N}b'_{2k}b'_{3k}+4u^2f_3\end{pmatrix}.\end{aligned}$$

By using Lemma 6 again, the maximum pole of $\int\|\Psi''\|^{2z}u^{3N}db$ is that of $J(z)=\int\|\Psi'''\|^{2z}u^{3N}\mathrm{d}b$, where $\Psi'''=u^2\begin{pmatrix} b''_{21}\\ b''_{31}\\ g_1\end{pmatrix}$, and

$$g_1=(\sum_{k=2}^{N}b'_{1k}b'_{2k}+u^2f_1)(\sum_{k=2}^{N}b'_{1k}b'_{3k}+u^2f_2)+\frac{\det S}{4}(\sum_{k=2}^{N}b'_{2k}b'_{3k}+u^2f_3).$$

Construct the blow-up of Ψ''' along the submanifold $\{b''_{21}=0, b''_{31}=0, b'_{3k}=0, 2\leq k\leq N\}$. Then we have cases (I) and (II).

(I) Let $b'_{32}=v$, $b''_{21}=vb'''_{21}$, $b''_{31}=vb'''_{21}$ and $b'_{3k}=vb''_{3k}$ for $3\leq k\leq N$. Then $\Psi'''=u^2v\begin{pmatrix} b'''_{21}\\ b'''_{31}\\ g'_1\end{pmatrix}$, where $g'_1=(\sum_{k=2}^{N}b'_{1k}b'_{2k}+u^2f_1)(b'_{12}+\sum_{k=3}^{N}b'_{1k}b''_{3k}+u^2f_2/v)+\frac{\det S}{4}(b'_{22}+\sum_{k=3}^{N}b'_{2k}b''_{3k}+u^2f_3/v)$.

By Theorem 9 (5), we can set $f_2 = vf_2'$ and $f_3 = vf_3'$, where f_2' and f_3' are polynomials.
We have

$$(\sum_{k=2}^{N} b'_{1k}b'_{2k})(b'_{12}+\sum_{k=3}^{N} b'_{1k}b''_{3k})+\frac{\det S}{4}(b'_{22}+\sum_{k=3}^{N} b'_{2k}b''_{3k})$$

$$=(b'_{2,2},b'_{2,3},\cdots,b'_{2,N})\left(\begin{pmatrix} b'_{1,2}\\ b'_{1,3}\\ \vdots\\ b'_{1,N}\end{pmatrix}(b'_{1,2},b'_{1,3},\cdots,b'_{1,N})+\frac{\det S}{4}E\right)\begin{pmatrix} 1\\ b''_{3,3}\\ \vdots\\ b''_{3,N}\end{pmatrix}.$$

Since $\begin{pmatrix} b'_{1,2}\\ b'_{1,3}\\ \vdots\\ b'_{1,N}\end{pmatrix}(b'_{1,2},b'_{1,3},\cdots,b'_{1,N})+\frac{\det S}{4}E$ is regular, we can change variables from $(b'_{2,2},b'_{2,3},\cdots,b'_{2,N})$ to $(b''_{2,2},b''_{2,3},\cdots,b''_{2,N})$ by

$$(b''_{2,2},b''_{2,3},\cdots,b''_{2,N})=(b'_{2,2},b'_{2,3},\cdots,b'_{2,N})\left(\begin{pmatrix} b'_{1,2}\\ b'_{1,3}\\ \vdots\\ b'_{1,N}\end{pmatrix}(b'_{1,2},b'_{1,3},\cdots,b'_{1,N})+\frac{\det S}{4}E\right).$$

Moreover, let $b'''_{22}=b''_{2,2}+b''_{2,3}b''_{3,3}+\cdots+b''_{2,N}b''_{3,N}$.

Then, we have

$$\Psi''' = u^2 v\begin{pmatrix} b'''_{21}\\ b'''_{31}\\ b'''_{22}+u^2 f_4\end{pmatrix},$$

where f_4 is a polynomial. Therefore, we have the poles $-\frac{3N}{4}, -\frac{N+1}{2}$ and $-\frac{3}{2}$.
(II) Let $b''_{21}=v$, $b''_{31}=vb'''_{21}$ and $b'_{3k}=vb''_{3k}$ for $2\le k\le N$. Then we have the poles $-\frac{3N}{4}$ and $-\frac{N+1}{2}$.
Q.E.D.

Appendix C.

Definition 10 *(1) Let $R=(r_{ij})$ be an $H\times H'$ matrix, I an element of $\{0,1\}^H$, and $f(R,r')$ an analytic function of $r_{11},r_{21},\ldots,r_{HH'},r'_1,\ldots,r'_k$, where $r'=(r'_1,\ldots,r'_k)$. $f(R,r')$ is an I-type function of $(r_{ij})_{i'\le i\le H,1\le j\le H'}$, if for any $I_{i_0}=1$ with $i_0\ge i'$,*

$$f(r_{11},\cdots,r_{1N},r_{21},\cdots,r_{i_0-1,N},ur_{i_0,1},\cdots,ur_{i_0,N},r_{i_0+1,1},\cdots,r_{M,N},r')/u,$$

is an analytic function of u, where u is a variable.
(2) Let $I,I'\in\{0,1\}^H$. We denote $I\le I'$ if $I_i\le I'_i$ for all $i=1,\ldots,H$, and denote $I<I'$ if $I\le I'$ and $I\ne I'$.

For example, $B^I = B^I_N$ is an I'-type function of B for all $I' \le I$ ($I' \in \{0,1\}^M$).

Let $I_{ij} = (0, \ldots, 0, \overset{i}{1}, 0, \ldots, 0, \overset{j}{1}, 0, \ldots, 0)$, for $i < j$.

Proof of Theorem 3

Assume that $a^* = 0$.

Let $B'^{I_{ij}} = B^{I_{ij}} - \sum_{k=1}^N b_{ik} b_{jk}$, which is a polynomial of at least degree four.

For $I \in I$, let $I^{(s)} \in \{0,1\}^M$ be $I^{(s)}_i = \begin{cases} 0, & \text{if } i \le s, \\ I_i, & \text{if } i > s. \end{cases}$

We set $I' = I - \{I_{ij} : 1 \le i < j \le M\}$.

By using a blowing up process together with an inductive method of s, we have functions (5) and (6) below.

$$\int \{ \sum_{1 \le i < j \le M} (B^{I_{ij}}_{(s)})^2 + \sum_{I \in I'} (B^I_{(s)})^2 \}^z \\ u_1^{MN-1} u_2^{(M-1)N-1} \cdots u_s^{(M-s+1)N-1} \prod_{i=1}^s v_i^{N-i-1} \mathrm{d}u \mathrm{d}b^{(s)} \mathrm{d}v, \tag{5}$$

where

$$B^{I_{ij}}_{(s)} = \begin{cases} u_1^2 u_2^2 \cdots u_i^2 u_{i+1} \cdots u_j \{ f^{(s)}_{ij} + b^{(s)}_{ji} + u_1^2 B'^{I_{ij}}_{(s)} \}, & i < j \le s, \\ u_1^2 u_2^2 \cdots u_i^2 u_{i+1} \cdots u_s \{ f^{(s)}_{ij} + b^{(s)}_{ji} + u_1^2 B'^{I_{ij}}_{(s)} \}, & i \le s < j, \\ u_1^2 u_2^2 \cdots u_s^2 \{ f^{(s)}_{ij} + \sum_{k=s+1}^N b^{(s)}_{ik} b^{(s)}_{jk} + u_1^2 B'^{I_{ij}}_{(s)} \}, & s < i < j, \end{cases}$$

$$B^I_{(s)} = \prod_{k=1}^s u_k^{\sum_{k'=k}^M I_i} B'^I_{(s)}, \text{ for } I \in I',$$

$f^{(s)}_{ij}$ is an $I^{(s)}_{ij}$-type function of $(b^{(s)}_{kl})_{s+1 \le k \le M, 1 \le l \le N}$,

$$f^{(s)}_{ij}|_{b^{(s)}_{i1} = \cdots = b^{(s)}_{i,\min\{i-1,s\}} = b^{(s)}_{j1} = \cdots = b^{(s)}_{j,\min\{i-1,s\}} = 0} = 0,$$

$B'^{I_{ij}}_{(s)}$ is an $I^{(s)}_{ij}$-type function of $(b^{(s)}_{kl})_{s+1 \le k \le M, 1 \le l \le N}$, and $B'^I_{(s)}$ ($I \in I'$) is an $I^{(s)}$-type function of $(b^{(s)}_{kl})_{s+1 \le k \le M, 1 \le l \le N}$.

For $s+1 \le \ell \le M$ and $1 \le \ell' \le s$,

$$\int \{ \sum_{\{i<j\le s\} \cup \{i \le s < j, i < \ell'\} \cup \{i = \ell', j = \ell\}} (B^{I_{ij}}_{(s)})^2 \}^z u_1^{MN-1} u_2^{(M-1)N-1} \cdots u_s^{(M-s+1)N-1} u_{s+1}^{(M-s)N-1} \mathrm{d}u \mathrm{d}\tilde{b}, \tag{6}$$

where

$$B^{I_{ij}}_{(s)} = \begin{cases} u_1^2 u_2^2 \cdots u_i^2 u_{i+1} \cdots u_j \tilde{b}_{ji}, & \text{if } i < j \le s, \\ u_1^2 u_2^2 \cdots u_i^2 u_{i+1} \cdots u_{s+1} \tilde{b}_{ji}, & \text{if } i \le s < j, i < \ell', \\ u_1^2 u_2^2 \cdots u_{\ell'}^2 u_{\ell'+1} \cdots u_{s+1}, & \text{if } i = \ell', j = \ell. \end{cases}$$

C.1 Step 1

Construct the blow-up of function (3) along the submanifold $\{b_{ij}=0, 1\leq i\leq M, 1\leq j\leq N\}$.

Let $b_{11}=u_1, b_{ij}=u_1 b'_{ij}, (i,j)\neq(1,1)$.

Then we have $B^{I_{1j}}=u_1^2(b'_{j1}+\sum_{k=2}^N b'_{1k}b'_{jk}+B'^{I_{1j}}/u_1^2)$ for $j\geq 2$ and $B^{I_{ij}}=u_1^2(\sum_{k=1}^N b'_{ik}b'_{jk}+B'^{I_{ij}}/u_1^2)$ for $2\leq i<j$.

Let $b''_{j1}=b'_{j1}+\sum_{k=2}^N b'_{1k}b'_{jk}$ for $j\geq 2$.

Then for $2\leq i<j$,

$$
\begin{aligned}
&\sum_{k=1}^N b'_{ik}b'_{jk}=(b''_{i1}-\sum_{k=2}^N b'_{1k}b'_{ik})(b''_{j1}-\sum_{k=2}^N b'_{1k}b'_{jk})+\sum_{k=2}^N b'_{ik}b'_{jk}\\
=\;& b''_{i1}(b''_{j1}-\sum_{k=2}^N b'_{1k}b'_{jk})-(\sum_{k=2}^N b'_{1k}b'_{ik})b''_{j1}+(\sum_{k=2}^N b'_{1k}b'_{ik})(\sum_{k=2}^N b'_{1k}b'_{jk})+\sum_{k=2}^N b'_{ik}b'_{jk}\\
=\;& f_{ij}^{(1)}+(b'_{i2},\ldots,b'_{iN})\begin{pmatrix} b'_{12}\\ b'_{13}\\ \vdots\\ b'_{1N}\end{pmatrix}(b'_{12},\ldots,b'_{1N})\begin{pmatrix} b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix}\\
&+(b'_{i2},\ldots,b'_{iN})\begin{pmatrix} b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix},
\end{aligned}
$$

where $f_{ij}^{(1)}=b''_{i1}(b''_{j1}-\sum_{k=2}^N b'_{1k}b'_{jk})-(\sum_{k=2}^N b'_{1k}b'_{ik})b''_{j1}$ is an $I_{ij}^{(1)}$-type function of $\begin{pmatrix} b''_{21} & b'_{22} & \cdots & b'_{2N}\\ b''_{31} & b'_{32} & \cdots & b'_{3N}\\ \vdots & \vdots & \vdots & \vdots\\ b''_{M1} & b'_{M2} & \cdots & b'_{MN}\end{pmatrix}$ with $f_{ij}^{(1)}|_{b''_{i1}=b''_{j1}=0}=0$.

Next, construct the blow-up along the submanifold $\{b'_{12}=b'_{13}=\cdots=b'_{1N}=0\}$.

Let $b'_{12}=v_1, b'_{13}=v_1 b''_{13}, \cdots, b'_{1N}=v_1 b''_{1N}$.

Then we have, for $2\leq i<j$,

$$
\begin{aligned}
&(b'_{i2},\ldots,b'_{iN})\begin{pmatrix} b'_{12}\\ b'_{13}\\ \vdots\\ b'_{1N}\end{pmatrix}(b'_{12},\ldots,b'_{1N})\begin{pmatrix} b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix}+(b'_{i2},\ldots,b'_{iN})\begin{pmatrix} b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix}\\
=\;&(b'_{i2},\ldots,b'_{iN})\left(v_1^2\begin{pmatrix} 1\\ b''_{13}\\ \vdots\\ b''_{1N}\end{pmatrix}(1,b''_{13},\cdots,b''_{1N})+\begin{pmatrix} 1 & 0 & \cdots & 0\\ 0 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & 1\end{pmatrix}\right)\begin{pmatrix} b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix}.
\end{aligned}
$$

Let $Q_i = \sqrt{1+b''^2_{13}+\cdots+b''^2_{1i}}$ and

$$G = \begin{pmatrix} \frac{1}{Q_N} & \frac{-b''_{13}}{Q_3} & \cdots & \frac{-b''_{1i}}{Q_{i-1}Q_i} & \cdots & \frac{-b''_{1N}}{Q_{N-1}Q_N} \\ \frac{b''_{13}}{Q_N} & \frac{1}{Q_3} & \cdots & \frac{-b''_{13}b''_{1i}}{Q_{i-1}Q_i} & \cdots & \frac{-b''_{13}b''_{1N}}{Q_{N-1}Q_N} \\ \frac{b''_{14}}{Q_N} & 0 & \cdots & \frac{-b''_{14}b''_{1i}}{Q_{i-1}Q_i} & \cdots & \frac{-b''_{14}b''_{1N}}{Q_{N-1}Q_N} \\ \vdots & \vdots & & \vdots & & \vdots \\ & & & \frac{-b''_{1,i-1}b''_{1i}}{Q_{i-1}Q_i} & & \\ & & & \frac{Q_{i-1}}{Q_i} & & \\ & & & 0 & & \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{b''_{1N}}{Q_N} & 0 & \cdots & 0 & \cdots & \frac{Q_{N-1}}{Q_N} \end{pmatrix}.$$

Then we have

$$\begin{aligned} & v_1^2 \begin{pmatrix} 1 \\ b''_{13} \\ \vdots \\ b''_{1N} \end{pmatrix} (1, b''_{13}, \cdots, b''_{1N}) + \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \\ = \; & v_1^2 G \begin{pmatrix} 1+b''^2_{13}+\cdots+b''^2_{1N} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} G^t + \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \\ = \; & G \left(v_1^2 \begin{pmatrix} 1+b''^2_{13}+\cdots+b''^2_{1N} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \right) G^t \\ = \; & G \begin{pmatrix} 1+v_1^2(1+b''^2_{13}+\cdots+b''^2_{1N}) & 0 & \cdots & 0 \\ 0 & 1 & 0\cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} G^t. \end{aligned}$$

Therefore, we can change the variables from $(b'_{i2}, b'_{i3}, \cdots, b'_{iN})$ to $(b''_{i2}, b''_{i3}, \cdots, b''_{iN})$ by

$$\begin{pmatrix} b''_{i2} \\ b''_{i3} \\ \vdots \\ b''_{iN} \end{pmatrix} = \begin{pmatrix} \sqrt{1+v_1^2(1+b''^2_{13}+\cdots+b''^2_{1N})} & 0 & \cdots & 0 \\ 0 & 1 & 0\cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} G^t \begin{pmatrix} b'_{i2} \\ b'_{i3} \\ \vdots \\ b'_{iN} \end{pmatrix}.$$

We have

$$(b'_{i2},b'_{i3},\cdots,b'_{iN})\left(v_1^2\begin{pmatrix}1\\ b'_{13}\\ \vdots\\ b'_{1N}\end{pmatrix}(1,b'_{13},\cdots,b'_{1N})+\begin{pmatrix}1&0&\cdots&0\\ 0&1&\cdots&0\\ \vdots&\vdots&\vdots&\vdots\\ 0&0&\cdots&1\end{pmatrix}\right)\begin{pmatrix}b'_{j2}\\ b'_{j3}\\ \vdots\\ b'_{jN}\end{pmatrix}$$
$$= \sum_{k=2}^{N} b''_{ik}b''_{jk}.$$

Let $b^{(1)}_{ik} = b''_{ik}$ for $1 \le i \le M, 1 \le k \le N$ and $(i,k) \ne (1,1)$, $B^{\prime I_{ij}}_{(1)} = B^{\prime I_{ij}}/u_1^4$ for $1 \le i < j \le M$ and $B^{I}_{(1)} = B^{I}/u_1^{\sum_{k=1}^{M} I_i}$ for $I \in I'$.

We have $B^{I_{1j}} = u_1^2(b^{(1)}_{j1} + u_1^2 B^{\prime I_{1j}}_{(1)})$ for $1 < j$, $B^{I_{ij}} = u_1^2(f^{(1)}_{ij} + \sum_{k=2}^{N} b^{(1)}_{ik}b^{(1)}_{jk} + u_1^2 B^{\prime I_{ij}}_{(1)})$ for $1 < i < j$ and $B^{I} = u_1^{\sum_{k=1}^{M} I_i} B^{I}_{(1)}$ for $I \in I'$, where $f^{(1)}_{ij}$ is an $I^{(1)}_{ij}$-type function of $(b^{(1)}_{kl})_{2\le k\le M, 1\le l\le N}$ with $f^{(1)}_{ij}|_{b^{(1)}_{i1}=b^{(1)}_{j1}=0} = 0$, $B^{\prime I_{ij}}_{(1)}$ is an $I^{(1)}_{ij}$-type function of $(b^{(1)}_{kl})_{2\le k\le M, 1\le l\le N}$, and $B^{\prime I}_{(1)}$ is an $I^{(1)}$-type function of $(b^{(1)}_{kl})_{2\le k\le M, 1\le l\le N}$.

C.2 Step 2

Assume Eq. (5). Construct the blow-up of function (5) along the submanifold $\{b^{(s)}_{ij} = 0, s+1 \le i \le M, 1 \le j \le N\}$.

Let $b^{(s)}_{ij} = u_{s+1} b'^{(s)}_{ij}$.

We have

$$B^{I_{ij}}_{(s)} = \begin{cases} u_1^2u_2^2\cdots u_i^2u_{i+1}\cdots u_j\{f^{(s)}_{ij} + b^{(s)}_{ji} + u_1^2 B^{\prime I_{ij}}_{(s)}\}, & i < j \le s, \\ u_1^2u_2^2\cdots u_i^2u_{i+1}\cdots u_su_{s+1}\{f^{(s)}_{ij}/u_{s+1} + b'^{(s)}_{ji} + u_1^2 B^{\prime I_{ij}}_{(s)}/u_{s+1}\}, & i \le s < j, \\ u_1^2u_2^2\cdots u_s^2u_{s+1}^2\{f^{(s)}_{ij}/u_{s+1}^2 + \sum_{k=s+1}^{N} b'^{(s)}_{ik}b'^{(s)}_{jk} + u_1^2 B^{\prime I_{ij}}_{(s)}/u_{s+1}^2\}, & s < i < j, \end{cases}$$

and

$$B^{I}_{(s)} = \prod_{k=1}^{s+1} u_k^{\sum_{k'=k}^{N} I_i} B'^{I}_{(s)}/u_{s+1}^{\sum_{k'=s+1}^{N} I_i}, I \in I'.$$

We may consider $b'^{(s)}_{s+1,s+1} = 1$ or $b'^{(s)}_{\ell,\ell'} = 1$ for some $s+1 \le \ell \le M$ and $1 \le \ell' \le s$.

If $b'^{(s)}_{\ell,\ell'} = 1$ for some $s+1 \le \ell \le M$ and $1 \le \ell' \le s$, then we have function (6) by using

$$f^{(s)}_{ij}|_{b^{(s)}_{i1}=\cdots=b^{(s)}_{i,\min\{i-1,s\}}=b^{(s)}_{j1}=\cdots=b^{(s)}_{j,\min\{i-1,s\}}=0} = 0,$$

and Lemma 6.

Let $b'^{(s)}_{s+1,s+1} = 1$.

Set $b''^{(s)}_{j,s+1} = b'^{(s)}_{j,s+1} + \sum_{k=s+2}^{N} b'^{(s)}_{s+1,k}b'^{(s)}_{jk}$ for $j \ge s+2$.

Then for $s+2 \leq i < j$,

$$\begin{aligned}
& f_{ij}^{(s)}/u_{s+1}^2 + \sum_{k=s+1}^{N} b'^{(s)}_{ik} b'^{(s)}_{jk} \\
= \; & f_{ij}^{(s)}/u_{s+1}^2 + (b''^{(s)}_{i,s+1} - \sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{ik})(b''^{(s)}_{j,s+1} - \sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{jk}) + \sum_{k=s+2}^{N} b'^{(s)}_{ik} b'^{(s)}_{jk} \\
= \; & f_{ij}^{(s)}/u_{s+1}^2 + b''^{(s)}_{j,s+1}(b''^{(s)}_{i,s+1} - \sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{ik}) - b''^{(s)}_{i,s+1}(\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{jk}) \\
& + (\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{ik})(\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{jk}) + \sum_{k=s+2}^{N} b'^{(s)}_{ik} b'^{(s)}_{jk}.
\end{aligned}$$

Let $f_{ij}^{(s+1)} = f_{ij}^{(s)}/u_{s+1}^2 + b''^{(s)}_{j,s+1}(b''^{(s)}_{i,s+1} - \sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{ik}) - b''^{(s)}_{i,s+1}(\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{jk})$. Then $f_{ij}^{(s+1)}$ is an $I_{ij}^{(s+1)}$-type function of $\begin{pmatrix} b''^{(s)}_{s+2,1} & b'^{(s)}_{s+2,2} & \cdots & b'^{(s)}_{s+2,N} \\ \vdots & \vdots & \vdots & \vdots \\ b''^{(s)}_{M1} & b'^{(s)}_{M2} & \cdots & b'^{(s)}_{MN} \end{pmatrix}$ with

$$f_{ij}^{(s+1)}|_{b''^{(s)}_{i1} = \cdots = b''^{(s)}_{i,s+1} = b''^{(s)}_{j1} = \cdots = b''^{(s)}_{j,s+1} = 0} = 0.$$

Next, construct the blow-up along the submanifold $\{b'^{(s)}_{s+1,s+2} = b'^{(s)}_{s+1,s+3} = \cdots = b'^{(s)}_{s+1,N} = 0\}$.
Let $b'^{(s)}_{s+1,s+2} = v_s$, $b'^{(s)}_{s+1,s+3} = v_s b''^{(s)}_{s+1,s+3}, \cdots,$
$b'^{(s)}_{s+1,N} = v_s b''^{(s)}_{s+1,N}$.
Let $Q_i^{(s)} = \sqrt{1 + {b''^{(s)}_{s+1,s+3}}^2 + \cdots + {b''^{(s)}_{s+1,i}}^2}$ and

$$G^{(s)} = \begin{pmatrix}
\frac{1}{Q_N} & \frac{-b''^{(s)}_{s+1,s+3}}{Q_{s+3}^{(s)}} & \cdots & \frac{-b''^{(s)}_{s+1,i}}{Q_{i-1}^{(s)} Q_i^{(s)}} & \cdots & \frac{-b''^{(s)}_{s+1,N}}{Q_{N-1}^{(s)} Q_N^{(s)}} \\
\frac{b''^{(s)}_{s+1,s+3}}{Q_N^{(s)}} & \frac{1}{Q_{s+3}^{(s)}} & \cdots & \frac{-b''^{(s)}_{s+1,s+3} b''^{(s)}_{s+1,i}}{Q_{i-1}^{(s)} Q_i^{(s)}} & \cdots & \frac{-b''^{(s)}_{s+1,s+3} b''^{(s)}_{s+1,N}}{Q_{N-1}^{(s)} Q_N^{(s)}} \\
\frac{b''^{(s)}_{s+1,s+4}}{Q_N^{(s)}} & 0 & \cdots & \frac{-b''^{(s)}_{s+1,s+4} b''^{(s)}_{s+1,i}}{Q_{i-1}^{(s)} Q_i^{(s)}} & \cdots & \frac{-b''^{(s)}_{s+1,s+4} b''^{(s)}_{s+1,N}}{Q_{N-1}^{(s)} Q_N^{(s)}} \\
\vdots & \vdots & & \vdots & & \vdots \\
 & & & \frac{-b''^{(s)}_{s+1,i-1} b''^{(s)}_{s+1,i}}{Q_{i-1}^{(s)} Q_i^{(s)}} & & \\
 & & & \frac{Q_{i-1}^{(s)}}{Q_i^{(s)}} & & \\
 & & & 0 & & \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
\frac{b''^{(s)}_{s+1,N}}{Q_N^{(s)}} & 0 & \cdots & 0 & \cdots & \frac{Q_{N-1}^{(s)}}{Q_N^{(s)}}
\end{pmatrix}.$$

Change variables from $(b'^{(s)}_{i,s+2}, b'^{(s)}_{i,s+3}, \cdots, b'^{(s)}_{iN})$ to $(b''^{(s)}_{i,s+2}, b''^{(s)}_{i,s+3}, \cdots, b''^{(s)}_{iN})$ by

$$\begin{pmatrix} b''^{(s)}_{i,s+2} \\ b''^{(s)}_{i,s+3} \\ \vdots \\ b''^{(s)}_{iN} \end{pmatrix} = \begin{pmatrix} \sqrt{1+v^2(1+{b''^{(s)}_{s+1,s+3}}^2+\cdots+{b''^{(s)}_{s+1,N}}^2)} & 0 & \cdots & 0 \\ 0 & & 1 & 0\cdots 0 \\ \vdots & & \vdots & \cdots \vdots \\ 0 & & 0 & \cdots 1 \end{pmatrix} G^{(s)^t} \begin{pmatrix} b'^{(s)}_{i,s+2} \\ b'^{(s)}_{i,s+3} \\ \vdots \\ b'^{(s)}_{iN} \end{pmatrix}.$$

We have

$$\begin{aligned} & (\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{ik})(\sum_{k=s+2}^{N} b'^{(s)}_{s+1,k} b'^{(s)}_{jk}) + \sum_{k=s+2}^{N} b^{(s)}_{ik} b^{(s)}_{jk} \\ = & \sum_{k=s+2}^{N} b''^{(s)}_{ik} b''^{(s)}_{jk}. \end{aligned}$$

Let $b^{(s+1)}_{ji} = b^{(s)}_{ji}$ for $i, j \leq s$ and $b^{(s+1)}_{ji} = b''^{(s)}_{ji}$ for $i, j > s$ and $(i, j) \neq (s+1, s+1)$. Also let $f^{(s+1)}_{ij} = f^{(s)}_{ij}$ for $i < j \leq s$, $f^{(s+1)}_{ij} = f^{(s)}_{ij}/u_{s+1}$ for $i \leq s < j$, $f^{(s+1)}_{s+1,j} = f^{(s)}_{s+1,j}/u^2_{s+1}$ for $j > s+1$, $B'^{I_{ij}}_{(s+1)} = B'^{I_{ij}}_{(s)}$ for $i < j \leq s$, $B'^{I_{ij}}_{(s+1)} = B'^{I_{ij}}_{(s)}/u_{s+1}$ for $i \leq s < j$, $B'^{I_{ij}}_{(s+1)} = B'^{I_{ij}}_{(s)}/u^2_{s+1}$ for $s < i < j$ and $B'^{I}_{(s+1)} = B'^{I}_{(s)}/u^{\sum_{k'=s+1}^{N} I_i}_{s+1}$ for $I \in I'$. Then we have Eq. (5) with $s+1$.

C.3 Step 3

From the above induction ($1 \leq s \leq N+1$), we finally have Eq. (6) since we assume that $N < M$.

Note that we have the same inductive results for

$$\int \{ \sum_{1 \leq i < j \leq M} (\sum_{k=1}^{N} b_{ik} b_{jk})^2 \}^z \mathrm{d}b, \tag{7}$$

instead of the function in Eq. (3). This means that the maximum pole, and its order, of the function in Eq. (3) are those of the function in Eq. (7).

Now we again consider the maximum pole of the function in Eq. (7) and its order.

In Step3, we use the same symbol b rather than $b^{(s)}$ for the sake of simplicity.

We need to consider the following function with the inductive method with s.

$$\begin{aligned} \int \{ u_1^4 u_2^4 \cdots u_s^4 \sum_{1 \leq i < j \leq M, i \leq s} b^2_{ji} + u_1^4 u_2^4 \cdots u_s^4 \sum_{s+1 \leq i < j \leq M} (\sum_{k=s+1}^{N} b_{ik} b_{jk})^2 \}^z \\ \prod_{k=1}^{s} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u \mathrm{d}b. \end{aligned} \tag{8}$$

First, we set the variables the same as in Step 1. Then we have

$$\int \{ u_1^4 \sum_{2 \leq j \leq M} b^2_{j1} + u_1^4 \sum_{2 \leq i < j \leq M} (f^{(1)}_{ij} + \sum_{k=2}^{N} b_{ik} b_{jk})^2 \}^z u_1^{MN-1} \mathrm{d}u \mathrm{d}b.$$

By using Lemma 6 again, we need to consider

$$\int \{u_1^4 \sum_{2\le j\le M} b_{j1}^2 + u_1^4 \sum_{2\le i<j\le M} (\sum_{k=2}^{N} b_{ik}b_{jk})^2\}^z u_1^{MN-1} \mathrm{d}u\mathrm{d}b.$$

Assume Eq. (8). Construct the blow-up of function (8) along the submanifold $\{b_{ji} = 0, 1 \le i < j \le M, i \le s, b_{kl} = 0, s+1 \le k \le M, s+1 \le l \le N\}$.

Then we have

$$\int \{u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^2 \sum_{1\le i<j\le M, i\le s} b_{ji}^2 + u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^4 \sum_{s+1\le i<j\le M} (\sum_{k=s+1}^{N} b_{ik}b_{jk})^2\}^z$$
$$u_{s+1}^{(M-s)(N-s)+(2M-1-s)s/2-1} \prod_{k=1}^{s} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u\mathrm{d}b,$$

where we can set $b_{21} = 1$ or $b_{s+1,s+1} = 1$.

If $b_{21} = 1$, we have the poles

$$\frac{(M-k+1)(N-k+1)+(2M-k)(k-1)}{4}, k = 1, \ldots, s$$

and

$$\frac{(M-s)(N-s)+(2M-1-s)s/2}{2}.$$

If $b_{s+1,s+1} = 1$, then by setting the variables the same as in Step 2 and by using Lemma 6, we have

$$\int \{u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^2 \sum_{1\le i<j\le M, i\le s} b_{ji}^2$$
$$+ u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^4 (\sum_{s+1<j\le M} b_{j,s+1}^2 + \sum_{s+2\le i<j\le M} (\sum_{k=s+2}^{N} b_{ik}b_{jk})^2)\}^z$$
$$u_{s+1}^{(M-s)(N-s)+(2M-1-s)s/2-1} \prod_{k=1}^{s} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u\mathrm{d}b. \tag{9}$$

Construct the blow-up of function (9) along the submanifold $\{b_{ji} = 0, 1 \le i < j \le M, i \le s, u_{s+1} = 0\}$.

Then we have Eq. (8) with $s+1$, that is,

$$\int \{u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^4 \sum_{1\le i<j\le M, i\le s} b_{ji}^2$$
$$+ u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^4 (\sum_{s+1<j\le M} b_{j,s+1}^2 + \sum_{s+2\le i<j\le M} (\sum_{k=s+2}^{N} b_{ik}b_{jk})^2)\}^z$$
$$u_{s+1}^{(M-s)(N-s)+(2M-1-s)s-1} \prod_{k=1}^{s} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u\mathrm{d}b.$$

or

$$\int \{u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^2 b_{21}^4 (1 + \sum_{1 \le i < j \le M, i \le s, (i,j) \ne (1,2)} b_{ji}^2)$$
$$+ u_1^4 u_2^4 \cdots u_s^4 u_{s+1}^4 b_{21}^4 (\sum_{s+1 < j \le M} b_{j,s+1}^2 + \sum_{s+2 \le i < j \le M} (\sum_{k=s+2}^{N} b_{ik} b_{jk})^2)\}^z$$
$$u_{s+1}^{(M-s)(N-s)+(2M-1-s)s/2-1} b_{21}^{(M-s)(N-s)+(2M-1-s)s-1}$$
$$\prod_{k=1}^{s} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u\mathrm{d}b,$$

which have the poles

$$\frac{(M-k+1)(N-k+1)+(2M-k)(k-1)}{4}, k = 1, \ldots, s+1,$$

and

$$\frac{(M-s)(N-s)+(2M-1-s)s/2}{2}.$$

Finally, we have

$$\int \{u_1^4 u_2^4 \cdots u_N^4 \sum_{1 \le i < j \le M, i \le N} b_{ji}^2\}^z \prod_{k=1}^{N} u_k^{(M-k+1)(N-k+1)+(2M-k)(k-1)-1} \mathrm{d}u\mathrm{d}b,$$

and obtain the poles

$$\frac{(M-k+1)(N-k+1)+(2M-k)(k-1)}{4}, k = 1, \ldots, N,$$

and

$$\frac{(2M-1-N)N}{4}.$$

Therefore, since we assume that $M > N$, we have the maximum pole $-\lambda = -\frac{MN}{4}$ and its order $\theta = \begin{cases} 1, & \text{if } M > N+1, \\ M, & \text{if } M = N+1. \end{cases}$ Q.E.D.

Proof of Theorem 4

Assume that $a^* = 0$

By the proof of Theorem 3, the maximum pole of $\int \{\sum_{1 \le i < j \le M} (B^{l_{ij}})^2\}^z db$ is that of $\int \{\sum_{1 \le i < j \le M} (\sum_{k=1}^{N} b_{ik} b_{jk})^2\}^z db$ even for $M \le N$. If $M \le N$ then the maximum pole of $\int \{\sum_{1 \le i < j \le M} (\sum_{k=1}^{N} b_{ik} b_{jk})^2\}^z db$ is $-M(M-1)/4$. Therefore the maximum pole $-\lambda$ of $\int \{\sum_{I \ne 0 \in I} (B^I)^2\}^z db$ satisfies $\lambda \ge M(M-1)/4$, since $\sum_{1 \le i < j \le M} (B^{l_{ij}})^2 \le \sum_{I \ne 0 \in I} (B^I)^2$.

Next let us prove that $\lambda \le \frac{2N+(M-1)(M-2)}{4}$. Consider Eq. (6) with $\ell = M$, $\ell' = M-1$ and $s = M-1$.

Let $\tilde{b}_{ji} = u_M \tilde{b}'_{ji}$ for $i < j < M$. Then we have the pole

$$\frac{N+(M-2)(M-1)/2}{2}.$$

For $a^* \ne 0$, Lemma 7 yields the statement.

Q.E.D.

References

H. Akaike. A new look at the statistical model identification. *IEEE Trans. on Automatic Control*, 19:716–723, 1974.

H. Akaike. Likelihood and bayes procedure. In J. M. Bernald, editor, *Bayesian Statistics*, pages 143–166. University Press, Valencia, Spain, 1980.

S. Amari and N. Murata. Statistical theory of learning curves under entropic loss. *Neural Computation*, 5:140–153, 1993.

S. Amari, N. Fujita, and S. Shinomoto. Four types of learning curves. *Neural Computation*, 4(4): 608–618, 1992.

M. Aoyagi. The zeta function of learning theory and generalization error of three layered neural perceptron. *RIMS Kokyuroku, Recent Topics on Real and Complex Singularities*, 1501:153–167, 2006.

M. Aoyagi and S. Watanabe. Stochastic complexities of reduced rank regression in bayesian estimation. *Neural Networks*, 18:924–933, 2005b.

M. Aoyagi and S. Watanabe. Resolution of singularities and the generalization error with bayesian estimation for layered neural network. *IEICE Trans. J88-D-II*, 10:2112–2124, 2005a.

K. Fukumizu. A regularity condition of the information matrix of a multilayer perceptron network. *Neural Networks*, 9(5):871–879, 1996.

W. Fulton. *Introduction to toric varieties, Annals of Mathematics Studies*. Princeton University Press, 1993.

K. Hagiwara, N. Toda, and S. Usui. On the problem of applying aic to determine the structure of a layered feed-forward neural network. In *Proceedings of IJCNN Nagoya Japan*, volume 3, pages 2263–2266, 1993.

E. J. Hannan and B. G. Quinn. The determination of the order of an autoregression. *Journal of Royal Statistical Society Series B*, 41:190–195, 1979.

J. A. Hartigan. A failure of likelihood asymptotics for normal mixtures. In *Proceedings of the Berkeley Conference in Honor of J.Neyman and J.Kiefer*, volume 2, pages 807–810, 1985.

G. E. Hinton. Reinforcement learning with factored states and actions. *Brian Sallans*, 5:1063–1088, 2004.

H. Hironaka. Resolution of singularities of an algebraic variety over a field of characteristic zero. *Annals of Math*, 79:109–326, 1964.

E. Levin, N. Tishby, and S. A. Solla. A statistical approaches to learning and generalization in layered neural networks. In *Proceedings of IEEE*, volume 78, pages 1568–1674, 1990.

D. J. Mackay. Bayesian interpolation. *Neural Computation*, 4(2):415–447, 1992.

N. J. Murata, S. G. Yoshizawa, and S. Amari. Network information criterion - determining the number of hidden units for an artificial neural network model. *IEEE Trans. on Neural Networks*, 5(6):865–872, 1994.

K. Nagata and S. Watanabe. A proposal and effectiveness of the optimal approximation for bayesian posterior distribution. In *Workshop on Information-Based Induction Sciences*, pages 99–104, 2005.

K. Nagata and S. Watanabe. Analysis of exchange ratio for exchange monte carlo method. In *Proceedings of The First IEEE Symposium on Foundations of Computational Intelligence*, pages 434–439, 2007.

Y. Nishiyama and S. Watanabe. Asymptotic behavior of free energy of general boltzmann machines in mean field approximation. *Technical report of IEICE, NC2006*, 38:1–6, 2006.

J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Trans. on Information Theory*, 30(4):629–636, 1984.

J. Rissanen. Stochastic complexity and modeling. *Annals of Statistics*, 14:1080–1100, 1986.

D. Rusakov and D. Geiger. Asymptotic model selection for naive bayesian networks. *Journal of Machine Learning Research*, 6:1–35, 2005.

R. Salakhutdinov, A. Mnih, and G. E. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning*, pages 791–798. Omni Press, Zoubin Ghahramani, 2007.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

H. J. Sussmann. Uniqueness of the weights for minimal feed-forward nets with a given input-output map. *Neural Networks*, 5:589–593, 1992.

S. Takamatsu, S. Nakajima, and S. Watanabe. Generalization error of localized bayes estimation in reduced rank regression. In *Workshop on Information-Based Induction Sciences*, pages 81–86, 2005.

K. Takeuchi. Distribution of an information statistic and the criterion for the optimal model. *Mathematical Science*, 153:12–18, 1976.

S. Watanabe. Algebraic analysis for nonidentifiable learning machines. *Neural Computation*, 13 (4):899–933, 2001a.

S. Watanabe. Algebraic geometrical methods for hierarchical learning machines. *Neural Networks*, 14(8):1049–1060, 2001b.

S. Watanabe. Algebraic geometry of learning machines with singularities and their prior distributions. *Journal of Japanese Society of Artificial Intelligence*, 16(2):308–315, 2001c.

S. Watanabe. *Algebraic Geometry and Statistical Learning Theory*, volume 25. Cambridge Monographs on Applied and Computational Mathematics, 2009.

S. Watanabe, K. Hagiwara, S. Akaho, Y. Motomura, K. Fukumizu, M. Okada, and M. Aoyagi. *Theory and Application of Learning System*. Morikita Press, 2005.

K. Yamanishi. A decision-theoretic extension of stochastic complexity and its applications to learning. *IEEE Trans. on Information Theory*, 44(4):1424–1439, 1998.

K. Yamazaki and S. Watanabe. Singularities in complete bipartite graph-type boltzmann machines and upper bounds of stochastic complexities. *IEEE Trans. on Neural Networks*, 16(2):312–324, 2005.

Journal of Machine Learning Research 11 (2010) 1273-1296 Submitted 2/09; Revised 1/10; Published 4/10

Approximate Inference on Planar Graphs using Loop Calculus and Belief Propagation

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

Michael Chertkov CHERTKOV@LANL.GOV
Theoretical Division and Center for Nonlinear Studies
Los Alamos National Laboratory
Los Alamos, NM 87545

Editor: Tommi Jaakkola

Abstract

We introduce novel results for approximate inference on planar graphical models using the loop calculus framework. The loop calculus (Chertkov and Chernyak, 2006a) allows to express the exact partition function of a graphical model as a finite sum of terms that can be evaluated once the belief propagation (BP) solution is known. In general, full summation over all correction terms is intractable. We develop an algorithm for the approach presented in Chertkov et al. (2008) which represents an efficient truncation scheme on planar graphs and a new representation of the series in terms of Pfaffians of matrices. We analyze the performance of the algorithm for models with binary variables and pairwise interactions on grids and other planar graphs. We study in detail both the loop series and the equivalent Pfaffian series and show that the first term of the Pfaffian series for the general, intractable planar model, can provide very accurate approximations. The algorithm outperforms previous truncation schemes of the loop series and is competitive with other state of the art methods for approximate inference.

Keywords: belief propagation, loop calculus, approximate inference, partition function, planar graphs, Ising model

1. Introduction

Graphical models are popular tools widely used in many areas which require modeling of uncertainty. They provide an effective approach through a compact representation of the joint probability distribution. The two most common types of graphical models are Bayesian networks (BN) and Markov random fields (MRFs).

The partition function of a graphical model, which plays the role of normalization constant in a MRF or probability of evidence (likelihood) in a BN is a fundamental quantity which arises in many contexts such as hypothesis testing or parameter estimation. Exact computation of this quantity is only feasible when the graph is not too complex, or equivalently, when its tree-width is small. Currently many methods are devoted to approximate this quantity.

The belief propagation (BP) algorithm (Pearl, 1988) is at the core of many of these approximate inference methods. Initially thought as an exact algorithm for tree graphs, it is widely used as an

approximation method for loopy graphs (Murphy et al., 1999; Frey and MacKay, 1998). The exact partition function is explicitly related to the BP approximation through the loop calculus framework introduced by Chertkov and Chernyak (2006a). Loop calculus allows to express the exact partition function as a finite sum of terms (loop series) that can be evaluated once the BP solution is known. Each term maps uniquely to a subgraph, also denoted as a generalized loop, where the connectivity of any node within the subgraph is *at least* degree two. Summation of the entire loop series is a hard combinatorial task since the number of generalized loops is typically exponential in the size of the graph (see also Watanabe and Fukumizu, 2009). However, different approximations can be obtained by considering different subsets of generalized loops in the graph.

It has been shown empirically (Gómez et al., 2007; Chertkov and Chernyak, 2006b) that truncating this series may provide efficient corrections to the initial BP approximation. More precisely, whenever BP performs satisfactorily, which occurs in the case of sufficiently weak interactions between variables (or short-range influence of loops), accounting for only a small number of terms is sufficient to recover the exact result. On the other hand, for those cases where BP requires many iterations to converge, many terms of the series are required to improve substantially the approximation. Nevertheless, a formal characterization of the classes of problems which are tractable via loop calculus still remains an open question.

A step toward this goal with a focus on the class of planar graphs has been done in Chertkov et al. (2008). A graph is said to be planar if it can be embedded into a plane without crossing edges. Chertkov et al. (2008) showed that under this condition summation of certain (large) subset of terms can be performed in polynomial time via mapping to the problem of counting perfect matchings. A perfect matching is a subset of edges where every vertex has exactly one attached edge in the subset. Furthermore, the full loop series can be expressed as a sum over certain Pfaffians (or determinants), where each Pfaffian may account for a large number of loops and is solvable in polynomial time as well.

The approach of Chertkov et al. (2008) builds upon classical results from Fisher (1961), Kasteleyn (1961) and Temperley and Fisher (1961) who addressed the question of counting the number of perfect matchings on a planar grid, also known as the dimer covering problem in statistical physics. These classical results are consistent with the following related statement: the partition function of a *planar graphical model defined in terms of binary variables* can be solved in polynomial time by computing an appropriate Pfaffian under the key restriction that pairwise interactions only depend on agreement or disagreement between the signs of their variables, and not on their individual values. Such a model is known in statistical physics as planar Ising model without external field and in the machine learning community as planar, binary MRF with pure interaction potentials. Notice that exact inference for a general binary graphical model on a planar graph, namely Ising model *with* external field, is intractable, in particular #P (Barahona, 1982). A gentle overview of counting perfect matchings in planar graphs can be found in Jerrum (2003).

Recently, other inference methods models based on the Fisher-Temperley-Kasteleyn (FTK) method have been introduced in the machine learning community. Globerson and Jaakkola (2007) obtained upper bounds on the partition function for non-planar graphs with binary variables by decomposition of the partition function into a weighted sum over partition functions of spanning tractable (zero field) planar models. The resulting problem is a convex optimization problem and, since exact inference can be done in each planar *sub*-model via the FTK method, the bound can be calculated in polynomial time.

Another example is the work of Schraudolph and Kamenetsky (2008) which provides a framework for exact inference on a restricted class of planar graphs using directly the FTK approach. Since a planar Ising model can be transformed to a model *without* external field at the cost of introducing additional edges, one can allow local fields for a subset $\mathcal{B}$ of variables. If the graphical model is $\mathcal{B}-$outerplanar, which means that there exists a planar embedding in which the subset $\mathcal{B}$ of the nodes lie on the same face, the FTK technique can still be applied.

Contrary to these two approaches, which rely on exact inference on a tractable planar model, the loop calculus directly leads to a framework for approximate inference on general planar graphs. Truncating the loop series according to Chertkov et al. (2008) already gives the exact result in the zero external field case. In the general planar case, however, this truncation may result into an accurate approximation that can be incrementally corrected by considering subsequent terms in the series.

In the next Section we review the main theoretical results of the loop calculus approach for planar graphs and introduce the proposed algorithm for approximating the partition function and single-variable marginals. In Section 3 we provide experimental results for regular grids and other types of planar graphs. We focus on a planar-intractable binary model with symmetric pairwise interactions but nonzero single variable potentials.[1] We end this manuscript with conclusions and discussion of future work in Section 4.

2. Belief Propagation and Loop Series for Planar Graphs

We consider the Forney graph representation, also called general vertex model (Jr., 2001; Loeliger, 2004), of a probability distribution $p(\boldsymbol{\sigma})$ defined over a vector $\boldsymbol{\sigma}$ of binary variables (vectors are denoted using bold symbols). Forney graphs are associated with general graphical models which subsume other factor graphs, for instance those correspondent to BNs and MRFs. In Appendix A we explain how our approach is related to the more common bipartite factor graph model.

A binary Forney graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ consists of a set of nodes $\mathcal{V}$ where each node $a \in \mathcal{V}$ represents an interaction and each edge $(a,b) \in \mathcal{E}$ represents a binary variable ab which take values $\sigma_{ab} := \{\pm 1\}$. We denote $\bar{a}$ the set of neighbors of node a. Interactions $f_a(\boldsymbol{\sigma}_a)$ are arbitrary functions defined over typically small subsets of variables where $\boldsymbol{\sigma}_a$ is the vector of variables associated with node a, that is, $\boldsymbol{\sigma}_a := (\sigma_{ab_1}, \sigma_{ab_2}, \dots)$ where $b_i \in \bar{a}$.

The joint probability distribution of such a model factorizes as:

$$p(\boldsymbol{\sigma}) = Z^{-1} \prod_{a \in \mathcal{V}} f_a(\boldsymbol{\sigma}_a), \qquad Z = \sum_{\boldsymbol{\sigma}} \prod_{a \in \mathcal{V}} f_a(\boldsymbol{\sigma}_a), \tag{1}$$

where Z is the normalization factor, also called the partition function.

From a variational perspective, a fixed point of the BP algorithm represents a stationary point of the Bethe "free energy" approximation under proper constraints (Yedidia et al., 2000). In the Forney style notation:

$$\begin{aligned} Z^{BP} &= \exp\left(-F^{BP}\right), \\ F^{BP} &= \sum_{a} \sum_{\boldsymbol{\sigma}_a} \tau_a(\boldsymbol{\sigma}_a) \ln\left(\frac{\tau_a(\boldsymbol{\sigma}_a)}{f_a(\boldsymbol{\sigma}_a)}\right) - \sum_{b \in \bar{a}} \sum_{\sigma_{ab}} \tau_{ab}(\sigma_{ab}) \ln \tau_{ab}(\sigma_{ab}), \end{aligned} \tag{2}$$

1. The source code used in this paper is freely available at `http://www.mbfys.ru.nl/staff/v.gomez/`.

where $\tau_a(\boldsymbol{\sigma}_a)$ and $\tau_{ab}(\sigma_{ab})$ are the beliefs (pseudo-marginals) associated with each node $a \in \mathcal{V}$ and edge $(a,b) \in \mathcal{E}$. For graphs without loops, Equation (2) coincides with the Gibbs "free energy" and therefore Z^{BP} coincides with the exact partition function Z. If the graph contains loops, Z^{BP} is just an approximation critically dependent on how strong the influence of the loops is.

We introduce now some convenient definitions related to the loop calculus framework.

Definition 1 *A **generalized loop** in a graph $\mathcal{G} := \langle \mathcal{V}, \mathcal{E} \rangle$ is any subgraph $C := \langle V', E' \rangle$, $V' \subseteq \mathcal{V}, E' \subseteq (V' \times V') \cap \mathcal{E}$ such that each node in V' has degree two or larger.*

For simplicity, we will use the term "loop", instead of "generalized loop", in the rest of this manuscript. Loop calculus allows to represent Z explicitly in terms of the BP approximation via the loop series expansion:

$$Z = Z^{BP} \cdot z, \qquad z = \left(1 + \sum_{C \in \mathcal{C}} r_C\right), \qquad r_C = \prod_{a \in C} \mu_{a;\bar{a}_C}, \tag{3}$$

where $\mathcal{C}$ is the set of all the loops within the graph and $\bar{a}_C$ the set of neighbors of node a within the loop C. Each term r_C is a product of terms $\mu_{a;\bar{a}_C}$ associated with every node a of the loop C:

$$\mu_{a;\bar{a}_C} = \frac{\mathbb{E}_{\tau_a}\left[\prod_{b \in \bar{a}_C} (\sigma_{ab} - m_{ab})\right]}{\sqrt{\prod_{b \in \bar{a}_C} \mathrm{Var}_{\tau_{ab}}(\sigma_{ab})}}, \qquad m_{ab} = \mathbb{E}_{\tau_{ab}}\left[\sigma_{ab}\right], \tag{4}$$

where we have used $\mathbb{E}_{\tau}[\cdot]$ to denote expectation with respect to the pseudo-marginal distribution τ. Equation (4) states that all terms of the expansion can be written as correlation functions between associated variables defined on the BP pseudo-marginals. In the case of $\{\pm 1\}$ alphabet we have:

$$\mu_{a;\bar{a}_C} = \frac{\sum_{\boldsymbol{\sigma}_a} \left(\tau_a(\boldsymbol{\sigma}_a) \prod_{b \in \bar{a}_C} (\sigma_{ab} - m_{ab})\right)}{\prod_{b \in \bar{a}_C} \sqrt{1 - m_{ab}^2}}, \qquad m_{ab} = \tau_{ab}(+1) - \tau_{ab}(-1). \tag{5}$$

In this work we consider planar graphs where nodes have degree at most three, that is $|\bar{a}_C| \leq 3$. We denote by *triplet* a node with degree exactly three in the graph. In Appendix A.2 we show how a graphical model stated in a bipartite factor graph representation can be converted to a Forney graph which satisfies this constraint at the cost of introducing auxiliary nodes.

Definition 2 *A **2-regular loop** is a loop in which all nodes have degree* exactly *two*.

Definition 3 *The **2-regular partition function** $Z_\emptyset$ is the truncated form of* (3) *which sums all* 2*-regular loops only:*[2]

$$\begin{aligned} Z_\emptyset &= Z^{BP} \cdot z_\emptyset, \\ z_\emptyset &= 1 + \sum_{\substack{C \in \mathcal{C}, \\ |\bar{a}_C| = 2, \forall a \in C}} r_C. \end{aligned} \tag{6}$$

As an example, Figure 1a shows a small Forney graph and Figure 1c shows seven loops found in the corresponding 2-regular partition function.

2. Notice that this is the *single-connected partition function* in Chertkov et al. (2008). We use the term 2-regular partition function instead because loops with more than one connected component are also included in this part of the series.

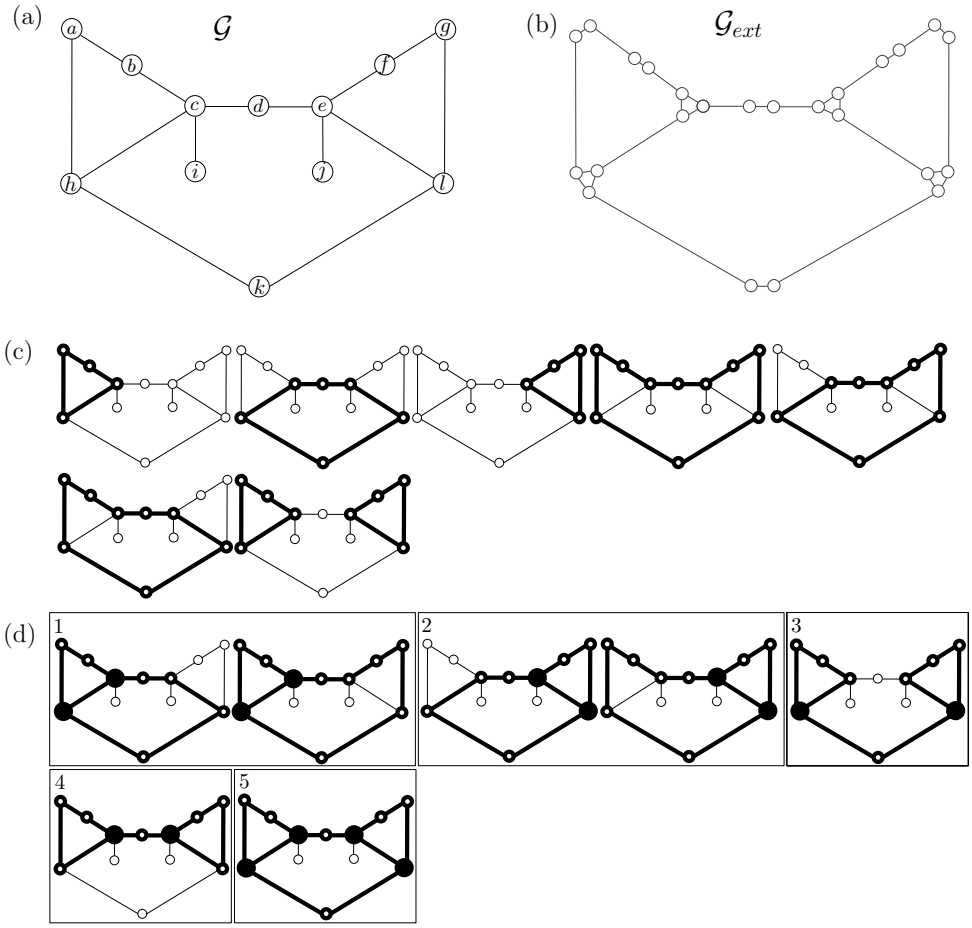


Figure 1: Example. **(a)** A Forney graph. **(b)** Corresponding extended graph. **(c)** Loops (in bold) included in the 2-regular partition function. **(d)** Loops (in bold) not included in the 2-regular partition function. Marked nodes denote triplets. Grouped in gray squares, the loops considered in different subsets Ψ of triplets: (d.1) $\Psi = \{c,h\}$, (d.2) $\Psi = \{e,l\}$, (d.3) $\Psi = \{h,l\}$, (d.4) $\Psi = \{c,e\}$ and (d.5) $\Psi = \{c,e,h,l\}$ (see Section 2.3).

Definition 4 *Consider the set P of all permutations α of the set $\mathcal{S} = \{1,\ldots,2n\}$ in pairs: $\alpha = (i_1,j_1),\ldots,(i_n,j_n))$, $i_k < j_k, \forall k = 1,\ldots,n$. The Pfaffian of a skew-symmetric matrix $A = (A_{ij})_{1\le i<j\le 2n}$ with ($A_{ij} = -A_{ji}$) is:*

$$\text{Pfaffian}(A) = \sum_{\alpha\in P} \text{sign}(\alpha) \prod_{(i,j)\in\alpha} A_{ij},$$

where the sign of a permutation α is -1 if the number of transpositions to get α from $\mathcal{S}$ is odd and $+1$ otherwise. The following identity allows to obtain the Pfaffian *up to a sign* by computing the determinant:

$$\text{Pfaffian}^2(A) = \text{Det}(A).$$

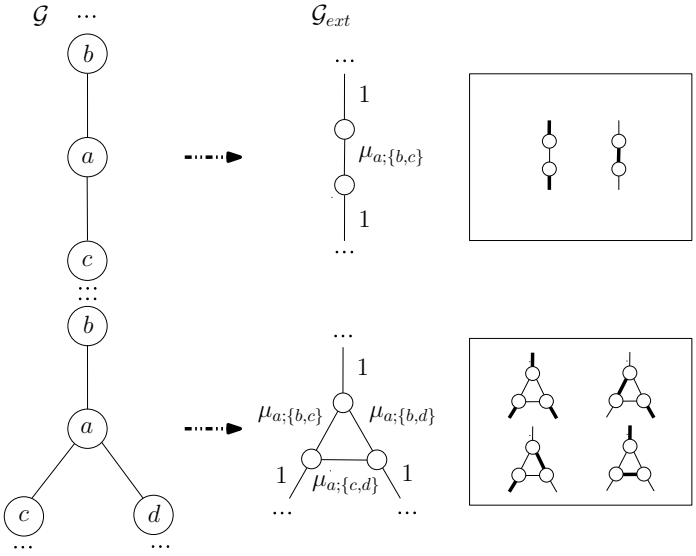


Figure 2: Fisher's rules. **(Top)** A node a of degree two in $\mathcal{G}$ is split in two nodes in $\mathcal{G}_{ext}$. **(Bottom)** A node a of degree three in $\mathcal{G}$ is split in three nodes in $\mathcal{G}_{ext}$. The squares on the right indicate all possible matchings in $\mathcal{G}_{ext}$ related to node a. Note that the rules preserve planarity.

2.1 Computing the 2-regular Partition Function Using Perfect Matchings

In Chertkov et al. (2008) it has been shown that computation of $Z_\emptyset$ in a Forney graph $\mathcal{G}$ can be mapped to the computation of the sum of all weighted perfect matchings within another extended weighted graph $\mathcal{G}_{ext} := (\mathcal{V}_{\mathcal{G}_{ext}}, \mathcal{E}_{\mathcal{G}_{ext}})$. A perfect matching is defined as a subset of edges such that each node neighbors exactly one edge from the subset and its weight is the product of weights of edges in it. The key idea of this mapping is that 2-regular loops in $\mathcal{G}$ are in one-to-one correspondence to perfect matchings in $\mathcal{G}_{ext}$ (see Figures 1b and c for an illustration). If $\mathcal{G}_{ext}$ is planar, the sum of its weighted perfect matchings can be calculated in polynomial time via the FTK approach, evaluating the Pfaffian of an associated matrix. Here we reproduce these results with little variations and emphasis on the algorithmic aspects.

Given a Forney graph $\mathcal{G}$ and the BP approximation, we obtain the 2-core of $\mathcal{G}$ by removing nodes of degree 1 recursively. The 2-core excludes all nodes that do not appear in any loop. After this step, $\mathcal{G}$ is either the null graph (and then BP is exact) or it is only composed of vertices of degree two or three.

To construct $\mathcal{G}_{ext}$ we split each node in $\mathcal{G}$ according to the rules introduced by Fisher (1966) and illustrated in Figure 2. The procedure results in an extended graph of $|\mathcal{V}_{\mathcal{G}_{ext}}| \leq 3|\mathcal{V}|$ nodes and $|\mathcal{E}_{\mathcal{G}_{ext}}| \leq 3|\mathcal{E}|$ edges. To see that each 2-regular loop in $\mathcal{G}$ is associated with a perfect matching in $\mathcal{G}_{ext}$ consider, for instance, the vertex of degree three in the bottom of Figure 2. Given a 2-regular loop C, vertex a can appear in four different configurations: either node a does not appear in C, or C contains one of the following three paths: -b-a-c-, -b-a-d- or -c-a-d-. These four cases correspond to node terms in a loop with values 1, $\mu_{a;\{b,c\}}$, $\mu_{a;\{b,d\}}$ and $\mu_{a;\{c,d\}}$ respectively, and coincide with

the matchings in $\mathcal{G}_{ext}$ shown within the box on the bottom-right. An simpler argument applies to the vertex of degree two of the top portion of Figure 2.

Therefore, if we associate to each internal edge (new edge in $\mathcal{G}_{ext}$ not in $\mathcal{G}$) of each split node a the corresponding weight $\mu_{a;\bar{a}_C}$ of Equation (4) and to the external edges (existing edges already in $\mathcal{G}$) weight 1, the sum over all weighted perfect matchings defined on $\mathcal{G}_{ext}$ is precisely $z_\emptyset$:

$$z_\emptyset = \sum \text{perfect matchings in } \mathcal{G}_{ext}.$$

The 2-regular partition function $Z_\emptyset$ is obtained using Equation (6).

Kasteleyn (1963) provided a method to compute this sum in polynomial time for planar graphs. First, edges are properly oriented in such a way that for every face (except possibly the external face) the number of clockwise oriented edges is odd. We use the linear time algorithm in Karpinski and Rytter (1998) described here as Algorithm 1 to produce such an orientation. It receives an undirected graph $\mathcal{G}_{ext}$ and constructs a copy $\mathcal{G}'_{ext} := (\mathcal{V}_{\mathcal{G}'_{ext}}, \mathcal{E}_{\mathcal{G}'_{ext}})$ with properly oriented edges $\mathcal{E}_{\mathcal{G}'_{ext}}$. It is convenient that $\mathcal{G}_{ext}$ is bi-connected, that is, has no articulation points. If needed, we add edges with zero weight which do not alter Z.

Algorithm 1 Pfaffian orientation

Arguments: undirected bi-connected extended graph $\mathcal{G}_{ext}$.

1: Construct a planar embedding $\bar{\mathcal{G}}_{ext}$ of $\mathcal{G}_{ext}$.
2: Construct a spanning tree T of $\bar{\mathcal{G}}_{ext}$.
3: Construct a graph H having vertices corresponding to the faces of $\bar{\mathcal{G}}_{ext}$:
   Connect two vertices in H if the respective face boundaries share an edge not in T.
   H is a tree. Root H to the external face.
4: $\mathcal{G}'_{ext} := T$.
5: Orient all edges in $\mathcal{G}'_{ext}$ arbitrarily.
6: **for all** face (vertex in H) traversed in post-order **do**
7: Add to $\mathcal{G}'_{ext}$ the unique edge not in $\mathcal{G}'_{ext}$.
8: Orient it such that the number of clock-wise oriented edges is odd.
9: **end for**
10: **RETURN** directed bi-connected extended graph $\mathcal{G}'_{ext}$.

Second, denote μ_{ij} the weight of the edge between nodes i and j in $\mathcal{G}'_{ext}$ and define the following skew-symmetric matrix:

$$\hat{A}_{ij} = \begin{cases} +\mu_{ij} & \text{if } (i,j) \in \mathcal{E}_{\mathcal{G}'_{ext}} \\ -\mu_{ij} & \text{if } (j,i) \in \mathcal{E}_{\mathcal{G}'_{ext}} \\ 0 & \text{otherwise} \end{cases}.$$

The Pfaffian of $\hat{A}_{ij}$ is the weighted sum of all perfect matchings. Calculation of $z_\emptyset$ can be performed in time $O(N^3_{\mathcal{G}_{ext}})$:

$$z_\emptyset = \sqrt{\text{Det}(\hat{A})}.$$

For the tractable case, namely, Ising model without external field or pairwise MRF with pure interaction potentials, the 2-regular partition function coincides with the exact partition function $Z =$

$Z_\emptyset = Z^{BP} \cdot z_\emptyset$ since other terms in the loop series vanish. In this case, it can be shown that all terms $\mu_{a;\{b,c,d\}}$ associated with triplets a (vertices with degree three in $\mathcal{G}$) are zero using the following argument: first, note that a BP fixed point is attained when all variables have zero magnetizations, that is, uniform beliefs $\tau_{ab}(\sigma_{ab}) = 1/2$, which reduces Equation (5) to:

$$\mu_{a;\{b,c,d\}} = \sum_{\boldsymbol{\sigma}_a} \tau_a(\boldsymbol{\sigma}_a)\, \sigma_{ab}\sigma_{ac}\sigma_{ad}. \tag{7}$$

Second, since by construction pseudo-marginals on the triplets $\tau_a(\boldsymbol{\sigma}_a)$ only depend on agreement or disagreement between the variables:

$$\tau_a(\boldsymbol{\sigma}_a) = \begin{cases} \tau_a(=), & \text{for } \sigma_{ab} = \sigma_{ac} = \sigma_{ad}, \\ \tau_a(\neq), & \text{otherwise,} \end{cases}$$

it suffices to see that for each term $\tau_a(\boldsymbol{\sigma}_a)\sigma_{ab}\sigma_{ac}\sigma_{ad}$ in the sum, the "symmetric" term where σ_{ab}, σ_{ac} and σ_{ad} are replaced by their opposite values, has same absolute value but different sign, therefore the sum (7) is zero.

2.2 Computing Estimates of Marginal Distributions

Given the estimate $Z_\emptyset$, estimates for the marginals $P_{ab}(\sigma_{ab})$ can be obtained:

$$P_{ab}(\sigma_{ab}) = \left.\frac{\partial \log Z_\emptyset(\theta_{ab})}{\partial \theta_{ab}(\sigma_{ab})}\right|_{\theta_{ab}\to 0}, \qquad \text{where} \qquad Z_\emptyset(\theta_{ab}) := \sum_{\boldsymbol{\sigma}} \exp(\theta_{ab}\sigma_{ab}) \prod_{a\in\mathcal{V}} f_a(\boldsymbol{\sigma}_a)$$

is the partition sum of the network perturbed with respect to an additional local field potential θ_{ab} of variable σ_{ab}.

Alternatively, one can compute different partition functions for different settings of the variables $Z_\emptyset^{\sigma_{ab}=+1}$ and $Z_\emptyset^{\sigma_{ab}=-1}$, and derive the marginals from respective ratios:

$$P_{ab}(\sigma_{ab} = +1) = \frac{Z_\emptyset^{\sigma_{ab}=+1}}{Z_\emptyset^{\sigma_{ab}=+1} + Z_\emptyset^{\sigma_{ab}=-1}},$$

where $Z_\emptyset^{\sigma_{ab}=+1}$ indicates the 2-regular partition function calculated from the same model conditioning on variable σ_{ab}, that is, with variable σ_{ab} fixed (clamped) to $+1$. Therefore, approximation errors in the computation of any marginal can be related to approximation errors in the computation of $Z_\emptyset$. In the following we will mainly focus on evaluating $Z_\emptyset$, although marginal probabilities will be discussed as well.

2.3 Computing the Full Loop Series Using Perfect Matchings

Chertkov et al. (2008) established that $z_\emptyset$ is just the first term of a finite sum involving Pfaffians. We briefly reproduce their results here and provide an algorithm for computing the full loop series as a Pfaffian series.

Consider $\mathcal{T}$ defined as the set of all possible triplets in the original graph $\mathcal{G}$. For each possible subset $\Psi \in \mathcal{T}$, including an *even* number of triplets, there exists a unique correspondence between loops in $\mathcal{G}$ including the triplets in Ψ and perfect matchings in another extended graph $\mathcal{G}_{ext_\Psi}$ constructed after removal of the triplets Ψ in $\mathcal{G}$. Using this representation the full loop series can be

represented as a Pfaffian series, where each term Z_Ψ is tractable and is a product of the respective Pfaffian and the $\mu_{a;\bar{a}}$ terms associated with each triplet of Ψ:[3]

$$z = \sum_{\Psi} Z_\Psi \qquad Z_\Psi = z_\Psi \prod_{a \in \Psi} \mu_{a;\bar{a}} \qquad (8)$$
$$z_\Psi = \text{sign}\left(\text{Pfaffian}\left(\hat{B}_\Psi\right)\right) \cdot \text{Pfaffian}\left(\hat{A}_\Psi\right).$$

where $\hat{B}_\Psi$ corresponds to the original Kasteleyn matrix with weights $+1$ instead of $+\mu_{ij}$ and -1 instead of $-\mu_{ij}$ and we make explicit use of the Pfaffians to correct for the sign. The correction sign(Pfaffian($\hat{B}_\Psi$)) is necessary because loop terms can be negative and even evaluating Pfaffian($\hat{A}_\Psi$) with the correct sign would only give the contribution up to a sign. In the previous Subsection, we assumed $z_\emptyset$ positive.

The 2-regular partition function thus corresponds to $\Psi = \emptyset$. We refer to the remaining terms of the series ($\Psi \neq \emptyset$) as higher order terms. Notice that matrices $\hat{A}_\Psi$ and $\hat{B}_\Psi$ depend on the removed triplets and therefore each z_Ψ requires different matrices and different edge orientations. In addition, after removal of vertices in $\mathcal{G}$, the resulting extended graph may be disconnected. As before, in these cases we add dummy edges with zero weight so that $\mathcal{G}_{ext}$ remains bi-connected.

Figure 1d shows loops corresponding to the higher order Pfaffian terms on our illustrative example. The first and second subsets of triplets (d.1 and d.2) include summation over two loops whereas the remaining Pfaffian terms include uniquely one loop.

Exhaustive enumeration of all subsets of triplets leads to a $2^{|\mathcal{T}|}$ time algorithm, which is prohibitive. However, many triplet combinations may lead to forbidden configurations. Experimentally, we found that a principled way to look for higher order Pfaffian terms with large contribution is to search first for pairs of triplets, then groups of four, and so on. For large graphs, this also becomes intractable. However, the key advantage of the Pfaffian representation is that $Z_\emptyset$ is always the term that accounts for the largest number of loop terms in the series. In this work we do not derive any heuristic for searching higher order Pfaffian terms with larger contributions. Instead, in Section 3.1 we study the full Pfaffian series and subsequently we restrict our attention to analyze the accuracy of $Z_\emptyset$.

Algorithm 2 describes the procedure for computing all terms using Equation (8). The main loop of the algorithm can be interrupted at any time, thus leading to a sequence of algorithms producing corrections incrementally.

3. Experiments

In this Section we study numerically the proposed algorithm. To facilitate the evaluation and the comparison with other algorithms we focus on the pairwise binary Ising model, a particular case of the model (1) where factors only depend on the disagreement between variables and take the form: $\log f_a(\sigma_{ab}, \sigma_{ac}) = \phi_a \sigma_{ab} \sigma_{ac}$. We consider also nonzero local potentials in all variables parameterized by: $\log f_a(\sigma_{ab}) = \theta_{ab} \sigma_{ab}$ so that the model becomes planar-intractable.

We create different inference problems by choosing different interactions ϕ_a and local field parameters θ_{ab}. To generate them we draw independent samples from a normal distribution $\phi_a \sim \mathcal{N}(0, \beta/2)$ and $\theta_{ab} \sim \mathcal{N}(0, \beta\Theta)$, where Θ and β determine how difficult the inference problem is. Generally, for $\Theta = 0$ the planar problem is tractable (zero field). For $\Theta > 0$, small values of β result

3. We omit the loop index in the triplet term $\mu_{a;\bar{a}}$ because nodes have at most degree three and therefore the set $\bar{a}$ always coincide in all loops which contain that triplet.

Algorithm 2 Pfaffian series

Arguments: Forney graph $\mathcal{G}$

1: $z := 0$.
2: **for all** $\{\Psi \in \mathcal{T}, |\Psi| \text{even}\}$ **do**
3: Build extended graph $\mathcal{G}_{ext_\Psi}$ applying rules of Figure 2.
4: Set Pfaffian orientation in $\mathcal{G}_{ext_\Psi}$ according to Algorithm 1.
5: Build matrices $\hat{A}$ and $\hat{B}$.
6: Compute Pfaffian with sign correction z_Ψ according to Equation (8).
7: $z := z + z_\Psi \prod_{a \in \Psi} \mu_{a;\bar{a}}$.
8: **end for**
9: **RETURN** $Z^{BP} \cdot z$.

in weakly coupled variables (easy problems) and large values of β in strongly coupled variables (hard problems). Larger values of Θ result in easier inference problems.

In the next Subsection we analyze the full Pfaffian series using a small example and compare it with the original representation based on the loop series. Next, we compare our algorithm with the following ones:[4]

Truncated Loop-Series for BP (TLSBP) (Gómez et al., 2007), which accounts for a certain number of loops by performing depth-first-search on the factor graph and then merging the found loops iteratively. We adapted TSLBP as an any-time algorithm (**anyTLSBP**) in a way that the length of the loop is used as the only parameter instead of the two parameters S and M (see Gómez et al., 2007, for details). This is equivalent to setting $M = 0$ and discarding S. In this way, anyTLSBP does not compute all possible loops of a certain length (in particular, complex loops[5] are not included), but search can be performed faster.

Cluster Variation Method (**CVM-Loopk**) A double-loop implementation of CVM (Heskes et al., 2003). This algorithm is a special case of generalized belief propagation (Yedidia et al., 2005) with convergence guarantees. We use as outer clusters all (maximal) factors together with loops of four (k=4) or six (k=6) variables in the factor graph.

Tree-Structured Expectation Propagation (**TreeEP**) (Minka and Qi, 2004). This method performs exact inference on a base tree of the graphical model and approximates the other interactions. The method yields good results if the graphical model is very sparse.

When possible, we also compare with the following two variational methods which provide upper bounds on the partition function:

Tree Reweighting (**TRW**) (Wainwright et al., 2005) which decomposes the parameterization of a probabilistic graphical model as a mixture of spanning trees of the model, and then uses the convexity of the partition function to get an upper bound.

Planar graph decomposition (**PDC**) (Globerson and Jaakkola, 2007) which decomposes the parameterization of a probabilistic graphical model as a mixture of tractable planar graphs (with zero local field).

4. We use the libDAI library (Mooij, 2008) for algorithms **CVM-Loopk**, **TreeEP** and **TRW**.
5. A complex loop is defined as a loop which can not be expressed as the union of two or more circuits or simple loops.

To evaluate the accuracy of the approximations we consider errors in Z and, when possible, computational cost as well. As shown in Gómez et al. (2007), errors in Z, obtained from a truncated form of the loop series, are very similar to errors in single variable marginal probabilities, which can be obtained by conditioning over the variables under interest. We only consider tractable instances for which Z can be computed via the junction tree algorithm (Lauritzen and Spiegelhalter, 1988) using 8GB of memory.

The error measure for a given approximation Z' of Z is:

$$\text{error } Z' = \frac{|\log Z - \log Z'|}{\log Z},$$

and given $P'_{ab}(\sigma_{ab})$ estimates for the exact marginals $P_{ab}(\sigma_{ab})$, the error is:

$$\text{error marginals} = \text{mean}_{\substack{(a,b)\in\mathcal{E} \\ \sigma_{ab}=\pm 1}} |P'_{ab}(\sigma_{ab}) - P_{ab}(\sigma_{ab})|.$$

As in Gómez et al. (2007), we use four different message updates for BP: fixed and random sequential updates, parallel (or synchronous) updates, and residual belief propagation (RBP), a method proposed by Elidan et al. (2006) which selects the next message to be updated which has maximum *residual*, a quantity defined as an upper bound on the distance of the current messages from the fixed point. We report non-convergence when none of the previous methods converged. We report convergence at iteration t when the maximum absolute value of the updates of all the messages from iteration $t-1$ to t is smaller than a threshold $\vartheta = 10^{-14}$.

3.1 Full Pfaffian Series

In the previous Section we have described two equivalent representations for Z in terms of the loop series and the Pfaffian series. Here we analyze numerically how these two representations differ using an example, shown in Figure 3 as a bipartite factor graph, for which all terms of both series can be computed. We analyze a single instance, parameterized using $\Theta = 0.1$ and different pairwise interactions $\beta \in \{0.1, 0.5, 1.5\}$.

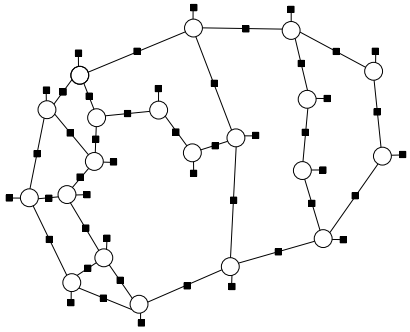


Figure 3: Planar bipartite factor graph used to compare the full Pfaffian series with the loop series. Circles and black squares denote variables and factors respectively.

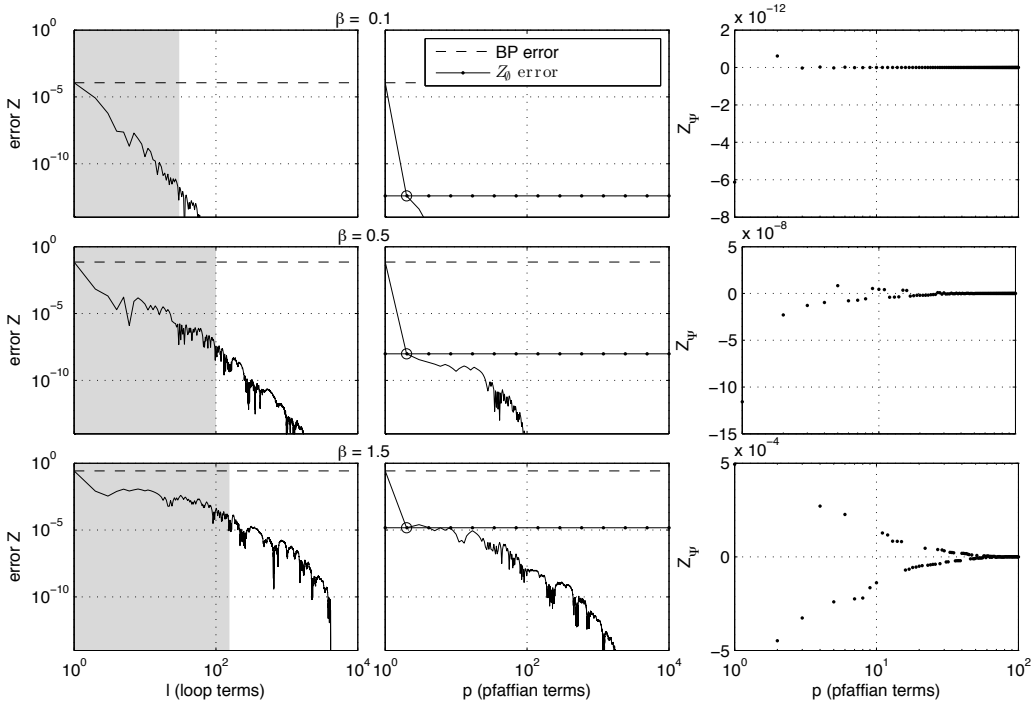


Figure 4: Comparison between the full loop series and the full Pfaffian series. Each row corresponds to a different value of the interaction strength β. **Left column** shows the error, considering loop terms $Z^{TLSBP}(l)$ in log-log scale. Shaded regions include all loop terms (not necessarily 2-regular loops) required to achieve the same (or better) accuracy than the accuracy of the 2-regular partition function $Z_\emptyset$. **Middle column** shows the error considering Pfaffian terms $Z^{Pf}(p)$ also in log-log scale. The first Pfaffian term corresponds to $Z_\emptyset$, marked by a circle. **Right column** shows the values of the first 100 Pfaffian terms sorted in descending order in $|Z_\Psi|$ and excluding $z_\emptyset$.

We use TLSBP to retrieve all loops, 8123 for this example, and Algorithm 2 to compute all Pfaffian terms. To compare the two approximations we sort all contributions, either loops or Pfaffians, by their absolute values in descending order, and then analyze how the errors are corrected as more terms are included in the approximation. We define partition functions for the truncated series in the following way:

$$Z^{TLSBP}(l) = Z^{BP}\left(1 + \sum_{i=1\ldots l} r_{C_i}\right), \qquad Z^{Pf}(p) = Z^{BP}\left(\sum_{i=1\ldots p} Z_{\Psi_i}\right).$$

Then $Z^{TLSBP}(l)$ accounts for the l most contributing loops and $Z^{Pf}(p)$ accounts for the p most contributing Pfaffian terms. In all cases, the Pfaffian term with largest absolute value Z_{Ψ_1} corresponds to $z_\emptyset$.

Figure 4 shows the error Z^{TLSBP} (first column) and Z^{Pf} (second column) for both representations. For weak interactions ($\beta = 0.1$) BP converges fast and provides an accurate approximation

with an error of order 10^{-4}. Summation of less than 50 loop terms (top-left panel) is enough to obtain machine precision accuracy. Notice that the error is almost reduced totally with $Z_\emptyset$ (top-middle panel). In this scenario, higher order terms of the Pfaffian series are negligible (top-right panel).

As we increase β, the quality of the BP approximation decreases. The number of loop corrections required to correct the BP error then increases. In this example, for intermediate interactions ($\beta = 0.5$) the first Pfaffian term $z_\emptyset$ improves considerably, more than five orders of magnitude, on the BP error, whereas approximately 100 loop terms are required to achieve a similar correction (gray region of middle-left panel).

For strong interactions ($\beta = 1.5$) BP converges after many iterations and gives a poor approximation. In this scenario also a larger proportion of loop terms (bottom-left panel) is necessary to correct the BP result up to machine precision. Looking at the bottom-left panel we find that approximately 200 loop terms are required to achieve the same correction as the one obtained by $Z_\emptyset$, which is quite accurate (bottom-middle panel).

As the right panels show, higher order Pfaffian contributions change progressively from a flat sequence of small terms to an alternating sequence of positive and negative terms which grow in absolute value as β increases. These oscillations are also present in the loop series expansion.

In general, we conclude that the $Z_\emptyset$ correction to the BP approximation can give a significant improvement even in hard problems for which BP converges after many iterations. Notice again that calculating $Z_\emptyset$, the most contributing term in the Pfaffian series, does not require explicit search for loop nor Pfaffian terms.

3.2 Grids

After analyzing the full Pfaffian series on a small random example we now restrict our attention to the first Pfaffian correction using grids (nearest neighbor connectivity). First, we compare this approximation, for both $Z_\emptyset$ and single-variable marginals, with other inference methods for different types of interactions (attractive or mixed) and then study the scalability of the method with the size of the graphs.

3.2.1 ATTRACTIVE INTERACTIONS

We first focus on binary models with "ferromagnetic" tendency, which favors alignment of neighboring variables, $\phi_a > 0$. If local fields are also positive $\theta_{ab} > 0$, Sudderth et al. (2007) showed that, under some additional condition, the BP approximation gives a *lower-bound* for the exact partition function and all loops (and therefore Pfaffian terms too) have the same sign.[6] Although this is not formally proved for general models with attractive interactions regardless of the sign of the local fields, numerical results suggest that this property holds as well for this more general type of models.

We generate grids with positive interactions and local fields, that is $|\phi_a| \sim \mathcal{N}(0, \beta/2)$ and $|\theta_{ab}| \sim \mathcal{N}(0, \beta\Theta)$, and study performance of the algorithms for various values of β, as well as for strong $\Theta = 1$ and weak $\Theta = 0.1$ local fields.

Figure 5 shows the average error over 50 instances reported by different methods. For this setup, BP converged in all instances using random sequential updates of the messages. The error curves of all methods show an initial growth and a subsequent decrease, a fact explained by the phase

6. The condition is that all single variable beliefs at the BP fixed point must satisfy $m_{ab} = \tau_{ab}(+1) - \tau_{ab}(-1) > 0, \forall (a,b) \in \mathcal{E}$.

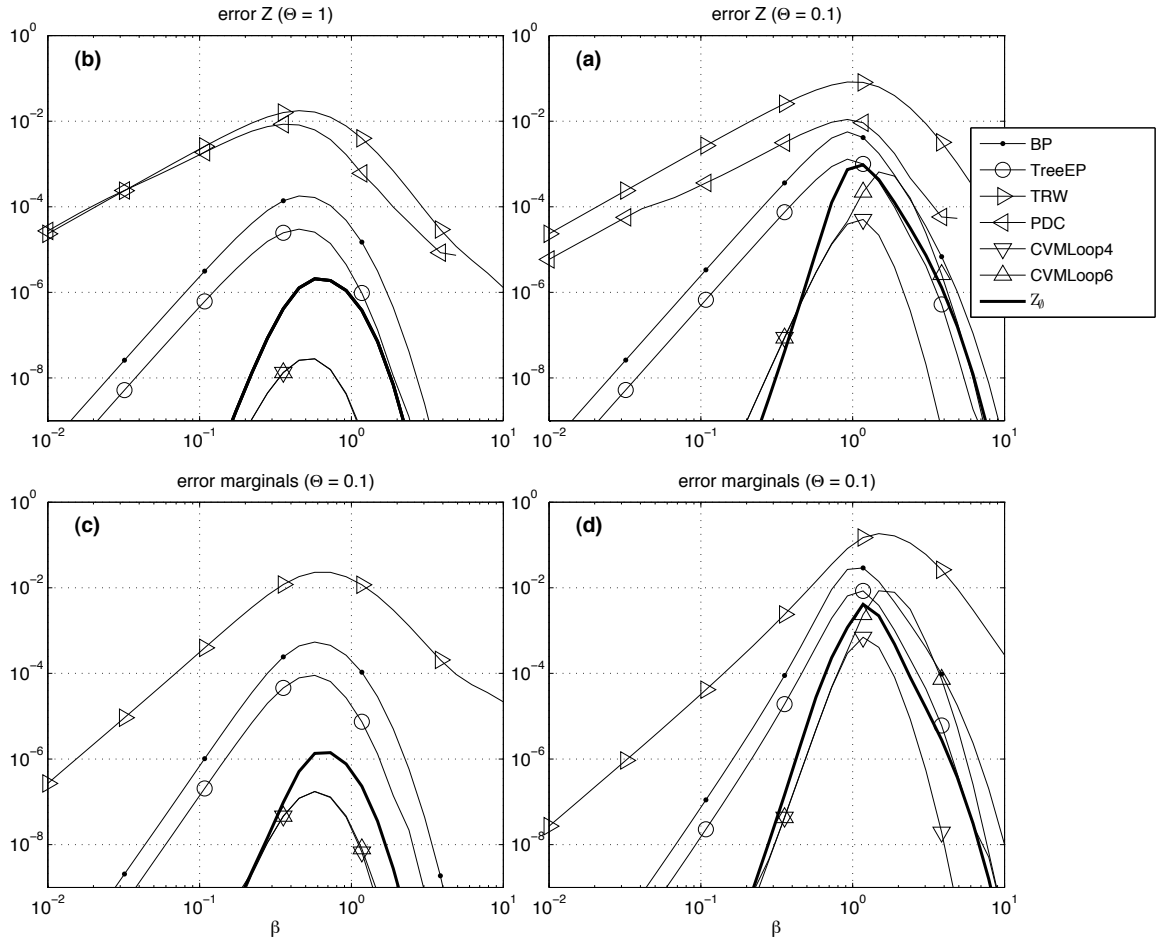


Figure 5: 7x7 grid *attractive* interactions and positive local fields. BP converged always. Errors are averages over 50 random instances for fixed β and Θ. Error in partition function Z for **(a)** strong local fields $\Theta = 1$ and **(b)** weak local fields $\Theta = 0.1$. Error in marginals for **(c)** strong local fields $\Theta = 1$ and **(d)** weak local fields $\Theta = 0.1$.

transition for $\Theta = 0$ and $\beta \approx 1$ (Mooij and Kappen, 2005). Figures suggest that errors are larger as Θ approaches zero. Notice that $Z_\emptyset = Z$ for $\Theta = 0$.

We observe that for all the instances $Z_\emptyset$ is *always an improvement* over the BP approximation. Corrections are most significant for weak interactions $\beta < 1$ and strong local fields. For strong interactions $\beta > 1$ and weak local fields, the improvement is less significant.

It appears that the $Z_\emptyset$ approximation performs better than TreeEP in all cases except for very strong couplings, where the two algorithms show very similar results. For weak local fields $Z_\emptyset$ performs similar to CVM-Loop4 which is known to be a very accurate approximation for this type of model, see Yedidia et al. (2000) for instance. Selecting larger outer-clusters such as loops of

length 6 does not necessarily leads to improvements although it leads to a dramatic increase in the computational cost.

The methods which provide upper bounds on Z (PDC and TRW) report the largest average error. PDC performs slightly better than TRW, as was shown in Globerson and Jaakkola (2007) for the case of mixed interactions. We note that the worse performance of PDC for strong couplings might be attributed to implementation artifacts, since for $\beta > 4$ the algorithm suffers from numerical precision errors. In general, both empirical upper bounds are significantly less tight than the lower bounds provided by BP and $Z_\emptyset$.

Finally, bottom plots show that errors in marginals behave very similar to errors in Z.

3.2.2 MIXED INTERACTIONS

We now analyze a more general Ising model where interactions and local fields can have mixed signs. In that case, Z^{BP} and $Z_\emptyset$ are no longer lower bounds on Z and loop terms can be positive or negative. Figure 6 shows results for this setup.

For strong local fields (subplots a,c,e), we observe that $Z_\emptyset$ improvements over BP results become less significant as β increases. It is important to note that $Z_\emptyset$ always improves the BP result, even when the couplings are very strong ($\beta = 10$) and BP fails to converge for a small percentage of instances. $Z_\emptyset$ performs very similar to CVM-Loop4 and substantially better than TreeEP for small and intermediate β. As in the case of attractive interactions, the best results are attained using CVM-loop4. CVM-loop6 gives worse estimates for $\beta > 1$.

For the case of weak local fields (subplots b,d,f), $Z_\emptyset$ is the best approximation in the weak coupling regime. BP fails to converge near the transition to the spin-glass phase. For $\beta = 10$, BP converges only in less than 25% of the instances. In the most difficult domain, $\beta > 22$, all methods under consideration give similar results (all comparable to BP). Moreover, it may happen that $Z_\emptyset$ degrades the Z^{BP} approximation because loops of alternating signs have major influence in the series. This result was also reported in Gómez et al. (2007) when loop terms, instead of Pfaffian terms, where considered.

Finally, as in the case of attractive interactions, errors in marginals behave similar to errors in Z.

3.2.3 SCALING WITH GRAPH SIZE

We now study how the accuracy of the $Z_\emptyset$ approximation changes as we increase the size of the grid. We generate random grids with mixed couplings for $\sqrt{N} = \{4, ..., 18\}$ and focus on a regime of very weak local fields $\Theta = 0.01$ and strong couplings $\beta = 1$, a difficult configuration according to the previous results. We compare $Z_\emptyset$ also with anyTLSBP, a variant of our previous algorithm for truncating the loop series. We run anyTLSBP by selecting loops shorter than a given length, and the length is increased progressively. To provide a fair comparison between both methods, we run anyTLSBP for the same amount of CPU time as the one required to obtain $Z_\emptyset$.

Figure 7a shows a comparison of the errors reported by the different algorithms. Since variability in the errors is larger than before, we take the median for comparison. In order of increasing accuracy we get BP, TreeEP, anyTLSBP, CVM-Loop6, CVM-Loop4 and $Z_\emptyset$. We note again that using larger clusters in CVM does not necessarily result in better performance.

Overall, we can see that results are roughly independent of the network size N in almost all methods that we compare. The error of anyTLSBP starts being the smallest but soon increases because the proportion of loops captured decreases very fast. For $N > 64$, anyTLSBP performs

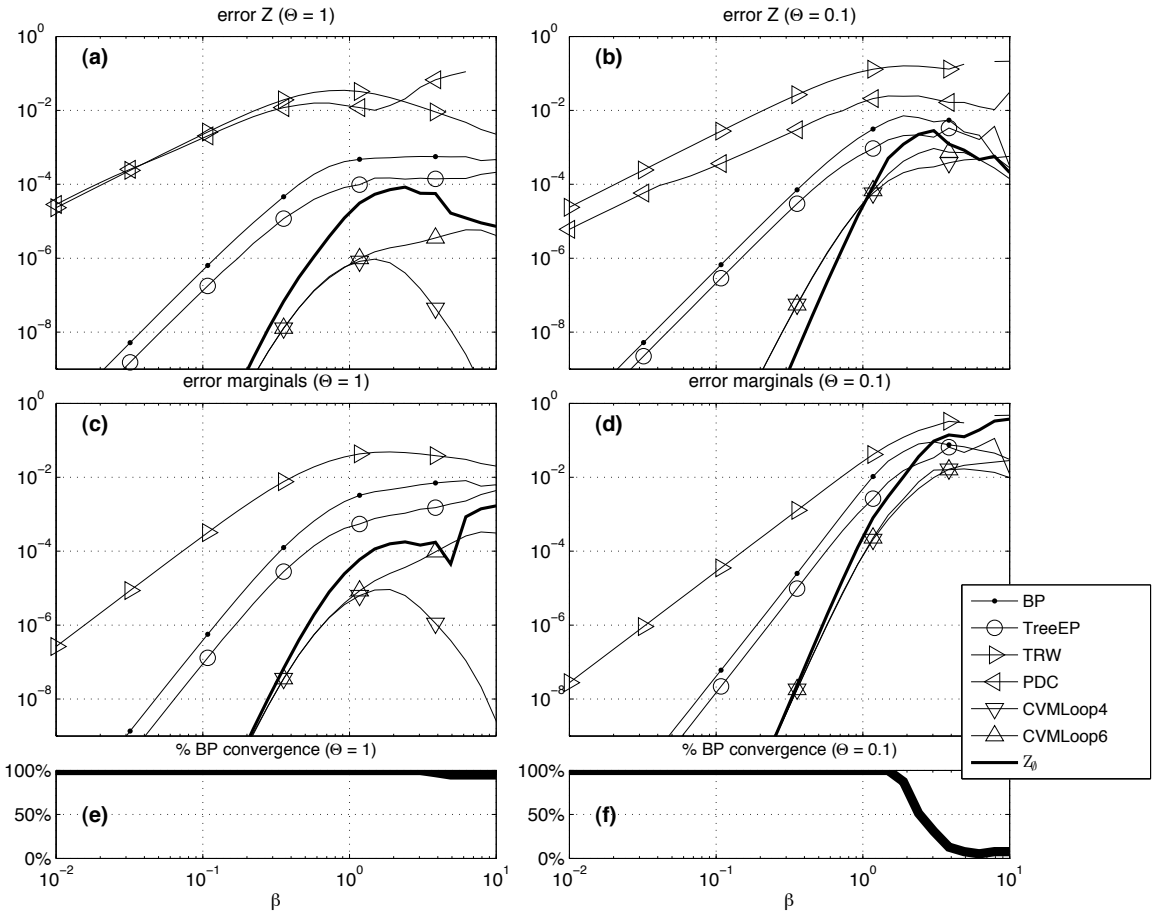


Figure 6: 7x7 grid *mixed* interactions and positive local fields. Errors are averages over 50 random instances for fixed β and Θ. Error in partition function Z for **(a)** strong local fields $\Theta = 1$ and **(b)** weak local fields $\Theta = 0.1$. Error in marginals for **(c)** strong local fields $\Theta = 1$ and **(d)** weak local fields $\Theta = 0.1$. Bottom panels show percentage of cases when BP converges using at least one of the methods described above for **(e)** strong local fields and **(f)** weak local fields.

worse than CVM. The $Z_\emptyset$ correction, on the other hand, stays flat and we can conclude that it scales reasonably well. Interestingly, although $Z_\emptyset$ and anyTLSBP use different ways to truncate the loop series, both methods show similar scaling behaviour for large graphs.

Figure 7b shows the CPU time for all the tested approaches averaged over all the cases. The CPU time of the junction tree method quickly increases with the tree-width of the graphs. For large enough N, exact solution via the junction tree method is no longer feasible because of the memory requirements. In contrast, for all the approximate inference methods, memory demands do not represent a limitation.

In order of increasing cost we have BP, $Z_\emptyset$ with anyTLSBP, TreeEP, CVM-Loop4 and CVM-Loop6. The $Z_\emptyset$ therefore is a very efficient correction to Z^{BP}.

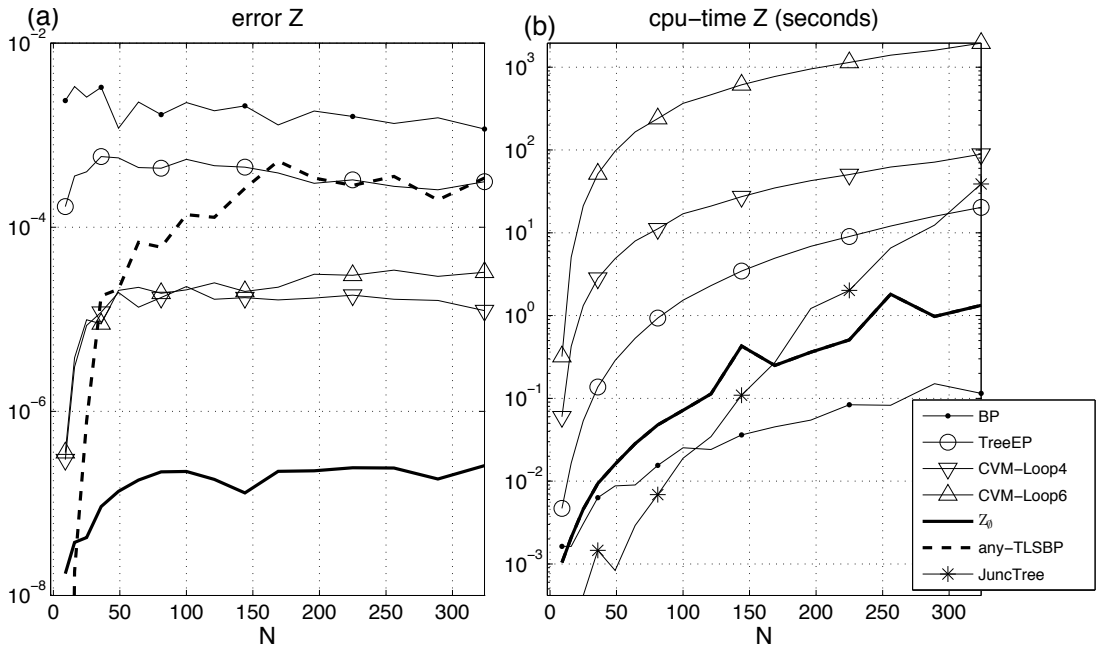


Figure 7: Results on regular grids: scaling with grid size for strong interactions $\beta = 1$ and very weak local fields $\Theta = 0.01$. Error medians over 50 random instances. BP converged always. **(a)** Partition function Z error. **(b)** CPU time to compute Z.

3.3 Radial Grid Graphs

In the previous Subsection we analyzed the quality of the $Z_\emptyset$ correction for graphs with a regular grid structure. Here, we carry over the analysis of the $Z_\emptyset$ correction using planar graphs which consist of concentric polygons with a variable number of sides. Figure 8 illustrates these spider-web graphs. We generate them as factor graphs with pairwise interactions which we subsequently convert to an equivalent Forney graph using the procedure described in Appendix A.2. Again, local field potentials are parameterized using $\Theta = 0.01$ and interactions using $\beta = 1$. We analyze the error in Z as a function of the degree d of the central node.

Figure 9a shows the median of errors in Z for 50 random instances. First, we see that the variability of all the methods, in particular anyTLSBP, is larger than in the regular grid scenario. CVM-Loop6 does not converge for instances with $d > 4$ before 10^4 seconds and it is thus not included in the analysis. We can say that all approaches scale reasonably well, and as d grows, the errors become independent of d.

The $Z_\emptyset$ approximation is the best method compared to the other tested approaches. The improvements of $Z_\emptyset$ over CVM-Loop4 (the second best method) can be more than two orders of magnitude and more than three orders of magnitude compared to BP.

Computational costs are shown in 9b. Again, for larger graphs, exact solution via the junction tree is not feasible due to the large tree-width and $Z_\emptyset$ represent the most efficient correction which improves BP of all approximate methods we compared.

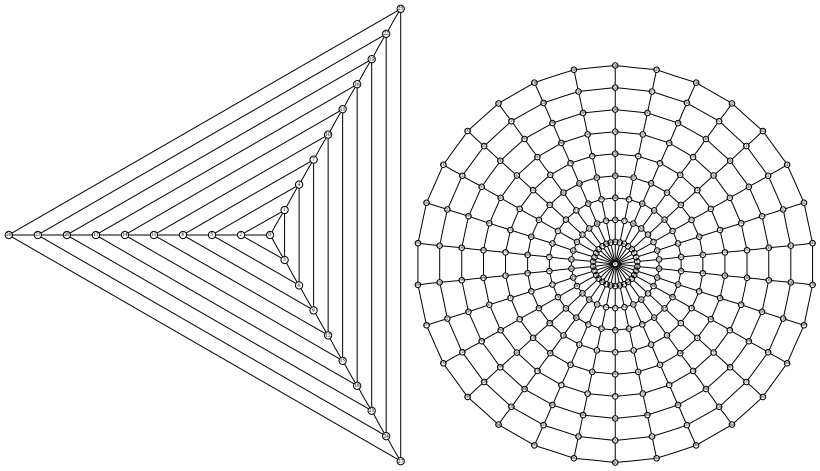


Figure 8: Two examples of planar graphs used for comparison between methods. We fix the number of concentric polygons to 9 and change the degree d of the central node within the range $[3,...,25]$. **(left)** Graph for $d = 3$. **(right)** Graph for $d = 25$. Here nodes represent variables and edges pairwise interactions. We also add external fields which depend on the state of each node (not drawn).

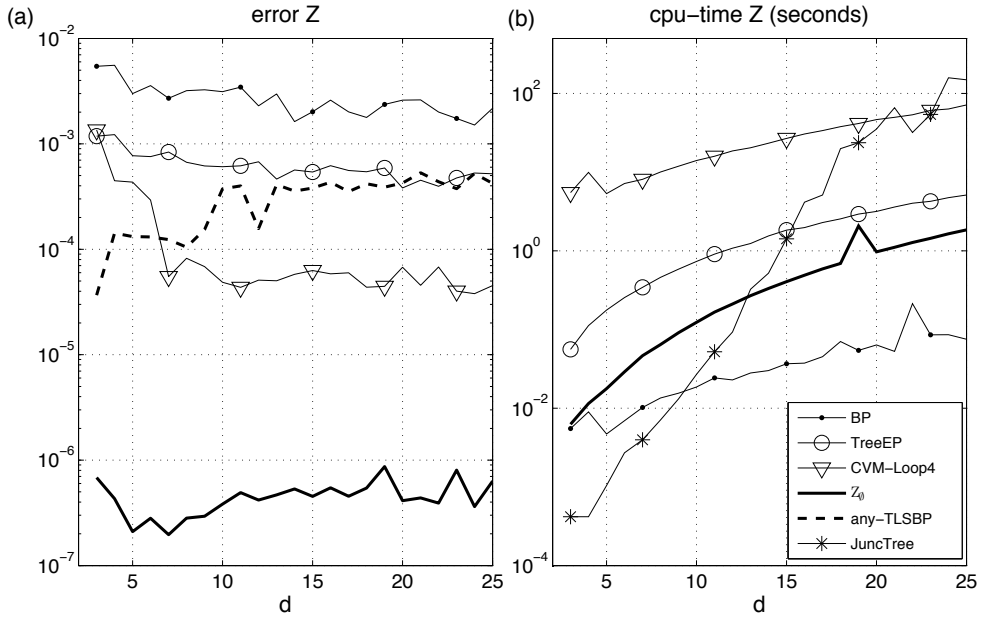


Figure 9: Results on spider-web graphs: scaling with the degree d of the central node for $\beta = 1$ and $\Theta = 0.01$. BP converged always. **(a)** Median of the error in the partition function Z over 50 random instances. **(b)** CPU time required to compute Z.

4. Discussion

We have presented an approximate algorithm based on the exact loop calculus framework for inference on planar graphical models defined in terms of binary variables. The proposed approach improves the estimate for the partition function provided by BP without an explicit search for loops.

The algorithm is illustrated on the example of ordered and disordered Ising model on a planar graph. Performance of the method is analyzed in terms of its dependence on the system size. The complexity of the partition function computation is exponential in the general case, unless the local fields are zero, when it becomes polynomial. We tested our algorithm on regular grids and planar graphs with different structures. Our experiments on regular grids show that significant improvements over BP are always obtained if single variable potentials (local magnetic fields) are sufficiently large. The quality of this correction degrades with decrease in the amplitude of external field, to become exact at zero external fields. This suggests that the difficulty of the inference task changes from very easy, with no local fields, to very hard, with small local fields, and then decays again as external fields become larger.

The $Z_\emptyset$ correction turns out to be competitive with other state of the art methods for approximate inference of the partition function. First of all, we showed that $Z_\emptyset$ is much more accurate than upper bounds based methods such as TRW or PDC, illustrating that such methods come at the cost of accuracy. We have also shown that for the case of grids with attractive symmetric interactions and positive local fields, the lower bound provided by $Z_\emptyset$ is the most accurate.

We also found that $Z_\emptyset$ performs much better than treeEP for weak and intermediate interactions and similar for strong interactions. Comparing with CVM, we have found that $Z_\emptyset$ presented better results for very small local fields. Using larger outer clusters in CVM does not necessarily lead to better approximations.

Finally, we have presented a comparison of $Z_\emptyset$ with TLSBP, which is another algorithm based on the loop series expansion for BP that uses the loop length as truncation parameter. On one hand, the calculation of $Z_\emptyset$ involves a *re-summation* of many loop terms which in the case of TLSBP are summed individually. This consideration favors the $Z_\emptyset$ approach. On the other hand, $Z_\emptyset$ is restricted to the class of 2-regular loops whereas TLSBP also accounts for terms corresponding to more complex loop structures in which nodes can have degree larger than two. Overall, for planar graphs, we have shown evidence that the $Z_\emptyset$ approach is better than TLSBP when the size of the graphs is not very small. We emphasize, however, that TLSBP can be applied to non-planar binary graphical models too.

Although planarity is a severe restriction, we emphasize that planar graphs appear in many contexts such as computer vision and image processing, magnetic and optical recording, or network routing and logistics. We have focused on inference problems defined on planar graphs with symmetric pairwise interactions and, to make the problems difficult, we have introduced local field potentials. Notice however, that the algorithm can also be used to solve models with more complex interactions, that is, more than pairwise typical from the Ising model (see Chertkov et al., 2008, for a discussion of possible generalizations). This makes our approach more applicable than other approaches, namely, (Globerson and Jaakkola, 2007; Schraudolph and Kamenetsky, 2008), designed specifically for the pairwise interaction case.

Summarizing, among the compared methods and models, the introduced approach based on $Z_\emptyset$ represents the best estimate to the partition function as long as the given graph does not deviate too much from perfect planarity, that is, we are in the small local field regime. It also represents

an efficient correction to the BP estimate of the partition function, being much more efficient than CVM, the second best method we have analyzed. Finally, we have shown that estimates of single-variable marginals calculated using $Z_\emptyset$ are comparable to state of the art inference methods.

Acknowledgments

We acknowledge J. M. Mooij, A. Windsor and A. Globerson for providing their software, V. Y. Chernyak, J. K. Johnson and N. Schraudolph for interesting discussions and anonymous reviewers for valuable suggestions. This research is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024. The work at LANL was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at LANL under Contract No. DE-AC52-06NA25396.

Appendix A. Bipartite Factor Graph Representation

The Forney graph representation is convenient because each loop decomposes naturally in terms associated with nodes which have the same analytical form of Equation (4). However, probabilistic models are more frequently represented as bipartite factor graphs. In this Appendix we first show how the presented approach differs when it is directly applied to a bipartite factor graph and second, how to convert a bipartite factor graph to a Forney graph.

We consider binary variables i which take values $x_i \in \{\pm 1\}$ and factor functions $\psi_a(\mathbf{x}_a)$ defined over subsets a of variables which take values $\mathbf{x}_a := \{x_i | i \in a\}$. On a bipartite factor graph $\mathcal{G}_{fg} := (\mathcal{V}_{fg}, \mathcal{E}_{fg})$, the set $\mathcal{V}_{fg}$ consists of variable nodes $i \in I$ and factor nodes $a \in \mathcal{A}$, with an edge between i and a iff $i \in a$, that is i appears in the factor function ψ_a.

The joint probability distribution of vector $\mathbf{x} := \{x_i | i \in \mathcal{V}\}$ is specified as:

$$p(\mathbf{x}) = \frac{1}{Z(\psi)} \prod_{a \in \mathcal{A}} \psi_a(\mathbf{x}_a), \qquad Z(\psi) = \sum_{\mathbf{x}} \prod_{a \in \mathcal{A}} \psi_a(\mathbf{x}_a),$$

where we have stressed the dependency of Z on the potential functions ψ.

A.1 Loop Calculus for Standard Factor-Graph Model

Following Sudderth et al. (2007), one can use the reparameterized model in terms of the factor pseudo-marginals $\tau_a(\mathbf{x}_a)$ and the variable pseudo-marginals $\tau_i(x_i)$ associated with the minimum of the Bethe free energy:

$$p(\mathbf{x}) = \frac{1}{Z(\tau)} \prod_{i \in I} \tau_i(x_i) \prod_{a \in \mathcal{A}} \frac{\tau_a(\mathbf{x}_a)}{\prod_{j \in a} \tau_j(x_j)},$$

and express the relation between the exact $Z(\psi)$ and the Bethe partition functions $Z^{BP}(\psi;\tau)$ as $Z(\psi)/Z(\tau) = Z^{BP}(\psi;\tau)$. The loop series correction becomes:

$$Z(\tau) = 1 + \sum_{C \in \mathcal{C}} s_C, \qquad s_C = \prod_{a \in C} \mu_a(C) \prod_{i \in C} \mu_i(C),$$

where each term s_C is associated with a loop and can be specified as a product between terms $\mu_a(C)$ and $\mu_i(C)$ corresponding to factor nodes a and variable nodes i in C respectively:

$$\mu_a(C) = \frac{\mathbb{E}_{\tau_a}\left[\prod_{i\in a, i\in C}(x_i - m_i)\right]}{\prod_{i\in a, i\in C}\mathrm{Var}_{\tau_i}(x_i)} \qquad \mu_i(C) = \mathbb{E}_{\tau_i}\left[(x_i - m_i)^{d_i(C)}\right],$$

where $d_i(C)$ is the degree of variable node i in the loop C and m_i is again $\mathbb{E}_{\tau_i}[x_i]$. For the $x_i = \{\pm 1\}$ alphabet, we have $m_i = \tau_i(+1) - \tau_i(-1)$ and the formulas become:

$$\begin{aligned}
\mu_a(C) &= \frac{\sum_{\mathbf{x}_a}\tau_a(\mathbf{x}_a)\prod_{i\in a, i\in C}(x_i - m_i)}{\prod_{i\in a, i\in C}\tau_i(+1)(1-m_i)^2 + \tau_i(-1)(-1-m_i)^2} \\
&= \frac{\sum_{\mathbf{x}_a}\tau_a(\mathbf{x}_a)\prod_{i\in a, i\in C}(x_i - m_i)}{\prod_{i\in a, i\in C}\frac{1+m_i}{2}(1-m_i)^2 + \frac{1-m_i}{2}(1+m_i)^2} \\
&= \frac{\sum_{\mathbf{x}_a}\tau_a(\mathbf{x}_a)\prod_{i\in a, i\in C}(x_i - m_i)}{\prod_{i\in a, i\in C}(1-m_i^2)}, \qquad (9) \\
\mu_i(C) &= \tau_i(+1)(1-m_i)^{d_i(C)} + \tau_i(-1)(-1-m_i)^{d_i(C)} \\
&= \frac{1+m_i}{2}(1-m_i)(1-m_i)^{d_i(C)-1} + \frac{1-m_i}{2}(1+m_i)(-1)^{d_i(C)}(1+m_i)^{d_i(C)-1} \\
&= \frac{1-m_i^2}{2}\Big((1-m_i)^{d_i(C)-1} + (-1)^{d_i(C)}(1+m_i)^{d_i(C)-1}\Big). \qquad (10)
\end{aligned}$$

One can recover the original formulation of Chertkov and Chernyak (2006a, Pag.5) reallocating the denominator of (9) to (10) and simplifying.

Now consider Figure 2 where the construction of the extended graph $\mathcal{G}_{ext}$ is described. Clearly, the one-to-one correspondence from 2-regular loops in $\mathcal{G}$ to perfect matchings in $\mathcal{G}_{ext}$ is independent of whether $\mathcal{G}$ is expressed as a bipartite factor graph or a Forney graph. For a bipartite factor graph, the terms of Equation (5) are replaced with terms given by Equation (10) if the expanded node is a variable or by Equation (9) if the expanded node is a factor. As previously stated, the Forney style allows to express in the same form the weights of the extended graph $\mathcal{G}_{ext}$.

A.2 Converting a Bipartite Factor Graph to a Forney-Style Factor Graph.

We show here how to convert a bipartite factor graph defined in terms of binary variables to a more general Forney graph representation, for which the presented algorithm can be directly applied to.

We label variable and factor nodes using numbers and capital letters respectively. Thus $i \in I, i := \{1,2,\dots\}$ represents a variable and $a = \{A,B,\dots\}, a \in \mathcal{A}$ a factor. Given $\mathcal{G}_{fg} := (\mathcal{V}_{fg}, \mathcal{E}_{fg})$, a direct way to obtain an equivalent Forney graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ is: first, to create a node $\delta_i \in \mathcal{V}$ for each variable node $i \in \mathcal{V}_{fg}$, and second, to associate a new binary variable $\delta_i a$ with values $\sigma_{\delta_i a} = \{\pm 1\}$ to edges $(\delta_i, a) \in \mathcal{E}$. Nodes $\delta_i \in \mathcal{V}$ are *equivalent factor nodes* denoting the characteristic function: $f_{\delta_i}(\boldsymbol{\sigma}_{\delta_i}) = 1$ if $\sigma_{\delta_i a} = \sigma_{\delta_i b}, \forall a,b \in \bar{\delta}_i$ and zero otherwise. Finally, factor nodes $c \in \mathcal{V}_{fg}$ with associated functions $\psi_c(\mathbf{x}_c)$ correspond to the same factor nodes c in $\mathcal{V}$ with same associated functions $f_c(\boldsymbol{\sigma}_c)$ and defined in terms of the new variables $\delta_i c, \forall i \in \bar{c}$. Figure 10 shows an example of this transformation. Notice that, although we impose a direction in the edge labels, they remain undirected: $(\delta_i, a) = (a, \delta_i), \forall \delta_i, a \in \mathcal{V}$. For variables $i \in \mathcal{V}_{fg}$ which only appear in two factors, such as variable 3 in Figure 10a, the corresponding node δ_3 is redundant and can be removed. The joint

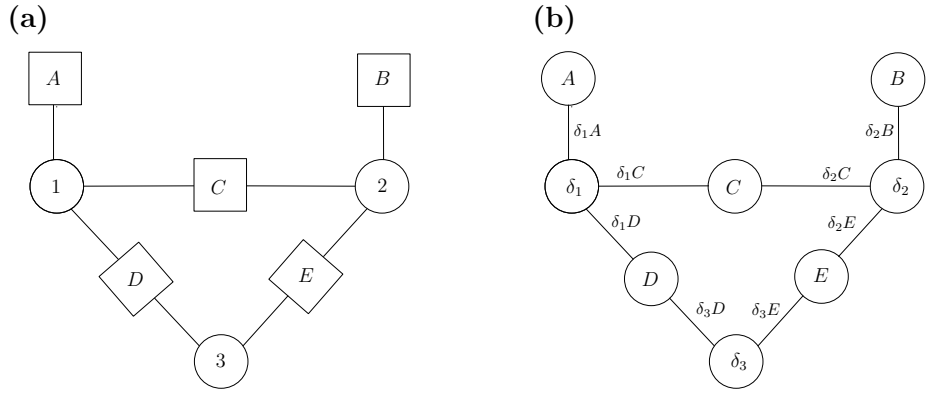


Figure 10: **(a)** Bipartite factor graph $\mathcal{G}_{fg}$ where squares represent factors and circles represent variables. **(b)** Equivalent Forney-style factor graph $\mathcal{G}$ where factors reside in the nodes and variables in the edges.

distribution of $\mathcal{G}_{fg}$ is related to the joint distribution of $\mathcal{G}$ by:

$$\begin{aligned}
&\frac{1}{Z}\psi_A(x_1)\psi_B(x_2)\psi_C(x_1,x_2)\psi_D(x_1,x_3)\psi_E(x_2,x_3)\\
\equiv&\frac{1}{Z}f_A(\sigma_{\delta_1 A})f_B(\sigma_{\delta_2 B})f_C(\sigma_{\delta_1 C},\sigma_{\delta_2 C})f_D(\sigma_{\delta_1 D},\sigma_{\delta_3 D})f_E(\sigma_{\delta_2 E},\sigma_{\delta_3 E})\\
&f_{\delta_1}(\sigma_{\delta_1 A},\sigma_{\delta_1 C},\sigma_{\delta_1 D})f_{\delta_2}(\sigma_{\delta_2 B},\sigma_{\delta_2 C},\sigma_{\delta_2 E})f_{\delta_3}(\sigma_{\delta_3 D},\sigma_{\delta_3 E}).
\end{aligned}$$

Once $\mathcal{G}$ has been generated following the previous procedure it may be the case that the nodes $\delta_i \in \mathcal{V}$ have degree three or larger. This happens if a variable i appears in more than three factors on $\mathcal{G}_{fg}$. It is easy to convert $\mathcal{G}$ to a graph were all δ_i nodes have maximum degree three by introducing new auxiliary variables $\delta_{i_1}, \delta_{i_2}, \ldots$ and new equivalent nodes. For instance, if variable $i \in \mathcal{V}_{fg}$ appears in four factors A, B, C, D:

$$f_{\delta_i}(\sigma_{\delta_i A},\sigma_{\delta_i B},\sigma_{\delta_i C},\sigma_{\delta_i D}) \equiv f_{\delta_{i_1}}(\sigma_{\delta_i A},\sigma_{\delta_i B},\sigma_{\delta_{i_1}})f_{\delta_{i_2}}(\sigma_{\delta_{i_1}},\sigma_{\delta_i C},\sigma_{\delta_i D}).$$

Notice that although the models are equivalent, the number of loops in $\mathcal{G}$ may be larger than in $\mathcal{G}_{fg}$. In the case that a factor in $\mathcal{G}_{fg}$ involves more than three variables, as sketched in Chertkov et al. (2008), one could split the node of degree N into auxiliary nodes of degree $N-1$ and compute $Z_\emptyset$ on the transformed model. Alternatively, one can reduce the number of variables that enter a factor conditioning over the variables.

References

F. Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982. URL `http://stacks.iop.org/0305-4470/15/3241`.

M. Chertkov and V. Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(06):P06009, 2006a.

M. Chertkov and V. Y. Chernyak. Loop calculus helps to improve belief propagation and linear programming decodings of LDPC codes. In *invited talk at 44th Allerton Conference*, September 2006b.

M. Chertkov, V. Y. Chernyak, and R. Teodorescu. Belief propagation and loop series on planar graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(05):P05003 (19pp), 2008. URL http://stacks.iop.org/1742-5468/2008/P05003.

G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, Boston, Massachussetts, 2006. AUAI Press.

M. E. Fisher. On the dimer solution of the planar Ising model. *Journal of Mathematical Physics*, 7 (10):1776–1781, 1966.

M. E. Fisher. Statistical mechanics of dimers on a plane lattice. *Physical Review*, 124:1664–1672, 1961.

B. J. Frey and D. J. C. MacKay. A revolution: belief propagation in graphs with cycles. In *Advances in Neural Information Processing Systems 10*, pages 479–486. MIT Press, Cambridge, MA, 1998.

A. Globerson and T. S. Jaakkola. Approximate inference using planar graph decomposition. In *Advances in Neural Information Processing Systems 19*, pages 473–480. MIT Press, Cambridge, MA, 2007.

V. Gómez, J. M. Mooij, and H. J. Kappen. Truncating the loop series expansion for belief propagation. *Journal of Machine Learning Research*, 8:1987–2016, 2007. ISSN 1533-7928.

T. Heskes, K. Albers, and H. J. Kappen. Approximate inference and constrained optimization. In *Proceedings of the 19th Annual conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 313–320, San Francisco, CA, 2003. Morgan Kaufmann Publishers.

M. Jerrum. *Counting, Sampling and Integrating : Algorithms and Complexity*. Lectures in Mathematics - ETH Zrich. Birkhuser, Basel, 2003.

G.D. Forney Jr. Codes on graphs: normal realizations. *IEEE Transactions on Information Theory*, 47(2):520–548, Feb 2001. ISSN 0018-9448. doi: 10.1109/18.910573.

M. Karpinski and W. Rytter. Fast parallel algorithms for graph matching problems. In *Oxford Lecture Series in Mathematics & Its Applications Number 9*, pages 164–170. Oxford University Press, USA, 1998.

P. W. Kasteleyn. The statistics of dimers on a lattice : I. The number of dimer arrangements on a quadratic lattice. *Physica*, 27:1209–1225, 1961.

P. W. Kasteleyn. Dimer statistics and phase transitions. *Journal of Mathematical Physics*, 4(2): 287–293, 1963. URL http://link.aip.org/link/?JMP/4/287/1.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical society. Series B-Methodological*, 50(2):154–227, 1988.

H.-A. Loeliger. An introduction to factor graphs. *Signal Processing Magazine, IEEE*, 21(1):28–41, 2004. ISSN 1053-5888. doi: 10.1109/MSP.2004.1267047.

T. Minka and Y. Qi. Tree-structured approximations by expectation propagation. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

J. M. Mooij. libDAI: A free/open source C++ library for discrete approximate inference methods, 2008. `http://mloss.org/software/view/77/`.

J. M. Mooij and H. J. Kappen. On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005 (11):P11012, 2005.

K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy Belief Propagation for approximate inference: An empirical study. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 467–475, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco, CA, 1988. ISBN 1558604790.

N. Schraudolph and D. Kamenetsky. Efficient exact inference in planar Ising models. In *Advances in Neural Information Processing Systems 22*. MIT Press, Cambridge, MA, 2008.

E. B. Sudderth, M. J. Wainwright, and A. S. Willsky. Loop series and Bethe variational bounds in attractive graphical models. In *Advances in Neural Information Processing Systems 20*, pages 1425–1432. MIT Press, Cambridge, MA, 2007.

H. N. V. Temperley and M. E. Fisher. Dimer problem in statistical mechanics - an exact result. *Philosophical Magazine*, 6:1061–1063, 1961.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.

Y. Watanabe and K. Fukumizu. Loop series expansion with propagation diagrams. *Journal of Physics A: Mathematical and Theoretical*, 42(4):045001 (18pp), 2009.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems 13*, pages 689–695, 2000.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.

Journal of Machine Learning Research 11 (2010) 1297-1322 Submitted 9/09; Revised 2/10; Published 4/10

Learning From Crowds

Vikas C. Raykar VIKAS.RAYKAR@SIEMENS.COM
Shipeng Yu SHIPENG.YU@SIEMENS.COM
CAD and Knowledge Solutions (IKM CKS)
Siemens Healthcare
Malvern, PA 19355 USA

Linda H. Zhao LZHAO@WHARTON.UPENN.EDU
Department of Statistics
University of Pennsylvania
Philadelphia, PA 19104 USA

Gerardo Hermosillo Valadez GERARDO.HERMOSILLOVALADEZ@SIEMENS.COM
Charles Florin CHARLES.FLORIN@SIEMENS.COM
Luca Bogoni LUCA.BOGONI@SIEMENS.COM
CAD and Knowledge Solutions (IKM CKS)
Siemens Healthcare
Malvern, PA 19355 USA

Linda Moy LINDA.MOY@NYUMC.ORG
Department of Radiology
New York University School of Medicine
New York, NY 10016 USA

Editor: David Blei

Abstract

For many supervised learning tasks it may be infeasible (or very expensive) to obtain objective and reliable labels. Instead, we can collect subjective (possibly noisy) labels from multiple experts or annotators. In practice, there is a substantial amount of disagreement among the annotators, and hence it is of great practical interest to address conventional supervised learning problems in this scenario. In this paper we describe a probabilistic approach for supervised learning when we have multiple annotators providing (possibly noisy) labels but no absolute gold standard. The proposed algorithm evaluates the different experts and also gives an estimate of the actual hidden labels. Experimental results indicate that the proposed method is superior to the commonly used majority voting baseline.

Keywords: multiple annotators, multiple experts, multiple teachers, crowdsourcing

1. Supervised Learning From Multiple Annotators/Experts

A typical supervised learning scenario consists of a training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ containing N instances, where $\boldsymbol{x}_i \in \mathcal{X}$ is an instance (typically a d-dimensional feature vector) and $y_i \in \mathcal{Y}$ is the corresponding known label. The task is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ which generalizes well on unseen data. Specifically for binary classification the supervision is from the set $\mathcal{Y} = \{0,1\}$, for multi-class classification $\mathcal{Y} = \{1,\ldots,K\}$, for ordinal regression $\mathcal{Y} = \{1,\ldots,K\}$ (with an ordering $1 < \ldots < K$), and $\mathcal{Y} = \mathbb{R}$ for regression.

However, for many real life tasks, it may not be possible, or may be too expensive (or tedious) to acquire the actual label y_i for training—which we refer to as the *gold standard* or the *objective ground truth*. Instead, we may have multiple (possibly noisy) labels $y_i^1, \ldots, y_i^R$ provided by R different experts or annotators. In practice, there is a substantial amount of disagreement among the experts, and hence it is of great practical interest to address conventional supervised learning algorithms in this scenario.

Our motivation for this work comes from the area of computer-aided diagnosis[1] (CAD), where the task is to build a classifier to predict whether a suspicious region on a medical image (like a X-ray, CT scan, or MRI) is malignant (cancerous) or benign. In order to train such a classifier, a set of images is collected from hospitals. The actual gold standard (whether it is cancer or not) can only be obtained from a biopsy of the tissue. Since it is an expensive, invasive, and potentially dangerous process, often CAD systems are built from labels assigned by *multiple radiologists* who identify the locations of malignant lesions. Each radiologist visually examines the medical images and provides a *subjective* (possibly noisy) version of the gold standard.[2] The radiologist also annotates various descriptors of the potentially malignant lesion, like the size (a regression problem), shape (a multi-class classification problem), and also degree of malignancy (an ordinal regression problem). The radiologists come from a diverse pool including luminaries, experts, residents, and novices. Very often there is lot of disagreement among the annotations.

For a lot of tasks the labels provided by the annotators are inherently *subjective* and there will be substantial variation among different annotators. The domain of text classification offers such a scenario. In this context the task is to predict the category for a token of text. The labels for training are assigned by human annotators who read the text and attribute their subjective category. With the advent of crowdsourcing (Howe, 2008) services like Amazon's Mechanical Turk,[3] Games with a Purpose,[4] and reCAPTCHA[5] it is quite inexpensive to acquire labels from a large number of annotators (possibly thousands) in a short time (Sheng et al., 2008; Snow et al., 2008; Sorokin and Forsyth, 2008). Websites such as Galaxy Zoo[6] allow the public to label astronomical images over the internet. In situations like these, the performance of different annotators can vary widely (some may even be malicious), and without the actual gold standard, it may not be possible to evaluate the annotators.

In this work, we provide principled probabilistic solutions to the following questions:

1. How to adapt conventional supervised learning algorithms when we have multiple annotators providing subjective labels but no objective gold standard?

2. How to evaluate systems when we do not have absolute gold-standard?

3. A closely related problem—particularly relevant when there are a large number of annotators—is to estimate how reliable/trustworthy is each annotator.

1. See Fung et al. (2009) for an overview of the data mining issues in this area.
2. Sometimes even a biopsy cannot confirm whether it is cancer or not and hence all we can hope to get is subjective ground truth.
3. Mechanical Turk found at `https://www.mturk.com`.
4. Games with a Purpose found at `http://www.gwap.com`.
5. reCAPTCHA found at `http://recaptcha.net/`.
6. Galaxy Zoo found at `http://galaxyzoo.org`.

1.1 The Problem With Majority Voting

When we have multiple labels a commonly used strategy is to use the labels on which the majority of them agree (or average for regression problem) as an estimate of the actual gold standard. For binary classification problems this amounts to using the majority label,[7] that is,

$$\hat{y}_i = \begin{cases} 1 & \text{if } (1/R)\sum_{j=1}^{R} y_i^j > 0.5 \\ 0 & \text{if } (1/R)\sum_{j=1}^{R} y_i^j < 0.5 \end{cases},$$

as an *estimate of the hidden true label* and use this estimate to learn and evaluate classifiers/annotators. Another strategy is that of considering every pair (instance, label) provided by each expert as a separate example. Note that this amounts to using a soft probabilistic estimate of the actual ground truth to learn the classifier, that is,

$$\Pr[y_i = 1 | y_i^1, \ldots, y_i^R] = (1/R)\sum_{j=1}^{R} y_i^j.$$

Majority voting assumes all experts are equally good. However, for example, if there is only one true expert and the majority are novices, and if novices give the same incorrect label to a specific instance, then the majority voting method would favor the novices since they are in a majority. One could address this problem by introducing a weight capturing how good each expert is. But how would one measure the performance of an expert when there is no gold standard available?

1.2 Proposed Approach and Organization

To address the apparent chicken-and-egg problem, we present a maximum-likelihood estimator that *jointly* learns the classifier/regressor, the annotator accuracy, and the actual true label. For ease of exposition we start with binary classification problem in § 2. The performance of each annotator is measured in terms of the sensitivity and specificity with respect to the unknown gold standard (§ 2.1). The proposed algorithm automatically discovers the best experts and assigns a higher weight to them. In order to incorporate prior knowledge about each annotator, we impose a beta prior on the sensitivity and specificity and derive the maximum-a-posteriori estimate (§ 2.6). The final estimation is performed by an Expectation Maximization (EM) algorithm that iteratively establishes a particular gold standard, measures the performance of the experts given that gold standard, and refines the gold standard based on the performance measures. While the proposed approach is described using logistic regression as the base classifier (§ 2.2), it is quite general, and can be used with any black-box classifier (§ 2.7), and can also handle missing labels (that is, each expert is not required to label all the instances). Furthermore, we extend the proposed algorithm to handle categorical (§ 3), ordinal (§ 4), and regression problems (§ 5). In § 6 section we extensively validate our approach using both simulated data and real data from different domains.

1.3 Related Work and Novel Contributions

We first summarize the novel contributions of this work in context of other related work in this emerging new area. There has been a long line of work in the biostatistics and epidemiology literature on latent variable models where the task is to get an estimate of the observer error rates based

7. When there is no clear majority among the multiple experts (that is, $\hat{y}_i = 0.5$) in CAD domain the final decision is often made by an adjudicator or a super-expert. When there is no adjudicator a fair coin toss is used.

on the results from multiple diagnostic tests without a gold standard (see Dawid and Skene, 1979, Hui and Walter, 1980, Hui and Zhou, 1998, Albert and Dodd, 2004 and references therein). In the machine learning community Smyth et al. (1995) first addressed the same problem in the context of labeling volcanoes in satellite images of Venus. We differ from this previous body of work in the following aspects:

1. Unlike Dawid and Skene (1979) and Smyth et al. (1995) which just focused on estimating the ground truth from multiple noisy labels, we specifically address the issue of *learning a classifier*. Estimating the ground truth and the annotator/classifer performance is a byproduct of our proposed algorithm.

2. In order to learn a classifier Smyth (1995) proposed to first estimate the ground truth (without using the features) and then use the probabilistic ground truth to learn a classifier. In contrast, our proposed algorithm *learns the classifier and the ground truth jointly*. Our experiments (§ 6.1.1) show that the classifier learnt and ground truth obtained by the proposed algorithm is superior to that obtained by other procedures which first estimates the ground truth and then learns the classifier.

3. Our solution is more general and can be easily extended to categorical(§ 3), ordinal(§ 4), and continuous data(§ 5). It can also be used in conjunction with any supervised learning algorithm. A preliminary version of this paper (Raykar et al., 2009) mainly discussed the binary classification problem.

4. Our proposed algorithm is also Bayesian—we impose a prior on the experts. The priors can potential capture the skill of different annotators. In this paper we refrain from doing a full Bayesian inference and use the mode of the posterior as a point estimate. A recent complete Bayesian generalization of these kind of models has been developed by Carpenter (2008).

5. The EM approach used in this paper is similar to that proposed by Jin and Ghahramani (2003). However their motivation is somewhat different. In their setting, each training example is annotated with a set of possible labels, only one of which is correct.

There has been recent interest in the natural language processing (Sheng et al., 2008; Snow et al., 2008) and computer vision (Sorokin and Forsyth, 2008) communities where they use Amazon's Mechanical Turk to collect annotations from many people. They show that it can be potentially as good as that provided by an expert. Sheng et al. (2008) analyzed when it is worthwhile to acquire new labels for some of the training examples. There is also some theoretical work (see Lugosi, 1992 and Dekel and Shamir, 2009a) dealing with multiple experts. Recently Dekel and Shamir (2009b) presented an algorithm which does not resort to repeated labeling, that is, each example does not have to be labeled by multiple teachers. Donmez et al. (2009) address the issue of active learning in this scenario—How to jointly learn the accuracy of labeling sources and obtain the most informative labels for the active learning task? There has also been some work in the medical imaging community (Warfield et al., 2004; Cholleti et al., 2008).

2. Binary Classification

We first describe our proposed noise model for the annotators. The performance of each annotator is measured in terms of the sensitivity and specificity with respect to the unknown gold standard.

2.1 A Two-coin Model for Annotators

Let $y^j \in \{0,1\}$ be the label assigned to the instance $\boldsymbol{x}$ by the j^{th} annotator/expert. Let y be the actual (unobserved) label for this instance. Each annotator provides a version of this hidden true label based on two biased coins. If the true label is one, she flips a coin with bias α^j (*sensitivity*). If the true label is zero, she flips a coin with bias β^j (*specificity*). In each case, if she gets heads she keeps the original label, otherwise she flips the label.

If the true label is one, the sensitivity (true positive rate) for the j^{th} annotator is defined as the probability that she labels it as one.

$$\alpha^j := \Pr[y^j = 1 | y = 1]. \tag{1}$$

On the other hand, if the true label is zero, the specificity (1−false positive rate) is defined as the probability that she labels it as zero.

$$\beta^j := \Pr[y^j = 0 | y = 0]. \tag{2}$$

The assumption introduced is that α^j and β^j do not depend on the instance $\boldsymbol{x}$. For example, in the CAD domain, this means that the radiologist's performance is consistent across different sub-groups of data.[8]

2.2 Classification Model

While the proposed method can be used for any classifier, for ease of exposition, we consider the family of linear discriminating functions: $\mathcal{F} = \{f_{\boldsymbol{w}}\}$, where for any $\boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^d$, $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}$. The final classifier can be written in the following form: $\hat{y} = 1$ if $\boldsymbol{w}^\top \boldsymbol{x} \geq \gamma$ and 0 otherwise. The threshold γ determines the operating point of the classifier. The Receiver Operating Characteristic (ROC) curve is obtained as γ is swept from $-\infty$ to ∞. The probability for the positive class is modeled as a *logistic sigmoid* acting on $f_{\boldsymbol{w}}$, that is,

$$\Pr[y = 1 | \boldsymbol{x}, \boldsymbol{w}] = \sigma(\boldsymbol{w}^\top \boldsymbol{x}),$$

where the logistic sigmoid function is defined as $\sigma(z) = 1/(1+e^{-z})$. This classification model is known as *logistic regression*.

2.3 Estimation/Learning Problem

Given the training data $\mathcal{D}$ consisting of N instances with annotations from R annotators, that is, $\mathcal{D} = \{\boldsymbol{x}_i, y_i^1, \ldots, y_i^R\}_{i=1}^N$, the task is to estimate the weight vector $\boldsymbol{w}$ and also the sensitivity $\boldsymbol{\alpha} = [\alpha^1, \ldots, \alpha^R]$ and the specificity $\boldsymbol{\beta} = [\beta^1, \ldots, \beta^R]$ of the R annotators. It is also of interest to get an estimate of the unknown gold standard $y_1, \ldots, y_N$.

2.4 Maximum Likelihood Estimator

Assuming the training instances are independently sampled, the likelihood function of the parameters $\theta = \{\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\}$ given the observations $\mathcal{D}$ can be factored as

$$\Pr[\mathcal{D}|\theta] = \prod_{i=1}^{N} \Pr[y_i^1, \ldots, y_i^R | \boldsymbol{x}_i, \theta].$$

8. While this is a reasonable assumption, it is not entirely true. It is known that some radiologists are good at detecting certain kinds of malignant lesions based on their training and experience.

Conditioning on the true label y_i, and also using the assumption y_i^j is conditionally independent (of everything else) given α^j, β^j and y_i, the likelihood can be decomposed as

$$\begin{aligned}\Pr[\mathcal{D}|\theta] &= \prod_{i=1}^{N}\big\{\Pr[y_i^1,\ldots,y_i^R|y_i=1,\boldsymbol{\alpha}]\Pr[y_i=1|\boldsymbol{x_i},\boldsymbol{w}] \\ &+ \Pr[y_i^1,\ldots,y_i^R|y_i=0,\boldsymbol{\beta}]\Pr[y_i=0|\boldsymbol{x_i},\boldsymbol{w}]\big\}.\end{aligned}$$

Given the true label y_i, we assume that $y_i^1,\ldots,y_i^R$ are independent, that is, the annotators make their decisions independently.[9] Hence,

$$\Pr[y_i^1,\ldots,y_i^R|y_i=1,\boldsymbol{\alpha}] = \prod_{j=1}^{R}\Pr[y_i^j|y_i=1,\alpha^j] = \prod_{j=1}^{R}[\alpha^j]^{y_i^j}[1-\alpha^j]^{1-y_i^j}.$$

Similarly, we have

$$\Pr[y_i^1,\ldots,y_i^R|y_i=0,\boldsymbol{\beta}] = \prod_{j=1}^{R}[\beta^j]^{1-y_i^j}[1-\beta^j]^{y_i^j}.$$

Hence the likelihood can be written as

$$\Pr[\mathcal{D}|\theta] = \prod_{i=1}^{N}\big[a_i p_i + b_i(1-p_i)\big],$$

where we have defined

$$\begin{aligned}p_i &:= \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i). \\ a_i &:= \prod_{j=1}^{R}[\alpha^j]^{y_i^j}[1-\alpha^j]^{1-y_i^j}. \\ b_i &:= \prod_{j=1}^{R}[\beta^j]^{1-y_i^j}[1-\beta^j]^{y_i^j}.\end{aligned}$$

The maximum-likelihood estimator is found by maximizing the log-likelihood, that is,

$$\hat{\theta}_{\mathrm{ML}} = \{\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}},\hat{\boldsymbol{w}}\} = \arg\max_{\theta}\{\ln\Pr[\mathcal{D}|\theta]\}.$$

2.5 The EM Algorithm

This maximization problem can be simplified a lot if we use the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). The EM algorithm is an efficient iterative procedure to compute the maximum-likelihood solution in presence of missing/hidden data. We will use the unknown hidden true label y_i as the missing data. If we know the missing data $\boldsymbol{y} = [y_1,\ldots,y_N]$ then the complete likelihood can be written as

$$\ln\Pr[\mathcal{D},\boldsymbol{y}|\theta] = \sum_{i=1}^{N} y_i \ln p_i a_i + (1-y_i)\ln(1-p_i)b_i.$$

9. This assumption is not true in general and there is some correlations among the labels assigned by multiple annotators. For example in the CAD domain if the cancer is in advanced stage (which is very easy to detect) almost all the radiologists assign the same label.

Each iteration of the EM algorithm consists of two steps: an Expectation(E)-step and a Maximization(M)-step. The M-step involves maximization of a lower bound on the log-likelihood that is refined in each iteration by the E-step.

1. **E-step.** Given the observation $\mathcal{D}$ and the current estimate of the model parameters θ, the conditional expectation (which is a lower bound on the true likelihood) is computed as

$$\mathbb{E}\{\ln \Pr[\mathcal{D}, \boldsymbol{y}|\theta]\} = \sum_{i=1}^{N} \mu_i \ln p_i a_i + (1-\mu_i)\ln(1-p_i)b_i, \tag{3}$$

   where the expectation is with respect to $\Pr[\boldsymbol{y}|\mathcal{D},\theta]$, and $\mu_i = \Pr[y_i = 1|y_i^1,\ldots,y_i^R,\boldsymbol{x}_i,\theta]$. Using Bayes' theorem we can compute

$$\begin{aligned} \mu_i &\propto \Pr[y_i^1,\ldots,y_i^R|y_i=1,\theta]\cdot\Pr[y_i=1|\boldsymbol{x_i},\theta] \\ &= \frac{a_i p_i}{a_i p_i + b_i(1-p_i)}. \end{aligned}$$

2. **M-step.** Based on the current estimate μ_i and the observations $\mathcal{D}$, the model parameters θ are then estimated by maximizing the conditional expectation. By equating the gradient of (3) to zero we obtain the following estimates for the sensitivity and specificity:

$$\alpha^j = \frac{\sum_{i=1}^N \mu_i y_i^j}{\sum_{i=1}^N \mu_i}, \qquad \beta^j = \frac{\sum_{i=1}^N (1-\mu_i)(1-y_i^j)}{\sum_{i=1}^N (1-\mu_i)}.$$

   Due to the non-linearity of the sigmoid, we do not have a closed form solution for $\boldsymbol{w}$ and we have to use gradient ascent based optimization methods. We use the Newton-Raphson update given by $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \boldsymbol{H}^{-1}\mathbf{g}$, where $\boldsymbol{g}$ is the gradient vector, $\boldsymbol{H}$ is the Hessian matrix, and η is the step length. The gradient vector is given by

$$\boldsymbol{g}(w) = \sum_{i=1}^{N} \left[\mu_i - \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\right] \boldsymbol{x}_i.$$

   The Hessian matrix is given by

$$\boldsymbol{H}(w) = -\sum_{i=1}^{N} \left[\sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\right]\left[1-\sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\right] \boldsymbol{x}_i \boldsymbol{x}_i^\top.$$

   Essentially, we are estimating *a logistic regression model with probabilistic labels* μ_i.

These two steps (the E- and the M-step) can be iterated till convergence. The log-likelihood increases monotonically after every iteration, which in practice implies convergence to a local maximum. The EM algorithm is only guaranteed to converge to a local maximum. In practice multiple restarts with different initializations can potentially mitigate the local maximum problem. In this paper we use majority voting $\mu_i = 1/R\sum_{j=1}^R y_i^j$ as the initialization for μ_i to start the EM-algorithm.

2.6 A Bayesian Approach

In some applications we may want to trust a particular expert more than the others. One way to achieve this is by imposing priors on the sensitivity and specificity of the experts. Since α_j and β_j represent the probability of a binary event, a natural choice of prior is the beta prior. The beta prior is also conjugate to the binomial distribution. For any $a > 0$, $b > 0$, and $\delta \in [0,1]$ the beta distribution is given by

$$\text{Beta}(\delta|a,b) = \frac{\delta^{a-1}(1-\delta)^{b-1}}{\text{B}(a,b)},$$

where $\text{B}(a,b) = \int_0^1 \delta^{a-1}(1-\delta)^{b-1} d\delta$ is the beta function.We assume a beta prior[10] for both the sensitivity and the specificity as

$$\begin{aligned} \Pr[\alpha_j|a_1^j,a_2^j] &= \text{Beta}(\alpha_j|a_1^j,a_2^j). \\ \Pr[\beta_j|b_1^j,b_2^j] &= \text{Beta}(\beta_j|b_1^j,b_2^j). \end{aligned}$$

For sake of completeness we also assume a zero mean Gaussian prior on the weights $\boldsymbol{w}$ with inverse covariance matrix $\boldsymbol{\Gamma}$, that is, $\Pr[\boldsymbol{w}] = \mathcal{N}(\boldsymbol{w}|0,\boldsymbol{\Gamma}^{-1})$. Assuming that $\{\alpha_j\}$, $\{\beta_j\}$, and $\boldsymbol{w}$ have independent priors, the maximum-a-posteriori (MAP) estimator is found by maximizing the log-posterior, that is,

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta} \{\ln \Pr[\mathcal{D}|\theta] + \ln \Pr[\theta]\}.$$

An EM algorithm can be derived in a similar fashion for MAP estimation by relying on the interpretation of Neal and Hinton (1998). The final algorithm is summarized below:

1. Initialize $\mu_i = (1/R)\sum_{j=1}^R y_i^j$ based on majority voting.

2. Given μ_i, estimate the sensitivity and specificity of each annotator/expert as follows.

$$\begin{aligned} \alpha^j &= \frac{a_1^j - 1 + \sum_{i=1}^N \mu_i y_i^j}{a_1^j + a_2^j - 2 + \sum_{i=1}^N \mu_i}. \\ \beta^j &= \frac{b_1^j - 1 + \sum_{i=1}^N (1-\mu_i)(1-y_i^j)}{b_1^j + b_2^j - 2 + \sum_{i=1}^N (1-\mu_i)}. \end{aligned} \tag{4}$$

   The Newton-Raphson update for optimizing $\boldsymbol{w}$ is given by $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \boldsymbol{H}^{-1}\mathbf{g}$, with step length η, gradient vector

$$\boldsymbol{g}(w) = \sum_{i=1}^N \Big[\mu_i - \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\Big] \boldsymbol{x}_i - \boldsymbol{\Gamma}\boldsymbol{w},$$

   and Hessian matrix

$$\boldsymbol{H}(w) = -\sum_{i=1}^N \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\Big[1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\Big] \boldsymbol{x}_i \boldsymbol{x}_i^\top - \boldsymbol{\Gamma}.$$

10. It may be convenient to specify a prior in terms of the mean μ and variance σ^2. The mean and the variance for a beta prior are given by $\mu = a/(a+b)$ and $\sigma^2 = ab/((a+b)^2(a+b+1))$. Solving for a and b we get $a = (-\mu^3 + \mu^2 - \mu\sigma^2)/\sigma^2$ and $b = a(1-\mu)/\mu$.

3. Given the sensitivity and specificity of each annotator and the model parameters, update μ_i as

$$\mu_i = \frac{a_i p_i}{a_i p_i + b_i(1-p_i)}, \tag{5}$$

where

$$\begin{aligned} p_i &= \sigma(\boldsymbol{w}^\top \boldsymbol{x}_i). \\ a_i &= \prod_{j=1}^{R} [\alpha^j]^{y_i^j} [1-\alpha^j]^{1-y_i^j}. \\ b_i &= \prod_{j=1}^{R} [\beta^j]^{1-y_i^j} [1-\beta^j]^{y_i^j}. \end{aligned} \tag{6}$$

Iterate (2) and (3) till convergence.

2.7 Discussions

1. **Estimate of the gold standard** The value of the posterior probability μ_i is a soft probabilistic estimate of the actual ground truth y_i, that is, $\mu_i = \Pr[y_i = 1 | y_i^1, \ldots, y_i^R, \boldsymbol{x}_i, \theta]$. The actual hidden label y_i can be estimated by applying a threshold on μ_i, that is, $y_i = 1$ if $\mu_i \geq \gamma$ and zero otherwise. We can use $\gamma = 0.5$ as the threshold. By varying γ we can change the misclassification costs and obtain a ground truth with large sensitivity or large specificity. Because of this in our experimental validation we can actually draw an ROC curve for the estimated ground truth.

2. **Log-odds of $\boldsymbol{\mu}$** A particularly revealing insight can be obtained in terms of the log-odds or the *logit* of the posterior probability μ_i. From (5) the logit of μ_i can be written as

$$\begin{aligned} \text{logit}(\mu_i) = \ln \frac{\mu_i}{1-\mu_i} &= \ln \frac{\Pr[y_i = 1 | y_i^1, \ldots, y_i^R, \boldsymbol{x}_i, \theta]}{\Pr[y_i = 0 | y_i^1, \ldots, y_i^R, \boldsymbol{x}_i, \theta]} \\ &= \boldsymbol{w}^\top \boldsymbol{x}_i + c + \sum_{j=1}^{R} y_i^j [\text{logit}(\alpha^j) + \text{logit}(\beta^j)]. \end{aligned}$$

where $c = \sum_{j=1}^{R} \log \frac{1-\alpha^j}{\beta^j}$ is a constant term which does not depend on i. This indicates that the estimated ground truth (in the logit form of the posterior probability) is a *weighted linear combination* of the labels from all the experts. The weight of each expert is the sum of the logit of the sensitivity and specificity.

3. **Using any other classifier** For ease of exposition we used logistic regression. However, the proposed algorithm can be used with any generalized linear model or in fact with any classifier that can be trained with soft probabilistic labels. In each step of the EM-algorithm, the classifier is trained with instances sampled from μ_i. This modification is easy for most probabilistic classifiers. For general black-box classifiers where we cannot tweak the training algorithm an alternate approach is to replicate the training examples according to the soft label. For example a probabilistic label $\mu_i = 0.8$ can be effectively simulated by adding 8 training examples with deterministic label 1 and 2 examples with label 0.

4. **Obtaining ground truth with no features** In some scenarios we may not have features $\boldsymbol{x}_i$ and we wish to obtain an estimate of the actual ground truth based only on the labels from multiple annotators. Here instead of learning a classifier we estimate p which is the prevalence of the positive class, that is, $p = \Pr[y_i = 1]$. We further assume a beta prior for the prevalence, that is, $\text{Beta}(p|p_1, p_2)$. The algorithm simplifies as follows.

   (a) Initialize $\mu_i = (1/R)\sum_{j=1}^{R} y_i^j$ based on majority voting.

   (b) Given μ_i, estimate the sensitivity and specificity of each annotator using (4). The prevalence of the positive class is estimated as follows.

$$p = \frac{p_1 - 1 + \sum_{i=1}^{N} \mu_i}{p_1 + p_2 - 2 + N}.$$

   (c) Given the sensitivity and specificity of each annotator and prevalence, refine μ_i as follows.

$$\mu_i = \frac{a_i p}{a_i p + b_i (1-p)}.$$

   Iterate (2) and (3) till convergence. This algorithm is similar to the one proposed by Dawid and Skene (1979) and Smyth et al. (1995).

5. **Handling missing labels** The proposed approach can easily handle missing labels, that is, when the labels from some experts are missing for some instances. Let R_i be the number of radiologists labeling the i^{th} instance, and let N_j be the number of instances labeled by the j^{th} radiologist. Then in the EM algorithm, we just need to replace N by N_j for estimating the sensitivity and specificity in (4), and replace R by R_i for updating μ_i in (6).

6. **Evaluating a classifier** We can use the probability scores μ_i directly to evaluate classifiers. If z_i are the labels obtained from any other classifier, then sensitivity and specificity can be estimated as

$$\alpha = \frac{\sum_{i=1}^{N} \mu_i z_i}{\sum_{i=1}^{N} \mu_i}, \quad \beta = \frac{\sum_{i=1}^{N} (1-\mu_i)(1-z_i)}{\sum_{i=1}^{N} (1-\mu_i)}.$$

7. **Posterior approximation** At the end of each EM iteration a crude approximation to the posterior is obtained as

$$\begin{aligned} \alpha_j &\sim \text{Beta}\left(\alpha_j | a_1^j + \sum_{i=1}^{N} \mu_i y_i^j, a_2^j + \sum_{i=1}^{N} \mu_i (1 - y_i^j)\right), \\ \beta_j &\sim \text{Beta}\left(\beta_j | b_1^j + \sum_{i=1}^{N} (1-\mu_i)(1-y_i^j), b_2^j + \sum_{i=1}^{N} (1-\mu_i) y_i^j\right). \end{aligned}$$

3. Multi-class Classification

In this section we describe how the proposed approach for binary classification can be extended to categorical data. Suppose there are $K \geq 2$ categories. An example for categorical data from the CAD domain is in LungCAD, where the radiologist needs to label whether a nodule (known to be precursors of cancer) is a solid, a part-solid, or a ground glass opacity—which are three

different kinds on nodules. We can extend the previous model and introduce a vector of multinomial parameters $\boldsymbol{\alpha}_c^j = (\alpha_{c1}^j, \ldots, \alpha_{cK}^j)$ for each annotator, where

$$\alpha_{ck}^j := \Pr[y^j = k | y = c]$$

and $\sum_{k=1}^K \alpha_{ck}^j = 1$. Here α_{ck}^j denotes the probability that the annotator j assigns class k to an instance given the true class is c. When $K = 2$, α_{11}^j and α_{00}^j are sensitivity and specificity, respectively. A similar EM algorithm can be derived. In the E-step, we estimate

$$\Pr[y_i = c | \mathcal{Y}, \boldsymbol{\alpha}] \propto \Pr[y_i = c | \boldsymbol{x}_i] \prod_{j=1}^R \prod_{k=1}^K (\alpha_{ck}^j)^{\delta(y_i^j, k)},$$

where $\delta(u, v) = 1$ if $u = v$ and 0 otherwise and in the M-step we learn a multi-class classifier and update the multinomial parameter as

$$\alpha_{ck}^j = \frac{\sum_{i=1}^N \Pr[y_i = c | \mathcal{Y}, \boldsymbol{\alpha}] \delta(y_i^j, k)}{\sum_{i=1}^N \Pr[y_i = c | \mathcal{Y}, \boldsymbol{\alpha}]}.$$

One can also assign a Dirichlet prior for the multinomial parameters, and this results in a smoothing term in the above updates in the MAP estimate.

4. Ordinal Regression

We now consider the situation where the outputs are categorical and have an ordering among the labels. In the CAD domain the radiologist often gives a score (for example, 1 to 5 from lowest to highest) to indicate how likely she thinks it is malignant. This is different from a multi-class setting in which we do not have any preference among the multiple class labels.

Let $y_i^j \in \{1, \ldots, K\}$ be the label assigned to the i^{th} instance by the j^{th} expert. Note that there is an ordering in the labels $1 < \ldots < K$. A simple approach is to convert the ordinal data into a series of binary data (Frank and Hall, 2001). Specifically the K class ordinal labels are transformed into $K - 1$ binary class labels as follows:

$$y_i^{jc} = \begin{cases} 1 & \text{if } y_i^j > c \\ 0 & \text{otherwise} \end{cases} \qquad c = 1, \ldots, K-1.$$

Applying the same procedure used for binary labels we can estimate $\Pr[y_i > c]$ for $c = 1, \ldots, K-1$. The probability of the actual class values can then be obtained as

$$\Pr[y_i = c] = \Pr[y_i > c - 1 \text{ and } y_i \leq c] = \Pr[y_i > c - 1] - \Pr[y_i > c].$$

The class with the maximum probability is assigned to the instance.

5. Regression

In this section we develop a similar algorithm to learn a regression function using annotations from multiple experts. In the CAD domain as a part of the annotation process a common task for a radiologist is to measure the diameter of a suspicious lesion.

5.1 Model for Annotators

Let $y_i^j \in \mathbb{R}$ be the continuous target value assigned to the i^{th} instance by the j^{th} annotator. Our model is that the annotator provides a noisy version of the actual true value y_i. For the j^{th} annotator we will assume a Gaussian noise model with mean y_i (the true unknown value) and inverse-variance (precision) τ^j, that is,

$$\Pr[y_i^j|y_i,\tau^j] = \mathcal{N}(y_i^j|y_i,1/\tau^j), \tag{7}$$

where the Gaussian distribution is defined as $\mathcal{N}(z|m,\sigma^2) = (2\pi\sigma^2)^{-1/2}\exp(-(z-m)^2/2\sigma^2)$. The unknown inverse-variance τ^j measures the accuracy of each annotator—the larger the value of τ^j the more accurate the annotator. We have assumed that τ^j does not depend on the instance $\boldsymbol{x}_i$. For example, in the CAD domain, this means that the radiologist's accuracy does not depend on the nodule she is measuring. While this a practical assumption, it is not entirely true. It is known that some nodules are harder to measure than others.

5.2 Linear Regression Model for Features

As before we consider the family of linear regression functions: $\mathcal{F} = \{f_{\boldsymbol{w}}\}$, where for any $\boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^d$, $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}$. We assume that the actual target response y_i is given by the deterministic regression function $f_{\boldsymbol{w}}$ with additive Gaussian noise, that is,

$$y_i = \boldsymbol{w}^\top \boldsymbol{x}_i + \varepsilon,$$

where ε is a zero-mean Gaussian random variable with inverse-variance (precision) γ. Hence

$$\Pr[y_i|\boldsymbol{x}_i,\boldsymbol{w},\gamma] = \mathcal{N}(y_i|\boldsymbol{w}^\top \boldsymbol{x}_i, 1/\gamma). \tag{8}$$

5.3 Combined Model

Combining both the annotator (7) and the regressor (8) model we have

$$\Pr[y_i^j|\boldsymbol{x}_i,\boldsymbol{w},\tau^j,\gamma] = \int \Pr[y_i^j|y_i,\tau^j]\Pr[y_i|\boldsymbol{x}_i,\boldsymbol{w},\gamma]dy_i = \mathcal{N}(y_i^j|\boldsymbol{w}^\top \boldsymbol{x}_i, 1/\gamma + 1/\tau^j).$$

Since the two precision terms (γ and τ_j) are grouped together they are not uniquely identifiable. Hence we will define a new precision term λ^j as

$$\frac{1}{\lambda^j} = \frac{1}{\gamma} + \frac{1}{\tau^j}.$$

So we have the following model

$$\Pr[y_i^j|\boldsymbol{x}_i,\boldsymbol{w},\lambda^j] = \mathcal{N}(y_i^j|\boldsymbol{w}^\top \boldsymbol{x}_i, 1/\lambda^j). \tag{9}$$

5.4 Estimation/Learning Problem

Given the training data $\mathcal{D}$ consisting of N instances with annotations from R experts, that is, $\mathcal{D} = \{\boldsymbol{x}_i, y_i^1, \ldots, y_i^R\}_{i=1}^N$, the task is to estimate the weight vector $\boldsymbol{w}$ and the precision $\boldsymbol{\lambda} = [\lambda^1, \ldots, \lambda^R]$ of all the annotators.

5.5 Maximum-likelihood Estimator

Assuming the instances are independent the likelihood of the parameters $\theta = \{\boldsymbol{w}, \boldsymbol{\lambda}\}$ given the observations $\mathcal{D}$ can be factored as

$$\Pr[\mathcal{D}|\theta] = \prod_{i=1}^{N} \Pr[y_i^1, \ldots, y_i^R | \boldsymbol{x}_i, \theta].$$

Conditional on the instance $\boldsymbol{x}_i$ we assume that $y_i^1, \ldots, y_i^R$ are independent, that is, the annotators provide their responses independently. Hence from (9) the likelihood can be written as

$$\Pr[\mathcal{D}|\theta] = \prod_{i=1}^{N} \prod_{j=1}^{R} \mathcal{N}(y_i^j | \boldsymbol{w}^\top \boldsymbol{x}_i, 1/\lambda^j).$$

The maximum-likelihood estimator is found by maximizing the log-likelihood

$$\widehat{\theta}_{\text{ML}} = \{\widehat{\boldsymbol{\lambda}}, \widehat{\boldsymbol{w}}\} = \arg\max_{\theta} \{\ln \Pr[\mathcal{D}|\theta]\}.$$

By equating the gradient of the log-likelihood to zero we obtain the following update equations for the precision and the weight vector.

$$\frac{1}{\widehat{\lambda}^j} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i^j - \widehat{\boldsymbol{w}}^\top \boldsymbol{x_i} \right)^2. \tag{10}$$

$$\widehat{\boldsymbol{w}} = \left(\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^\top \right)^{-1} \sum_{i=1}^{N} \boldsymbol{x}_i \left(\frac{\sum_{j=1}^{R} \widehat{\lambda}^j y_i^j}{\sum_{j=1}^{R} \widehat{\lambda}^j} \right). \tag{11}$$

As the parameters $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{\lambda}}$ are coupled together we iterate these two steps till convergence.

5.6 Discussions

1. **Is this standard least-squares?** Define the design matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^\top$ and the response vector for each annotator as $\boldsymbol{y}^j = [y_1^j, \ldots, y_N^j]^\top$. Using matrix notation Equation 11 can be written as

$$\widehat{\boldsymbol{w}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \widehat{\boldsymbol{y}} \quad \text{where} \quad \widehat{\boldsymbol{y}} = \frac{\sum_{j=1}^{R} \widehat{\lambda}^j \boldsymbol{y}^j}{\sum_{j=1}^{R} \widehat{\lambda}^j}. \tag{12}$$

   Equation 12 is essentially the solution to a standard linear regression model, except that we are training a linear regression model with $\widehat{\boldsymbol{y}}$ as the ground truth, which is a precision weighted mean of the response vectors from all the annotators. The variance of each annotator is estimated using (10). The final algorithm iteratively establishes a particular gold standard ($\widehat{\boldsymbol{y}}$), measures the performance of the annotators and learns a regressor given that gold standard, and refines the gold standard based on the performance measures.

2. **Are we better than the best annotator?** If we assume $\widehat{\boldsymbol{\lambda}}$ is fixed (i.e., we ignore the variability and assume that it is well estimated) then $\hat{\boldsymbol{w}}$ is an unbiased estimator of $\boldsymbol{w}$ and the covariance matrix is given by

$$\text{Cov}(\hat{\boldsymbol{w}}) = \text{Cov}(\widehat{\boldsymbol{y}}) \left(\boldsymbol{X}^\top \boldsymbol{X} \right)^{-1} = \frac{1}{\sum_{j=1}^{R} \widehat{\lambda}_j} \left(\boldsymbol{X}^\top \boldsymbol{X} \right)^{-1}.$$

Since $\sum_{j=1}^{R} \widehat{\lambda}_j > \max_j(\widehat{\lambda}_j)$ the proposed method has a lower variance than the regressor learnt with the best annotator (i.e., the one with the minimum variance).

3. **Are we better than the average?** For a fixed $\boldsymbol{X}$ the error in $\widehat{\boldsymbol{w}}$ depends only on the variance of $\widehat{y}^j$. If we know the true λ^j then $\widehat{y}_i$ is the best linear unbiased estimator for y_i which minimizes the variance. To see this consider any linear estimator of the form $\widehat{y}_i = \sum_j a^j (y_i^j - b^j)$. The variance is given by $\mathrm{Var}[\widehat{y}_i] = \sum_j (a^j)^2/\lambda_j$. Since $\mathrm{E}[\widehat{y}_i] = y_i \sum_j a^j$, for the bias of this estimator to be zero we require that $\sum_j a^j = 1$. Solving the constrained minimization problem we see that $a_j = \lambda^j / \sum_j \lambda_j$ minimizes the variance.

4. **Obtaining a consensus without features** When no features are available the same algorithm can be simplified to get a consensus estimate of the actual ground truth and also evaluate the annotators. Essentially we have to iterate the following two updates till convergence

$$\widehat{y}_i = \frac{\sum_{j=1}^{R} \widehat{\lambda}^j y_i^j}{\sum_{j=1}^{R} \widehat{\lambda}^j} \quad \frac{1}{\widehat{\lambda}^j} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i^j - \widehat{y}_i \right)^2.$$

6. Experimental Validation

We now experimentally validate the proposed algorithms on both simulated and real data.

6.1 Classification Experiments

We use two CAD and one text data set in our experiments. The CAD data sets include a digital mammography data set and a breast MRI data set, both of which are biopsy proven, that is, the gold standard is available. For the digital mammography data set we simulate the radiologists in order to validate our methods. The breast MRI data has annotations from four radiologists. We also report results on a Recognizing Textual Entailment data collected by Snow et al. (2008) using the Amazon's Mechanical Turk which has annotations from 164 annotators.

6.1.1 DIGITAL MAMMOGRAPHY WITH SIMULATED RADIOLOGISTS

Mammograms are used as a screening tool to detect early breast cancer. CAD systems search for abnormal areas (*lesions*) in a digitized mammographic image. These lesions generally indicate the presence of malignant cancer. The CAD system then highlights these areas on the images, alerting the radiologist to the need for a further diagnostic mammogram or a biopsy. In classification terms, given a set of descriptive morphological features for a region on a image, the task is to predict whether it is potentially malignant (1) or not (0). In order to train such a classifier, a set of mammograms is collected from hospitals. The ground truth (whether it is cancer or not) is obtained from biopsy. Since biopsy is an expensive, tedious, and an invasive process, very often CAD systems are built from labels collected from *multiple expert radiologists* who visually examine the mammograms and mark the lesion locations—this constitutes our ground truth (multiple labels) for learning.

In this experiment we use a proprietary biopsy-proven data set (Krishnapuram et al., 2008) containing 497 positive and 1618 negative examples. Each instance is described by a set of 27 morphological features. In order to validate our proposed algorithm, we simulate multiple radiologists according to the two-coin model described in § 2.1. Based on the labels from multiple radiologists,

we can simultaneously (1) learn a logistic-regression classifier, (2) estimate the sensitivity and specificity of each radiologist, and (3) estimate the golden ground truth. We compare the results with the classifier trained using the biopsy proved ground truth as well as the majority-voting baseline. For the first set of experiments we use 5 radiologists with sensitivity $\boldsymbol{\alpha} = [0.90\ 0.80\ 0.57\ 0.60\ 0.55]$ and specificity $\boldsymbol{\beta} = [0.95\ 0.85\ 0.62\ 0.65\ 0.58]$. This corresponds to a scenario where the first two radiologists are experts and the last three are novices. Figure 1 summarizes the results. We compare on three different aspects: (1) How good is the learnt classifier? (2) How well can we estimate the sensitivity and specificity of each radiologist? (3) How good is the estimated ground truth? The following observations can be made.

1. **Classifier performance** Figure 1(a) plots the ROC curve of the learnt classifier on the training set. The dotted (black) line is the ROC curve for the classifier learnt using the actual ground truth. The solid (red) line is the ROC curve for the proposed algorithm and the dashed (blue) line is for the classifier learnt using the majority-voting scheme. The classifier learnt using the proposed method is as good as the one learnt using the golden ground truth. The area under the ROC curve (AUC) for the proposed algorithm is around 3.5% greater than that learnt using the majority-voting scheme.

2. **Radiologist performance** The actual sensitivity and specificity of each radiologist is marked as a black $\times$ in Figure 1(b). The end of the solid red line shows the estimates of the sensitivity and specificity from the proposed method. We used a uniform prior on all the parameters. The ellipse plots the contour of one standard deviation as obtained from the beta posterior estimates. The end of the dashed blue line shows the estimate obtained from the majority-voting algorithm. We see that the proposed method is much closer to the actual values of sensitivity and specificity.

3. **Actual ground truth** Since the estimates of the actual ground truth are probabilistic scores, we can also plot the ROC curves of the estimated ground truth. From Figure 1(b) we can see that the ROC curve for the proposed method dominates the majority voting ROC curve. Furthermore, the area under the ROC curve (AUC) is around 3% higher. The estimate obtained by majority voting is closer to the novices since they form a majority (3/5). It does not have an idea of who is an expert and who is a novice. The proposed algorithm appropriately weights each radiologist based on their estimated sensitivity and specificity. The improvement obtained is quite large in Figure 2 which corresponds a situation where we have only one expert and 7 novices.

4. **Joint Estimation** To learn a classifier, Smyth et al. (1995) proposed to first estimate the golden ground truth and then use the probabilistic ground truth to learn a classifier. In contrast, our proposed algorithm learns the classifier and the ground truth *jointly* as a part of the EM algorithm. Figure 3 shows that the classifier and the ground truth learnt obtained by the proposed algorithm is superior than that obtained by other procedures which first estimates the ground truth and then learns the classifier.

6.1.2 BREAST MRI

In this example, each radiologist reviews the breast MRI data and assesses the malignancy of each lesion on a BIRADS scale of 1 to 5. The BIRADS scale is defined as follows: 1 Negative, 2 Benign,

Majority Voting	True 1	True 2	True 3	True 4	True 5
Estimated 1	x	0.0217	0	x	0.0000
Estimated 2	x	**0.5869**	0	x	0.1785
Estimated 3	x	0.2391	0	x	0.1071
Estimated 4	x	0.1521	1	x	0.2500
Estimated 5	x	0.0000	0	x	**0.4642**
EM algorithm	**True 1**	**True 2**	**True 3**	**True 4**	**True 5**
Estimated 1	x	0.0000	0	x	0.0000
Estimated 2	x	**0.6957**	0	x	0.1428
Estimated 3	x	0.1304	0	x	0.0000
Estimated 4	x	0.1739	1	x	0.3214
Estimated 5	x	0.0000	0	x	**0.5357**

Table 1: The confusion matrix for the estimate obtained using majority voting and the proposed EM algorithm. The x indicates that there was no such category in the true labels (the gold standard). The gold-standard is obtained by the biopsy which can confirm whether it is benign (BIRADS=2) or malignant (BIRADS=5).

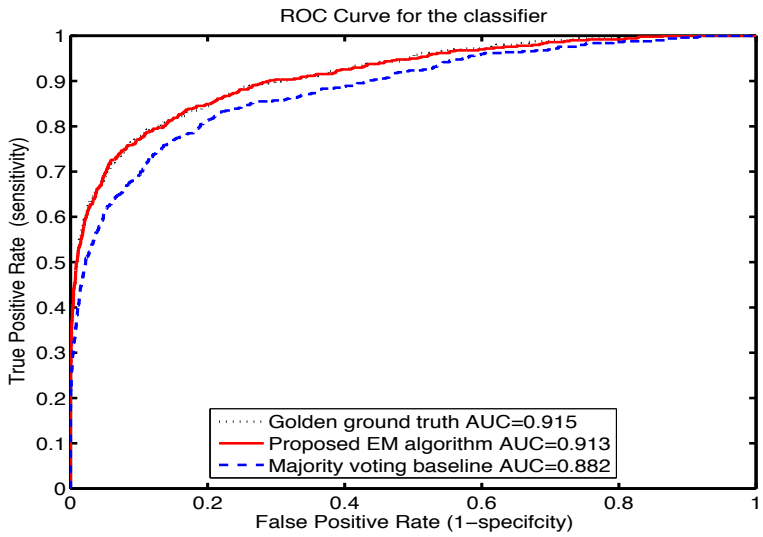
3 Probably Benign, 4 Suspicious abnormality, and 5 Highly suggestive of malignancy. Our data set comprises of 75 lesions with annotations from four radiologists, and the true labels from biopsy. Based on eight morphological features, we have to predict whether a lesion is malignant or not.

For the first experiment we reduce the BIRADS scale to a binary one: any lesion with a BIRADS > 3 is considered malignant and benign otherwise. The set included 28 malignant and 47 benign lesions. Figure 4 summarizes the results. We show the leave-one-out cross validated ROC for the classifier. The cross-validated AUC of the proposed method is approximately 6% better than the majority voting baseline.

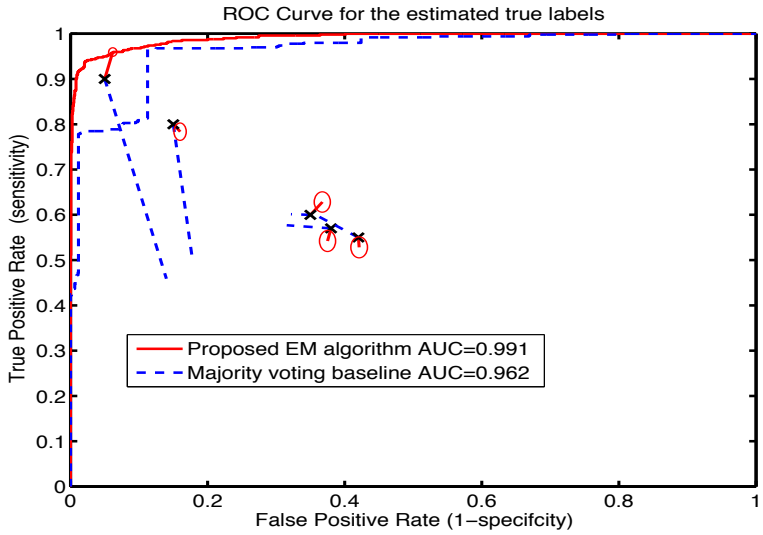
We also consider the BIRADS labels as a set of ordinal measurements since there is an ordering among the BIRADS label. The confusion matrix in Table 1 shows that the EM algorithm is significantly superior than the majority voting in estimating the true BIRADS.

6.1.3 RECOGNIZING TEXTUAL ENTAILMENT

Finally we report results on Recognizing Textual Entailment data collected by Snow et al. (2008) using the Amazon's Mechanical Turk. In this task, the annotator is presented with two sentences and given a choice of whether the second sentence can be inferred from the first. The data has 800 tasks and 164 distinct readers, with 10 annotations per task along with the golden ground truth. The majority of the entries (94 %) in the 800x164 matrix are missing. There is one annotator who has labeled all the tasks. We use this data set to obtain an estimate of the actual ground truth. Figure 5 plots the accuracy of the estimated ground truth as a function of the number of annotators. The proposed EM algorithm achieves a higher accuracy than majority voting. In other words to achieve a desired accuracy the proposed algorithm needs fewer annotators than the majority voting scheme.

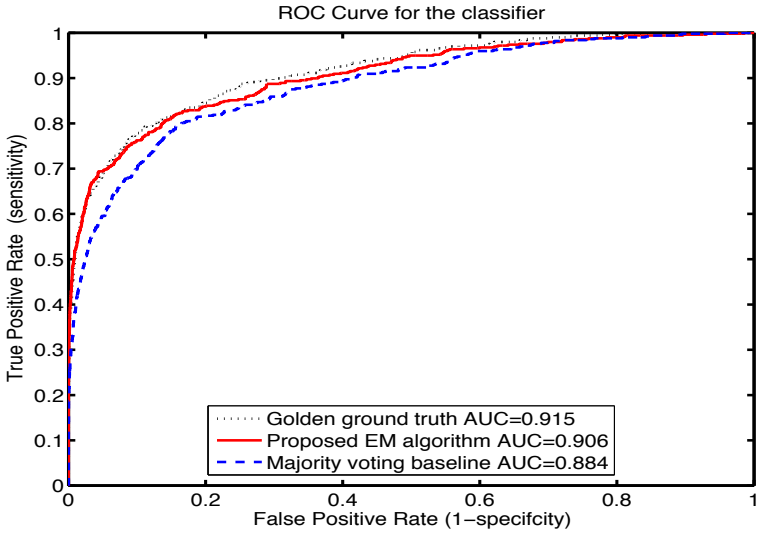


(a)

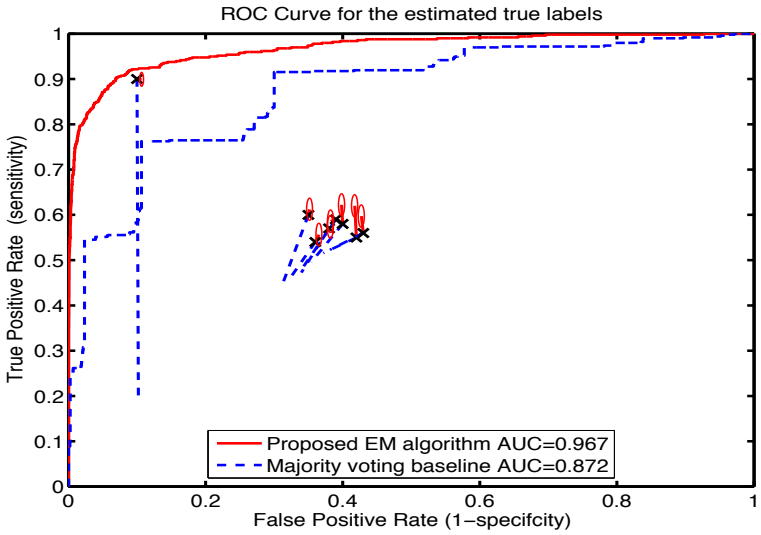


(b)

Figure 1: Results for the digital mammography data set with annotations from 5 simulated radiologists. (a) The ROC curve of the learnt classifier using the golden ground truth (dotted black line), the majority voting scheme (dashed blue line), and the proposed EM algorithm (solid red line). (b) The ROC curve for the estimated ground truth. The actual sensitivity and specificity of each of the radiologists is marked as a $\times$. The end of the dashed blue line shows the estimates of the sensitivity and specificity obtained from the majority voting algorithm. The end of the solid red line shows the estimates from the proposed method. The ellipse plots the contour of one standard deviation.

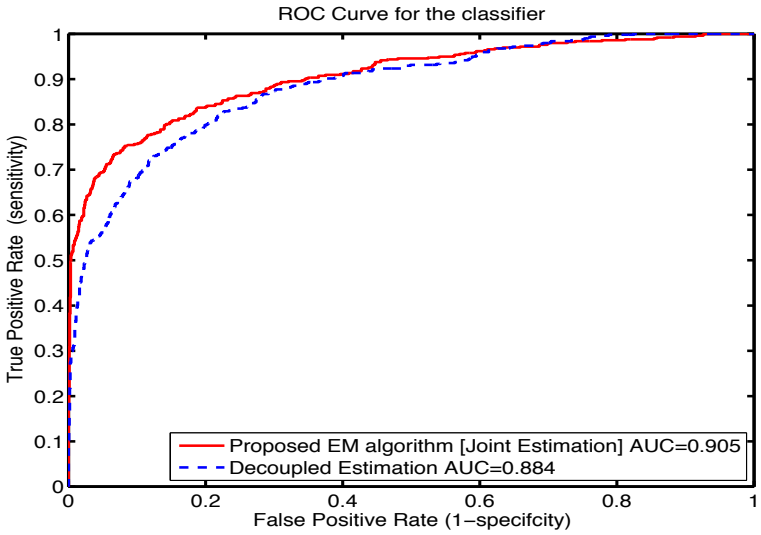


(a)

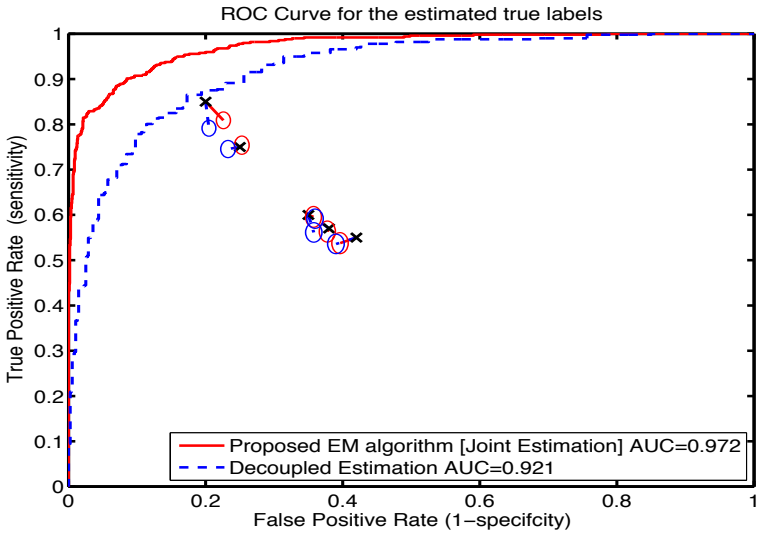


(b)

Figure 2: Same as Figure 1 except with 8 different radiologist annotations.

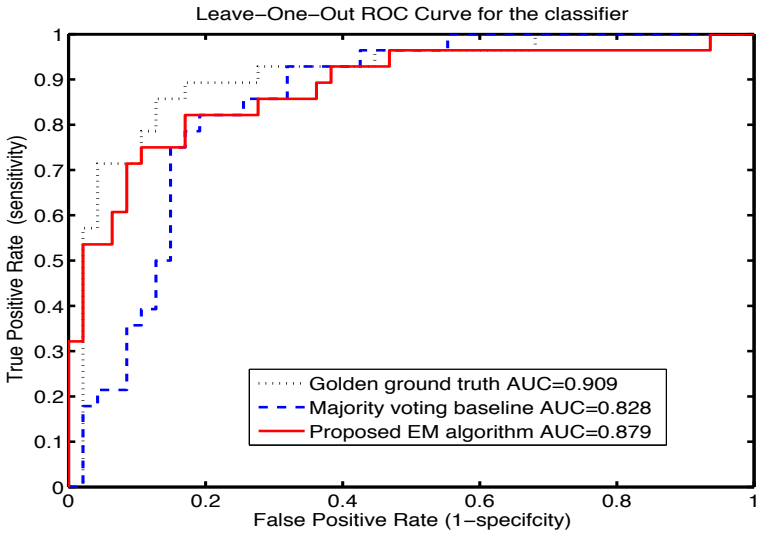


(a)

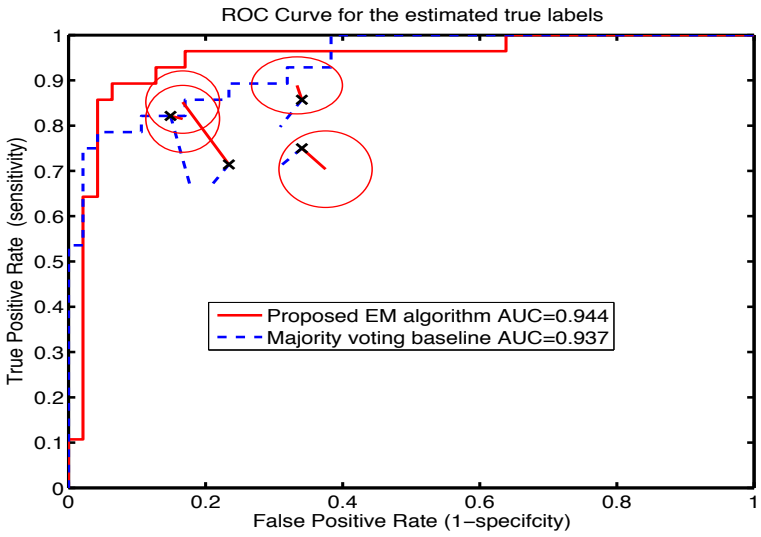


(b)

Figure 3: ROC curves comparing the proposed algorithm (solid red line) with the *Decoupled Estimation* procedure (dotted blue line), which refers to the algorithm where the ground truth is first estimated using just the labels from the five radiologists and then a logistic regression classifier is trained using the soft probabilistic labels. In contrast the proposed EM algorithm estimates the ground truth and learns the classifier simultaneously during the EM algorithm.



(a)



(b)

Figure 4: Breast MRI results. (a) The leave-one-out cross validated ROC. (b) ROC for the estimated ground truth.

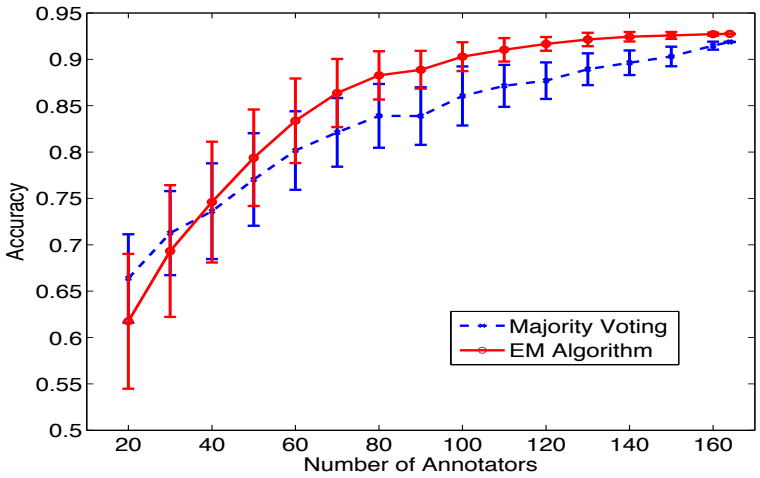


Figure 5: The mean and the one standard deviation error bars for the accuracy of the estimated ground truth for the Recognizing Textual Entailment task as a function of the number of annotators. The plot was generated by randomly sampling the annotators 100 times.

6.2 Regression Experiments

We first illustrate the algorithm on a toy dataset and then present a case study for automated polyp measurements.

6.2.1 ILLUSTRATION

Figure 6 illustrates the the proposed algorithm for regression on a one-dimensional toy data set with three annotators. The actual regression model (shown as a blue dotted line) is given by $y = 5x - 2$. We simulate 20 samples from three annotators with precisions 0.01, 0.1, and 1.0. The data are shown by the annotators's number. While we can fit a regression model using each annotators's response, we see that only the model for annotator three (with highest precision) is close to the true regression model. The green dashed line shows the model learnt using the average response from all the three annotators. The red line shows the model learnt by the proposed algorithm.

6.2.2 AUTOMATED POLYP MEASUREMENTS

Colorectal polyps are small colonic findings that may develop into cancer at a later stage. The diameter of the polyp is one of the key factors which decides the malignancy of a suspicious polyp. Hence accurate size estimation is crucial to decide the action to be taken on a polyp. We have developed various algorithms to segment a polyp. Multiple segmentation algorithms give rise to a set of features which are correlated with the diameter of the polyp. We want to learn a regression function which can predict the diameter of a polyp as a function of these features. In order to learn a regression function we collect our ground truth by asking many radiologists to manually measure the the diameter of the polyps from the three-dimensional images. In practice there is a lot of disagreement among the radiologists as to the actual size of the polyp.

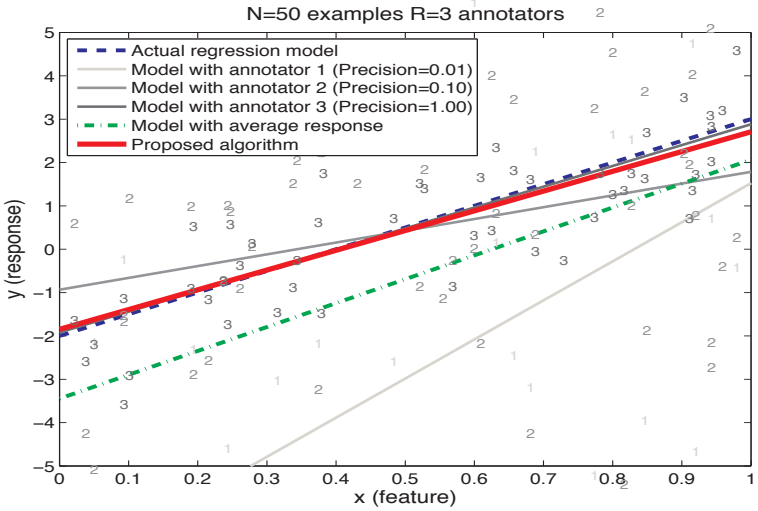
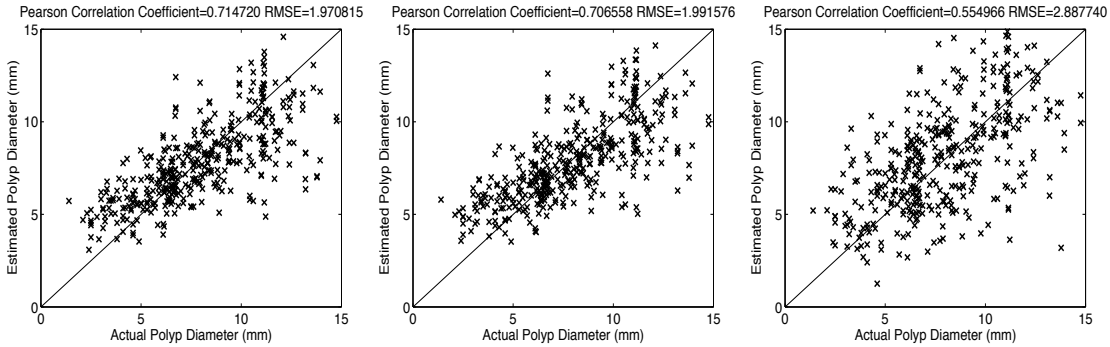


Figure 6: Illustration of the proposed algorithm on a one-dimensional toy data set. The actual regression model (shown as a blue dotted line) is given by $y = 5x - 2$. We simulate 50 samples from three annotators with precisions 0.01, 0.1, and 1.0. The data are shown by the annotators's number. While we can fit a regression model using each annotators's response, we see that only the model for annotator three (with highest precision) is close to the true regression model. The green dashed line shows the model learnt using the average response from all the three annotators. The red line shows the model learnt by the proposed algorithm.



(a) Gold standard model (b) Proposed model (c) Average model

Figure 7: Scatter plot of the actual polyp diameter vs the diameter predicted by the models learnt using (a) the actual gold standard, (b) the proposed algorithm with annotations from five radiologists, and (c) the average of the radiologist's annotations. (See § 6.2.2 for a description of the experimental setup.)

We use a proprietary data set containing 393 examples (which point to 285 distinct polyps—the segmentation algorithms generally return multiple marks on the same polyp.) along with the measured diameter (ranging from 2mm to 15mm) as our training set. Each example is described by a set of 60 morphological features which are correlated to the diameter of the polyp. In order to validate the feasibility of our proposed algorithm, we simulate five radiologists according to the noisy model described in § 5.1 with $\tau = [0.001\ 0.01\ 0.1\ 1\ 10]$. This corresponds to a situation where the first three radiologists are extremely noisy and the last two are quite accurate. Based on the measurements from multiple radiologists, we can simultaneously (1) learn a linear regressor and (2) estimate the precision of each radiologist. We compare the results with the classifier trained using the actual golden ground truth as well as the regressor learnt using the average of the radiologists measurements. The results are validated on an independent test set containing 397 examples (which point to 298 distinct polyps).

Figure 7 shows the scatter plot of the actual polyp diameter vs the diameter predicted by the three different models. We compare the performance based on the root mean squared error (RMSE) and also the Pearson's correlation coefficient. The regressor learnt using the proposed iterative algorithm (Figure 7(b)) is almost as good as the one learnt using the golden ground truth (Figure 7(a)). The correlation coefficient for the proposed algorithm is significantly larger than that learnt using the average of the radiologists response. The estimate obtained by averaging is closer to the novices since they form a majority (3/5). The proposed algorithm appropriately weights each radiologist based on their estimated precisions.

7. Conclusions and Future Work

In this paper we proposed a probabilistic framework for supervised learning with multiple annotators providing labels but no absolute gold standard. The proposed algorithm iteratively establishes a particular gold standard, measures the performance of the annotators given that gold standard, and then refines the gold standard based on the performance measures. We specifically discussed binary/categorical/ordinal classification and regression problems.

We made two key assumptions: (1) the performance of each annotator does not depend on the feature vector for a given instance and (2) conditional on the truth the experts are independent, that is, they make their errors independently. As we pointed out earlier these assumptions are not true in practice. The annotator performance depends on the instance he is labeling and there is some degree of correlation among the annotators. We briefly discuss some strategies to relax these two assumptions.

7.1 Instance Difficulty

One drawback of the current model is that it doesn't estimate difficulty of items. It is often observed that for the easy instances all the annotators agree on the labels—thus violating our conditional independence assumption. The difficulty of annotating an item can be captured by another latent variable γ_i for each instance—which modulates the annotators performance. Models for this have been developed in the area of item-response theory (Baker and Kim, 2004) and also in epidemiology (Uebersax and Grove, 1993)—see also Whitehill et al. (2009) for a recent paper in the machine learning community. While these models do not take into account the available features our pro-

posed model for sensitivity and specificity can be extended as follows (in place of (1) and (2)):

$$\alpha^j(\gamma_i) := \Pr[y_i^j = 1 | y_i = 1, \gamma_i] = \sigma(a_{j1} + b_{j1}\gamma_i).$$

$$\beta^j(\gamma_i) := \Pr[y_i^j = 0 | y_i = 0, \gamma_i] = \sigma(a_{j0} + b_{j0}\gamma_i).$$

Here the parameters a_{j1} and a_{j0} are related to the sensitivity and specificity of the j^{th} annotator, while the latent term γ_i captures the difficulty of the instance. The key assumption here is that the annotators are independent conditional on both y_i and γ_i. Various assumptions can be made on two parameters b_{j1} and b_{j0} to simplify these models further—for example we could set $b_{j1} = b_1$ and $b_{j0} = b_0$ for all the annotators.

7.2 Annotators Actually Look at the Data

In our model we made the assumption that the sensitivity α^j and the specificity β^j of the j^{th} annotator does not depend on the feature vector $\boldsymbol{x}_i$. For example, in the CAD domain, this meant that the radiologist's performance is consistent across different sub-groups of data—which is not entirely true. It is known that some radiologists are good at detecting certain kinds of malignant lesions based on their training and experience. We can extend the previous model such that the sensitivity and the specificity depends on the feature vector $\boldsymbol{x}_i$ explicitly as follows

$$\alpha^j(\gamma_i, \boldsymbol{x}_i) := \Pr[y_i^j = 1 | y_i = 1, \gamma_i, \boldsymbol{x}_i] = \sigma(a_{j1} + b_{j1}\gamma_i + \boldsymbol{w}_\alpha^{j\top} \boldsymbol{x}_i).$$

$$\alpha^j(\gamma_i, \boldsymbol{x}_i) := \Pr[y_i^j = 0 | y_i = 0, \gamma_i, \boldsymbol{x}_i] = \sigma(a_{j0} + b_{j0}\gamma_i + \boldsymbol{w}_\beta^{j\top} \boldsymbol{x}_i).$$

However this change increases the number of parameters to be learned.

References

P. S. Albert and L. E. Dodd. A cautionary note on the robustness of latent class models for estimating diagnostic error without a gold standard. *Biometrics*, 60:427–435, 2004.

F. B. Baker and S. Kim. *Item Response Theory: Parameter Estimation Techniques*. CRC Press, 2 edition, 2004.

B. Carpenter. Multilevel bayesian models of categorical data annotation. Technical Report available at http://lingpipe-blog.com/lingpipe-white-papers/, 2008.

S. R. Cholleti, S. A. Goldman, A. Blum, D. G. Politte, and S. Don. Veritas: Combining expert opinions without labeled data. In *Proceedings of the 2008 20th IEEE international Conference on Tools with Artificial intelligence*, 2008.

A. P. Dawid and A. M. Skene. Maximum likeihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28, 1979.

O. Dekel and O. Shamir. Vox Populi: Collecting high-quality labels from a crowd. In *COLT 2009: Proceedings of the 22nd Annual Conference on Learning Theory*, 2009a.

O. Dekel and O. Shamir. Good learners for evil teachers. In *ICML 2009: Proceedings of the 26th International Conference on Machine Learning*, pages 233–240, 2009b.

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.

P. Donmez, J. G. Carbonell, and J. Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. In *KDD 2009: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 259–268, 2009.

E. Frank and M. Hall. A simple approach to ordinal classification. *Lecture Notes in Computer Science*, pages 145–156, 2001.

G. Fung, B. Krishnapuram, J. Bi, M. Dundar, V. C. Raykar, S. Yu, R. Rosales, S. Krishnan, and R. B. Rao. Mining medical images. In *Fifteenth Annual SIGKDD International Conference on Knowledge Discovery and Data Mining: Third Workshop on Data Mining Case Studies and Practice Prize*, 2009.

J. Howe. *Crowd sourcing: Why the Power of the Crowd Is Driving the Future of Business*. 2008.

S. L. Hui and S. D. Walter. Estimating the error rates of diagnostic tests. *Biometrics*, 36:167–171, 1980.

S. L. Hui and X. H. Zhou. Evaluation of diagnostic tests without gold standards. *Statistical Methods in Medical Research*, 7:354–370, 1998.

R. Jin and Z. Ghahramani. Learning with multiple labels. In *Advances in Neural Information Processing Systems 15*, pages 897–904. 2003.

B. Krishnapuram, J. Stoeckel, V. C. Raykar, R. B. Rao, P. Bamberger, E. Ratner, N. Merlet, I. Stainvas, M. Abramov, and A. Manevitch. Multiple-instance learning improves CAD detection of masses in digital mammography. In *IWDM 2008: Proceedings of the 9th international workshop on Digital Mammography*, pages 350–357. 2008.

G. Lugosi. Learning with an unreliable teacher. *Pattern Recognition*, 25(1):79–87, 1992.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

V. C. Raykar, S. Yu, L .H. Zhao, A. Jerebko, C. Florin, G. H. Valadez, L. Bogoni, and L. Moy. Supervised learning from multiple experts: Whom to trust when everyone lies a bit. In *ICML 2009: Proceedings of the 26th International Conference on Machine Learning*, pages 889–896, 2009.

V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 614–622, 2008.

P. Smyth. Learning with probabilistic supervision. In *Computational Learning Theory and Natural Learning Systems 3*, pages 163–182. MIT Press, 1995.

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 7*, 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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, 2008.

A. Sorokin and D. Forsyth. Utility data annotation with Amazon Mechanical Turk. In *Proceedings of the First IEEE Workshop on Internet Vision at CVPR 08*, pages 1–8, 2008.

J. S. Uebersax and W. M. Grove. A latent trait finite mixture model for the analysis of rating agreement. *Biometrics*, 49:823–835, 1993.

S. K. Warfield, K. H. Zou, and W. M. Wells. Simultaneous truth and performance level estimation (STAPLE): an algorithm for the validation of image segmentation. *IEEE Transactions on Medical Imaging*, 23(7):903–921, 2004.

J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043. 2009.

Journal of Machine Learning Research 11 (2010) 1323-1351 Submitted 6/09; Revised 11/09; Published 4/10

Unsupervised Supervised Learning I: Estimating Classification and Regression Errors without Labels

Pinar Donmez PINARD@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Guy Lebanon LEBANON@CC.GATECH.EDU
Krishnakumar Balasubramanian KRISHNAKUMAR3@GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Nicolò Cesa-Bianchi

Abstract

Estimating the error rates of classifiers or regression models is a fundamental task in machine learning which has thus far been studied exclusively using supervised learning techniques. We propose a novel unsupervised framework for estimating these error rates using only unlabeled data and mild assumptions. We prove consistency results for the framework and demonstrate its practical applicability on both synthetic and real world data.

Keywords: classification and regression, maximum likelihood, latent variable models

1. Introduction

A common task in machine learning is predicting a response variable $y \in \mathcal{Y}$ based on an explanatory variable $x \in \mathcal{X}$. Assuming a joint distribution $p(x,y)$ and a loss function $L(y,\hat{y})$, a predictor $f : \mathcal{X} \to \mathcal{Y}$ is characterized by an expected loss or risk function

$$R(f) = \mathsf{E}_{p(x,y)}\{L(y, f(x))\}.$$

For example, in classification we may have $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{1,\ldots,l\}$, and $L(y,\hat{y}) = I(y \neq \hat{y})$ where $I(A) = 1$ if A is true and 0 otherwise. The resulting risk is known as the 0-1 risk or simply the classification error rate

$$R(f) = P(f \text{ predicts the wrong class}).$$

In regression we may have $\mathcal{X} = \mathcal{Y} = \mathbb{R}$, and $L(y,\hat{y}) = (y-\hat{y})^2$. The resulting risk is the mean squared error

$$R(f) = \mathsf{E}_{p(x,y)}(y - f(x))^2.$$

We consider the case where we are provided with k predictors $f_i : \mathcal{X} \to \mathcal{Y}$, $i = 1,\ldots,k$ ($k \geq 1$) whose risks are unknown. The main task we are faced with is estimating the risks $R(f_1),\ldots,R(f_k)$ without using any labeled data whatsoever. The estimation of $R(f_i)$ is rather based on an estimator $\hat{R}(f_i)$ that uses unlabeled data $x^{(1)},\ldots,x^{(n)} \overset{\text{iid}}{\sim} p(x)$.

A secondary task that we consider is obtaining effective schemes for combining k predictors $f_1,\ldots,f_k$ in a completely unsupervised manner. We refer to these two tasks of risk estimation and predictor combination as unsupervised-supervised learning since they refer to unsupervised analysis of supervised prediction models.

It may seem surprising that unsupervised risk estimation is possible at all. After all in the absence of labels there is no ground truth that guides us in estimating the risks. However, as we show in this paper, if the marginal $p(y)$ is known it is possible in some cases to obtain a consistent estimator for the risks using only unlabeled data, that is,

$$\lim_{n\to\infty} \hat{R}(f_i;x^{(1)},\ldots,x^{(n)}) = R(f_i) \quad \text{with probability 1}, \quad i=1,\ldots,k.$$

In addition to demonstrating consistency, we explore the asymptotic variance of the risk estimators and how it is impacted by changes in n (amount of unlabeled data), k (number of predictors), and $R(f_1),\ldots,R(f_k)$ (risks). We also demonstrate that the proposed estimation technique works well in practice on both synthetic and real world data.

The assumption that $p(y)$ is known seems restrictive, but there are plenty of cases where it holds. Examples include medical diagnosis ($p(y)$ is the well known marginal disease frequency), handwriting recognition/OCR ($p(y)$ is the easily computable marginal frequencies of different English letters), regression model for life expectancy ($p(y)$ is the well known marginal life expectancy tables). In these and other examples $p(y)$ is obtained from extremely accurate histograms.

There are several reasons that motivate our approach of using exclusively unlabeled data to estimate the risks. Labeled data may be unavailable due to privacy considerations where the predictors are constructed by organizations using training sets with private labels. For example, in medical diagnosis prediction, the predictors $f_1,\ldots,f_k$ may be obtained by k different hospitals, each using a private internal labeled set. Following the training stage, each hospital releases its predictor to the public who then proceed to estimate $R(f_1),\ldots,R(f_k)$ using a separate unlabeled data set.

Another motivation for using unlabeled data is domain adaptation where predictors that are trained on one domain, are used to predict data from a new domain from which we have only unlabeled data. For example, predictors are often trained on labeled examples drawn from the past but are used at test time to predict data drawn from a new distribution associated with the present. Here the labeled data used to train the predictors will not provide an accurate estimate due to differences in the test and train distributions.

Another motivation is companies releasing predictors to clients as black boxes (without their training data) in order to protect their intellectual property. This is the situation in business analytics and consulting. In any case, it is remarkable that without labels we can still accurately estimate supervised risks.

The collaborative nature of this diagnosis is especially useful for multiple predictors as the predictor ensemble $\{f_1,\ldots,f_k\}$ diagnoses itself. However, our framework is not restricted to a large k and works even for a single predictor with $k=1$. It may further be extended to the case of active learning where classifiers are queried for specific data and the case of semi-supervised learning where a small amount of labeled data is augmented by massive unlabeled data.

We proceed in the next section to describe the general framework and some important special cases. In Section 3 we discuss extensions to the general framework and in Section 4-5 we discuss the theory underlying our estimation process. In Section 6 we discuss practical optimization algorithms. Section 7 contains an experimental study. We conclude with a discussion in Section 8.

2. Unsupervised Risk Estimation Framework

We adopt the framework presented in Section 1 with the added requirement that the predictors $f_1,\ldots,f_k$ are stochastic, that is, their prediction $\hat{y} = f_i(x)$ (conditioned on x) is a random variable. Such stochasticity occurs if the predictors are conditional models predicting values according to their estimated probability, that is, f_i models a conditional distribution q_i and predicts y' with probability $q_i(y'|x)$.

As mentioned previously our goal is to estimate the risk associated with classification or regression models $f_1,\ldots,f_k$ based on unlabeled data $x^{(1)},\ldots,x^{(n)} \overset{\text{iid}}{\sim} p(x)$. The testing marginal and conditional distributions $p(x), p(y|x)$ may differ from the distributions used at training time for the different predictors. In fact, each predictor may have been trained on a completely different training distribution, or may have been designed by hand with no training data whatsoever. We consider the predictors as black boxes and do not assume any knowledge of their modeling assumptions or training processes.

At the center of our framework is the idea to define a parameter vector $\theta \in \Theta$ which characterizes the risks $R(f_1),\ldots,R(f_k)$, that is, $R(f_j) = g_j(\theta)$ for some function $g_j : \Theta \to \mathbb{R}$, $j = 1,\ldots,k$. The parameter vector θ is estimated from data by connecting it to the probabilities

$$p_j(y'|y) \overset{\text{def}}{=} p(f_j \text{ predicts } y' | \text{ true label is } y).$$

More specifically, we use a plug-in estimate $\hat{R}(f_j) = g_j(\hat{\theta})$ where $\hat{\theta}$ maximizes the likelihood of the predictor outputs $\hat{y}_j^{(i)} = f_j(x^{(i)})$ with respect to the model $p_\theta(\hat{y}) = \int p_\theta(\hat{y}|y)p(y)\,dy$. The precise equations are:

$$\hat{R}(f_j;\hat{y}^{(1)},\ldots,\hat{y}^{(n)}) = g_j(\hat{\theta}^{\text{mle}}(\hat{y}^{(1)},\ldots,\hat{y}^{(n)})) \quad \text{where} \tag{1}$$

$$\hat{y}^{(i)} \overset{\text{def}}{=} (\hat{y}_1^{(i)},\ldots,\hat{y}_k^{(i)})$$

$$\hat{y}_j^{(i)} \overset{\text{def}}{=} f_j(x^{(i)}),$$

$$\hat{\theta}^{\text{mle}}(\hat{y}^{(1)},\ldots,\hat{y}^{(n)}) = \arg\max \ell(\theta;\hat{y}^{(1)},\ldots,\hat{y}^{(n)}), \tag{2}$$

$$\ell(\theta;\hat{y}^{(1)},\ldots,\hat{y}^{(n)}) = \sum_{i=1}^{n} \log p_\theta(\hat{y}_1^{(i)},\ldots,\hat{y}_k^{(i)}) \tag{3}$$

$$= \sum_{i=1}^{n} \log \int_{\mathcal{Y}} p_\theta(\hat{y}_1^{(i)},\ldots,\hat{y}_k^{(i)}|y^{(i)})p(y^{(i)})\,d\mu(y^{(i)}).$$

The integral in (3) is over the unobserved label $y^{(i)}$ associated with $x^{(i)}$. It should be a continuous integral $\int_{y^{(i)}=-\infty}^{\infty}$ for regression and a finite summation $\sum_{y^{(i)}=1}^{l}$ for classification. For notational simplicity we maintain the integral sign for both cases with the understanding that it is over a continuous or discrete measure μ, depending on the topology of $\mathcal{Y}$. Note that (3) and its maximizer are computable without any labeled data. All that is required are the classifiers (as black boxes), unlabeled data $x^{(1)},\ldots,x^{(n)}$, and the marginal label distribution $p(y)$.

Besides being a diagnostic tool for the predictor accuracy, $\hat{\theta}^{\text{mle}}$ can be used to effectively aggregate $f_1,\ldots,f_j$ to predict the label of a new example x^{new}

$$\begin{aligned}\hat{y}^{\text{new}} &= \arg\max_{y\in\mathcal{Y}} p_{\hat{\theta}^{\text{mle}}}(y \mid f_1(x^{\text{new}}),\ldots,f_k(x^{\text{new}})) \\ &= \arg\max_{y\in\mathcal{Y}} p(y)\prod_{j=1}^{k} p_{\hat{\theta}_j^{\text{mle}}}(f_j(x^{\text{new}}) \mid y). \end{aligned} \tag{4}$$

As a result, our framework may be used to combine existing classifiers or regression models in a completely unsupervised manner.

There are three important research questions concerning the above framework. First, what are the statistical properties of $\hat{\theta}^{\text{mle}}$ and $\hat{R}$ (consistency, asymptotic variance). Second, how can we efficiently solve the maximization problem (2). And third, how does the framework work in practice. We address these three questions in Sections 4-5, 6, 7 respectively, We devote the rest of the current section to examine some important special cases of (2)-(3) and consider some generalizations in the next section.

2.1 Non-Collaborative Estimation of the Risks

In the non-collaborative case we estimate the risk of each one of the predictors $f_1,\ldots,f_k$ separately. This reduces the problem to that of estimating the risk of a single predictor, which is repeated k times for each one of the predictors. We thus assume in this subsection the framework (1)-(3) with $k=1$ with no loss of generality. For simplicity we denote the single predictor by f rather than f_1 and denote $g=g_1$ and $\hat{y}^{(i)}=\hat{y}_1^{(i)}$. The corresponding simplified expressions are

$$\begin{aligned}&\hat{R}(f;\hat{y}^{(1)},\ldots,\hat{y}^{(n)}) = g(\hat{\theta}^{\text{mle}}(\hat{y}^{(1)},\ldots,\hat{y}^{(n)})), \\ &\hat{\theta}^{\text{mle}}(\hat{y}^{(1)},\ldots,\hat{y}^{(n)}) = \arg\max_{\theta} \sum_{i=1}^{n} \log \int_{\mathcal{Y}} p_\theta(\hat{y}^{(i)}|y^{(i)})p(y^{(i)})\,d\mu(y^{(i)}) \end{aligned} \tag{5}$$

where $\hat{y}^{(i)}=f(x^{(i)})$.

We consider below several important special cases.

2.1.1 CLASSIFICATION

Assuming l labels $\mathcal{Y}=\{1,\ldots,l\}$, the classifier f defines a multivariate Bernoulli distribution $p_\theta(\hat{y}|y)$ mapping the true label y to $\hat{y}$

$$p_\theta(\hat{y}|y) = \theta_{\hat{y},y}. \tag{6}$$

where θ is the stochastic confusion matrix or noise model corresponding to the classifier f. In this case, the relationship between the risk $R(f)$ and the parameter θ is

$$R(f) = 1 - \sum_{y\in\mathcal{Y}} \theta_{yy}\,p(y). \tag{7}$$

Equations (6)-(7) may be simplified by assuming a symmetric error distribution (Cover and Thomas, 2005)

$$p_\theta(\hat{y}|y) = \theta^{I(\hat{y}=y)} \left(\frac{1-\theta}{l-1}\right)^{I(\hat{y}\neq y)}, \tag{8}$$
$$R(f) = 1-\theta$$

where I is the indicator function and $\theta \in [0,1]$ is a scalar corresponding to the classifier accuracy. Estimating θ by maximizing (5), with (6) or (8) substituting p_θ completes the risk estimation task.

In the simple binary case $l = 2, \mathcal{Y} = \{1,2\}$ with the symmetric noise model (8) the loglikelihood

$$\ell(\theta) = \sum_{i=1}^{n} \log \sum_{y^{(i)}=1}^{2} \theta^{I(\hat{y}^{(i)}=y^{(i)})}(1-\theta)^{I(\hat{y}^{(i)}\neq y^{(i)})} p(y^{(i)})$$

may be shown to have the following closed form maximizer

$$\hat{\theta}^{\text{mle}} = \frac{p(y=1) - m/n}{2p(y=1)-1}. \tag{9}$$

where $m \stackrel{\text{def}}{=} |\{i \in \{1,\ldots,n\} : \hat{y}^{(i)} = 2\}|$. The estimator (9) works well in practice and is shown to be a consistent estimator in the next section (i.e., it converges to the true parameter value). In cases where the symmetric noise model (8) does not hold, using (9) to estimate the classification risk may be misleading. For example, in some cases (9) may be negative. In these cases, using the more general model (6) instead of (8) should provide more accurate results. We discuss this further from theoretical and experimental perspectives in Sections 4-5, and 7 respectively.

2.1.2 REGRESSION

Assuming a regression equation

$$y = ax + \varepsilon, \qquad \varepsilon \sim N(0, \tau^2)$$

and an estimated regression model or predictor $\hat{y} = a'x$ we have

$$\hat{y} = a'x = a'a^{-1}(y-\varepsilon) = \theta y - \theta\varepsilon$$

where $\theta = a'a^{-1}$. Thus, in the regression case the distribution $p_\theta(\hat{y}|y)$ and the relationship between the risk and the parameter $R(f) = g(\theta)$ are

$$p_\theta(\hat{y}|y) = (2\pi\theta^2\tau^2)^{-1/2} \exp\left(-\frac{(\hat{y}-\theta y)^2}{2\theta^2\tau^2}\right), \tag{10}$$
$$R(f|y) = \mathsf{bias}^2(f) + \mathsf{Var}(f) = (1-\theta)^2 y^2 + \theta^2\tau^2,$$
$$R(f) = \theta^2\tau^2 + (1-\theta)^2 \mathsf{E}_{p(y)}(y^2).$$

Note that we consider regression as a stochastic estimator in that it predicts $y = a'x + \varepsilon$ or $y|x \sim N(a'x, \tau^2)$.

Assuming $p(y) = N(\mu_y, \sigma_y^2)$ (as is often done in regression analysis) we have

$$p_\theta(\hat{y}^{(i)}) = \int_{\mathbb{R}} p_\theta(\hat{y}^{(i)}|y)p(y)dy = (2\pi\theta^2\tau^2 2\pi\sigma_y^2)^{-1/2} \int_{\mathbb{R}} \exp\left(-\frac{(\hat{y}-\theta y)^2}{2\theta^2\tau^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}\right) dy \quad (11)$$

$$= \frac{1}{\theta\sqrt{2\pi(\tau^2+\sigma_y^2)}} \exp\left(\frac{(\hat{y}^{(i)})^2}{2\theta^2\tau^2}\left(\frac{\sigma_y^2}{\sigma_y^2+\tau^2}-1\right) + \frac{\mu_y^2}{2\sigma_y^2}\left(\frac{\tau^2}{\sigma_y^2+\tau^2}-1\right) + \frac{\hat{y}^{(i)}\mu_y}{\theta\left(\tau^2+\sigma_y^2\right)}\right)$$

where we used the following lemma in the last equation.

Lemma 1 (e.g., Papoulis, 1984)

$$\int_{-\infty}^{\infty} A e^{-Bx^2+Cx+D}\, dx = A\sqrt{\frac{\pi}{B}} \exp\left(C^2/4B + D\right)$$

where A, B, C, D are constants that do not depend on x.

In this case the loglikelihood simplifies to

$$\ell(\theta) = -n\log\left(\theta\sqrt{2\pi(\tau^2+\sigma_y^2)}\right) - \left(\frac{\sum_{i=1}^n (\hat{y}^{(i)})^2}{2(\tau^2+\sigma_y^2)}\right)\frac{1}{\theta^2} + \left(\frac{\mu_y\sum_{i=1}^n \hat{y}^{(i)}}{\tau^2+\sigma_y^2}\right)\frac{1}{\theta} - n\frac{\mu_y^2}{2(\sigma_y^2+\tau^2)}$$

which can be shown to have the following closed form maximizer

$$\hat{\theta}^{\text{mle}} = -\frac{\mu_y\sum_{i=1}^n \hat{y}^{(i)}}{2n(\tau^2+\sigma_y^2)} \pm \sqrt{\frac{\left(\mu_y\sum_{i=1}^n \hat{y}^{(i)}\right)^2}{4n^2(\tau^2+\sigma_y^2)^2} + \frac{\sum_{i=1}^n (\hat{y}^{(i)})^2}{n(\tau^2+\sigma_y^2)}}$$

where the two roots correspond to the two cases where $\theta = a'/a > 0$ and $\theta = a'/a < 0$.

The univariate regression case described above may be extended to multiple explanatory variables, that is, $y = Ax + \varepsilon$ where y, x, ε are vectors and A is a matrix. This is an interesting extension which falls beyond the scope of the current paper.

2.1.3 NOISY GAUSSIAN CHANNEL

In this case our predictor f corresponds to a noisy channel mapping a real valued signal y to its noisy version $\hat{y}$. The aim is to estimate the mean squared error or noise level $R(f) = \mathsf{E}\,\|y-\hat{y}\|^2$. In this case the distribution $p_\theta(\hat{y}|y)$ and the relationship between the risk and the parameter $R(f) = g(\theta)$ are

$$p_\theta(\hat{y}|y) = (2\pi\theta^2)^{-1/2}\exp\left(-\frac{(\hat{y}-y)^2}{2\theta^2}\right),$$
$$R(f|y) = \theta^2,$$
$$R(f) = \theta^2 \mathsf{E}_{p(y)}(y).$$

The loglikelihood and other details in this case are straightforward variations on the linear regression case described above. We therefore concentrate in this paper on the classification and linear regression cases.

As mentioned above, in both classification and regression, estimating the risks for $k \geq 2$ predictors rather than a single one may proceed by repeating the optimization process described above for each predictor separately. That is $\hat{R}(f_j) = g_j(\hat{\theta}_j^{\text{mle}})$ where $\hat{\theta}_1^{\text{mle}}, \ldots, \hat{\theta}_k^{\text{mle}}$ are estimated by maximizing k different loglikelihood functions. In some cases the convergence rate to the true risks can be accelerated by jointly estimating the risks $R(f_1), \ldots, R(f_k)$ in a collaborative fashion. Such collaborative estimation is possible under some assumptions on the statistical dependency between the noise processes defining the k predictors. We describe below such an assumption followed by a description of more general cases.

2.2 Collaborative Estimation of the Risks: Conditionally Independent Predictors

We have previously seen how to estimate the risks of k predictors by separately applying (1) to each predictor. If the predictors are known to be conditionally independent given the true label, that is, $p_\theta(\hat{y}_1, \ldots, \hat{y}_k|y) = \prod_j p_{\theta_j}(\hat{y}_j|y)$ the loglikelihood (3) simplifies to

$$\ell(\theta) = \sum_{i=1}^{n} \log \int_{\mathcal{Y}} \prod_{j=1}^{k} p_{\theta_j}(\hat{y}_j^{(i)}|y^{(i)}) p(y^{(i)}) \, d\mu(y^{(i)}), \quad \text{where} \quad \hat{y}_j^{(i)} = f_j(x^{(i)}) \tag{12}$$

and p_{θ_j} above is (6) or (8) for classification and (10) for regression. Maximizing the loglikelihood (12) jointly over $\theta_1, \ldots, \theta_k$ results in estimators $\hat{R}(f_1), \ldots, \hat{R}(f_k)$ that converge to the true value faster than the non-collaborative MLE (5) (more on this in Section 7). Equation (12) does not have a closed form maximizer requiring the use of iterative computational techniques.

The conditional independence of the predictors is a much weaker condition than the independence of the predictors which is very unlikely to hold. In our case, each predictor f_j has its own stochastic noise operator $T_j(r,s) = p(\hat{y} = r|y = s)$ (regression) or matrix $[T_j]_{rs} = p_j(\hat{y} = r|y = s)$ (classification) where $T_1, \ldots, T_k$ may be arbitrarily specified. In particular, some predictors may be similar, for example, $T_i \approx T_j$, and some may be different, for example, $T_i \not\approx T_j$. The conditional independence assumption that we make in this subsection is that conditioned on the latent label y the predictions of the predictors proceed stochastically according to $T_1, \ldots, T_k$ in an independent manner.

Figure 1 displays the loglikelihood functions $\ell(\theta)$ for three different data set sizes $n = 100, 250, 500$. As the size n of the unlabeled data grows the curves become steeper and $\hat{\theta}_n^{\text{mle}}$ approach θ^{true}. Figure 2 displays a similar figure for $k = 1$ in the case of regression.

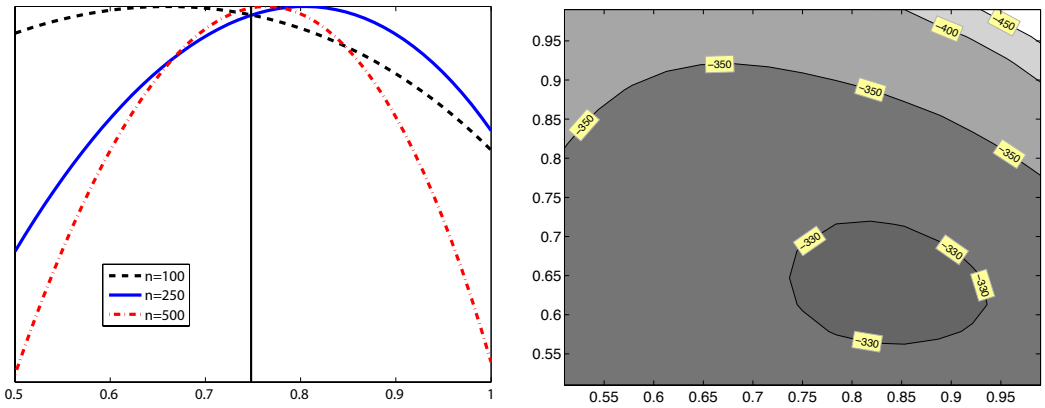


Figure 1: A plot of the loglikelihood functions $\ell(\theta)$ in the case of classification for $k = 1$ (left, $\theta^{\text{true}} = 0.75$) and $k = 2$ (right, $\theta^{\text{true}} = (0.8, 0.6)^\top$). The loglikelihood was constructed based on random samples of unlabeled data with sizes $n = 100, 250, 500$ (left) and $n = 250$ (right) and $p(y = 1) = 0.75$. In the left panel the y values of the curves were scaled so their maxima would be aligned. For $k = 1$ the estimators $\hat{\theta}^{\text{mle}}$ (and their errors $|\hat{\theta}^{\text{mle}} - 0.75|$) for $n = 100, 250, 500$ are 0.6633 (0.0867), 0.8061 (0.0561), 0.765 (0.0153). As additional unlabeled examples are added the loglikelihood curves become steeper and their maximizers become more accurate and closer to θ^{true}.

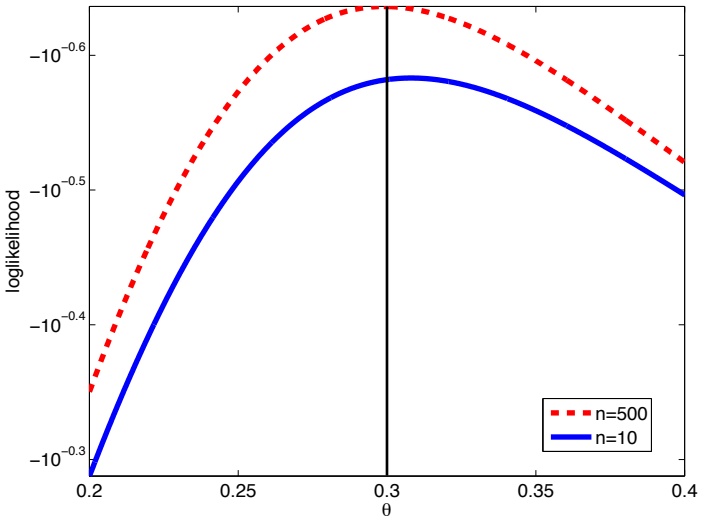


Figure 2: A plot of the loglikelihood function $\ell(\theta)$ in the case of regression for $k = 1$ with $\theta^{\text{true}} = 0.3, \tau = 1, \mu_y = 0$ and $\sigma_y = 0.2$. As additional unlabeled examples are added the loglikelihood curve become steeper and their maximizers get closer to the true parameter θ^{true} resulting in a more accurate risk estimate.

In the case of regression (12) involves an integral over a product of $k+1$ Gaussians, assuming that $y \sim N(\mu_y, \sigma_y^2)$. In this case the integral in (12) simplifies to

$$\begin{aligned}
p_\theta(\hat{y}_1^{(i)}, \ldots, \hat{y}_k^{(i)}) &= \int_{-\infty}^{\infty} \left(\prod_{j=1}^{k} \frac{1}{\theta_j \tau \sqrt{2\pi}} e^{-\left(\hat{y}_j^{(i)} - \theta_j y^{(i)}\right)^2 \Big/ 2\theta_j^2 \tau^2} \right) \frac{1}{\sigma_y \sqrt{2\pi}} e^{-\left(y^{(i)} - \mu_y\right)^2 \Big/ 2\sigma_y^2} dy^{(i)} \\
&= \frac{1}{\tau^k (\sqrt{2\pi})^{k+1} \sigma_y \prod_{j=1}^{k} \theta_j} \int_{-\infty}^{\infty} \exp\left[-\frac{1}{2} \left(\left(\frac{y^{(i)} - \mu_y}{\sigma_y} \right)^2 + \sum_{j=1}^{k} \left(\frac{y^{(i)}}{\tau} - \frac{\hat{y}_j^{(i)}}{\tau \theta_j} \right)^2 \right) \right] dy^{(i)} \\
&= \frac{\int_{-\infty}^{\infty} \exp\left(-\frac{1}{2} \left(\frac{1}{\sigma_y^2} + \frac{k}{\tau^2} \right) (y^{(i)})^2 + \left(\frac{\mu_y}{\sigma_y^2} + \sum_{j=1}^{k} \frac{\hat{y}_j^{(i)}}{\tau^2 \theta_j} \right) y^{(i)} - \frac{1}{2} \left(\frac{\mu_y^2}{\sigma_y^2} + \sum_{j=1}^{k} \frac{(\hat{y}_j^{(i)})^2}{\tau^2 \theta_j^2} \right) \right)}{\tau^k (\sqrt{2\pi})^{k+1} \sigma_y \prod_{j=1}^{k} \theta_j} \\
&= \frac{\sqrt{\pi} \left[\frac{1}{2} \left(\frac{1}{\sigma_y^2} + \frac{k}{\tau^2} \right) \right]^{-1/2}}{\tau^k (\sqrt{2\pi})^{k+1} \sigma_y \prod_{j=1}^{k} \theta_j} \exp\left(\frac{\left(\frac{\mu_y}{\sigma_y^2} + \sum_{j=1}^{k} \frac{\hat{y}_j^{(i)}}{\tau^2 \theta_j} \right)^2}{2 \left(\frac{1}{\sigma_y^2} + \frac{k}{\tau^2} \right)} - \sum_{j=1}^{k} \frac{(\hat{y}_j^{(i)})^2}{2\tau^2 \theta_j^2} - \frac{\mu_y^2}{2\sigma_y^2} \right)
\end{aligned} \tag{13}$$

where the last equation was obtained using Lemma 1 concerning Gaussian integrals. Note that this equation does not have a closed form maximizer requiring the use of iterative computational techniques.

2.3 Collaborative Estimation of the Risks: Conditionally Correlated Predictors

In some cases the conditional independence assumption made in the previous subsection does not hold and the factorization (12) is violated. In this section, we discuss how to relax this assumption in the classification case. A similar approach may also be used for regression. We omit the details here due to notational clarity.

There are several ways to relax the conditional independence assumption. Most popular, perhaps, is the mechanism of hierarchical loglinear models for categorical data (Bishop et al., 1975). For example, generalizing our conditional independence assumption to second-order interaction log-linear models we have

$$\log p(\hat{y}_1, \ldots, \hat{y}_k | y) = \alpha_y + \sum_{i=1}^{l} \beta_{i,\hat{y}_i,y} + \sum_{i<j} \gamma_{i,j,\hat{y}_i,\hat{y}_j,y} \tag{14}$$

where the following ANOVA-type parameter constraints are needed (Bishop et al., 1975)

$$\begin{aligned}
0 &= \sum_{\hat{y}_i} \beta_{i,\hat{y}_i,y} \quad \forall i, y, \\
0 &= \sum_{\hat{y}_i} \gamma_{i,j,\hat{y}_i,\hat{y}_j,y} = \sum_{\hat{y}_j} \gamma_{i,j,\hat{y}_i,\hat{y}_j,y} \quad \forall i, j, y.
\end{aligned} \tag{15}$$

The β parameters in (14) correspond to the order-1 interaction between the variables $\hat{y}_1, \ldots, \hat{y}_k$, conditioned on y. They correspond to the θ_i in the independent formulation (6)-(8). The γ parameters capture two-way interactions which do not appear in the conditionally independent case. Indeed, setting $\gamma_{i,j,\hat{y}_i,\hat{y}_j,y} = 0$ retrieves the independent models (6)-(8).

In the case of classification, the number of degrees of freedom or free unconstrained parameters in (14) depends on whether the number of classes is 2 or more and what additional assumptions exist on β and γ. For example, assuming that the probability of f_i, f_j making an error depends on the true class y but not on the predicted classes $\hat{y}_i, \hat{y}_j$ results in a $k+k^2$ parameters. Relaxing that assumption but assuming binary classification results in $2k+4k^2$ parameters. The estimation and aggregation techniques described in Section 2.1 work as before with a slight modification of replacing (6)-(8) with variations based on (14) and enforcing the constraints (15).

Equation (14) captures two-way interactions but cannot model higher order interactions. However, three-way and higher order interaction models are straightforward generalizations of (14) culminating in the full loglinear model which does not make any assumption on the statistical dependency of the noise operators $T_1, \ldots, T_k$. However, as we weaken the assumptions underlying the loglinear models and add higher order interactions the number of parameters increases adding to the difficulty in estimating the risks $R(f_1), \ldots, R(f_k)$.

In our experiments on real world data (see Section 7), it is often the case that maximizing the loglikelihood under the conditionally independent assumption (12) provides adequate accuracy and there is no need for the more general (14)-(15). Nevertheless, we include here the case of loglinear models as it may be necessary in some situations.

3. Extensions: Missing Values, Active Learning, and Semi-Supervised Learning

In this section, we discuss extensions to the current framework. Specifically, we consider extending the framework to the cases of missing values, active and semi-supervised learning.

Occasionally, some predictors are unable to provide their output over specific data points. That is assuming a data set $x^{(1)}, \ldots, x^{(n)}$ each predictor may provide output on an arbitrary subset of the data points $\{f_j(x^{(i)}) : i \in S_j\}$, where $S_j \subset \{1, \ldots, n\}$, $j = 1, \ldots, k$.

Commonly referred to as a missing value situation, this scenario may apply in cases where different parts of the unlabeled data are available to the different predictors at test time due to privacy, computational complexity, or communication cost. Another example where this scenario applies is active learning where operating f_j involves a certain cost $c_j \geq 0$ and it is not advantageous to operate all predictors with the same frequency for the purpose of estimating the risks $R(f_1), \ldots, R(f_k)$. Such is the case when f_j corresponds to judgments obtained from human experts or expensive machinery that is busy serving multiple clients. Active learning fits into this situation with S_j denoting the set of selected data points for each predictor.

We proceed in this case by defining indicators β_{ji} denoting whether predictor j is available to emit $f_j(x^{(i)})$. The risk estimation proceeds as before with the observed likelihood modified to account for the missing values.

In the case of collaborative estimation with conditional independence, the estimator and loglikelihood become

$$\begin{aligned}
\hat{\theta}_n^{\text{mle}} &= \arg\max_{\theta} \ell(\theta), \\
\ell(\theta) &= \sum_{i=1}^{n} \log \sum_{r:\beta_{ri}=0} \int_{\mathcal{Y}} p_\theta(\hat{y}_1^{(i)}, \ldots, \hat{y}_k^{(i)})\, d\mu(\hat{y}_r^{(i)}) \qquad (16) \\
&= \sum_{i=1}^{n} \log \sum_{r:\beta_{ri}=0} \iint_{\mathcal{Y}^2} p_\theta(\hat{y}_1^{(i)}, \ldots, \hat{y}_k^{(i)} | y^{(i)}) p(y^{(i)})\, d\mu(\hat{y}_r^{(i)}) d\mu(y^{(i)})
\end{aligned}$$

where p_θ may be further simplified using the non-collaborative approach, or using the collaborative approach with conditional independence or loglinear model assumptions.

In the case of semi-supervised learning a small set of labeled data is augmented by a large set of unlabeled data. In this case our framework remains as before with the likelihood summing over the observed labeled and unlabeled data. For example, in the case of collaborative estimation with conditional independence we have

$$\ell(\theta) = \sum_{i=1}^{n} \log \int_{\mathcal{Y}} \prod_{j=1}^{k} p_{\theta_j}(\hat{y}_j^{(i)}|y^{(i)}) p(y^{(i)})\, d\mu(y^{(i)}) + \sum_{i=n+1}^{m} \log \prod_{j=1}^{k} p_{\theta_j}(\hat{y}_j^{(i)}|y^{(i)}) p(y^{(i)}).$$

The different variations concerning missing values, active learning, semi-supervised learning, and non-collaborative or collaborative estimation with conditionally independent or correlated noise processes can all be combined in different ways to provide the appropriate likelihood function. This provides substantial modeling flexibility.

4. Consistency of $\hat{\theta}_n^{\text{mle}}$ and $\hat{R}(f_j)$

In this and the next section we consider the statistical behavior of the estimator $\hat{\theta}_n^{\text{mle}}$ defined in (2) and the risk estimator $\hat{R}(f_j) = g_j(\hat{\theta}^{\text{mle}})$ defined in (1). The analysis is conducted under the assumption that the vectors of observed predictors outputs $\hat{y}^{(i)} = (\hat{y}_1^{(i)}, \ldots, \hat{y}_k^{(i)})$ are iid samples from the distribution

$$p_\theta(\hat{y}) = p_\theta(\hat{y}_1, \ldots, \hat{y}_k) = \int_{\mathcal{Y}} p_\theta(\hat{y}_1, \ldots, \hat{y}_k|y) p(y)\, d\mu(y).$$

We start by investigating whether estimator $\hat{\theta}^{\text{mle}}$ in (2) converges to the true parameter value. More formally, strong consistency of the estimator $\hat{\theta}_n^{\text{mle}} = \hat{\theta}(\hat{y}^{(1)}, \ldots, \hat{y}^{(n)})$, $\hat{y}^{(1)}, \ldots, \hat{y}^{(n)} \overset{\text{iid}}{\sim} p_{\theta_0}$ is defined as strong convergence of the estimator to θ_0 as $n \to \infty$ (Ferguson, 1996)

$$\lim_{n\to\infty} \hat{\theta}_n^{\text{mle}}(\hat{y}^{(1)}, \ldots, \hat{y}^{(n)}) = \theta_0 \text{ with probability 1.}$$

In other words as the number of samples n grows, the estimator will surely converge to the true parameter θ_0 governing the data generation process.

Assuming that the risks $R(f_j) = g_j(\theta)$ are defined using continuous functions g_j, strong consistency of $\hat{\theta}^{\text{mle}}$ implies strong convergence of $\hat{R}(f_j)$ to $R(f_j)$. This is due to the fact that continuity preserves limits. Indeed, as the g_j functions are continuous in both the classification and regression cases, strong consistency of the risk estimators $\hat{R}(f_j)$ reduces to strong consistency of the estimators $\hat{\theta}^{\text{mle}}$.

It is well known that the maximum likelihood estimator is often strongly consistent. Consider, for example, the following theorem.

Proposition 2 (e.g., Ferguson, 1996) *Let* $\hat{y}^{(1)}, \ldots, \hat{y}^{(n)} \overset{\text{iid}}{\sim} p_{\theta_0}$, $\theta_0 \in \Theta$. *If the following conditions hold*

1. Θ *is compact* *(compactness)*
2. $p_\theta(\hat{y})$ *is upper semi-continuous in* θ *for all* $\hat{y}$ *(continuity)*
3. *There exists a function* $K(\hat{y})$ *such that* $E_{p_{\theta_0}}|K(\hat{y})| < \infty$ *and* $\log p_\theta(\hat{y}) - \log p_{\theta_0}(\hat{y}) \le K(\hat{y}) \quad \forall \hat{y} \quad \forall \theta$ *(boundedness)*
4. *For all* θ *and sufficiently small* $\rho > 0$, $\sup_{|\theta'-\theta|<\rho} p_{\theta'}(\hat{y})$ *is measurable in* $\hat{y}$ *(measurability)*
5. $p_\theta \equiv p_{\theta_0} \Rightarrow \theta = \theta_0$ *(identifiability)*

then the maximum likelihood estimator is strongly consistent, that is, $\hat{\theta}^{mle} \to \theta_0$ *as* $n \to \infty$ *with probability 1.*

Note that $p_\theta(\hat{y})$ in the proposition above corresponds to $\int_{\mathcal{Y}} p_\theta(\hat{y}|y)p(y)\,d\mu(y)$ in our framework. That is the MLE operates on the observed data or predictor output $\hat{y}^{(1)},\ldots,\hat{y}^{(n)}$ that is sampled iid from the distribution $p_{\theta_0}(\hat{y}) = \int_{\mathcal{Y}} p_{\theta_0}(\hat{y}|y)p(y)\,d\mu(y)$.

Of the five conditions above, the last condition of identifiability is the only one that is truly problematic. The first condition of compactness is trivially satisfied in the case of classification. In the case of regression it is satisfied assuming that the regression parameter and model parameter are finite and $a \neq 0$ as the estimator $\hat{\theta}^{\text{mle}}$ will eventually lie in a compact set. The second condition of continuity is trivially satisfied in both classification and regression as the function $\int_{\mathcal{Y}} p_\theta(\hat{y}|y)p(y)\,d\mu(y)$ is continuous in θ once $\hat{y}$ is fixed. The third condition is trivially satisfied for classification (finite valued y). In the case of regression due to conditions 1,2 (compactness and semi-continuity) we can replace the quantifier $\forall\theta$ with a particular value $\theta' \in \Theta$ representing worst case situation in the bound of the logarithm difference. Then, the bound K may be realized by the difference of log terms (with respect to that worst case θ') whose expectation converges to the KL divergence which in turn is never ∞ for Gaussian distributions or its derivatives. The fourth condition of measurability follows as p_θ is specified in terms of compositions, summations, multiplications, and point-wise limits of well-known measurable functions.

The fifth condition of identifiability states that if $p_\theta(\hat{y})$ and $p_{\theta_0}(\hat{y})$ are identical as functions, that is, they are identical for every value of $\hat{y}$, then necessarily $\theta = \theta_0$. This condition does not hold in general and needs to be verified in each one of the special cases.

We start with establishing consistency in the case of classification where we rely on a symmetric noise model (8). The non-symmetric case (6) is more complicated and is treated afterwards. We conclude the consistency discussion with an examination of the regression case.

4.1 Consistency of Classification Risk Estimation

Proposition 3 *Let* $f_1,\ldots,f_k$ *be classifiers* $f_i : \mathcal{X} \to \mathcal{Y}$, $|\mathcal{Y}| = l$, *with conditionally independent noise processes described by* (8). *If the classifiers are weak learners, that is,* $1/l < 1 - err(f_i) < 1$ *and* $p(y)$ *is not uniform the unsupervised collaborative diagnosis model is identifiable.*

Corollary 4 *Let* $f_1,\ldots,f_k$ *be classifiers* $f_i : \mathcal{X} \to \mathcal{Y}$ *with* $|\mathcal{Y}| = l$ *and noise processes described by* (8). *If the classifiers are weak learners, that is,* $1/l < 1 - err(f_i) < 1$, *and* $p(y)$ *is not uniform the unsupervised non-collaborative diagnosis model is identifiable.*

Proof Proving identifiability in the non-collaborative case proceeds by invoking Proposition 3 (whose proof is given below) with $k = 1$ separately for each classifier. The conditional independence assumption in Proposition 3 becomes redundant in this case of a single classifier, resulting in identifiability of $p_{\theta_j}(\hat{y}_j)$ for each $j = 1,\ldots,k$ ■

Corollary 5 *Under the assumptions of Proposition 3 or Corollary 4 the unsupervised maximum likelihood estimator is consistent, that is,*

$$P\left(\lim_{n\to\infty} \hat{\theta}_n^{mle}(\hat{y}^{(1)},\ldots,y^{(n)}) = (\theta_1^{true},\ldots,\theta_k^{true})\right) = 1.$$

Consequentially, assuming that $R(f_j) = g_j(\theta), j = 1,\ldots,k$ with continuous g_j we also have

$$P\left(\lim_{n\to\infty} \hat{R}(f_j; y^{(1)},\ldots,y^{(n)}) = R(f_j), \quad \forall j = 1,\ldots,k\right) = 1.$$

Proof Proposition 3 or Corollary 4 establishes identifiability, which in conjunction with Proposition 2 proves the corollary. ■

Proof (**for Proposition 3**) We prove identifiability by induction on k. In the base case of $k = 1$, we have a set of l equations, corresponding to $i = 1,2\ldots l$,

$$\begin{aligned} p_\theta(\hat{y}_1 = i) &= p(y=i)\theta_1 + \left(\sum_{j\neq i} p(y=j)\right)\frac{(1-\theta_1)}{(l-1)} \\ &= p(y=i)\theta_1 + (1-p(y=i))\frac{(1-\theta_1)}{(l-1)} \\ &= \frac{\theta_1(lp(y=i)-1)+1-p(y=i)}{(l-1)} \end{aligned}$$

from which we can see that if $\eta \neq \theta$ and $p(y=i) \neq 1/l$ then $p_\theta(\hat{y}_1) \neq p_\eta(\hat{y}_1)$. This proves identifiability for the base case of $k = 1$.

Next, we assume identifiability holds for k and prove that it holds for $k+1$. We do so by deriving a contradiction from the assumption that identifiability holds for k but not for $k+1$. We denote the parameters corresponding to the k labelers by the vectors $\theta, \eta \in [0,1]^k$ and the parameters corresponding the additional $k+1$ labeler by θ_{k+1}, η_{k+1}.

In the case of k classifiers we have

$$p_\theta(\hat{y}_1,\ldots,\hat{y}_k) = \sum_{i=1}^{l} p_\theta(\hat{y}_1,\ldots,\hat{y}_k|y=i)p(y=i) = \sum_{i=1}^{l} G(\mathcal{A}_i,\theta)$$

where

$$\begin{aligned} G(\mathcal{A}_i,\theta) &\stackrel{\text{def}}{=} p(y=i)\prod_{j\in\mathcal{A}_i}\theta_j \cdot \prod_{j\notin\mathcal{A}_i}\frac{(1-\theta_j)}{(l-1)}, \\ \mathcal{A}_i &\stackrel{\text{def}}{=} \{j \in \{1,2...,k\} : \hat{y}_j = i\}. \end{aligned}$$

Note that the $\mathcal{A}_1,\ldots,\mathcal{A}_l$ form a partition of $\{1,\ldots,k\}$, that is, they are disjoint and their union is $\{1,\ldots,k\}$.

In order to have unidentifiability for the $k+1$ classifiers we need $(\theta,\theta_{k+1}) \neq (\eta,\eta_{k+1})$ and the following l equations (corresponding to $\hat{y}_{k+1} = 1,2,\ldots,l$) to hold for any $\hat{y}_1,\ldots,\hat{y}_k$ which corre-

sponds to any partition $\mathcal{A}_1,\ldots,\mathcal{A}_l$

$$\theta_{k+1}G(\mathcal{A}_1,\theta)+\frac{(1-\theta_{k+1})}{(l-1)}\sum_{i\neq 1}G(\mathcal{A}_i,\theta)=\eta_{k+1}G(\mathcal{A}_1,\eta)+\frac{(1-\eta_{k+1})}{(l-1)}\sum_{i\neq 1}G(\mathcal{A}_i,\eta),$$
$$\theta_{k+1}G(\mathcal{A}_2,\theta)+\frac{(1-\theta_{k+1})}{(l-1)}\sum_{i\neq 2}G(\mathcal{A}_i,\theta)=\eta_{k+1}G(\mathcal{A}_2,\eta)+\frac{(1-\eta_{k+1})}{(l-1)}\sum_{i\neq 2}G(\mathcal{A}_i,\eta),$$
$$\vdots$$
$$\theta_{k+1}G(\mathcal{A}_l,\theta)+\frac{(1-\theta_{k+1})}{(l-1)}\sum_{i\neq l}G(\mathcal{A}_i,\theta)=\eta_{k+1}G(\mathcal{A}_l,\eta)+\frac{(1-\eta_{k+1})}{(l-1)}\sum_{i\neq l}G(\mathcal{A}_i,\eta).$$

We consider two cases in which $(\theta,\theta_{k+1})\neq(\eta,\eta_{k+1})$: (a) $\theta\neq\eta$, and (b) $\theta=\eta,\theta_{k+1}\neq\eta_{k+1}$. In the case of (a) we add the l equations above which marginalizes $\hat{y}_{k+1}$ out of $p_\theta(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1})$ and $p_\eta(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1})$ to provide

$$\sum_{i=1}^{l}G(\mathcal{A}_i,\theta)=\sum_{i=1}^{l}G(\mathcal{A}_i,\eta)$$

which together with $\theta\neq\eta$ contradicts the identifiability for the case of k classifiers.

In case (b) we have from the l equations above

$$\theta_{k+1}G(\mathcal{A}_t,\theta)+\frac{1-\theta_{k+1}}{l-1}\left(\sum_{i=1}^{l}G(\mathcal{A}_i,\theta)-G(\mathcal{A}_t,\theta)\right)$$
$$=\eta_{k+1}G(\mathcal{A}_t,\eta)+\frac{1-\eta_{k+1}}{l-1}\left(\sum_{i=1}^{l}G(\mathcal{A}_i,\eta)-G(\mathcal{A}_t,\eta)\right)$$

for any $t\in\{1,\ldots,l\}$ which simplifies to

$$0=(\theta_{k+1}-\eta_{k+1})\left(lG(\mathcal{A}_t,\theta)-\sum_{i=1}^{l}G(\mathcal{A}_i,\theta)\right)\quad t=1,\ldots,k.$$

As we assume at this point that $\theta_{k+1}\neq\eta_{k+1}$ the above equality entails

$$lG(\mathcal{A}_t,\theta)=\sum_{i=1}^{l}G(\mathcal{A}_i,\theta). \tag{17}$$

We show that (17) cannot hold by examining separately the cases $p(y=t)>1/l$ and $p(y=t)<1/l$. Recall that there exists a t for which $p(y=t)\neq 1/l$ since the proposition requires that $p(y)$ is not uniform.

If $p(y=t)>1/l$ we choose $\mathcal{A}_t=\{1,\ldots,k\}$ and obtain

$$lp(y=t)\prod_{j=1}^{k}\theta_j=\sum_{i\neq t}p(y=i)\prod_{j=1}^{k}\frac{1-\theta_j}{l-1}+p(y=t)\prod_{j=1}^{k}\theta_j$$
$$(l-1)p(y=t)\prod_{j=1}^{k}\theta_j=(1-p(y=t))\prod_{j=1}^{k}\frac{1-\theta_j}{l-1}$$
$$p(y=t)\prod_{j=1}^{k}\theta_j=\frac{(1-p(y=t))}{(l-1)}\prod_{j=1}^{k}\frac{1-\theta_j}{l-1}$$

which cannot hold as the term on the left hand side is necessarily larger than the term on the right hand side (if $p(y=t) > 1/l$ and $\theta_j > 1/l$). In the case $p(y=t) < 1/l$ we choose $\mathcal{A}_s = \{1,\ldots,k\}$, $s \neq t$ to obtain

$$lp(y=t)\prod_{j=1}^{k}\frac{1-\theta_j}{l-1} = \sum_{i\neq s} p(y=i)\prod_{j=1}^{k}\frac{1-\theta_j}{l-1} + p(y=s)\prod_{j=1}^{k}\theta_j$$

$$(lp(y=t) - p(y\neq s))\prod_{j=1}^{k}\frac{1-\theta_j}{l-1} = p(y=s)\prod_{j=1}^{k}\theta_j$$

which cannot hold as the term on the left hand side is necessarily smaller than the term on the right hand side (if $p(y=t) < 1/l$ and $\theta_j > 1/l$).

Since we derived a contradiction to the fact that we have k-identifiability but not $k+1$ identifiability, the induction step is proven which establishes identifiability for any $k \geq 1$. ■

The conditions asserted above that $p(y) \neq 1/l$ and $1/l < 1 - \mathrm{err}(f_i) < 1$ are intuitive. If they are violated a certain symmetry may emerge which renders the model non-identifiable and the MLE estimator not consistent.

In the case of the non-collaborative estimation for binary classification with the non-symmetric noise model, the matrix θ in (6) is a 2×2 matrix with two degrees of freedom as each row sums to one. In particular we have $\theta_{11} = p_\theta(\hat{y}=1|y=1)$, $\theta_{12} = p_\theta(\hat{y}=1|y=2)$, $\theta_{21} = p_\theta(\hat{y}=2|y=1)$, $\theta_{22} = p_\theta(\hat{y}=2|y=2)$ with the overall risk $R(f) = 1-\theta_{11}p(y=1)-\theta_{22}p(y=2)$. Unfortunately, the matrix θ is not identifiable in this case and neither is the scalar parameter $\theta_{11}p(y=1)+\theta_{22}p(y=2)$ that can be used to characterize the risk.

We can, however, obtain a consistent estimator for θ (and therefore for $R(f)$) by first showing that the parameter $\theta_{11}p(y=1)-\theta_{22}p(y=2)$ is identifiable and then taking the intersection of two such estimators.

Lemma 6 *In the case of the non-collaborative estimation for binary classification with the non-symmetric noise model and $p(y) \neq 0$, the parameter $\theta_{11}p(y=1)-\theta_{22}p(y=2)$ is identifiable.*

Proof For two different parameterizations θ, η we have

$$p_\theta(\hat{y}=1) = p(y=1)\theta_{11} + (1-p(y=1))(1-\theta_{22}), \tag{18}$$

$$p_\theta(\hat{y}=2) = p(y=1)(1-\theta_{11}) + (1-p(y=1))\theta_{22} \tag{19}$$

and

$$p_\eta(\hat{y}=1) = p(y=1)\eta_{11} + (1-p(y=1))(1-\eta_{22}), \tag{20}$$

$$p_\eta(\hat{y}=2) = p(y=1)(1-\eta_{11}) + (1-p(y=1))\eta_{22}. \tag{21}$$

Equating the two Equations (18) and (20) we have

$$p(y=1)(\theta_{11}+\theta_{22}) + 1 - p(y=1) - \theta_{22} = p(y=1)(\eta_{11}+\eta_{22}) + 1 - p(y=1) - \eta_{22}$$

$$p(y=1)\theta_{11} - (1-p(y=1))\theta_{22} = p(y=1)\eta_{11} - (1-p(y=1))\eta_{22}$$

$$p(y=1)\theta_{11} - p(y=2)\theta_{22} = p(y=1)\eta_{11} - p(y=2)\eta_{22}$$

Similarly, equating Equation (19) and Equation (21) also results in $p(y=1)\theta_{11} - p(y=2)\theta_{22} = p(y=1)\eta_{11} - p(y=2)\eta_{22}$. As a result, we have

$$p_\theta \equiv p_\eta \quad \Rightarrow \quad p(y=1)\theta_{11} - p(y=2)\theta_{22} = p(y=1)\eta_{11} - p(y=2)\eta_{22}.$$

■

The above lemma indicates that we can use the maximum likelihood method to obtain a consistent estimator for the parameter $\theta_{11}p(y=1) - \theta_{22}p(y=2)$. Unfortunately the parameter $\theta_{11}p(y=1) - \theta_{22}p(y=2)$ does not have a clear probabilistic interpretation and does not directly characterize the risk. As the following proposition shows we can obtain a consistent estimator for the risk $R(f)$ if we have two populations of unlabeled data drawn from distributions with two distinct marginals $p_1(y)$ and $p_2(y)$.

Proposition 7 *Consider the case of the non-collaborative estimation of binary classification risk with the non-symmetric noise model. If we have access to two unlabeled data sets drawn independently from two distributions with different marginals, that is,*

$$x^{(1)}, \ldots, x^{(n)} \overset{\text{iid}}{\sim} p_1(x) = \sum_y p(x|y)p_1(y),$$
$$x'^{(1)}, \ldots, x'^{(m)} \overset{\text{iid}}{\sim} p_2(x) = \sum_y p(x|y)p_2(y)$$

we can obtain a consistent estimator for the classification risk $R(f)$.

Proof Operating the classifier f on both sets of unlabeled data we get two sets of observed classifier outputs $\hat{y}^{(1)}, \ldots, \hat{y}^{(n)}$, $\hat{y}'^{(1)}, \ldots, \hat{y}'^{(m)}$ where $\hat{y}^{(i)} \overset{\text{iid}}{\sim} \sum_y p_\theta(\hat{y}|y)p_1(y)$ and $\hat{y}'^{(i)} \overset{\text{iid}}{\sim} \sum_y p_\theta(\hat{y}|y)p_2(y)$. In particular, note that the marginal distributions $p_1(y)$ and $p_2(y)$ are different but the parameter matrix θ is the same in both cases as we operate the same classifier on samples from the same class conditional distribution $p(x|y)$.

Based on Lemma 6 we construct a consistent estimator for $p_1(y=1)\theta_{11} - p_1(y=2)\theta_{22}$ by maximizing the likelihood of $\hat{y}^{(1)}, \ldots, \hat{y}^{(n)}$. Similarly, we construct a consistent estimator for $p_2(y=1)\theta_{11} - p_2(y=2)\theta_{22}$ by maximizing the likelihood of $\hat{y}'^{(1)}, \ldots, \hat{y}'^{(m)}$. Note that $p_1(y=1)\theta_{11} - p_1(y=2)\theta_{22}$ and $p_2(y=1)\theta_{11} - p_2(y=2)\theta_{22}$ describe two lines in the 2-D space $(\theta_{11}, \theta_{22})$. Since the true value of θ_{11}, θ_{22} represent a point in that 2-D space belonging to both lines, it is necessarily the intersection of both lines (the lines cannot be parallel since their linear coefficients are distributions which are assumed to be different).

As n and m increase to infinity, the two estimators converge to the true parameter values. As a result, the intersection of the two lines described by the two estimators converges to the true values of $(\theta_{11}, \theta_{22})$ thus allowing reconstruction of the matrix θ and the risk $R(f)$. ■

Clearly, the conditions for consistency in the asymmetric case are more restricted than in the symmetric case. However, situations such as in Proposition 7 are not necessarily unrealistic. In many cases it is possible to identify two unlabeled sets with different distributions. For example, if y denotes a medical condition, it may be possible to obtain two unlabeled sets from two different

hospitals or two different regions with different marginal distribution corresponding to the frequency of the medical condition.

As indicated in the previous section, the risk estimation framework may be extended beyond non-collaborative estimation and collaborative conditionally independent estimation. In these extensions, the conditions for identifiability need to be determined separately, in a similar way to Corollary 4. A systematic way to do so may be obtained by noting that the identifiability equations

$$0 = p_\theta(\hat{y}_1,\ldots,\hat{y}_k) - p_\eta(\hat{y}_1,\ldots,\hat{y}_k) \quad \forall \hat{y}_1,\ldots,\hat{y}_k$$

is a system of polynomial equations in (θ,η). As a result, demonstrating lack of identifiability becomes equivalent to obtaining a solution to a system of polynomial equations. Using Hilbert's Nullstellensatz theorem we have that a solution to a polynomial system exists if the polynomial system defines a proper ideal of the ring of polynomials (Cox et al., 2006). As k increases the chance of identifiability failing decays dramatically as we have a system of l^k polynomials with $2k$ variables. Such an over-determined system with substantially more equations than variables is very unlikely to have a solution.

These observations serve as both an interesting theoretical connection to algebraic geometry as well as a practical tool due to the substantial research in computational algebraic geometry. See Sturmfels (2002) for a survey of computational algorithms and software associated with systems of polynomial equations.

4.2 Consistency of Regression Risk Estimation

In this section, we prove the consistency of the maximum likelihood estimator $\hat{\theta}^{\text{mle}}$ in the regression case. As in the classification case our proof centers on establishing identifiability.

Proposition 8 *Let $f_1,\ldots,f_k$ be regression models $f_i(x) = a_i'x$ with $y \sim N(\mu_y,\sigma_y^2)$, $y = ax+\varepsilon$. Assuming that $a \neq 0$ the unsupervised collaborative estimation model assuming conditionally independent noise processes* (12) *is identifiable.*

Corollary 9 *Let $f_1,\ldots,f_k$ be regression models $f_i(x) = a_i'x$ with $y \sim N(\mu_y,\sigma_y^2)$, $y = ax+\varepsilon$. Assuming that $a \neq 0$ the unsupervised non-collaborative estimation model* (12) *is identifiable.*

Proof Proving identifiability in the non-collaborative case proceeds by invoking Proposition 8 (whose proof is given below) with $k = 1$ separately for each regression model. The conditional independence assumption in Proposition 8 becomes redundant in this case of a single predictor, resulting in identifiability of $p_{\theta_j}(\hat{y}_j)$ for each $j = 1,\ldots,k$. ■

Corollary 10 *Under the assumptions of Proposition 8 or Corollary 9 the unsupervised maximum likelihood estimator is consistent, that is,*

$$P\left(\lim_{n\to\infty} \hat{\theta}_n^{mle}(\hat{y}^{(1)},\ldots,y^{(n)}) = (\theta_1^{true},\ldots,\theta_k^{true})\right) = 1.$$

Consequentially, assuming that $R(f_j) = g_j(\theta)$, $j = 1,\ldots,k$ with continuous g_j we also have

$$P\left(\lim_{n\to\infty} \hat{R}(f_j;y^{(1)},\ldots,y^{(n)}) = R(f_j), \quad \forall j = 1,\ldots,k\right) = 1.$$

Proof Proposition 8 or Corollary 9 establish identifiability, which in conjunction with Proposition 2 completes the proof. ■

Proof (of Proposition 8).

We will proceed, as in the case of classification, with induction on the number of predictors k. In the base case of $k=1$ we have derived $p_{\theta_1}(\hat{y}_1)$ in Equation (11). Substituting in it $\hat{y}_1 = 0$ we get

$$P_{\theta_1}(\hat{y}_1 = 0) = \frac{1}{\theta_1\sqrt{2\pi(\tau^2+\sigma_y^2)}} \exp\left(\frac{\mu_y^2}{2\sigma_y^2}\left(\frac{\tau^2}{\sigma_y^2+\tau^2}-1\right)\right),$$

$$P_{\eta_1}(\hat{y}_1 = 0) = \frac{1}{\eta_1\sqrt{2\pi(\tau^2+\sigma_y^2)}} \exp\left(\frac{\mu_y^2}{2\sigma_y^2}\left(\frac{\tau^2}{\sigma_y^2+\tau^2}-1\right)\right).$$

The above expression leads to $\theta_1 \neq \eta_1 \Rightarrow p_{\theta_1}(\hat{y}_1 = 0) \neq p_{\eta_1}(\hat{y}_1 = 0)$ which implies identifiability.

In the induction step we assume identifiability holds for k and we prove that it holds also for $k+1$ by deriving a contradiction to the assumption that it does not hold. We assume that identifiability fails in the case of $k+1$ due to differing parameter values, that is,

$$p_{(\theta,\theta_{k+1})}(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1}) = p_{(\eta,\eta_{k+1})}(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1}) \;\forall \hat{y}_j \in \mathbb{R}\; j = 1,\ldots,k+1 \tag{22}$$

with $(\theta,\theta_{k+1}) \neq (\eta,\eta_{k+1})$ where $\theta,\eta \in \mathbb{R}^k$. There are two cases which we consider separately: (a) $\theta \neq \eta$ and (b) $\theta = \eta$.

In case (a) we marginalize both sides of (22) with respect to $\hat{y}_{k+1}$ which leads to a contradiction to our assumption that identifiability holds for k

$$\int_{-\infty}^{\infty} p_{(\theta,\theta_{k+1})}(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1})d\hat{y}_{k+1} = \int_{-\infty}^{\infty} p_{(\eta,\eta_{k+1})}(\hat{y}_1,\ldots,\hat{y}_k,\hat{y}_{k+1})d\hat{y}_{k+1}$$
$$p_\theta(\hat{y}_1,\ldots,\hat{y}_k) = p_\eta(\hat{y}_1,\ldots,\hat{y}_k).$$

In case (b) $\theta = \eta$ and $\theta_{k+1} \neq \eta_{k+1}$. Substituting $\hat{y}_1 = \cdots = \hat{y}_{k+1} = 0$ in (22) (see (13) for a derivation) we have

$$P_{(\theta,\theta_{k+1})}(\hat{y}_1 = 0,\ldots,\hat{y}_{k+1} = 0) = P_{(\eta,\eta_{k+1})}(\hat{y}_1 = 0,\ldots,\hat{y}_{k+1} = 0)$$

or

$$\frac{\sqrt{\pi}\left[\frac{1}{2}\left(\frac{1}{\sigma_y^2}+\frac{k+1}{\tau^2}\right)\right]^{-1/2}}{\tau^{k+1}(\sqrt{2\pi})^{k+2}\sigma_y\theta_{k+1}\prod_{j=1}^k\theta_j} \exp\left(\frac{\left(\frac{\mu_y}{\sigma_y^2}\right)^2}{2\left(\frac{1}{\sigma_y^2}+\frac{k+1}{\tau^2}\right)} - \frac{\mu_y^2}{2\sigma_y^2}\right)$$
$$= \frac{\sqrt{\pi}\left[\frac{1}{2}\left(\frac{1}{\sigma_y^2}+\frac{k+1}{\tau^2}\right)\right]^{-1/2}}{\tau^{k+1}(\sqrt{2\pi})^{k+2}\sigma_y\eta_{k+1}\prod_{j=1}^k\eta_j} \exp\left(\frac{\left(\frac{\mu_y}{\sigma_y^2}\right)^2}{2\left(\frac{1}{\sigma_y^2}+\frac{k+1}{\tau^2}\right)} - \frac{\mu_y^2}{2\sigma_y^2}\right)$$

which cannot hold if $\theta = \eta$ but $\theta_{k+1} \neq \eta_{k+1}$. ■

5. Asymptotic Variance of $\hat{\theta}_n^{\text{mle}}$ and $\hat{R}$

A standard result from statistics is that the MLE has an asymptotically normal distribution with mean vector θ^{true} and variance matrix $(nJ(\theta^{\text{true}}))^{-1}$, where $J(\theta)$ is the $r \times r$ Fisher information matrix

$$J(\theta) = \mathsf{E}_{p_\theta}\{\nabla \log p_\theta(\hat{y})(\nabla \log p_\theta(\hat{y}))^\top\}$$

with $\nabla \log p_\theta(\hat{y})$ represents the $r \times 1$ gradient vector of $\log p_\theta(\hat{y})$ with respect to θ. Stated more formally, we have the following convergence in distribution as $n \to \infty$ (Ferguson, 1996)

$$\sqrt{n}\,(\hat{\theta}_n^{\text{mle}} - \theta_0) \rightsquigarrow N(0, J^{-1}(\theta^{\text{true}})). \tag{23}$$

It is instructive to consider the dependency of the Fisher information matrix, which corresponds to the asymptotic estimation accuracy, on $n, k, p(y), \theta^{\text{true}}$.

In the case of classification considering (8) with $k = 1$ and $\mathcal{Y} = \{1, 2\}$ it can be shown that

$$J(\theta) = \frac{\alpha(2\alpha-1)^2}{(\theta(2\alpha-1)-\alpha+1)^2} - \frac{(2\alpha-1)^2(\alpha-1)}{(\alpha-\theta(2\alpha-1))^2} \tag{24}$$

where $\alpha = P(y = 1)$. As Figure 3 (right) demonstrates, the asymptotic accuracy of the MLE (as indicated by J) tends to increase with the degree of non-uniformity of $p(y)$. Recall that since identifiability fails for a uniform $p(y)$ the risk estimate under a uniform $p(y)$ is not consistent. The above derivation (24) is a quantification of that fact reflecting the added difficulty in estimating the risk as we move closer to a uniform label distribution $\alpha \to 1/2$. The dependency of the asymptotic accuracy on θ^{true} is more complex, tending to favor θ^{true} values close to 1 or 0.5. Figure 3 (left) displays the empirical accuracy of the estimator as a function of $p(y)$ and θ^{true} and shows remarkable similarity to the contours of the Fisher information (see Section 7 for more details on the experiments). In particular, whenever the estimation error is high the asymptotic variance of the estimator is high (or equivalently, the Fisher information is low). For instance, the top contours in the left panel have smaller estimation error on the top right than in the top left. Similarly, the top contours in the right panel have smaller asymptotic variance on the top right than on the top left. We thus conclude that the Fisher information provides practical, as well as theoretical insight into the estimation accuracy.

Similar calculations of $J(\theta^{\text{true}})$ for collaborative classification case or for the regression case result in more complicated but straightforward derivations. It is important to realize that consistency is ensured for any identifiable $\theta^{\text{true}}, p(y)$. The value $(J(\theta^{\text{true}}))^{-1}$ is the constant dominating that consistency convergence.

A similar distributional analysis can be derived for the risk estimator. Applying Cramer's theorem (Ferguson, 1996) to $\hat{R}(f_j) = g_j(\hat{\theta}^{\text{mle}})$, $j = 1, \ldots, k$ and (23) we have

$$\sqrt{n}(\hat{R}(f) - R(f)) \rightsquigarrow N\left(0, \nabla g(\theta^{\text{true}}) J(\theta^{\text{true}}) \nabla g(\theta^{\text{true}})^\top\right)$$

where $R(f), \hat{R}(f)$ are the vectors of true risk and risk estimates for the different predictors $f_1, \ldots, f_k$ and $\nabla g(\theta^{\text{true}})$ is the Jacobian matrix of the mapping $g = (g_1, \ldots, g_k)$ evaluated at θ^{true}.

For example, in the case of classification with $k = 1$ we have $R(f_j) = 1 - \theta_j$ and the Jacobian matrix is -1, leading to an identical asymptotic distribution to that of the MLE (23)-(24)

$$\sqrt{n}(\hat{R}(f) - R(f)) \rightsquigarrow N\left(0, \left(\frac{\alpha(2\alpha-1)^2}{(\theta(2\alpha-1)-\alpha+1)^2} - \frac{(2\alpha-1)^2(\alpha-1)}{(\alpha-\theta(2\alpha-1))^2}\right)^{-1}\right).$$

6. Optimization Algorithms

Recall that we obtained closed forms for the likelihood maximizers in the cases of non-collaborative estimation for binary classifiers and non-collaborative estimation for one dimensional regression models. The lack of closed form maximizers in the other cases necessitates iterative optimization techniques.

One class of technique for optimizing nonlinear loglikelihoods is the class of gradient based methods such as gradient descent, conjugate gradients, and quasi Newton methods. These techniques proceed iteratively following a search direction; they often have good performance and are easy to derive. The main difficulty with their implementation is the derivation of the loglikelihood and its derivatives. For example, in the case of collaborative estimation of classification ($l \geq 2$) with symmetric noise model and missing values the loglikelihood gradient is

$$\frac{\partial \ell}{\partial \theta_j} = \sum_{i=1}^{n} \frac{\sum_{y^{(i)}} p(y^{(i)}) \sum_{r:\beta_{ri}=0} \sum_{\hat{y}_r^{(i)}} \prod_{p \neq j} h_{pi} (I(\hat{y}_j^{(i)} = y^{(i)}) - \theta_j)((l-1)\theta_j)^{I(\hat{y}_j^{(i)}=y^{(i)})-1}(1-\theta_j)^{-I(\hat{y}_j^{(i)}=y^{(i)})}}{\sum_{y^{(i)}} p(y^{(i)}) \sum_{r:\beta_{ri}=0} \sum_{\hat{y}_r^{(i)}} \prod_{p=1}^{k} h_{pi}},$$

$$h_{pi} = \theta_p^{I(\hat{y}_p^{(i)}=y^{(i)})} \left(\frac{1-\theta_p}{l-1}\right)^{I(\hat{y}_p^{(i)} \neq y^{(i)})}$$

Similar derivations may be obtained in the other cases in a straightforward manner.

An alternative iterative optimization technique for finding the MLE is expectation maximization (EM). The derivation of the EM update equations is again relatively straightforward. For example in the above case of collaborative estimation of classification ($l \geq 2$) with symmetric noise model and missing values the EM update equations are

$$\begin{aligned}
\theta^{(t+1)} &= \arg\max_{\theta} \sum_{i=1}^{n} \sum_{y^{(i)}} \sum_{r:\beta_{ri}=0} \sum_{\hat{y}_r^{(i)}} q^{(t)}(\hat{y}_r^{(i)}, y^{(i)}) \sum_{j=1}^{k} \log p_j(\hat{y}_j^{(i)} | y^{(i)}) \\
&= \frac{1}{n} \sum_{i=1}^{n} \sum_{y^{(i)}} \sum_{r:\beta_{ri}=0} \sum_{\hat{y}_r^{(i)}} q^{(t)}(\hat{y}_r^{(i)}, y^{(i)}) I(\hat{y}_j^{(i)} = y^{(i)}), \\
q^{(t)}(\hat{y}_r^{(i)}, y^{(i)}) &= \frac{p(y^{(i)}) \prod_{j=1}^{k} p_j(\hat{y}_j^{(i)} | y^{(i)}, \theta^{(t)})}{\sum_{y^{(i)}} \sum_{r:\beta_{ri}=0} \sum_{\hat{y}_r^{(i)}} p(y^{(i)}) \prod_{j=1}^{k} p_j(\hat{y}_j^{(i)} | y^{(i)}, \theta^{(t)})}.
\end{aligned}$$

where $q^{(t)}$ is the conditional distribution defining the EM bound over the loglikelihood function.

If all the classifiers are always observed, that is, $\beta_{ri} = 1 \ \forall r, i$ Equation (16) reverts to (12), and the loglikelihood and its gradient may be efficiently computed in $O(nlk^2)$. In the case of missing classifier outputs a naive computation of the gradient or EM step is exponential in the number of missing values $R = \max_i \sum_r \beta_{ri}$. This, however, can be improved by careful dynamic programming. For example, the nested summations over the unobserved values in the gradient may be computed using a variation of the elimination algorithm in $O(nlk^2R)$ time.

7. Empirical Evaluation

We start with some experiments demonstrating our framework using synthetic data. These experiments are meant to examine the behavior of the estimators in a controlled setting. We then describe

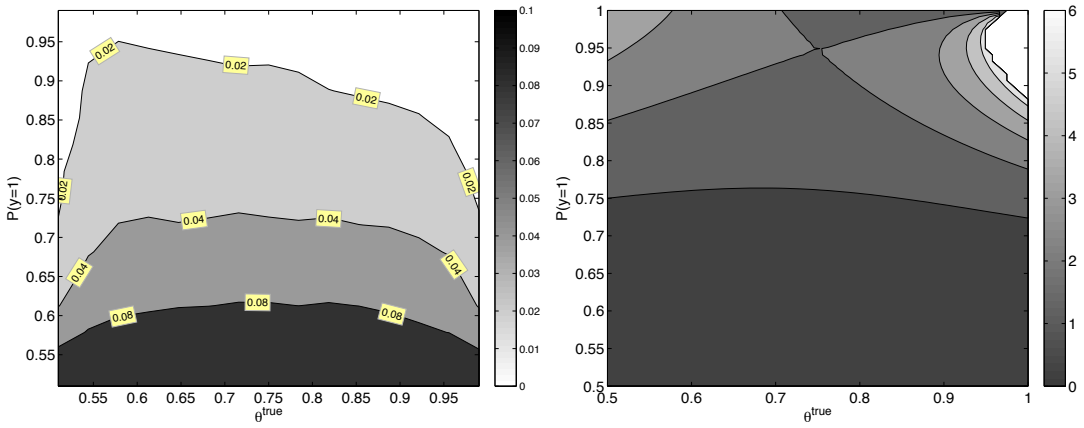


Figure 3: Left: Average value of $|\hat{\theta}_n^{\text{mle}} - \theta^{\text{true}}|$ as a function of θ^{true} and $p(y=1)$ for $k=1$ classifier and $n=500$ (computed over a uniform spaced grid of 15×15 points). The plot illustrates the increased accuracy obtained by a less uniform $P(y)$. Right: Fisher information $J(\theta)$ for $k=1$ as a function of θ^{true} and $P(y)$. The asymptotic variance of the estimator is $J^{-1}(\theta)$ which closely matches the experimental result in the left panel.

some experiments using several real world data sets. In these experiments we examine the behavior of the estimators in an uncontrolled setting where some of the underlying assumptions may be violated. In most of the experiments we consider the mean absolute error (mae) or the ℓ_1 error as a metric that measures the estimation quality

$$\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}}) = \frac{1}{k} \sum_{i=1}^{k} \left| \theta_i^{\text{true}} - \hat{\theta}_i^{\text{mle}} \right|.$$

In the non-collaborative case (which is equivalent to the collaborative case with $k=1$) this translates into the absolute deviation of the estimated parameter from the true parameter.

In Figure 3 (left) we display $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ for classification with $k=1$ as a function of θ^{true} and $p(y)$ for $n=500$ simulated data points. The estimation error, while overall relatively small, decays as $p(y)$ diverges from the uniform distribution. The dependency on θ^{true} indicates that the error is worst for θ^{true} around 0.75 and it decays as $|\theta^{\text{true}} - 0.75|$ increases with a larger decay attributed to higher θ^{true}. These observations are remarkably consistent with the developed theory as Figure 3 (right) shows by demonstrating the value of the inverse asymptotic variance $J(\theta)$ which agrees nicely with the empirical measurement in the left panel.

Figure 4 (left) contains a scatter plot contrasting values of θ^{true} and $\hat{\theta}^{\text{mle}}$ for $k=1$ classifier and $p(y=1)=0.8$. The estimator was constructed based on 500 simulated data points. We observe a symmetric Gaussian-like distribution of estimated values $\hat{\theta}^{\text{mle}}$, conditioned on specific values of θ^{true}. This is in agreement with the theory predicting an asymptotic Gaussian distribution for the mle, centered around the true value θ^{true}. A similar observation is made in Figure 5 (left) which contains a similar scatter plot in the regression case ($k=1$, $\sigma_y=1$, $n=1000$). In both figures, the striped effect is due to selection of θ^{true} over a discrete grid with a small perturbation for increased visibility. Similar plots of larger and smaller n values (not shown) verify that the variation of $\hat{\theta}^{\text{mle}}$

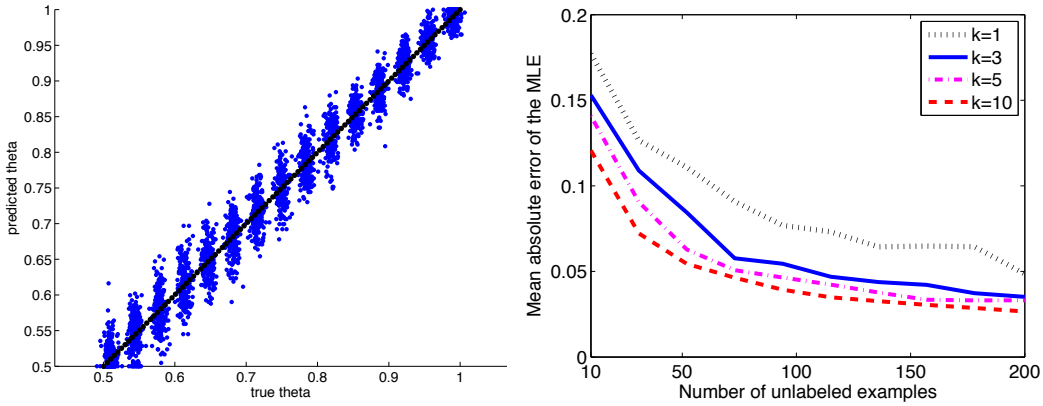


Figure 4: Left: Scatter plot contrasting the true and predicted values of θ in the case of a single classifier $k = 1$, $p(y = 1) = 0.8$, and $n = 500$ unlabeled examples. The displayed points were perturbed for improved visualization and the striped effect is due to empirical evaluation over a discrete grid of θ^{true} values. Right: $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ as a function of the number of unlabeled examples for different number of classifiers ($\theta_i^{\text{true}} = p(y = 1) = 0.75$) in the collaborative case. The estimation error decreases as more classifiers are used due to the collaborative nature of the estimation process.

around θ^{true} decreases as n increases. This agrees with the theory that indicates a $O(n^{-1})$ rate of decay for the variance of the asymptotic distribution.

Figures 4 and 5 (right) show the $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ for various k values in classification and regression, respectively. In classification, $\hat{\theta}^{\text{mle}}$ was obtained by sampling data from $p(y = 1) = 0.75 = \theta_i^{\text{true}}, \forall i$. In regression, the data was sampled from the regression equation with $\theta_i^{\text{true}} = 1$ and $p(y) = N(0,1)$. In both cases, the mae error decays with n as expected from the consistency proof and with k as a result of the collaborative estimation effect.

To further illustrate the effect of the collaboration on the estimation accuracy, we estimated the error rates individually (non-collaboratively) for 10 predictors and compared their mae to that of the collaborative estimation case in Figure 6. This shows that each of the classifiers have a similar mae curve when non-collaborative estimation is used. However, all of these curves are higher than the collaborative mae curve (solid black line in Figure 6) demonstrating the improvement of the collaborative process.

We compare in Figure 7 the proposed unsupervised estimation framework with supervised estimation that takes advantage of labeled information to determine the classifier accuracy. We conducted this study using equal number of examples for both supervised and unsupervised cases. Clearly, this is an unfair comparison if we assume that labeled data is unavailable or is difficult to obtain. The unsupervised estimation does not perform as well as the supervised version especially in general. Nevertheless, the unsupervised estimation accuracy improves significantly with increasing number of classifiers and finally reaches the performance level of the supervised case due to collaborative estimation.

In Figure 8 we report the effect of misspecification of the marginal $p(y)$ on the estimation accuracy. More specifically, we generated synthetic data using a true marginal distribution but

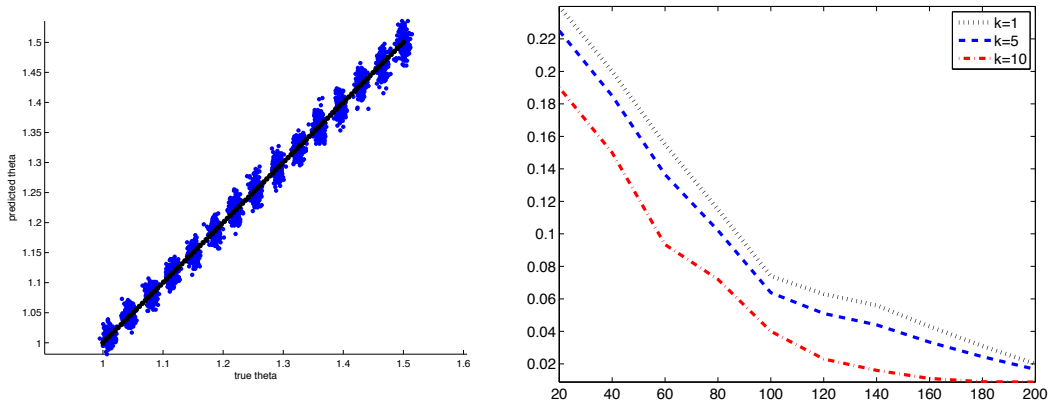


Figure 5: Left: Scatter plot contrasting the true and predicted values of θ in the case of a single regression model $k = 1$, $\sigma_y = 1$, and $n = 1000$ unlabeled examples. The displayed points were perturbed for improved visualization and the striped effect is due to empirical evaluation over a discrete grid of θ^{true} values. Right: $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ as a function of the number of unlabeled examples for different number of regression models ($\theta_i^{\text{true}} = \sigma_y = 1$) in the collaborative case. The estimation error decreases as more regression models are used due to the collaborative nature of the estimation process.

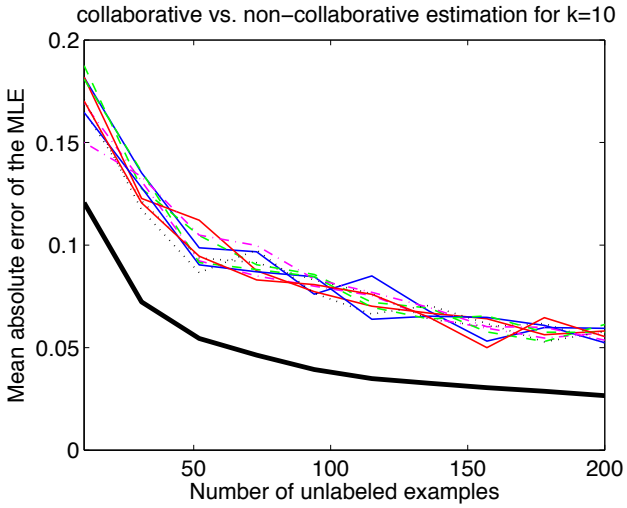


Figure 6: Comparison of collaborative and non-collaborative estimation for $k = 10$ classifiers. $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ as a function of n is reported for $\theta_i^{\text{true}} = 0.75\ \forall k_i$ and $P(y = 1) = 0.75$. The colored lines represent the estimation error for each individual classifier and the solid black line represents the collaborative estimation for all classifiers. The estimation converges to the truth faster in the collaborative case than in the non-collaborative case.

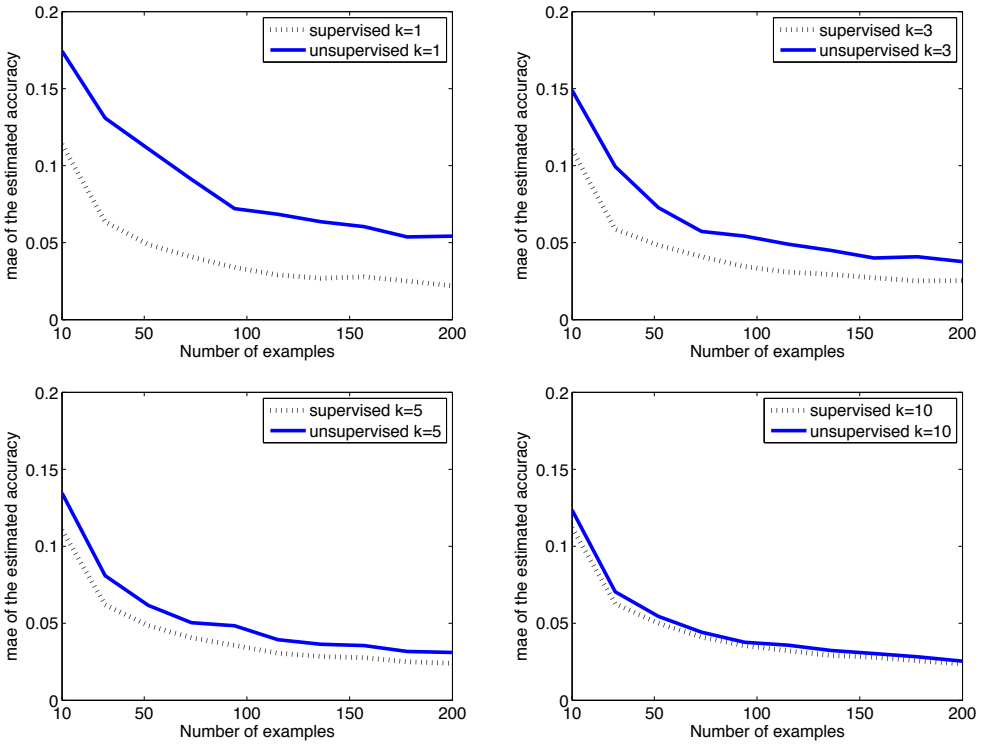


Figure 7: Comparison of supervised and unsupervised estimation for different values of classifiers with $k = 1,3,5,10$. Supervised estimation uses the true labels to determine the accuracy of the classifiers whereas in the unsupervised case the estimation proceeds according to the collaborative estimation framework. Despite the fact that the supervised case uses labels the unsupervised framework reaches similar levels by increasing the number of classifiers.

estimated the classifier accuracy on this data assuming a misspecified marginal. Generally, the estimation framework is robust to small perturbations while over-specifying tends to hurt less than under-specifying (misspecification closer to uniform distribution).

Figure 9 shows the mean prediction accuracy for the unsupervised predictor combination scheme in (4) for synthetic data. The left panel displays classification accuracy and the right panel displays the regression accuracy as measured by $1 - \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i^{\text{new}} - y_i^{\text{new}})^2$. The graphs show that in both cases the accuracy increases with k and n in accordance with the theory and the risk estimation experiments. The parameter θ_i^{true} was chosen uniformly in the range $(0.5,1)$, and $P(y=1) = 0.75$ for classification and $\theta_i^{\text{true}} = 0.3$, $p(y) = N(0,1)$ in the case of regression.

We also experimented with the natural language understanding data set introduced in Snow et al. (2008). This data was created using the Amazon Mechanical Turk (AMT) for data annotation. AMT is an online tool that uses paid employees to complete small labeling and annotation tasks. We selected two binary tasks from this data: the textual entailment recognition (RTE) and temporal event recognition (TEMP) tasks. In the former task, the annotator is presented with two sentences for each question. He needs to decide whether the second sentence can be inferred from the first.

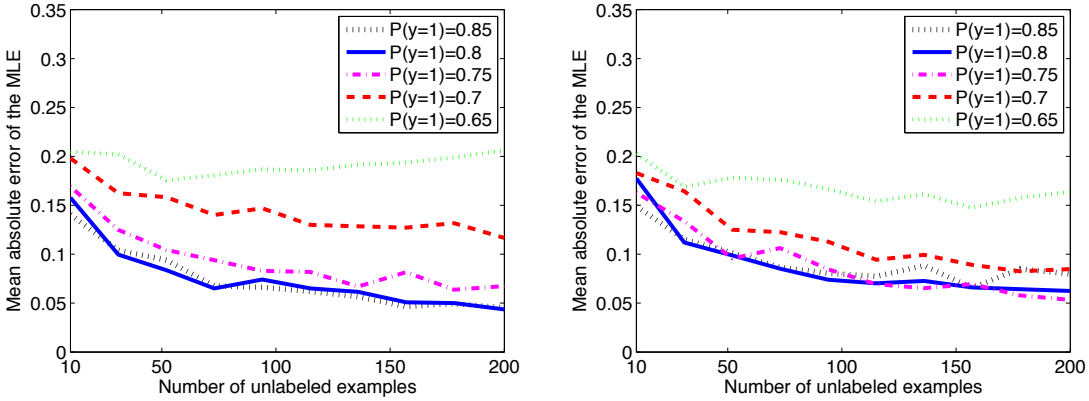


Figure 8: The figure compares the estimator accuracy assuming that the marginal $p(y)$ is misspecified. The plots draw $\mathrm{mae}(\hat{\theta}^{\mathrm{mle}}, \theta^{\mathrm{true}})$ as a function of n for $k = 1$ and $\theta^{\mathrm{true}} = 0.75$ when $P^{\mathrm{true}}(y = 1) = 0.8$ (left) and $P^{\mathrm{true}}(y = 1) = 0.75$ (right). Small perturbations in $P^{\mathrm{true}}(y)$ do not affect the results significantly; interestingly over-specifying $P^{\mathrm{true}}(y = 1)$ leads to more accurate estimates than under-specifying (misspecification closer to uniform distribution)

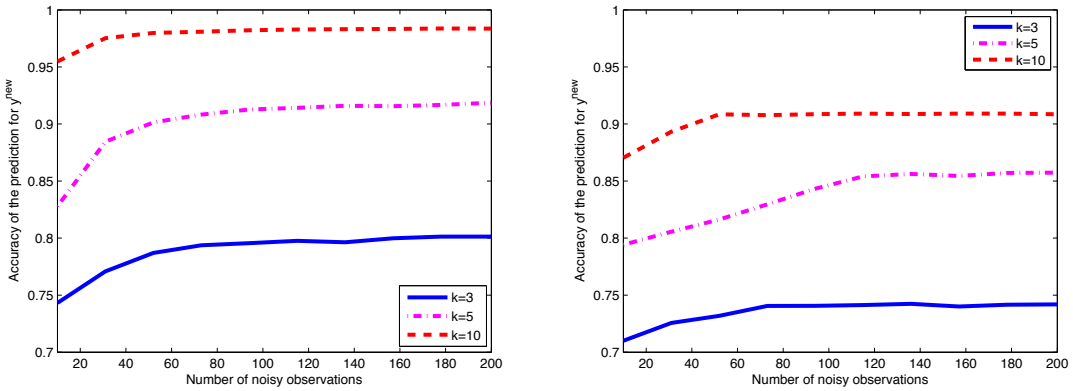


Figure 9: Mean prediction accuracy for the unsupervised predictor combination scheme in (4) for synthetic data. The left panel displays classification accuracy and the right panel displays the regression accuracy as measured by $1 - \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i^{\mathrm{new}} - y_i^{\mathrm{new}})^2$. The graphs show that in both cases the accuracy increases with k and n in accordance with the theory and the risk estimation experiments.

The original data set contains 800 sentence pairs with a total of 165 annotators. The latter task involves recognizing the temporal relation in verb-event pairs. The annotator is forced to decide whether the event described by the first verb occurs before or after the second. The original data set contains 462 pairs and 76 annotators. In both data sets, most of the annotators have completed only a handful of tasks. Therefore, we selected a subset of these annotators for each task such that each annotator has completed at least 100 problems and has differing accuracies. The data sets contain ground truth labels which are used solely to calculate the annotator accuracy and not used

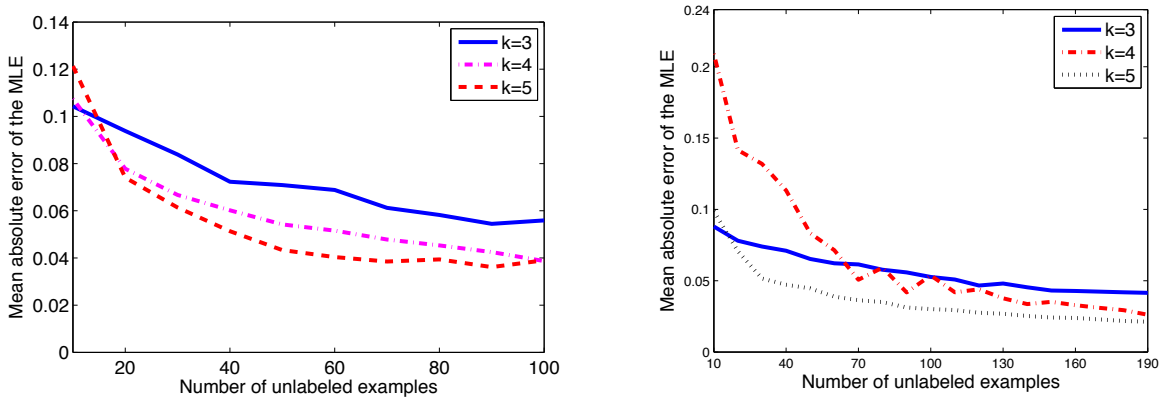


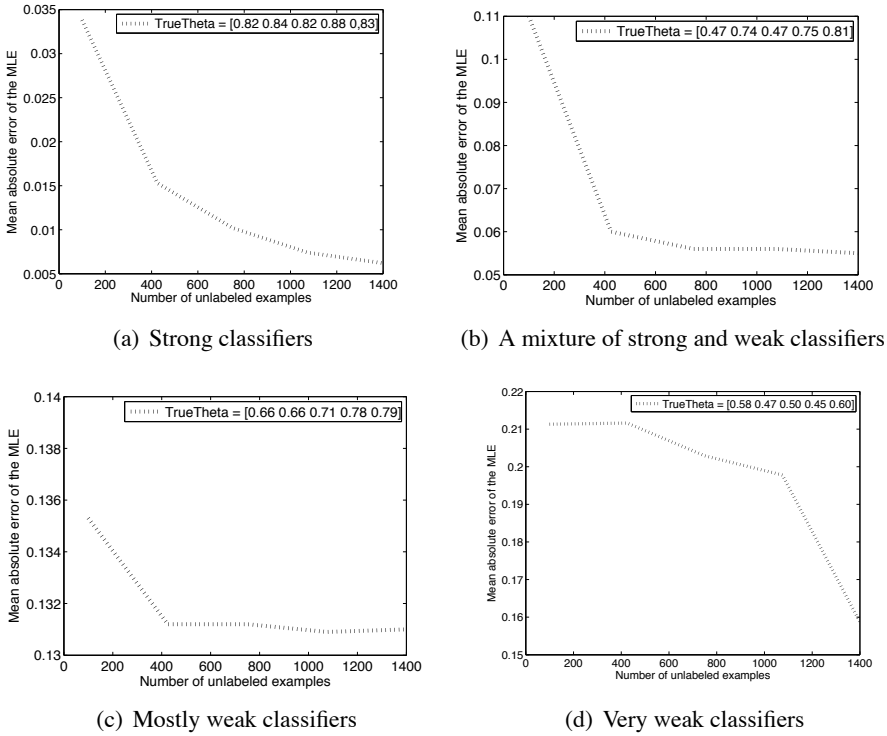
Figure 10: $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ as a function of n for different number of annotators k on RTE (left) and TEMP (right) data sets. Left: $n = 100$, $P(y = 1) = 0.5$ and $\theta^{\text{true}} = \{0.85, 0.92, 0.58, 0.5, 0.51\}$. Right: $n = 190$, $P(y = 1) = 0.56$ and $\theta^{\text{true}} = \{0.93, 0.92, 0.54, 0.44, 0.92\}$. The classifiers were added in the order specified.

at all during the estimation process. For efficiency, we selected only the instances for which all annotators provide an answer. This resulted in $n = 100, 190$ for RTE and TEMP, respectively.

In Figure 10 we display $\text{mae}(\theta^{\text{true}}, \hat{\theta}^{\text{mle}})$ for these data sets as function of n for different values of k. These plots generated from real-world data show similar trend to the synthetic experiments. The estimation errors decay to 0 as n increases and generally tend to decrease as k increases. This correspondence is remarkable since two of the labelers have worse than random accuracy and since it is not clear whether the conditional independence assumption actually holds in reality for these data sets. Nevertheless, the collaborative estimation error behaves in accordance with the synthetic data experiments and the theory. This shows that the estimation framework is robust to the breakdown of the assumption that the classifier accuracy must be higher than random choice. Also, whether the conditional independence assumption holds or not is not crucial in this case.

We further experimented with classifiers trained on different representations of the same data set and estimated their error rates. We adopted the Ringnorm data set generated by Breiman (1996). Ringnorm is a 2-class artificial data set with 20 dimensions where each class is drawn from a multivariate normal distribution. One class has zero mean and a covariance $\Sigma = 4I$ where I is the identity matrix. The other class has unit covariance and a mean $\mu = (\frac{2}{\sqrt{20}}, \frac{2}{\sqrt{20}}, \ldots, \frac{2}{\sqrt{20}})$. The total size is 7400. We created 5 different representations of the data by projecting it onto mutually exclusive sets of principal components obtained by Principal Component Analysis (PCA). We trained an SVM classifier (with 2-degree polynomial kernel) (Vapnik, 2000; Joachims, 1999) on samples from each representation while holding out 1400 examples as the test set resulting in a total of 5 classifiers. We tested each of the 5 classifiers on the test set and used their outputs to estimate the corresponding parameters. The true labels of the test set examples were used as ground truth to calculate the mae of the mle estimators.

The mae curves for this data set appear in Figure 11 as a function of the number n of unlabeled examples. When all classifiers are highly accurate (upper left panel), the collaborative unsupervised estimator is reliable, see Figure 11(a). With a mixture of weak and strong classifiers (upper right panel), the collaborative unsupervised estimator is also reliable. This is despite the fact that some of



(a) Strong classifiers

(b) A mixture of strong and weak classifiers

(c) Mostly weak classifiers

(d) Very weak classifiers

Figure 11: $\text{mae}(\theta^{\text{true}}, \hat{\theta}^{\text{mle}})$ as a function of the test set size on the Ringnorm data set. $p(y=1) = 0.47$, and θ^{true} is indicated in the legend in each plot. The four panels represent mostly strong classifiers (upper left), a mixture of strong and weak classifiers (upper right), mostly weak classifiers (bottom left), and mostly very weak classifiers (bottom right). The figure shows that the framework is robust to occasional deviations from the assumption regarding better than random guess classification accuracy (upper right panel). However, as most of the classifiers become weak or very weak, the collaborative unsupervised estimation framework results in worse estimation error.

the weak classifiers in Figure 11(b) have worse than random accuracy which violates the assumptions in the consistency proposition. This shows again that the estimation framework is robust to occasional deviations from the requirement concerning better than random classification accuracies. On the other hand, as most of the classifiers become worse (bottom row), the accuracy of the unsupervised estimator decreases, in accordance with the theory developed in Sections 5 (recall the Fisher information contour plot).

Our experiments thus far assumed the symmetric noise model (8). Despite it not being always applicable for real world data and classifiers, it did result in good estimation accuracy in some of the cases described thus far. However, in some cases this assumption is grossly violated and the more general noise model is needed (6). For this reason, we conducted two experiments using real world data assuming the more general (6).

The first experiment concerned domain adaptation (Blitzer et al., 2007) for Amazon's product reviews in four different product domains: books, DVDs, electronics and kitchen appliances. Each

	book	dvd	kitchen	electronics	20newsgroup
training error	0.22	0.23	0.26	0.30	0.028
non-collaborative	**0.04**	**0.04**	**0.08**	**0.06**	**0.006**
collaborative	0.10	0.10	0.09	0.08	n/a

Figure 12: $\text{mae}(\hat{\theta}^{\text{mle}}, \theta^{\text{true}})$ for the domain adaptation ($n = 1000$, $p(y = 1) = 0.75$) and 20 newsgroup ($n = 15{,}000$, $p(y = 1) = 0.05$ for each one-vs-all data). The unsupervised non-collaborative estimator outperforms the collaborative estimator due to violation of the conditional independence assumption. Both unsupervised estimators perform substantially better than the baseline training error rate estimator. In both cases the results were averaged over 50 random train test splits.

domain consists of positive ($y = 1$) and negative ($y = 2$) reviews with $p(y = 1) = 0.75$. The task was to estimate the error rates of classifiers (linear SVM, Vapnik, 2000; Joachims, 1999) that are trained on 300 examples from one domain but tested on other domains. The mae values for the classification risks are displayed in Figure 12 with the columns indicating the test domain. In this case, the unsupervised non-collaborative estimator outperforms the collaborative estimator due to violation of the conditional independence assumption. Both unsupervised estimators perform substantially better than the baseline estimator that uses the training error on one domain to predict testing error on another domain.

In the second experiment using (6) we estimated the risk (non-collaboratively) of 20 one vs. all classifiers (trained to predict one class) on the 20 newsgroup data (Lang, 1995). The train set size was 1000 and the unlabeled data size was 15000. In this case the unsupervised non-collaborative estimator returned extremely accurate risk estimators. As a comparison, the risk estimates obtained from the training error are four times larger than the unsupervised MLE estimator (See Figure 12).

8. Discussion

We have demonstrated a collaborative framework for the estimation of classification and regression error rates for $k \geq 1$ predictors. In contrast to previous supervised risk estimation methods such as cross validation (Duda et al., 2001), bootstrap (Efron and Tibshirani, 1997), and others (Hand, 1986), our approach is fully unsupervised and thus able to use vast collections of unlabeled data. Other related work includes Smyth et al. (1995) and Sheng et al. (2008) which consider repeated labeling where each instance is labeled by multiple experts and the final label is decided based on a majority voting scheme. However, Smyth et al. and Sheng et al. fail to address estimating the risks of the predictors which is the main focus of our work.

We prove statistical consistency in the unsupervised case and derive the asymptotic variance. Our experiments on synthetic data demonstrate the effectiveness of the framework and verify the theoretical results. Experiments on real world data show robustness to underlying assumptions. The framework may be applied to estimate additional quantities in an unsupervised manner, including noise level in noisy communication channels (Cover and Thomas, 2005) and error rates in structured prediction problems.

Acknowledgments

The authors thank Jian Zhang and the reviewers for helpful discussions and suggestions. K. Balasubramanian and G. Lebanon were partially supported by NSF grant IIS-0906643.

References

Y. Bishop, S. Fienberg, and P. Holland. *Discrete Multivariate Analysis: Theory and Practice*. MIT press, 1975.

J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proc. of ACL '07*, 2007.

L. Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Statistics department, University of California, 1996.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, second edition, 2005.

D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2006.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley New York, 2001.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1997.

T. S. Ferguson. *A Course in Large Sample Theory*. Chapman & Hall, 1996.

D. J. Hand. Recent advances in error rate estimation. *Pattern Recognition Letters*, 4(5):335–346, 1986.

T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

K. Lang. Newsweeder: Learning to filter netnews. In *International Conference on Machine Learning*, 1995.

A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1984.

V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proc. of the 14th ACM SIGKDD Internation 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 7*, 1995.

R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast-but is it good? evaluating non-expert annotations for natural language tasks. In *Proc. of EMNLP*, 2008.

B. Sturmfels. *Solving Systems of Polynomial Equations*. American Mathematical Society, 2002.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, second edition, 2000.

Journal of Machine Learning Research 11 (2010) 1353-1390 Submitted 3/09; Revised 12/09; Published 4/10

Learning Translation Invariant Kernels for Classification

Kamaledin Ghiasi-Shirazi GHIASI@AUT.AC.IR
Reza Safabakhsh SAFA@AUT.AC.IR
Computer Engineering Department
Amirkabir University of Technology
Tehran, 15914, Iran

Mostafa Shamsi M_SHAMSI@AUT.AC.IR
Faculty of Mathematics and Computer Science
Amirkabir University of Technology
Tehran, 15914, Iran

Editor: John Shawe-Taylor

Abstract

Appropriate selection of the kernel function, which implicitly defines the feature space of an algorithm, has a crucial role in the success of kernel methods. In this paper, we consider the problem of optimizing a kernel function over the class of translation invariant kernels for the task of binary classification. The learning capacity of this class is invariant with respect to rotation and scaling of the features and it encompasses the set of radial kernels. We show that how translation invariant kernel functions can be embedded in a nested set of sub-classes and consider the kernel learning problem over one of these sub-classes. This allows the choice of an appropriate sub-class based on the problem at hand. We use the criterion proposed by Lanckriet et al. (2004) to obtain a functional formulation for the problem. It will be proven that the optimal kernel is a finite mixture of cosine functions. The kernel learning problem is then formulated as a semi-infinite programming (SIP) problem which is solved by a sequence of quadratically constrained quadratic programming (QCQP) sub-problems. Using the fact that the cosine kernel is of rank two, we propose a formulation of a QCQP sub-problem which does not require the kernel matrices to be loaded into memory, making the method applicable to large-scale problems. We also address the issue of including other classes of kernels, such as individual kernels and isotropic Gaussian kernels, in the learning process. Another interesting feature of the proposed method is that the optimal classifier has an expansion in terms of the number of cosine kernels, instead of support vectors, leading to a remarkable speedup at run-time. As a by-product, we also generalize the kernel trick to complex-valued kernel functions. Our experiments on artificial and real-world benchmark data sets, including the USPS and the MNIST digit recognition data sets, show the usefulness of the proposed method.

Keywords: kernel learning, translation invariant kernels, capacity control, support vector machines, classification, semi-infinite programming

1. Introduction

Kernel-based methods, such as support vector machines (SVM) and kernel principal component analysis (KPCA), increase the flexibility of machine learning algorithms by implicitly mapping the input data into a feature space and performing the algorithm in that space. This flexibility is achieved by a so called kernel function which substitutes the dot-product operation in an ordinary algorithm. The kernel function, by implicitly defining the feature space, plays a crucial role in the

success of kernel methods. In fact, as shown by Xiong et al. (2005), if the kernel function is not chosen appropriately, it may even worsen the performance of an algorithm. This significant impact on the performance of the kernel-based algorithms and the fact that the appropriate feature space is problem-dependent, have driven researchers to devise various algorithms to learn the kernel function from the problem data.

The earliest method for learning a kernel function is cross-validation which is very slow and is only applicable to kernels with a small number of parameters. Cristianini et al. (1998) proposed an algorithm for adapting kernel functions with only one unconstrained parameter. Instead of optimizing the parameters of a kernel, Amari and Wu (1999) suggested conformal transformation of the kernel function and proposed an algorithm for learning the parameters of the new kernel. Chapelle et al. (2002) devised a gradient-based algorithm for local optimization of a kernel with multiple unconstrained parameters. Glasmachers and Igel (2005) proposed a gradient-based method for learning the covariance matrix of Gaussian kernels (Note that since the covariance matrix of a Gaussian kernel is constrained to be positive semi-definite, the method of Chapelle et al. (2002) cannot be used for learning this matrix). Ong et al. (2005) introduced the notion of hyperkernels and used it for kernel learning. They formulated the kernel learning problem as a functional with three terms: an empirical quality functional, a regularization term that penalizes the functions in a reproducing kernel Hilbert space (RKHS), and another regularization term that penalizes the kernels in a hyper reproducing kernel Hilbert space.

A milestone in the kernel learning literature is the introduction of the multiple kernel learning (MKL) framework by Lanckriet et al. (2004). They considered the problem of finding the optimal convex combination of multiple kernels and formulated it as a quadratically constrained quadratic programming (QCQP) problem. They also introduced a generalized performance measure which encompasses the hard-margin, 1-norm soft-margin, and 2-norm soft-margin performance measures as special cases. Although these performance measures have extensively been used for learning the optimal separating hyperplane in SVMs, their use as performance measures for kernel selection was unprecedented. Since the formulation of the resulting QCQP requires storing several kernel matrices in memory, their method was only applicable to problems with a small number of training samples. Bach et al. (2004) introduced an SMO-based algorithm to widen the range of solvable MKL problems by using the Moreau-Yosida regularization technique. Sonnenburg et al. (2005, 2006) reformulated the MKL problem as a semi-infinite linear program (SILP) which was then reduced to training a sequence of classical SVMs with a single kernel for which several sophisticated large-scale algorithms exist. Rakotomamonjy et al. (2008) argued that the main difficulty with the SILP formulation of Sonnenburg et al. (2006) is that its objective function is non-smooth and introduced an equivalent convex formulation with a smooth objective function. Using convexity of the problem and the smoothness of the objective function, they proposed a reduced gradient algorithm for MKL which is also applicable to large-scale problems. The weakness of the reduced gradient algorithm is that, in contrast to to the SILP algorithm, it does not use the information collected in the previous points in the calculation of the next point. Combining the strengths of the SILP method of Sonnenburg et al. (2006) with those of the reduced gradient method of Rakotomamonjy et al. (2008), Xu et al. (2008) proposed an extended level method which is remarkably faster than both methods.

In their seminal work, Micchelli and Pontil (2005) generalized the class of admissible kernels to convex combination of an infinite number of kernels indexed by a compact set and applied their method to the problem of learning radial kernels (Argyriou et al., 2005, 2006). They used a classical

result proved by Schoenberg (1938) which states that every continuous radial kernel belongs to the convex hull of radial Gaussian kernels. They also proposed an efficient DC programming algorithm for numerically learning radial kernels in Argyriou et al. (2006). Gehler and Nowozin (2008) reformulated the optimization problem of Argyriou et al. (2006) as a semi-infinite programming problem and proposed the IKL (infinite kernel learning) framework for solving it numerically.

In this work, we consider the class of translation invariant kernel functions which encompasses the class of radial kernels as well as the class of anisotropic Gaussian kernel functions. This class contains exactly those kernels which can be defined solely based on the difference of kernel arguments; that is, the kernel functions with the property:

$$k(x,z) = \tilde{k}(x-z).$$

The general form of continuous translation invariant kernels on $\mathbb{R}^n$ was discovered by Bochner (1933). He proved that every function of the form[1]

$$k(x,z) = \tilde{k}(x-z) = \int_{\mathbb{R}^n} e^{j\gamma^T(x-z)}\, dV(\gamma) \tag{1}$$

is positive semi-definite, where V(.) is a monotonically increasing bounded function and the integration is in the Lebesgue-Stieltjes sense. He also proved that, conversely, every continuous translation invariant positive semi-definite kernel function can be represented in the above form. In statistics, the translation invariance property is referred to as the stationarity of the kernel function. Genton (2001) and Schölkopf and Smola (2002) give a list of the properties of this class along with important examples of stationary kernel functions, including the Gaussian, exponential, rational quadratic, and B_n spline kernels.

The rest of the paper proceeds as follows: Table 1 lists the choice of notations for familiar concepts in the field. Notations specific to this paper will be introduced in the course of discussions. Although the kernel learning formulation of Micchelli and Pontil (2005) contains a regularization term for controlling the complexity of the RKHS associated with the kernel function, there is no mechanism for controlling the capacity of the class of admissible kernels. In our formulation, we have provisioned a mechanism for controlling the complexity of the class of admissible kernels which is described in Section 2. The idea is to multiply a vanishing function inside the integral of Equation (1). In addition to controlling the capacity of the learning machine, this choice substitutes the compactness assumption of the integration region made by Micchelli and Pontil (2005). In Section 3, we propose a learning criterion which is essentially a reformulation of the generalized performance measure of Lanckriet et al. (2004). The proposed criterion ensures the compactness of the parameter space of SVM, and gives a probabilistic meaning to the regularization parameter of the 2-norm soft-margin SVM. The problem of finding an optimal kernel which minimizes this criterion over the class of translation invariant kernels leads to (4) which is our main variational problem.

In Section 4, we prove some important theorems which pave the way for an algorithmic solution to this problem. First, in Section 4.1 we prove the existence of an optimal solution for problem (4).

1. In this paper we will represent translation invariant kernels both as $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{C}$ with two arguments and as $\tilde{k} : \mathbb{R}^d \to \mathbb{C}$ with only one argument.

In Section 4.2, we prove that the min and max operations in (4) can be interchanged, provided that the integration region is replaced by a compact set. In addition, it will be shown that the optimal kernel is a finite mixture of the basic kernels of the form $k_\gamma(x,z) = \exp(j\gamma^T(x-z))$. In Section 4.3, it will be proved that the integration region can indeed be replaced by a compact set. To solve problem (4) numerically, we introduce a semi-infinite programming (SIP) formulation in Section 4.4. In Section 4.5, using a topological argument, the issue of including other classes of kernels in the learning process will be addressed. It is well known that the regularization parameter of the 2-norm SVM, usually denoted by τ, can be regarded as the weight of a Kronecker delta kernel function, which is incidently a translation invariant kernel, as well. In Section 4.4 we introduce the semi-infinite programming problem (19) which is our main numerical optimization problem and corresponds to the simultaneous learning of the optimal translation invariant kernel as well as the parameter τ. As another application of the discussion of Section 4.5, in Section 4.7 we will introduce a method for learning the best combination of stabilized translation invariant kernels and isotropic Gaussian kernels.

In Section 5, we address the problem of numerically solving (19) on a computer. The proposed optimization algorithm is a variant of the class of local-reduction-based algorithms for solving SIP problems. An important feature of the proposed optimization algorithm is that it does not require loading the kernel matrices into memory and so it is applicable to large-scale problems. As stated above, it will be shown in Section 4.2 that the optimal kernel is complex-valued. Since algorithms are usually designed for real Euclidean spaces, this complicates the application of the kernel trick to the optimal kernel (consider for example an algorithm that checks the sign of a dot product). In Section 6, we show that the feature space induced by the real part of a complex-valued kernel is essentially equivalent to the original complex-valued kernel and deduce that the optimal real-valued kernel is a mixture of cosines. Yet another astounding feature of the proposed method is concerned with the evaluation time of the classifier which is even faster than a classical SVM with a single Gaussian kernel. Usually, multiple kernel learning methods yield a model whose evaluation time is in the order of the number of kernels times the number of support vectors. In Section 7, we show that the evaluation time of the optimal translation invariant kernel is proportional to the number of cosine kernels, regardless of the number of support vectors. In Section 8, using a learning theory discussion, we show the necessity of controlling the complexity of the class of translation invariant kernels.

In Section 9, we will assess the practical usefulness of the proposed method on several data sets. In Section 9.1, we first perform some experiments on 13 artificial and real-world data sets collected from the UCI, DELVE, and STATLOG benchmark repositories by Rätsch et al. (2001).In Section 9.2, we perform experiments on the USPS handwritten digit recognition data set, comparing the proposed method with the MKL of Chapelle et al. (2002). In Section 9.3, we compare the proposed method with the DC method of Argyriou et al. (2006) on the MNIST handwritten digit recognition data set. In Section 9.4, we experimentally assess the role of the capacity control mechanism of Section 2. Finally, we conclude the paper in Section 10.

2. A Hierarchy of Classes for Translation Invariant Kernels

It is well-known in learning theory that to have a small generalization error, there should be a problem-dependent compromise between the complexity of the learning machine and the empirical error on the training data (Vapnik, 1998; Cucker and Zhou, 2007). In the previous section we saw

Symbol	Meaning	Symbol	Meaning
n	input space dimension	l	number of training data
l_1	number of training data with label +1	l_2	number of training data with label -1
X	input space	F	Feature space
Φ	feature map: $\Phi : X \to F$	nsv	number of support vectors
α	lagrange multipliers in SVM	C	regularization parameter in SVM
k	the kernel function	Δ	maximal margin
C^+	the set of data with label +1	C^-	the set of data with label -1
m	number of kernels in MKL framework	μ	weight of kernels in MKL
R	radius of the smallest ball surrounding the data in the feature space	$\Re(z)$	real part of the complex number z
		j	unit imaginary number: $\sqrt{-1}$
$\bar{z}$	the complex conjugate of z	S^c	the complement of set S

Table 1: Notations

that Equation (1) captures the general form of the large class of translation invariant kernels. To obtain the best tradeoff between approximation and estimation errors, we introduce a hierarchy of classes of translation invariant kernels. The appropriate class for a specific problem is then chosen by cross-validation. It must be mentioned that the idea of controlling the complexity of the class of admissible kernels has been previously used by Ong et al. (2005) for learning the kernels with hyperkernels. Although this type of complexity control is not provisioned by Micchelli and Pontil (2005), we, based on our experiments, believe that it is an important ingredient of the framework. To restrict the class of translation invariant kernels, we propose to limit the high frequency components of the kernel function. This is similar to the use of stabilizer functions in regularization theory (see Girosi et al., 1993). But, instead of adding a stabilizer function to the objective functional, which requires the determination of both the stabilizer and the regularization parameter, we explicitly define a nested class of translation invariant kernels $\mathcal{K}_\beta$ as:

$$\mathcal{K}_\beta := \left\{ k(x,z) = \int_{\mathbb{R}^n} e^{j\gamma^T(x-z)} G_\beta(\|\gamma\|)\, dp(\gamma) \quad : \text{ p is a probablity measure on } \mathbb{R}^n \right\}$$

where β is defined on an ordered set, and $G_\beta : \mathbb{R}_+ \to [0,1]$ is a decreasing continuous function with $G_\beta(0) = 1$ and $G_\beta(\infty) = 0$. In addition, to ensure that these classes of kernels are nested, we require that $G_{\beta_1}(r) \geq G_{\beta_2}(r)$ for every $r > 0$ and $\beta_1 \leq \beta_2$. Important candidates for $G_\beta(\|\gamma\|)$ are $\exp(-\beta\|\gamma\|^2)$ and $\|\gamma\|^{-\beta}$ for $\beta > 0$. Note that we have left the choice of the norm to the application. Two important candidates are L_1 and L_2 norms. In Section 5, we will also assume that the function $G_\beta(\|\gamma\|)$ is differentiable with respect to γ.

3. Kernel Selection Criterion

In the process of learning the kernel function, one needs a criterion for choosing a kernel (or equivalently, feature space) from the class of admissible kernels. In the classification task, the ideal criterion is the misclassification error. But since the probability density of data is unknown, this criterion cannot be computed. This problem has been circumvented by proposing upper bounds on the misclassification error which hold with high probability (see for example Vapnik, 1999 and Chapter 4 of Cristianini and Shawe-Taylor, 2000). In Chapelle et al. (2002), several criteria have

been studied and the authors suggested minimizing the radius-margin bound $(R/\Delta)^2$ as the preferred criterion.

Consider the training set of a classification task consisting of the input-output pairs $\{(x_1,y_1),...,(x_l,y_l)\}$, with $y_i \in \{-1,1\}$. For a fixed kernel, support vector machines compute the maximal hard/soft margin separating hyperplane. By generalizing hard margin, 1-norm soft margin, and 2-norm soft-margin objective functions, Lanckriet et al. (2004) obtained the following generalized criterion for $1/\Delta^2$ (which must be minimized):

$$\omega'_{C',\tau'}(k) := \max_{\substack{\alpha:\ 0\le\alpha\le C',\\ \alpha^T y=0}} \left\{ 2\alpha^T e - \sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u\alpha_v y_u y_v k(x_u,x_v) - \tau'\alpha^T\alpha \right\} \tag{2}$$

where e is the $n\times 1$ vector of ones and the prime sign is used to distinguish the parameters from those used in this paper. If (C',τ') is equal to $(\infty,0)$,$(C',0)$, and $(\infty,1/C')$, one obtains hard margin, 1-norm soft margin, and 2-norm soft-margin performance measures, respectively. Our first change to this criterion is adding the constraint $\alpha^T e = 2$. It has been shown (Crisp and Burges, 1999; Mavroforakis and Theodorodis, 2006) that by adjusting the parameters C and the offset b appropriately, this new constraint does not change the separating hyperplane. However, the new constraint plus $\alpha^T y = 0$ gives $\sum_{i\in C^-}\alpha_i = \sum_{i\in C^+}\alpha_i = 1$, which makes the exposition of our method simpler. Furthermore, we divide (2) by $1+\tau'$ and define $\tau := \frac{\tau'}{1+\tau'}$. So, in this paper we use the criterion:

$$\omega_{C,\tau}(k) := \min_{\alpha\in\mathcal{A}} \left\{ (1-\tau)\sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u\alpha_v y_u y_v k(x_u,x_v) + \tau\alpha^T\alpha \right\} \tag{3}$$

where $\mathcal{A} := \left\{\alpha\in\mathbb{R}^l :\ 0\le\alpha\le C,\ \alpha^T y = 0,\ \alpha^T e = 2\right\}$. Note that in the new criterion $\max\{\frac{1}{l_1},\frac{1}{l_2}\} \le C \le 2$ and $0\le\tau\le 1$. This criterion works well for a fixed kernel by maximizing the margin Δ. But in general, to minimize the radius-margin bound $(R/\Delta)^2$, one must impose some constraint on the radius R, as well. For translation invariant kernels we have $R^2 = \|\Phi(x)\|^2 = k(x,x) = \tilde{k}(0)$. Hence, bounding the radius R is equivalent to bounding $\tilde{k}(0)$. One can easily verify that bounding $trace\{K\}$, where K is the kernel matrix in the transductive framework (see Lanckriet et al., 2004, Equation 17), also leads to a bound on $\tilde{k}(0)$. Since by exploding R and $trace\{K\}$, the margin Δ also explodes by the same amount, whilst the radius-margin bound remains constant, we from now on assume that $\tilde{k}(0) = 1$ and obtain the following optimization problem:[2]

$$\begin{aligned} \sup_k \min_{\alpha\in\mathcal{A}} & \left\{ (1-\tau)\sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u\alpha_v y_u y_v k(x_u,x_v) + \tau\alpha^T\alpha \right\} \\ \text{s.t.}\quad & k(x,z) = \int_{\mathbb{R}^n} e^{j\gamma^T(x-z)} G_\beta(\|\gamma\|)\, dV(\gamma) \\ & \int_{\mathbb{R}^n} dV(\gamma) = 1, \\ & \text{V is monotonically increasing} \end{aligned}$$

2. Note that since k is a complex-valued positive semi-definite kernel, the objective function is real-valued and non-negative.

or equivalently

$$\sup_{p\in\mathcal{P}(\mathbb{R}^n)} \min_{\alpha\in\mathcal{A}} \int_{\mathbb{R}^n} \alpha^T H(\gamma)\alpha \, dp(\gamma) \tag{4}$$

where $\mathcal{P}(\Gamma)$ denotes the set of all probability measures on Γ and $H(\gamma)$ is defined below. Let $G(\gamma)$ be the $l\times l$ matrix whose (u,v)th entry is $y_u y_v \exp(j\gamma^T(x_u - x_v))G_\beta(\|\gamma\|)$. Define $H(\gamma) := (1-\tau)G(\gamma) + \tau I_l$ where I_l is the $l\times l$ identity matrix. The following equation shows an $O(l)$ computational method for computing $\alpha^T G(\gamma)\alpha$:

$$\alpha^T G(\gamma)\alpha = \left\| \sum_{u=1}^{l} \alpha_u y_u \exp(j\gamma^T x_u) \right\|_2^2 G_\beta(\|\gamma\|). \tag{5}$$

4. Variational Optimization

In this section, we first prove the existence of a solution to problem (4). Next we prove that (4) can be written as a min-max problem with integration replaced by summation. This allows us to introduce a SIP formulation of the problem. We then introduce another SIP problem for learning the optimal kernel and parameter τ.

4.1 Replacing Sup with Max

We will prove that the sup operation in (4) can be substituted by the max operation. Note that all the variability in the choice of a probability measure p from $\mathcal{P}(\mathbb{R}^n)$ collapses to the choice of an $l\times l$ matrix $\int_{\mathbb{R}^n} H(\gamma)\, dp(\gamma)$ from $S(\mathbb{C}^l)$, where $S(\mathbb{C}^l)$ is the space of all $l\times l$ Hermitian complex-valued matrices. So, it is sufficient to prove that the set of all these matrices is compact, which ensures that any sequence of these matrices has a convergent subsequence, and subsequently, the supremum value is achieved. Furthermore, by compacting the parameter space $\mathcal{A}$ in (3) there is no need to assume that the kernel matrices are strictly positive definite, as was done in Lemma 2 of Micchelli and Pontil (2005).

Let $C_0(\mathbb{R}^n)$ denote the function space of all continuous complex-valued functions defined on $\mathbb{R}^n$ which vanish at infinity, that is, $\lim_{\|\gamma\|\to\infty} g(\gamma) = 0$ for any $g \in C_0(\mathbb{R}^n)$. By Theorem 3.17 of Rudin (1987), the function space $C_0(\mathbb{R}^n)$ with the norm

$$\|g\| := \max_{\gamma\in\mathbb{R}^n} |g(\gamma)|$$

is a Banach space. Note that the use of max operation is justified by the continuity and vanishing properties of $g \in C_0(\mathbb{R}^n)$. Considering the above discussions, we need to prove the following theorem.

Theorem 1 *For any fixed sample data $Z = \{(x_1,y_1),...,(x_l,y_l)\}$, the set*

$$\mathcal{K}_Z := \left\{ \int_{\mathbb{R}^n} H(\gamma)\, dp(\gamma) \;\; : p \in \mathcal{P}(\mathbb{R}^n) \right\}$$

is a compact subset of $S(\mathbb{C}^l)$.[3]

Proof Consider any sequence (p_n) in $\mathcal{P}(\mathbb{R}^n)$. Define positive linear functionals $T_n : C_0(\mathbb{R}^n) \to \mathbb{C}$ by $T_n g := \int_{\mathbb{R}^n} g(\gamma)\, dp_n(\gamma)$. So, $T_n \in C_0'(\mathbb{R}^n)$, the dual space of $C_0(\mathbb{R}^n)$. Since for each n, $\|T_n\| = 1$, by Banach-Alaoglu theorem (see for example Theorem 3.15 of Rudin, 1991 or page 237 of Royden, 1988), there exists some $T \in C_0'(\mathbb{R}^n)$ with $\|T\| \leq 1$ and a subsequence (T_m) such that $T_m \to T$ in weak* topology. This means that for every $g \in C_0(\mathbb{R}^n)$, we have $T_m g \to Tg$. Since, each element of the matrix $H(\gamma)$ belongs to $C_0(\mathbb{R}^n)$, it follows that $\int_{\mathbb{R}^n} H(\gamma)\, dP_m(\gamma) \to TH$. One can easily prove by contradiction that $\|T\| = 1$ and T is in fact a positive linear functional. By the Riesz representation theorem (see Theorem 6.19 of Rudin, 1987), the functional T can be represented uniquely by a complex Borel measure μ ,with $\int_{\mathbb{R}^n} d|\mu| = \|T\| = 1$, in the sense that

$$Tg = \int_{\mathbb{R}^n} g\, d\mu \quad \text{for every } g \in C_0(\mathbb{R}^n).$$

The positivity of T implies that μ is a positive real measure. So, $\int_{\mathbb{R}^n} d\mu = 1$ and consequently μ is also a probability measure on $\mathbb{R}^n$. Thus,

$$\int_{\mathbb{R}^n} H(\gamma)\, dp_m(\gamma) \longrightarrow \int_{\mathbb{R}^n} H(\gamma)\, d\mu(\gamma)$$

which indicates that the set $\mathcal{K}_Z$ is compact. ■

4.2 Interchanging the min and max Operations

We first prove a theorem about interchanging the min and max operations which is an abstracted and generalized version of Theorem 20 in Micchelli and Pontil (2005).

Theorem 2 *Assume that* Γ *is a compact Hausdorff space and the function* $g : \Gamma \times \mathbb{R}^l \to \mathbb{R}$ *is continuous in the first parameter and convex and differentiable in the second parameter. Let* $\mathcal{E}$ *and* I *be finite index sets ,* a_i*, where* $i \in \mathcal{E} \cup I$*, be* $l \times 1$ *vectors, and* b_i*, where* $i \in \mathcal{E} \cup I$*, be real-valued scalars. Considering problem (6), assume that the Slater's condition (see Boyd and Vandenberghe, 2004) holds, that is, there exists some* α *such that* $a_i^T \alpha = b_i$ *for all* $i \in \mathcal{E}$ *and* $a_i^T \alpha < b_i$ *for all* $i \in I$*. Then there exist a discrete probability measure* $\tilde{p} \in \mathcal{P}(\Gamma)$ *with at most* $l+1$ *atoms and some feasible point* $\tilde{\alpha}$ *which solve the max-min problem*

$$\max_{p \in \mathcal{P}(\Gamma)} \min_{\substack{\alpha : a_i^T \alpha = b_i \text{ for } i \in \mathcal{E}, \\ a_i^T \alpha \geq b_i \text{ for } i \in I}} \int_\Gamma g(\gamma, \alpha)\, dp(\gamma) \tag{6}$$

and the min-max problem

$$\min_{\substack{\alpha : a_i^T \alpha = b_i \text{ for } i \in \mathcal{E}, \\ a_i^T \alpha \geq b_i \text{ for } i \in I}} \max_{p \in \mathcal{P}(\Gamma)} \int_\Gamma g(\gamma, \alpha)\, dp(\gamma) \tag{7}$$

simultaneously. In addition, each atom of $\tilde{p}$ *is a global maximum of* $g(\gamma, \tilde{\alpha})$ *as a function of* γ.

3. Note that the topology of $S(\mathbb{C}^l)$ is the same as that of $\mathbb{R}^{l^2}$.An $l \times l$ hermitian matrix contains l real-valued diagonal elements and $\frac{l^2-l}{2}$ independent complex-valued off-diagonal element.

Proof Assume that $\hat{\alpha}$ and $\hat{p}$ solve the problem (7). Define the function

$$\phi : \mathbb{R}^l \to \mathbb{R}$$
$$\phi(\alpha) := \max_{\gamma \in \Gamma} \{g(\gamma, \alpha)\}$$

and the set

$$\Gamma^* := \{\gamma : \gamma \in \Gamma, g(\gamma, \hat{\alpha}) = \phi(\hat{\alpha})\}.$$

By Lemma 24 of Micchelli and Pontil (2005), the directional derivative of ϕ along the direction $d \in \mathbb{R}^l$, denoted by $\phi'_+(\alpha; d)$, is given by:

$$\phi'_+(\alpha; d) = \max_{\gamma \in \Gamma^*} \left\{ d^T \nabla_\alpha g(\gamma, \alpha) \right\}.$$

Since $\hat{\alpha}$ minimizes (7), we have

$$\phi'_+(\hat{\alpha}; d) = \max_{\gamma \in \Gamma^*} \left\{ d^T \nabla_\alpha g(\gamma, \alpha) \mid_{\alpha = \hat{\alpha}} \right\} \geq 0 \tag{8}$$

for any direction d such that $a_i^T d = 0$ for $i \in \mathcal{E}$ and $a_i^T d \geq 0$ for $i \in I^*$, where $I^* := \{i \in I : a_i^T \alpha = b_i\}$.

Let $\mathcal{M}$ be the convex hull of the set of vectors $\mathcal{N} := \{\nabla_\alpha g(\gamma, \alpha) \mid_{\alpha = \hat{\alpha}} : \gamma \in \Gamma^*\} \subseteq \mathbb{R}^l$. Since $\mathcal{M} \subseteq \mathbb{R}^l$, by the Caratheodory theorem (see for example Section 17 of Rockafellar, 1970) every vector in $\mathcal{M}$ can be expressed as a convex combination of at most $l+1$ elements of $\mathcal{N}$. We claim that the set

$$O := \left\{ \sum_{i \in \mathcal{E} \cup I^*} \lambda_i a_i : \ \lambda_i \geq 0 \text{ for all } i \in I^* \right\}$$

intersects $\mathcal{M}$. Assume, on the contrary, that $\mathcal{M}$ and O are distinct. Since $\mathcal{M}$ is convex and compact and O is convex and closed, by the strict separating hyperplane theorem (see corollary 11.4.2 of Rockafellar, 1970), there exists a separating hyperplane $w^T \alpha + b = 0$, $w \in \mathbb{R}^l$, $b \in \mathbb{R}$, such that

$$\sum_{i \in \mathcal{E} \cup I^*} \lambda_i w^T a_i + b > 0, \qquad \forall \lambda : \ \lambda_i \geq 0 \text{ for } i \in I^*$$

and

$$w^T \nabla_\alpha g(\gamma, \alpha) \mid_{\alpha = \hat{\alpha}} + b < 0, \qquad \forall \gamma \in \Gamma^*. \tag{9}$$

The first condition, for $\lambda = 0$ implies that $b > 0$ and since λ_i can take any real value for $i \in \mathcal{E}$ and any nonnegative value for $i \in I^*$, we have

$$w^T a_i = 0, \quad \forall i \in \mathcal{E},$$
$$w^T a_i \geq 0, \quad \forall i \in I^*.$$

By combining these results with (9), we get

$$\max_{\gamma \in \Gamma^*} w^T \nabla_\alpha g(\gamma, \hat{\alpha})\,|_{\alpha=\hat{\alpha}} < 0.$$

This means that w is a feasible descent direction at $\hat{\alpha}$ which contradicts (8). So, the sets O and $\mathcal{M}$ intersect. This means that there exist real numbers $\hat{\lambda}_i,\ i \in \mathcal{E}$, nonnegative numbers $\hat{\lambda}_i,\ i \in I^*$, and a discrete probability measure $\hat{p}$ with at most $l+1$ atoms such that

$$\sum_{i \in \mathcal{E} \cup I^*} \hat{\lambda}_i a_i = \int_\Gamma \nabla_\alpha g(\gamma, \hat{\alpha})\, d\hat{p}. \tag{10}$$

Now, we turn our attention to the solution of problem (6) for $p = \hat{p}$. This is a convex optimization problem. Since by assumption the Slater's condition holds, the KKT conditions provide a necessary and sufficient condition for optimality (see Boyd and Vandenberghe, 2004, page 244) and therefore a solution $\check{\alpha}$ to problem (6) is found by solving the following KKT conditions:

$$\begin{aligned}
\sum_{i \in \mathcal{E} \cup I} \check{\lambda}_i a_i &= \int_\Gamma \nabla_\alpha g(\gamma, \check{\alpha}) d\hat{p} \\
a_i^T(\check{\alpha}) &= 0, \quad i \in \mathcal{E} \\
a_i^T(\check{\alpha}) &\geq 0, \quad i \in I \\
\check{\lambda}_i &\geq 0, \quad i \in I \\
\check{\lambda}_i \left(a_i^T \check{\alpha} - b_i\right) &= 0, \quad i \in I.
\end{aligned}$$

By defining $\hat{\lambda}_i = 0$ for $i \in I \setminus I^*$, using (10), and recalling the definition of I^*, it can be seen that $\hat{\alpha}, \hat{\lambda}$ are the unique solution to the above KKT conditions. Thus, $\tilde{\alpha} = \hat{\alpha}$ and $\tilde{p} = \hat{p}$ solve problems (6) and (7) simultaneously and the theorem follows. ■

Corollary 3 *Assume that Γ is any compact subset of $\mathbb{R}^n$ and the parameter C is chosen*[4] *such that $C > \max\{\frac{1}{l_1}, \frac{1}{l_2}\}$. Then, there exist $\tilde{\alpha} \in \mathbb{R}^n$ and $\tilde{p} \in \mathcal{P}(\Gamma)$ that solve problems (11) and (12) simultaneously. Furthermore, $\tilde{p}$ is a discrete probability measure with at most $l+1$ atoms and each atom of $\tilde{p}$ is a global maximum of $\tilde{\alpha}^T H(\gamma)\tilde{\alpha}$.*

$$\max_{p \in \mathcal{P}(\Gamma)} \min_{\alpha \in \mathcal{A}} \int_\Gamma \alpha^T H(\gamma) \alpha\, dp(\gamma), \tag{11}$$

$$\min_{\alpha \in \mathcal{A}} \max_{p \in \mathcal{P}(\Gamma)} \int_\Gamma \alpha^T H(\gamma) \alpha\, dp(\gamma). \tag{12}$$

4. For a similar constraint in the context of ν-SVMs see Section 4 of Crisp and Burges (1999).

Proof It is sufficient to show that the Slater's condition holds, that is, there exists $\alpha \in \mathbb{R}^l$ such that $\alpha^T e = 2$, $\alpha^T y = 0$, $\alpha > 0$, and $\alpha < C$. One can easily verify that the choice $\alpha_i = \frac{1}{l_1}$ for $i \in C^+$ and $\alpha_i = \frac{1}{l_2}$ for $i \in C^-$ satisfies these conditions. ■

In the rest of the paper, we assume that $C > \max\{\frac{1}{l_1}, \frac{1}{l_2}\}$.

4.3 Confining Integration to a Compact Region

To write (4) as a min-max optimization problem, we first proved in Section 4.1 that the sup operation can be replaced by the max operation. In the previous section we proved Corollary 3 which asserts that if the integration region of (4) could have been replaced by a compact set, then the max and min operations could also be interchanged. In this section, we show that the integration region of (4) can safely be confined to a compact subset of $\mathbb{R}^n$. Let us first prove two useful lemmas.

Lemma 4 *For arbitrary domains X and Y and every function $f : X \times Y \to \mathbb{R}$, the following inequality holds:*

$$\sup_{x\in X} \inf_{y\in Y} f(x,y) \leq \inf_{y\in Y} \sup_{x\in X} f(x,y).$$

Proof Assume on the contrary that

$$\sup_{x\in X} \inf_{y\in Y} f(x,y) > \inf_{y\in Y} \sup_{x\in X} f(x,y).$$

Then, there exist $\tilde{x} \in X$ and $\tilde{y} \in Y$ such that

$$\inf_{y\in Y} f(\tilde{x},y) > \sup_{x\in X} f(x,\tilde{y})$$

which contradicts with the existence of $f(\tilde{x},\tilde{y})$. ■

Lemma 5 *If $\tau < 1$, then there exists some compact subset Γ_β of $\mathbb{R}^n$, independent of α, where all γ's that maximize $\alpha^T H(\gamma)\alpha$ lie in it.*[5]

Proof For each $\alpha \in \mathbb{R}^l$ we have

$$\max_{\gamma\in\mathbb{R}^n} \alpha^T H(\gamma)\alpha = (1-\tau) \max_{\gamma\in\mathbb{R}^n} \left\{ \left\| \sum_{u=1}^{l} \alpha_u y_u \exp(j\gamma^T x_u) \right\|_2^2 G_\beta(\|\gamma\|) \right\} + \tau\alpha^T\alpha. \tag{13}$$

Let $t(\alpha)$ denote the maximum value of the term in the braces in the above equation. Since G_β is continuous and $G_\beta(0) = 1$, there is an open ball around zero in $\mathbb{R}^n$ such that $G_\beta(\|\gamma\|) > 0$. This fact plus the condition $\alpha^T e = 2$, ensure that the coefficient of at least one of the exponential terms in (13) is nonzero. Hence, the term in the braces in (13), as a function of γ, is never identically zero and thus $t(\alpha) > 0$. For all values of γ with $G_\beta(\|\gamma\|) < \frac{t}{4}$ we have

5. Note that by (3), the choice $\tau = 1$ is unrealistic.

$$\alpha^T H(\gamma)\alpha - \tau\alpha^T\alpha = \left\| \sum_{u=1}^{l} \alpha_u y_u \exp(j\gamma^T x_u) \right\|^2 G_\beta(\|\gamma\|) \leq 4G_\beta(\|\gamma\|) < t(\alpha)$$

and the lemma's assertion follows from $lim_{r\to\infty} G(r) = 0$. ■

Theorem 6 *There exists an optimal solution $(\tilde{\alpha}, \tilde{p})$ to problem (4) which is also a solution to it when the integration domain is confined to the compact set Γ_β.*

Proof Let $(\tilde{\alpha}, \tilde{p})$ be a solution to problems (6) and (7) with Γ replaced by Γ_β. By Lemma 4 and Corollary 3, we have the following sequence of inequalities:

$$\begin{aligned}
\max_{p\in\mathcal{P}(\Gamma_\beta)} \min_{\alpha\in\mathcal{A}} \int_{\Gamma_\beta} \alpha^T H(\gamma)\alpha\, dp(\gamma) \leq \\
\max_{p\in\mathcal{P}(\mathbb{R}^n)} \min_{\alpha\in\mathcal{A}} \int_{\mathbb{R}^n} \alpha^T H(\gamma)\alpha\, dp(\gamma) \leq \\
\min_{\alpha\in\mathcal{A}} \max_{p\in\mathcal{P}(\mathbb{R}^n)} \int_{\mathbb{R}^n} \alpha^T H(\gamma)\alpha\, dp(\gamma) \leq \\
\min_{\alpha\in\mathcal{A}} \max_{p\in\mathcal{P}(\mathbb{R}^n)} \int_{\mathbb{R}^n} \max_{\gamma\in\mathbb{R}^n} \alpha^T H(\gamma)\alpha\, dp(\gamma) = \\
\min_{\alpha\in\mathcal{A}} \max_{\gamma\in\mathbb{R}^n} \alpha^T H(\gamma)\alpha = \\
\min_{\alpha\in\mathcal{A}} \max_{p\in\mathcal{P}(\Gamma_\beta)} \int_{\Gamma_\beta} \alpha^T H(\gamma)\alpha\, dp(\gamma).
\end{aligned}$$

However, the first and the last terms are equal by Corollary 3.

4.4 Semi-infinite Programming Formulation

We have not yet addressed the problem of how (12) is to be really solved on a machine. In this section, we reformulate (12) as a semi-infinite programming problem for which many algorithms have been proposed (see Hettich and Kortanek, 1993; Reemtsen and Görner, 1998, for two reviews on the subject).

Theorem 7 *Let $\tilde{\alpha}$ and $\tilde{t}$ be a solution to the semi-infinite programming problem (14) and define the set $\Gamma_\beta^H(\alpha) := \left\{\gamma \in \Gamma_\beta : \alpha^T H(\gamma)\alpha = max_{\gamma\in\Gamma_\beta}\alpha^T H(\gamma)\alpha\right\}$. Let (Q) be the QCQP problem that is obtained by replacing Γ_β by $\Gamma_\beta^H(\tilde{\alpha}) \equiv \{\gamma_1, ..., \gamma_m\}$ in (14) and let $\tilde{\mu}_1, ..., \tilde{\mu}_m$ be a set of Lagrange multipliers associated with the constraints $t \geq \alpha^T H(\gamma_i)\alpha$, $1 \leq i \leq m$ which optimize the dual problem of (Q). If $\tilde{p}$ is the discrete probability measure defined by $\tilde{p}(\gamma_i) := \tilde{\mu}_i$ $i = 1, ..., m$, then $\tilde{\alpha}$ and $\tilde{p}$ solve the problem (4). In addition, there exists a solution pair $(\tilde{\alpha}^*, \tilde{p}^*)$ such that $\tilde{p}^*$ contains at most $l+1$ nonzero atoms.*

$$\begin{array}{ll} \min_{\alpha,t} & t \\ s.t. & t \geq \alpha^T H(\gamma)\alpha \;\; \textit{for all} \;\; \gamma \in \Gamma_\beta \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array} \tag{14}$$

Proof Since for all $\gamma \notin \Gamma_\beta^H(\tilde{\alpha})$ the strict inequality $\tilde{t} > \tilde{\alpha}^T H(\gamma)\tilde{\alpha}$ holds, $\tilde{\alpha}$ and $\tilde{t}$ also solve the following QCQP problem:

$$\begin{array}{ll} \min_{\alpha,t} & t \\ s.t. & t \geq \alpha^T H(\gamma_i)\alpha \;\; i = 1,...,m \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array}$$

In addition, from $\tilde{t} = \tilde{\alpha}^T H(\gamma_1)\tilde{\alpha} = ... = \tilde{\alpha}^T H(\gamma_m)\tilde{\alpha}$, it follows that $\tilde{\alpha}$ and $\tilde{t}$ also solve the following problem:

$$\min_{\alpha \in \mathcal{A}} \max_{\mu \geq 0,\, \sum_{i=1}^m \mu_i = 1} \alpha^T \left[\sum_{i=1}^m \mu_i H(\gamma_i) \right] \alpha \tag{15}$$

By Theorem 17 of Lanckriet et al. (2004), if $\tilde{\mu}$ is chosen as specified by the statement of this theorem, then $\tilde{\alpha}$ and $\tilde{\mu}$ simultaneously solve the min-max problem (15) and the following max-min problem:

$$\max_{\mu \geq 0,\, \sum_{i=1}^m \mu_i = 1} \min_{\alpha \in \mathcal{A}} \alpha^T \left[\sum_{i=1}^m \mu_i H(\gamma_i) \right] \alpha$$

which can also be written as

$$\max_{p \in \mathcal{P}(\Gamma_\beta^H(\tilde{\alpha}))} \min_{\alpha \in \mathcal{A}} \int \alpha^T H(\gamma)\alpha \, dP(\gamma)$$

The first assertion of the theorem follows from the following inequalities:

$$\begin{aligned} \tilde{t} = \max_{p \in \mathcal{P}(\Gamma_\beta^H(\tilde{\alpha}))} \min_{\alpha \in \mathcal{A}} \int \alpha^T H(\gamma)\alpha \, dP(\gamma) \leq \\ \max_{p \in \mathcal{P}(\Gamma_\beta)} \min_{\alpha \in \mathcal{A}} \int \alpha^T H(\gamma)\alpha \, dP(\gamma) \leq \\ \min_{\alpha \in \mathcal{A}} \max_{p \in \mathcal{P}(\Gamma_\beta)} \int \alpha^T H(\gamma)\alpha \, dP(\gamma) = \tilde{t} \end{aligned}$$

where the last equality follows from a simple reformulation of problem (14). The last part of the theorem follows from Corollary 3 and reversing the above proof. ■

4.5 Including Other Kernels in the Learning Process

Although the focus of this paper is on the task of learning translation invariant kernels, it is easy to furnish the set of admissible kernels with other kernel functions. For example, one may want to find the best convex combination of stabilized translation invariant kernels along with isotropic/non-isotropic Gaussian kernels, and polynomial kernels with degrees one to five.[6] In general, assume that we have M classes of kernels

$$K_i := \left\{k_\gamma(x,z) : \ \gamma \in \Gamma_i\right\} \qquad i = 1,...,M$$

where $\Gamma_1,...,\Gamma_M$ are distinct compact Hausdorff spaces. For $i \in 1,...,M$ let $G_i(\gamma)$ be the $l \times l$ matrix whose (u,v)'s entry is $y_u y_v k_\gamma(x_u,x_v)$ and define $H_i(\gamma) := (1-\tau)G_i(\gamma) + \tau I_l$. The problem of learning the best convex combination of kernels from these classes for classification with support vector machines can be stated as

$$\sup_{p\in\mathcal{P}(\Gamma_0)} \min_{\alpha\in\mathcal{A}} \int_{\Gamma_0} \alpha^T H_0(\gamma)\alpha \, dp(\gamma) \tag{16}$$

where $\Gamma_0 := \Gamma_1 \cup ... \cup \Gamma_M$ and

$$H_0(\gamma) := \begin{cases} H_1(\gamma) & if \gamma \in \Gamma_1 \\ H_2(\gamma) & if \gamma \in \Gamma_2 \\ \qquad \vdots & \\ H_M(\gamma) & if \gamma \in \Gamma_M \end{cases}.$$

The results of the previous sections will hold for this combined class of kernels if we prove that Γ_0 is a compact Hausdoff space and that $h_0(\gamma,\alpha) := \alpha^T H_0(\gamma)\alpha$ is continuous with respect to γ. Let $\mathcal{T}_i$ denote the topology on Γ_i for $i \in 1,...,M$. Define the set $\mathcal{T}_0$ of subsets of Γ_0 as:

$$\mathcal{T}_0 := \{O_1 \cup O_2 ... \cup O_M : O_1 \in \mathcal{T}_1, O_2 \in \mathcal{T}_2, ..., O_M \in \mathcal{T}_M\}.$$

Proposition 8 *$\mathcal{T}_0$ is a topology.*

Proof Clearly $\Gamma_0 \in \mathcal{T}_0$. Next we must show that $\mathcal{T}_0$ is closed under arbitrary union. Let $O = \bigcup_{i\in\mathfrak{I}} O_i$ where $O_i \in \mathcal{T}_0$ and $\mathfrak{I}$ is an arbitrary index set. We have

$$O = \bigcup_{j\in\{1,...,M\}} \left(\bigcup_{i\in\mathfrak{I}} O_i \cap \Gamma_j \right)$$

6. Although the class of translation invariant kernels includes the set of isotropic/non-isotropic Gaussian kernels, it is not the case for the stabilized class $\mathcal{K}_\beta$.

which shows that $O \in \mathcal{T}_0$. Finally, we should show that finite intersection of closed sets is a closed set. Assume that $C = \bigcap_{i=1}^{r} C_i$, where $r \in \mathbb{N}$ and $C_i^c \in \mathcal{T}_0$. We have

$$C^c = \bigcup_{i\in\{1,...,r\}} C_i^c = \bigcup_{i\in\{1,...,r\}} \bigcup_{j\in\{1,...,M\}} (C_i^c \cap \Gamma_j).$$

Since C_i^c and Γ_j are both open in $\mathcal{T}_j$, the set $C_i^c \cap \Gamma_j$ is open in $\mathcal{T}_j$. By the properties of topologies $\mathcal{T}_1, ..., \mathcal{T}_M$ and the definition of topology $\mathcal{T}_0$ it follows that $C^c \in \mathcal{T}_0$ which shows that C is closed in $\mathcal{T}_0$. ■

Proposition 9 *Assume that the functions $h_i(\gamma, \alpha) : \Gamma_i \times \mathbb{R}^l \to \mathbb{R}$ defined as $h_i(\gamma, \alpha) = \alpha^T H_i(\gamma)\alpha$, where $i = 1, ..., M$, are continuous in the first parameter. Then, the function $h_0(\gamma, \alpha) : \Gamma_0 \times \mathbb{R}^l \to \mathbb{R}$ defined as $h_0(\gamma, \alpha) = \alpha^T H(\gamma)\alpha$,where the topology of Γ_0 is $\mathcal{T}_0$, is also continuous in the first parameter.*

Proof Fix α to any value and define $\bar{h}_i(\gamma) := h_i(\gamma, \alpha)$ for $i = 0, ..., M$. Let O be an open subset of $\mathbb{R}$. We must show that $\bar{h}_0^{-1}(O)$ is open in topology $\mathcal{T}_0$. We have

$$\bar{h}_0^{-1}(O) = \bigcup_{i=1,...,M} \left(\bar{h}_0^{-1}(O) \cap \Gamma_i\right) = \bigcup_{i=1,..,M} \bar{h}_i^{-1}(O).$$

Since the set $\bar{h}_i^{-1}(O)$ is open in $\mathcal{T}_i$ for each $i \in \{1, ..., M\}$, it follows from the definition of $\mathcal{T}_0$ that the union of these sets is also open in $\mathcal{T}_0$. So, $\bar{h}_0^{-1}(O)$ is open in $\mathcal{T}_0$ and the result follows. ■

Proposition 10 *The set Γ_0 with topology $\mathcal{T}_0$ is compact in itself.*

Proof Let $O = \{O_i : i \in I_0\}$ be an open covering of Γ_0, where I_0 is some index set. Let j be any number in the set $\{1, ..., M\}$. Since $\Gamma_j \subseteq \Gamma_0$, the set O is also an open covering for Γ_j. By compactness of Γ_j, there exists a finite index set $I_j \subseteq I_0$ such that the set $\left\{O_i : i \in I_j\right\}$ is an open subcovering of Γ_j. Thus, the set $\{O_i : i \in I_1 \cup ... \cup I_M\}$ is a finite open subcovering of $\mathcal{T}_0$ which proves that Γ_0 is compact. ■

Proposition 11 *Assume that topologies $\mathcal{T}_1, ..., \mathcal{T}_M$ are Hausdorff. Then, so is the topology $\mathcal{T}_0$.*

Proof We must prove that for any two points $\gamma_1, \gamma_2 \in \Gamma_0$ there are disjoint open sets O_1 and O_2 such that $\gamma_1 \in O_1$ and $\gamma_2 \in O_2$. If γ_1 and γ_2 belong to the same set Γ_j for some $j \in 1, ..., M$, then the assertion follows from the Hausdorffness property of $\mathcal{T}_j$. Without loss of generality, assume that $\gamma_1 \in \Gamma_1$ and $\gamma_2 \in \Gamma_2$. The choice $O_1 = \Gamma_1$ and $O_2 = \Gamma_2$ completes the proof. ■

Theorem 12 *Assume that $\Gamma_1, ..., \Gamma_M$ are compact Hausdorff spaces and the matrices H_j are as defined previously in this section. Furthermore, assume that for $j = 1, ..., M$ the functions $h_j(\gamma, \alpha) : \Gamma_j \to \mathbb{R}^l$ defined by $h_j(\gamma, \alpha) := \alpha^T H_j(\gamma)\alpha$ are continuous in the first parameter. Let $\tilde{\alpha}$ and $\tilde{t}$ be a solution to the semi-infinite programming problem $P(\Gamma_1, ..., \Gamma_M)$, which is defined as*

$$
\begin{array}{rll}
 & \min_{\alpha,t} & t \\
 & s.t. & t \geq \alpha^T H_1(\gamma)\alpha \ \ \textit{for all} \ \ \gamma \in \Gamma_1 \\
 & & \vdots \\
P(\Gamma_1,...,\Gamma_M) := & & t \geq \alpha^T H_M(\gamma)\alpha \ \ \textit{for all} \ \ \gamma \in \Gamma_M \\
 & & 0 \leq \alpha \leq C \\
 & & \alpha^T y = 0 \\
 & & \alpha^T e = 2.
\end{array} \tag{17}
$$

Define the set $\Gamma_j^H(\alpha) := \left\{\gamma \in \Gamma_j : \alpha^T H_j(\gamma)\alpha = max_{\gamma\in\Gamma_j}\alpha^T H_j(\gamma)\alpha\right\}$. *Let (Q) be the QCQP problem that is obtained by replacing every* Γ_j *with* $j = 1,...,M$ *by* $\Gamma_j^H(\tilde{\alpha}) \equiv \left\{\gamma_1^j,...,\gamma_{m_j}^j\right\}$ *in (17). For any* $j \in \{1,...,M\}$ *let* $\tilde{\mu}_1^j,...,\tilde{\mu}_{m_j}^j$ *be a set of Lagrange multipliers associated with the constraints* $t \geq \alpha^T H_j(\gamma_i^j)\alpha,\ \ 1 \leq i \leq m_j$ *which optimize the dual problem of (Q). If* $\tilde{p}$ *is the discrete probability measure defined by* $\tilde{p}(\gamma_i^j) := \tilde{\mu}_i^j \ \ i = 1,...,m_j \ \ j = 1,...,M$, *then the pair* $(\tilde{\alpha},\tilde{p})$ *solves the problem (16). In addition, there exists a solution pair* $(\tilde{\alpha}^*,\tilde{p}^*)$ *such that* $\tilde{p}^*$ *contains at most* $l+1$ *nonzero atoms*.

Proof The result is immediately obtained by replacing the set Γ_β by Γ_0 in Theorem 7. ■

4.6 Automatic Adjustment of the Parameter τ

In this section, we consider the following problem:

$$
\max_{0\leq\tau\leq 1, p\in\mathcal{P}(\mathbb{R}^n)} \min_{\alpha\in\mathcal{A}} \int_{\mathbb{R}^n} \alpha^T H(\gamma)\alpha \, dP(\gamma). \tag{18}
$$

It is well known that the parameter τ can be envisioned as the weight of the kernel $\delta(x,z)$, where δ is the Kronecker delta function. So, the problem of learning the parameter τ is equivalent to choosing the best convex combination of the set of translation invariant kernels augmented with the delta kernel $\delta(x,z)$. By using Theorem 12 we get the following corollary.

Corollary 13 *Let* $\tilde{\alpha}$ *and* $\tilde{t}$ *be a solution to the semi-infinite programming problem* $P_\tau(\Gamma_\beta)$, *where for each compact set* Γ *the problem* $P_\tau(\Gamma)$ *is defined by (19). Define the set*

$$
\Gamma_\beta^G(\alpha) := \left\{\gamma \in \Gamma_\beta : \alpha^T G(\gamma)\alpha = max_{\gamma\in\Gamma_\beta}\alpha^T G(\gamma)\alpha\right\}.
$$

Let (Q) be the QCQP problem that is obtained by replacing Γ *by* $\Gamma_\beta^G(\tilde{\alpha}) \equiv \{\gamma_1,...,\gamma_m\}$ *in (19) and let* $\tilde{\mu}_1,...,\tilde{\mu}_m$ *be a set of Lagrange multipliers associated with the constraints* $t \geq \alpha^T G(\gamma_i)\alpha,\ \ 1 \leq i \leq m$ *which optimize the dual problem of (Q). In addition, let* $\tilde{\mu}_0$ *be a Lagrange multiplier associated with the constraint* $t \geq \alpha^T\alpha$ *in the dual problem of (Q). If* $\tilde{p}$ *is the discrete probability measure defined by* $\tilde{p}(\gamma_i) := \tilde{\mu}_i \ \ i = 1,...,m$ *and* $\tilde{\tau} := \tilde{\mu}_0$, *then* $\tilde{\alpha}$, $\tilde{\tau}$, *and* $\tilde{p}$ *solve the problem (18). In addition, there exist some solution* $\tilde{\alpha}^*$, $\tilde{\tau}^*$, *and* $\tilde{p}^*$ *such that* $\tilde{p}^*$ *contains at most* $l+1$ *nonzero atoms*.

$$P_\tau(\Gamma) := \begin{array}{ll} \min_{\alpha,t} & t \\ s.t. & t \geq \alpha^T G(\gamma)\alpha \;\; \textit{for all} \;\; \gamma \in \Gamma \\ & t \geq \alpha^T \alpha \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array} \tag{19}$$

Proof Let $\omega_0 \notin \Gamma_\beta$ and assign the kernel $\delta(x,z)$ to this point. The corollary is proved by applying theorem 12 to the sets Γ_β and $\{\omega_0\}$. ■

4.7 Furnishing the Class of Admissible Kernels with Isotropic Gaussian Kernels

Although the class of translation invariant kernels encompasses the class of Gaussian kernels with arbitrary covariance matrices, the stabilized class of translation invariant kernels $\mathcal{K}_\beta$ does not. So, it may be advantageous to combine the class $\mathcal{K}_\beta$ with the class of Gaussian kernels. In this section, we consider learning the best convex combination of kernels of the classes $\mathcal{K}_\beta$ and the stabilized class of isotropic Gaussian kernels

$$\mathcal{K}_\eta := \left\{ k(x,z) = \int_{\mathbb{R}} e^{-\omega\|x_u - x_v\|^2} e^{-\eta\|\omega\|^2} \, dp(\omega) \quad : \text{ p is a probablity measure on } \mathbb{R} \right\}$$

where $\eta > 0$. We also learn the parameter τ automatically. The proof that there exists some compact set $\Omega_\eta \subseteq R$ where we can confine the integration to it parallels the discussion of Section 4.3 and is omitted. By using Theorem 12, it follows that the expansion of the optimal kernel along with the weight of each kernel can be obtained by solving the SIP problem $P_\tau(\Gamma_\beta, \Omega_\eta)$, where $P_\tau(\Gamma, \Omega)$ is defined as:

$$P_\tau(\Gamma,\Omega) := \begin{array}{ll} \min_{\alpha,t} & t \\ s.t. & t \geq \alpha^T G(\gamma)\alpha \;\; \text{for all} \;\; \gamma \in \Gamma \\ & t \geq \sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u \alpha_v y_u y_v e^{-\omega\|x-z\|^2} e^{-\eta\|\omega\|^2} \;\; \text{for all} \;\; \omega \in \Omega \\ & t \geq \alpha^T \alpha \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array}$$

5. Optimization Algorithm

We now turn to the problem of numerically solving the nonlinear convex semi-infinite programming problem $P_\tau(\Gamma_\beta)$.[7] The term semi-infinite stems from the fact that whilst the number of variables is

7. Modifying the proposed algorithm to solve problem $P_\tau(\Gamma, \Omega)$ is straightforward.

finite, there is an infinite number of constraints which are indexed by the compact set Γ_β. Hopefully, for each finite set $\Gamma \subseteq \Gamma_\beta$, the problem $P_\tau(\Gamma)$ is QCQP and therefore convex. Hence, in principle, one can construct a sequence of QCQP problems with an increasing number of constraints such that their solutions converge to the solution of problem $P_\tau(\Gamma_\beta)$. This is the principle used by discretization and exchange algorithms (see Hettich and Kortanek, 1993; Reemtsen and Görner, 1998). On the other hand, by Corollary 13, only a finite number of constraints will be active in a solution. Furthermore, it is easy to show that this property is not limited to a solution point, and the active constraints at a solution also identify the active constraints in a neighborhood of it. This is the principle behind the methods based on local reduction (see Reemtsen and Görner, 1998; Hettich and Kortanek, 1993). Reemtsen and Görner (1998) proposed that to further speed up the methods based on local reduction, the set of active constraints be locally adapted. Combining these ideas with numerous experiments, we arrived at Algorithm 1. This algorithm is very similar to Algorithm 7 in Reemtsen and Görner (1998) which is based on local reduction.

5.1 Choosing the Initial Value of α

We choose the initial value of α such that maximizing the criterion (3) with respect to the kernel function k and the parameter τ correspond to maximizing the distance of the means of the two classes in a feature space. Let us first write the distance between means of two classes in the feature space of some kernel k'

$$\|m_1 - m_2\|^2 = \left\| \frac{1}{l_1} \sum_{u \in C^+} \Phi(x_u) - \frac{1}{l_2} \sum_{v \in C^-} \Phi(x_v) \right\|^2$$

$$= \frac{1}{l_1^2} \sum_{u \in C^+} \sum_{v \in C^-} k'(x_u, x_v) + 2\frac{1}{l_1 l_2} \sum_{u \in C^+} \sum_{v \in C^-} k'(x_u, x_v) + \frac{1}{l_2^2} \sum_{u \in C^-} \sum_{v \in C^-} k'(x_u, x_v).$$

By choosing $\alpha_i = 1/l_1$ for $i \in C^+$ and $\alpha_i = 1/l_2$ for $i \in C^-$, we have

$$\|m_1 - m_2\|^2 = \sum_{u=1}^{l} \sum_{v=1}^{l} \alpha_u \alpha_v y_u y_v k'(x_u, x_v).$$

Comparing the above equation with (3), we see that maximizing the criterion (3) with respect to the kernel function k and the parameter τ is equivalent to maximizing the distance between means of the samples of the two classes in the feature space of kernel $k'(x,z) = k(x,z) + \tau\delta(x,z)$. Note that this choice for α also satisfies the required conditions $\alpha^T e = 2$, $\alpha^T y = 0$, and $\max\{\frac{1}{l_1}, \frac{1}{l_2}\} \leq \alpha \leq C$; and thus $\alpha \in \mathcal{A}$.

5.2 Global Search for Local Maxima of $\alpha^T G(\gamma)\alpha$

The algorithm presented in this section attempts to gather a subset of unsatisfied constraints to be considered in the next iteration of Algorithm 1. Although at the solution point $\tilde{\alpha}$ the set of active constraints globally maximize $\tilde{\alpha}^T G(\gamma)\tilde{\alpha}$, for other choices of α it is possible that the constraints be violated by the local maxima of the function $\alpha^T G(\gamma)\alpha$. So, in Algorithm 2, we try to find the

values of γ which locally maximize the function $\alpha^T G(\gamma)\alpha$ for given α. Here we also assume that the function $G_\beta(\|\gamma\|)$ is differentiable with respect to γ. The choice of limited-memory BFGS algorithm (Nocedal, 1980; Liu and Nocedal, 1989; Nocedal and Wright, 2006) for this optimization is very important. For large-scale problems with large n, the memory needed to store the Hessian matrix and the associated computations become prohibitive. Although a gradient-ascent algorithm does not compute the Hessian matrix, its convergence rate is very slow. The limited-memory BFGS algorithm provides an excellent practical compromise between the computations at each step and the number of iterations till convergence, without storing the full Hessian matrix in memory.

Algorithm 1 General Optimization Algorithm

Require: T_1

1. $\Gamma^{(0)} \leftarrow \{\}$, $t^{(0)} \leftarrow \frac{1}{l_1} + \frac{1}{l_2}$
   {A lower bound for parameter t is the minimum value of $\alpha^T\alpha$ for $\alpha \in \mathcal{A}$}
2. Initialize $\alpha^{(0)}$ as described in Section 5.1
3. **for** $i = 1, 2, \ldots$ **do**
4. set R such that for all γ with $\|\gamma\| > R$, the relation $\alpha^T G(\gamma)\alpha < t^{(i-1)}$ holds for all α
5. $\Gamma_g^{(i)} \leftarrow GlobalSearchForLocals(\alpha^{(i-1)}, R)$
   {denote the maximum value obtained by the global search by $s^{(i)}$}
6. $\Gamma_s^{(i)} \leftarrow \Gamma^{(i-1)} \bigcup \Gamma_g^{(i)}$
7. Solve problem $P_\tau(\Gamma_s^{(i)})$ to obtain the optimal parameters $t_s^{(i)}, \alpha_s^{(i)}$ and $\mu_s^{(i)}$
   {see Section 5.3}
8. Locally adapt $\Gamma_s^{(i)}$ and $\mu_s^{(i)}$ to obtain the optimal parameters $t_l^{(i)}, \alpha_l^{(i)}, \Gamma_l^{(i)}$ and $\mu_l^{(i)}$
   {see Section 5.4}
9. $t^{(i)} \leftarrow t_l^{(i)}$, $\alpha^{(i)} \leftarrow \alpha_l^{(i)}$
10. Construct $\mu^{(i)}$ and $\Gamma^{(i)}$ by eliminating zero indices of $\mu_l^{(i)}$ along with the corresponding vectors in $\Gamma_l^{(i)}$
11. **if** $t^{(i)} - t^{(i-1)} < \varepsilon t^{(i-1)}$ or $i = T_1$ **then**
12. terminate algorithm with the kernel $k(x,z) := \sum_{j=1}^m \mu_j^{(i)} \cos\left((x-z)^T \gamma_j^{(i)}\right)$
    {we assume that $\Gamma^{(i)} = \left\{\gamma_1^{(i)}, \ldots, \gamma_m^{(i)}\right\}$ and that $\mu_j^{(i)}$ is the Lagrange multiplier associated with the constraint $\alpha^T G(\gamma_j^{(i)})\alpha \le t$ in problem $P_\tau(\Gamma^{(i)})$}
13. **end if**
14. **end for**

5.3 Solving the Problem $P_\tau(\Gamma)$ for Finite Set Γ

Let $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ be a finite subset of Γ_β. Since Γ is finite, the problem $P_\tau(\Gamma)$ can be written as the following QCQP problem:

Algorithm 2 GlobalSearchForLocals

Require: α, R, T_2, and T_3

1. $\Gamma = \{\}, i = 0$
2. **for** $j = 1, 2, \ldots$ **do**
3. generate a random point $x \in \mathbb{R}^n$
4. generate a random number $r \in [0, R]$
5. $\gamma_0 \leftarrow r \frac{x}{\|x\|}$
6. starting from γ_0 and using the limited-memory BFGS algorithm find a local maximum $\gamma^{(i)}$ for function $\alpha^T G(\gamma)\alpha$
7. **if** $\gamma^{(i)} \in \Gamma$ **then**
8. $i \leftarrow i + 1$ {count the number of repeating local maxima}
9. **end if**
10. $\Gamma \leftarrow \Gamma \cup \gamma^{(i)}$
11. **if** $(i - |\Gamma|) \geq T_2$ or $j = T_3$ **then**
12. **return** Γ
13. **end if**
14. **end for**

$$\begin{array}{ll} \min_{\alpha,t} & t \\ s.t. & t \geq \alpha^T G(\gamma_i)\alpha \quad \text{for all} \quad i = 1, \ldots, m \\ & t \geq \alpha^T \alpha \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array} \tag{20}$$

This problem has been studied in Section 4.6 of Lanckriet et al. (2004) and it has been suggested to store the $l \times l$ kernel matrices $G(\gamma_1), \ldots, G(\gamma_m)$ in memory and solve the problem with general purpose software packages. But, the memory requirement of this approach limits its applicability to small-sized problems. However, the facts that

$$\alpha^T \Im\{G(\gamma)\}\alpha = 0$$

and

$$\begin{aligned} \Re\{G(\gamma)\} &= v_c(\gamma)^T v_c(\gamma) + v_s(\gamma)^T v_s(\gamma), \\ v_c(\gamma) &:= \left[y_1 cos(\gamma^T x_1),\ y_2 cos(\gamma^T x_2),\ \ldots\ , y_l cos(\gamma^T x_l)\right], \\ v_s(\gamma) &:= \left[y_1 sin(\gamma^T x_1),\ y_2 sin(\gamma^T x_2),\ \ldots\ , y_l sin(\gamma^T x_l)\right] \end{aligned}$$

where $\Re\{z\}$ and $\Im\{z\}$ are the real and imaginary parts of z, respectively, show that the kernel matrices $G(\gamma_1),...,G(\gamma_m)$ appearing in problem (20) are effectively[8] of rank two. This allows us to reformulate (20) as a new QCQP problem as follows:

$$\begin{array}{ll} \min_{\alpha,t,c,s} & t \\ s.t. & t \geq c_i^2 + s_i^2 \quad i = 1,...,m \\ & c_i = \sum_{u=1}^{l} \alpha_i y_i \cos(\gamma_i^T x_u) \quad i = 1,...,m \\ & s_i = \sum_{u=1}^{l} \alpha_i y_i \sin(\gamma_i^T x_u) \quad i = 1,...,m \\ & t \geq \alpha^T \alpha \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{array} \tag{21}$$

Now, there is no need to load the kernel matrices into memory and so general-purpose QCQP solvers such as Mosek (Andersen and Andersen, 2000) can be used to solve (21) even when the training set size is huge.

5.4 Local Adaptation

As stated in the previous section, for any finite set $\Gamma = \{\gamma_1,...,\gamma_m\} \subseteq \Gamma_\beta$, the problem $P_\tau(\Gamma)$ is convex and so every local solution is also globally optimal. But, if we consider the values $\gamma_1,...,\gamma_m$ as points in the space $\mathbb{R}^n$, we get the following non-convex optimization problem:

$$\max_{\substack{\mu \geq 0, \mu^T e = 1, \\ \gamma_1,...,\gamma_m \in \mathbb{R}^n}} \min_{\alpha \in \mathcal{A}} \sum_{i=1}^{m} \mu_i \alpha^T G(\gamma_i) \alpha + \mu_0 \alpha^T \alpha. \tag{22}$$

Now, we can use the the solution of the problem $P_\tau(\Gamma)$ obtained in the previous section as the starting point for problem (22) and locally improve it by an ascent method. By unrolling (22), we obtain the following optimization problem:

$$\max_{\substack{\mu \in \mathbb{R}^{m+1}, \\ \gamma_1,...,\gamma_m \in \mathbb{R}^n}} \hat{f}(\mu, \gamma_1,...,\gamma_m) \quad s.t.\ \mu \geq 0, \mu^T e = 1 \tag{23}$$

where

8. In this paper, we say that a matrix G is effectively of rank r if there exists some matrix H of rank r such that for all vectors $\alpha \in \mathcal{A}$ we have $\alpha^T G \alpha = \alpha^T H \alpha$.

$\hat{J}(\mu,\gamma_1,...,\gamma_m) :=$

$$\min_{\alpha\in\mathcal{A}} \left\{ \sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u\alpha_v y_u y_v \sum_{i=1}^{m} \mu_i G_\beta\left(\|\gamma_i\|\right)\cos(\gamma_i^T(x_u-x_v)) + \mu_0 \sum_{u=1}^{l}\alpha_u^2 \right\}. \tag{24}$$

Problem (23), corresponds to adapting the kernel parameters $\gamma_1,...,\gamma_m$ and $\mu_0,...,\mu_m$ of the kernel function $k(x,z) = \sum_{i=1}^{m}\mu_i\cos(\gamma_i^T(x-z)) + \mu_0\delta(x-z)$ for the task of SVM classification, where δ denotes the Kronecker delta function. This problem has been previously studied by Chapelle et al. (2002) for general kernel functions with unconstrained parameters and they proved that the function $\hat{J}(.)$ is differentiable provided that problem (24) has a unique solution.[9] They also proposed a simple gradient-based iterative algorithm for adapting the kernel parameters. Recently, Rakotomamonjy et al. (2008) performed a more detailed analysis of this problem in MKL and proposed a reduced gradient algorithm with line search. They reasoned that since the computation of the function $\hat{J}$ is costly,[10] the overhead of a line search preserves the effort.

To avoid the difficulties of the constrained optimization, we replace the constrained vector μ by the unconstrained vector ρ, connected by the relation $\mu_i = \frac{\rho_i^2}{\rho^T\rho}$, $i = 0,...m$, and rewrite (23) as the following problem:

$$\max_{\substack{\rho\in\mathbb{R}^{m+1},\\ \gamma_1,...,\gamma_m\in\mathbb{R}^n}} \quad J(\rho,\gamma_1,...,\gamma_m) \tag{25}$$

where

$J(\rho,\gamma_1,...,\gamma_m) :=$

$$\min_{\alpha\in\mathcal{A}} \frac{1}{\rho^T\rho} \left\{ \sum_{u=1}^{l}\sum_{v=1}^{l} \alpha_u\alpha_v y_u y_v \sum_{i=1}^{m} \rho_i^2 G_\beta\left(\|\gamma_i\|\right)\cos(\gamma_i^T(x_u-x_v)) + \rho_0^2 \sum_{u=1}^{l}\alpha_u^2 \right\}. \tag{26}$$

We use the limited-memory BFGS algorithm to numerically solve (25). Our experiments on MKL tasks show that the method proposed in this section is several times fatser than the reduced gradient algorithm of Rakotomamonjy et al. (2008).[11] It has been also stated by Rakotomamonjy et al. (2008) that their method could be improved if the Hessian matrix could be computed efficiently. This is not the case for the problem (25) with $m\times(n+1)$ variables; where, even for moderate size problems, the storage of the Hessian matrix requires lots of memory.

5.5 Solving the Intermediate SVM Problem and Its Gradient

To compute the function $J(.)$ defined by Equation (26), we have to solve the following constrained quadratic programming problem:

9. Truely speaking, the proof should be credited to Danskin (1966).
10. Although the definition of the function J in Rakotomamonjy et al. (2008) differs from (24), computation of both functions corresponds to training a single-kernel SVM.
11. We leave this comparison along with some theoretical results to another paper.

$$\begin{aligned} \min_{\alpha} \quad & \alpha^T \left(\sum_{i=1}^{m} \frac{\rho_i^2}{\rho^T \rho} G(\gamma_i) \right) \alpha + \frac{\rho_0^2}{\rho^T \rho} \alpha^T \alpha \\ s.t. \quad & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{aligned} \tag{27}$$

Although the traditional algorithms for solving quadratic programming problems, such as active set methods, are fast, they need to store the kernel matrix in memory which prevents their application in large-scale problems. So, various algorithms for large-scale training of SVMs, such as SMO (Platt, 1999) or SVM^{Light} (Joachims, 1999), have been proposed. Again, since the effective rank of the kernels $G(\gamma_1), ..., G(\gamma_m)$ is two, we can re-state the problem (27) in a memory efficient manner as:

$$\begin{aligned} \min_{\alpha,c,s} \quad & \frac{\rho_0^2}{\rho^T \rho} \alpha^T \alpha + \sum_{i=1}^{m} \frac{\rho_i^2}{\rho^T \rho} \left(c_i^2 + s_i^2 \right) \\ s.t. \quad & c_i = \sum_{u=1}^{l} \alpha_u y_u \cos(\gamma_i^T x_u) \quad i = 1, ..., m \\ & s_i = \sum_{u=1}^{l} \alpha_u y_u \sin(\gamma_i^T x_u) \quad i = 1, ..., m \\ & 0 \leq \alpha \leq C \\ & \alpha^T y = 0 \\ & \alpha^T e = 2. \end{aligned}$$

In our experiments we have used the optimization software Mosek (Andersen and Andersen, 2000) to solve this problem. After computing the value of the function $J(.)$ and obtaining a solution $\tilde{\alpha}$ to (26), we compute the gradient using the following formulas:

$$\nabla_{\gamma_j} J = \frac{\rho_j^2}{\rho^T \rho} \nabla_{\gamma} \left\{ \tilde{\alpha}^T G(\gamma) \tilde{\alpha} \right\} |_{\gamma = \gamma_j} \quad j = 1, ..., m, \tag{28}$$

$$\nabla_{\rho_j} J = 2 \frac{\rho_j}{\rho^T \rho} \left\{ \tilde{\alpha}^T G(\gamma_j) \tilde{\alpha} - \sum_{i=1}^{m} \frac{\rho_i^2}{(\rho^T \rho)} \tilde{\alpha}^T G(\gamma_i) \tilde{\alpha} - \frac{\rho_0^2}{(\rho^T \rho)} \tilde{\alpha}^T \tilde{\alpha} \right\} \quad j = 1, ..., m. \tag{29}$$

Note that, in general, the computational complexity of computing formulas (28) and (29) is $O(m \times n \times nsv^2)$ as was pointed out by Rakotomamonjy et al. (2008).[12] The following formulas show an $O(m \times (nsv + n))$ method for computing the gradient of function $J(.)$.

$$\begin{aligned} \nabla_{\gamma_j} J &= \frac{\rho_j^2}{\rho^T \rho} \left(c_j^2 + s_j^2 \right) \nabla_{\gamma} G_{\beta}(\|\gamma\|)|_{\gamma = \gamma_j} + 2 \frac{\rho_j^2}{\rho^T \rho} \left(s'_j s_j + c'_j c_j \right) G_{\beta}(\|\gamma_j\|) \quad j = 1, ..., m, \\ \nabla_{\rho_j} J &= 2 \frac{\rho_j}{\rho^T \rho} \left\{ (c_j^2 + s_j^2) G_{\beta}(\|\gamma_j\|) - \sum_{i=1}^{m} \frac{\rho_i^2}{\rho^T \rho} (c_j^2 + s_j^2) G_{\beta}(\|\gamma_j\|) - \frac{\rho_0^2}{\rho^T \rho} \tilde{\alpha}^T \tilde{\alpha} \right\} \quad j = 1, ..., m \end{aligned}$$

where

12. Note that in Rakotomamonjy et al. (2008) the function $J(.)$ has only m variables, while here the number of variables is $m \times (n+1)$.

$$c_j = \sum_{u=1}^{nsv} \tilde{\alpha}_u y_u \cos(\gamma_j^T x_u) \quad j = 1, ..., m,$$
$$s_j = \sum_{u=1}^{nsv} \tilde{\alpha}_u y_u \sin(\gamma_j^T x_u) \quad j = 1, ..., m,$$
$$c'_j = -\sum_{u=1}^{nsv} \tilde{\alpha}_u y_u x_u \sin(\gamma_j^T x_u) \quad j = 1, ..., m,$$
$$s'_j = \sum_{u=1}^{nsv} \tilde{\alpha}_u y_u x_u \cos(\gamma_j^T x_u) \quad j = 1, ..., m.$$

5.6 Convergence Analysis

In this section, we study the convergence properties of Algorithm 1. We hope the contents of this section help the reader to get a better feeling of this algorithm. For any finite or infinite set Γ, we denote the solution of problem $P_\tau(\Gamma)$ by $S(\Gamma)$. Let us first prove a useful lemma.

Lemma 14 *If the loop inside Algorithm 1 is executed for the i'th iteration, then $S(\Gamma_l^{(i)}) = S(\Gamma^{(i)})$. In other words, removing the constraints where their associated Lagrange multipliers are zero, does not change $S(\Gamma_l^{(i)})$.*

Proof Assume that $\Gamma_l^{(i)} = \{\gamma_1, ..., \gamma_m\}$. Without loss of generality, we assume that $\Gamma^{(i)} = \{\gamma_1, ..., \gamma_{m'}\}$, where $m' < m$. Denote the Lagrangian of $P_\tau(\Gamma_l^{(i)})$ by $\mathcal{L}(\alpha, t, \mu, \lambda)$, where $\mu_1, ..., \mu_m$ are the Lagrange multipliers associated with the constraints $\alpha^T G(\gamma_1)\alpha \leq t$, ..., $\alpha^T G(\gamma_m)\alpha \leq t$, respectively, and $\lambda \in \Lambda$ denotes the Lagrange multipliers associated with all other constraints. Since $P_\tau(\Gamma_l^{(i)})$ is convex, by the strong duality we have

$$S(\Gamma_l^{(i)}) = \max_{\mu \geq 0, \lambda \in \Lambda} \min_{\alpha, t} \mathcal{L}(\alpha, t, \lambda, \mu) = \mathcal{L}(\alpha^*, t^*, \lambda^*, \mu^*)$$

where it is assumed that α^*, t^*, λ^*, and μ^* are a solution to problem $S(\Gamma_l^{(i)})$. Since $\mu^*_{m'+1}, ..., \mu^*_m$ are zero, we also have

$$S(\Gamma_l^{(i)}) = \max_{\mu_1 \geq 0, ..., \mu_{m'} \geq 0, \lambda \in \Lambda} \min_{\alpha, t} \mathcal{L}(\alpha, t, \lambda, \mu).$$

Since the strong duality also holds for $P_\tau(\Gamma^{(i)})$, the last expression is equal to $S(\Gamma^{(i)})$ and the lemma follows. ■

Now, we prove that for any $\varepsilon > 0$ the Algorithm 1 converges, even without limiting the maximum number of iterations.

Proposition 15 *The sequence of numbers $t^{(0)}, t^{(1)}, ...$ generated by Algorithm 1 is increasing and bounded.*

Proof By steps 6 and 8 of Algorithm 1, it follows that $S(\Gamma^{(i-1)}) \leq S(\Gamma_s^{(i)})$ and $S(\Gamma_s^{(i)}) \leq S(\Gamma_l^{(i)})$. By Lemma 14, $S(\Gamma_l^{(i)}) = S(\Gamma^{(i)})$. So, $t^{(i-1)} = S(\Gamma^{(i-1)}) \leq S(\Gamma^{(i)}) = t^{(i)}$. By Equation (5) and the fact that $\alpha^T e = 2$, it follows that $\alpha^T \alpha \leq 4$ and $\alpha^T G(\gamma)\alpha \leq 4$ for all choices of α and γ; and thus, the sequence is bounded. ■

Define $g(\alpha,\gamma) := \alpha^T G(\gamma)\alpha$. The following theorem is essentially Theorem 7.2 of Hettich and Kortanek (1993), where its proof is reconstructed here for the sake of completeness.

Theorem 16 *Assume that in every run of step 5 of Algorithm 1 at least one global maximizer of the function $g(\alpha^{(i-1)},\gamma)$ is found and that the steps 10-13 of the algorithm are omitted (Note that the key point is the omission of step 10). Let $\bar{\alpha}$ be any accumulation point of the sequence $\alpha^{(0)},\alpha^{(1)},...$ and assume that $t^{(i)} \nearrow \bar{t}$. Then the pair $(\bar{\alpha},\bar{t})$ is a solution of $P_\tau(\Gamma_\beta)$.*

Proof First note that since $\alpha^{(i)} \in \mathcal{A}$ and $\mathcal{A}$ is compact, a point of accumulation for the sequence $\alpha^{(0)},\alpha^{(1)},...$ always exists. Recall the definition of Ω_β from Section 4.6 and define the function $g(\alpha)$ as:

$$g(\alpha) := \max_{\gamma \in \Omega_\beta} \alpha^T G(\gamma)\alpha$$

For simplicity, assume that $\alpha^{(i)} \to \bar{\alpha}$. Let (α^*,t^*) denote a solution of problem $P_\tau(\Gamma_\beta)$. Clearly $\bar{t} \leq t^*$. If $\bar{t} = t^*$ then the theorem is proved. Assume on the contrary that $\bar{t} < t^*$. Then, there exists $\bar{\gamma} \in \Omega_\beta$ such that $\bar{t} < g(\bar{\alpha},\bar{\gamma}) = g(\bar{\alpha})$. But, since $\bar{\alpha}^T\bar{\alpha} \leq \bar{t}$, it follows that $\bar{\gamma} \in \Gamma_\beta$. For $i = 0,1,...$ choose $\gamma^{(i)} \in \Omega_\beta$ such that $g(\alpha^{(i)},\gamma^{(i)}) = g(\alpha^{(i)})$. We have

$$g(\bar{\alpha}) - \bar{t} = \left[g(\alpha^{(i)}) - \bar{t}\right] + \left[g(\bar{\alpha}) - g(\alpha^{(i)})\right] = \left[g(\alpha^{(i)},\gamma^{(i)}) - \bar{t}\right] + \left[g(\bar{\alpha}) - g(\alpha^{(i)})\right] \tag{30}$$

On the other hand, by omission of step 10 from Algorithm 1, all constraints of the previous iterations will continue to appear in the next iterations. Since $g(\alpha,\gamma)$ is continuous, $\bar{\alpha}$ is a feasible point for all problems $P_\tau(\Gamma^{(i)})$, where $i = 0,1,...,\infty$. Therefore,

$$g(\bar{\alpha},\gamma^{(i)}) \geq \sup_{j=1,...,\infty} t^{(j)} = \bar{t} \quad \text{for all } i = 0,1,... \tag{31}$$

Using (31) in (30) we obtain

$$\begin{aligned} g(\bar{\alpha}) - \bar{t} &= \left[g(\alpha^{(i)},\gamma^{(i)}) - \bar{t}\right] + \left[g(\bar{\alpha}) - g(\alpha^{(i)})\right] \\ &\leq \left[g(\alpha^{(i)},\gamma^{(i)}) - g(\bar{\alpha},\gamma^{(i)})\right] + \left[g(\bar{\alpha}) - g(\alpha^{(i)})\right]. \end{aligned} \tag{32}$$

By continuity of $g(.,.)$, the right hand side of (32) tends to zero, which contradicts the assumption $\bar{t} < g(\bar{\alpha})$. ■

6. Generalizing the Kernel Trick to Complex-valued Kernels

Consider a machine learning algorithm designed for a real-valued Euclidean space. The kernel trick for real-valued kernels states that if all geometric concepts of an algorithm are defined solely based on the dot-product operation, then by replacing all of these dot-products by a kernel function k, we arrive at a version of the very algorithm running in a feature space associated with kernel k. For complex-valued kernels, the dot-product of any two vectors may be complex-valued which makes the application of these kernels to machine learning algorithms more tricky. We now introduce a generalization of the kernel trick for complex-valued kernels. Assume that $k(x,z)$ is a complex-valued kernel. Then there exists at least one complex feature space, say F, and a mapping $\Phi : X \to F$ such that $k(x,z) = \langle \Phi(x), \Phi(z) \rangle_F$. Each axis in the complex feature space F can be substituted by two real-valued axes, one representing the real part and the other the imaginary part. Let us call this real-valued space G. To use the kernel trick, we replace the complex feature space F with the equivalent real feature space G. Now, we show that the dot product between elements of G can be computed by the real-valued kernel function $\Re\{k(x,z)\}$,[13] where $\Re\{z\}$ is the real part of z.

Theorem 17 *Let F be a complex Hilbert space of dimension N (possibly infinite) and G the corresponding $2N$-dimensional Hilbert space obtained by representing real and imaginary parts of F in separate real axes. Then $\langle x', z' \rangle_G = \Re\{\langle x, z \rangle_F\}$, where x' is the $2N$-dimensional vector obtained by concatenating real and imaginary parts of x.*

Proof For finite N we have

$$\begin{aligned}\Re\{\langle x,z\rangle_F\} &= \Re \sum_{i=1}^{N} x_i \bar{z}_i = \Re \sum_{i=1}^{N} (x_i^{re} + j x_i^{im})(z_i^{re} - j z_i^{im}) \\ &= \sum_{i=1}^{N} (x_i^{re} z_i^{re} + x_i^{im} z_i^{im}) = \sum_{i=1}^{2N} x'_i z'_i = \langle x', z' \rangle_G\end{aligned}$$

If F is infinite dimensional, then it has an orthonormal basis (see Kreyszig, 1989 p.168) and x and z can have at most countably many nonzero elements (see Kreyszig, 1989 p.165) which we indicate by index set $\mathfrak{I}$. So,

$$\begin{aligned}\Re\{\langle x,z\rangle_F\} &= \Re \sum_{i\in\mathfrak{I}} x_i \bar{z}_i = \Re \sum_{i\in\mathfrak{I}} (x_i^{re} + j x_i^{im})(z_i^{re} - j z_i^{im}) \\ &= \sum_{i\in\mathfrak{I}} (x_i^{re} z_i^{re} + x_i^{im} z_i^{im}) = \sum_{i\in\mathfrak{I}} x'_i z'_i = \langle x', z' \rangle_G.\end{aligned}$$

■

After fully developing the paper based on the complex-valued form of translation invariant kernels, one of the reviewers introduced us to the real-valued form of these kernels as was discovered by Bochner (1955). He proved that every continuous real-valued translation invariant positive definite kernel in $\mathbb{R}^n$ has the general form

13. The fact that real part of a complex kernel is a real kernel is not new (see Schölkopf and Smola, 2002 page 31). But, as far as we know, the relation between the corresponding Hilbert spaces, as stated in the theorem, is new.

$$k(x,z) = \tilde{k}(x-z) = \int_{\mathbb{R}^n} cos\gamma^T (x-z)\, dV(\gamma).$$

It is interesting that after applying the appropriate kernel trick to both real-valued and complex-valued forms of translation invariant kernels, the optimal kernel is found to be a mixture of cosines.

7. The Method at Runtime

One frequently denounced feature of SVMs is that the resulting classifier has an expansion based on support vectors. Although support vectors are considered to be sparse in the training set, the resulting classifier is usually slower than other competing methods such as neural networks (Schölkopf et al., 1998). In general, the computation of a support vector classifier requires $O(n \times nsv)$ steps. The problem becomes more severe when the kernel function becomes a combination of several kernels, where the computational complexity of evaluating the classifier grows up to $O(m \times n \times nsv)$. For some kernels, such as the Gaussian kernel with isotropic covariance matrix, the computation time can be reduced to $O((m+n) \times nsv)$. Our method has the eminent property that the resulting classifier is not expanded based on support vectors at all. Considering the SVM classifier $f(x) = \sum_{u=1}^{nsv} \alpha_u y_u k(x, x_u) + b$, we have

$$f(x) = \sum_{u=1}^{nsv} \alpha_u y_u k(x,x_u) + b = \sum_{u=1}^{nsv} \alpha_u y_u \left(\sum_{i=1}^{m} \mu_i \cos(\gamma_i^T (x - x_u)) G_\beta(\|\gamma_i\|) \right) + b$$

$$= \sum_{i=1}^{m} \mu_i G_\beta(\|\gamma_i\|) \sum_{u=1}^{nsv} \alpha_u y_u \left(\cos(\gamma_i^T x)\cos(\gamma_i^T x_u) + \sin(\gamma_i^T x)\sin(\gamma_i^T x_u) \right) + b$$

$$= \sum_{i=1}^{m} \mu_i G_\beta(\|\gamma_i\|) \left[\left(\sum_{u=1}^{nsv} \alpha_u y_u \cos(\gamma_i^T x_u) \right) \cos(\gamma_i^T x) + \left(\sum_{u=1}^{nsv} \alpha_u y_u \sin(\gamma_i^T x_u) \right) \sin(\gamma_i^T x) \right] + b.$$

But $\sum_{u=1}^{nsv} \alpha_u y_u \cos(\gamma_i^T x_u)$ and $\sum_{u=1}^{nsv} \alpha_u y_u \sin(\gamma_i^T x_u)$ are constant values. So, the computational complexity of evaluating the classifier of the proposed method is $O(m \times n)$. Note that the classifier has an expansion based on the number of kernels, instead of support vectors. In addition, by Theorem 2, the number of kernels is limited to $l+1$. Furthermore, since the deletion of non-support vector samples from the training set has no effect on the optimal classifier, it follows that $m \leq nsv + 1$. Although, theoretically, the number of kernels can reach the number of support vectors, our experiments show that the number of kernels is usually a fraction of the number of support vectors.

8. A Learning Theory Perspective

A common feature between the class of radial kernels, considered by Micchelli and Pontil (2005), and the class of translation invariant kernels, considered here, is that the kernels of both classes

have the property that $k(x,x) = 1$ for every $x \in \mathbb{R}^n$.[14] Micchelli et al. (2005b) used this feature along with a result from Yiming and Zhou (2007) to obtain a probably approximately correct (PAC) upper bound on the generalization error of their kernel learning framework over the class of radial kernel functions. They concluded that the regularization parameter of a single-kernel learning machine is sufficient for controlling the complexity of the class of radial kernels, rejecting the use of an auxiliary method for controlling the complexity of the class of radial kernels.

But, the situation for translation invariant kernels is completely different. It is well-known that the VC-dimension of the class of cosine functions with arbitrary frequencies is infinite (see Vapnik, 1998, page 160). In addition, the finiteness of the VC-dimension is a necessary and sufficient condition for distribution independent learning of binary classification tasks (see Vapnik, 1998, Theorem 4.5). So, controlling the complexity of the class of translation invariant kernels is a necessary ingredient of our framework. This discussion will be experimentally verified in Section 9, where we will show the vital role of the complexity control mechanism of Section 2.

9. Experimental Results

In this section we report the results of our experiments on several artificial and real-world benchmark data sets. In addition, we will experimentally investigate the role of the complexity control mechanism of Section 2. In all the experiments we have set $C = \infty$, $T_1 = 1000$, $T_2 = 4$, $T_3 = 500$, $G_\beta(\|\gamma\|_2) = \exp(-\beta\|\gamma\|_2^2))$ and the parameter τ is automatically learnt according to Algorithm 1. The implementation of this paper is packaged in the SIKL (Stabilized infinite kernel learning) toolbox and is available at *http://www.mloss.org*. We obtained the implementation of the limited memory BFGS algorithm from the website *http://www.chokkan.org/software/liblbfgs* which is a C++ translation of the original implementation made available by Nocedal in Fortran 77. For limited-memory BFGS algorithm, the 17 most recent curvature information are used and the maximum number of line-search tries is set to 20. We also changed the stopping condition of the algorithm from $\frac{\|\nabla x\|}{\|x\|} < \varepsilon$ to $\|\nabla x\| < \varepsilon$ to avoid the degradation of the accuracy of the global search algorithm for points far from the origin. The QCQP sub-problem of Algorithm 1 and the QP problem of Section 5.5 are solved by the optimization software Mosek (Andersen and Andersen, 2000). All the experiments have been performed on a 2.8GHz Pentium D computer with 2GB memory and running the Linux operating system.

9.1 Experiments on Small-size Benchmark Data Sets

In this section, we report our experiments on the benchmark data sets prepared by Rätsch et al. (2001). This benchmark consists of 13 data sets and there exist 100 splits of each data set into training and test sets. The classification error for each data set is obtained by averaging the classification error over these splits. For this experiment we set $\varepsilon = 0.001$. The comparison is among the following methods:

- **Single Gaussian (SG)** Rätsch et al. (2001) performed experiments with a single isotropic Gaussian kernel. The variance parameter σ of the isotropic Gaussian kernel and the parameter C of the 1-norm soft-margin SVM are optimized by performing 5-fold cross-validation on the first five instances of the training set.

14. In fact, the classes of radial/translation invariant kernel functions contain kernels with arbitrary positive values for $k(x,x)$. But, the constraint $k(x,x) = 1$ is imposed for reasons stated in Section 3.

- **Gaussian Mixture (GM)** A generalization of the method of Gehler and Nowozin (2008) is implemented and used for learning the optimal kernel over the class $\mathcal{K}_\eta$. The number of Gaussian kernels $\bar{m}$, their parameters, and the parameter τ are learnt automatically. The parameter η is set to 0.001 and the parameter C is learnt by performing 5-fold cross validation on the first five instances of the training set.

- **Cosine Mixture (CM)** Here, the method of Section 4.6 is used. The number of cosine kernels m, their parameters, and the parameter τ are learnt automatically. The parameter C is fixed to ∞ and the parameter β is optimized by performing 5-fold cross validation on the first five instances of the training set.

- **Cosine and Gaussian Mixture (CGM)** Here, the method of Section 4.7 is used. The number of cosine kernels m, the number of Gaussian kernels $\bar{m}$, the parameters of cosine and Gaussian kernels, and the parameter τ are learnt automatically. The parameter C is set to ∞ and the parameter η is set to 0.03. The parameter β is optimized by performing 5-fold cross validation on the first five instances of the training set.

To compare the training and evaluation times of these methods, we repeated the experiments of Rätsch et al. (2001) on our machine. For training a single-kernel SVM we used the implementation of SMO algorithm (Platt, 1999) contained in the Statistical Pattern Recognition Toolbox.[15] To keep the results reported by Rätsch et al. (2001) as reference, we neglect the accuracies obtained by the SG method.

Table 2 summarizes the test error rates and training times of the methods on each data set. It can be seen that the GM method has the worst performance and does not provide any improvement over other methods. The only benefit of the GM over SG is that while the latter requires specifying the kernel function by hand, GM learns the kernel function automatically. The SG and CM are the only methods of this experiment that do not store the kernel matrices in memory; and thus are applicable to large-scale problems. In addition, they have also the best training times. To our surprise, although the CM method solves a musch more difficult problem than SG, it has also improved the training time in some data sets. Considering the test error rates, the CGM method has the best overall performance. But, the SG method on the *F.Solar* data set, and CM method on the *Thyroid* data set provide significantly better results. For the *Ringnorm* data set, the CM method has obtained a high error rate of 8.5%. Interestingly, the number of training and testing samples of the *Ringnorm* data set are exactly equal to the *Twonorm* data set, for which the CM method has even improved the accuracy. The essential difference between the *Twonorm* and the *Ringnorm* data sets, where in both data sets each class has a multivariate normal distribution, is that in the *Twonorm* data set the classes have separate means, whilst in the *Ringnorm* data set the classes have separate covariance matrices. So, it seems that the Gaussian kernel is inherently much more suitable for solving the *Ringnorm* data set than the cosine kernel. In fact, this is exactly why combining several kernels is important. By combining the cosine and Gaussian kernels, the CGM method provides the best performance.

Table 3 compares the methods in terms of the evaluation time. For each method, the factors that influence the evaluation time are also reported. As can be seen, except for the *Ringnorm* data set, the CM is significantly faster at run-time than all other methods, including a classical SVM with Gaussian kernel. The best speedup is for the *Twonorm* data set for which, in addition to a lower test

15. Available at *http://cmp.felk.cvut.cz/cmp/software/stprtool*.

Data Set	Single Gaussian		Gaussian Mixture		Cosine Mixture		Cosine& Gaussian Mixture	
	error (%)	training (*sec*)	error (%)	training (*sec*)	error (%)	training (*sec*)	error (%)	training (*sec*)
Banana	11.5 ± 0.7	**1.1**	10.5 ± 0.5	26.4	10.7 ± 0.5	3.5	$\mathbf{10.4 \pm 0.5}$	21.3
B. Cancer	26.0 ± 4.7	**0.2**	26.7 ± 5.0	2.7	26.2 ± 4.9	1.2	$\mathbf{25.8 \pm 4.7}$	4.2
Diabetis	23.5 ± 1.7	**1.6**	23.7 ± 1.7	7.5	$\mathbf{23.2 \pm 1.9}$	2.3	$\mathbf{23.2 \pm 1.8}$	16.5
F. Solar	$\mathbf{32.4 \pm 1.8}$	9.1	35.4 ± 1.7	15.4	33.3 ± 1.8	**1.2**	33.9 ± 1.8	57.7
German	$\mathbf{23.6 \pm 2.1}$	4.4	25.3 ± 2.5	35.4	24.1 ± 2.2	**3.5**	23.7 ± 2.2	53.9
Heart	16.0 ± 3.3	**0.1**	17.0 ± 3.2	1.4	$\mathbf{15.6 \pm 3.2}$	1.1	16.0 ± 3.2	2.8
Image	3.0 ± 0.6	**16.0**	3.6 ± 1.3	178.9	$\mathbf{2.5 \pm 0.5}$	1057.1	$\mathbf{2.5 \pm 0.5}$	779.6
Ringnorm	$\mathbf{1.7 \pm 0.1}$	**2.9**	$\mathbf{1.7 \pm 0.1}$	9.7	8.5 ± 0.9	192.7	$\mathbf{1.7 \pm 0.1}$	17.7
Splice	10.9 ± 0.7	445.4	11.1 ± 0.7	91.8	9.7 ± 0.4	**43.3**	$\mathbf{9.3 \pm 0.5}$	187.0
Thyroid	4.8 ± 2.2	**0.1**	4.6 ± 2.2	1.8	$\mathbf{3.7 \pm 2.2}$	3.4	4.8 ± 2.1	3.1
Titanic	$\mathbf{22.4 \pm 1.0}$	**0.1**	23.2 ± 1.3	1.0	22.9 ± 1.2	0.9	22.9 ± 1.2	2.6
Twonorm	3.0 ± 0.2	**0.4**	2.7 ± 0.2	7.9	$\mathbf{2.4 \pm 0.1}$	5.4	2.7 ± 0.2	17.9
Waveform	9.9 ± 0.4	5.8	9.8 ± 0.4	8.5	10.0 ± 0.5	**2.6**	$\mathbf{9.7 \pm 0.4}$	17.7

Table 2: Test errors and training times of SG, GM, CM, and CGM methods on the data sets collected by Rätsch et al. (2001)

Data Set	Single Gaussian		Gaussian Mixture			Cosine Mixture		Cosine & Gaussian Mixture			
	testing (*ms*)	nsv	testing (*ms*)	nsv	$\bar{\mathbf{m}}$ *Gauss*	testing (*ms*)	**m** *cos*	testing (*ms*)	nsv	**m** *cos*	$\bar{\mathbf{m}}$ *Gauss*
Banana	144.7	153.1	615.9	375.7	2.1	**13.4**	13.6	611.3	393.1	2.0	2.5
B. Cancer	2.2	122.3	4.8	200.0	1.8	**0.4**	3.6	0.5	200.0	4.3	0.1
Diabetis	19.9	263.3	47.6	464.8	2.1	**0.8**	7.0	3.8	466.2	5.8	0.2
F. Solar	49.9	507.9	354.0	666.0	1.6	**0.6**	2.1	1.3	666	11.0	0.0
German	31.4	426.0	120.7	696.4	2.9	**1.0**	6.6	76.3	700.0	11.6	1.8
Heart	2.2	84.3	6.4	163.1	1.9	**0.3**	1.8	5.1	169.8	1.6	1.5
Image	205.4	700.7	765.1	1030.4	4.4	**56.6**	160.2	503.2	712.7	65.2	3.8
Ringnorm	**120.2**	64.4	487.0	156.2	1.9	228.5	91.7	610.8	200.2	0.0	2.0
Splice	357.7	385.4	1647.7	879.5	2.3	**51.4**	30.8	1293.6	755.7	13.5	1.6
Thyroid	0.5	22.0	3.1	84.3	3.0	**0.4**	6.4	3.1	112.0	1.0	2.1
Titanic	35.9	89.5	78.5	150.0	1.5	**1.5**	2.5	14.8	150.0	2.5	0.3
Twonorm	332.6	167.4	455.7	180.6	1.5	**3.5**	1.0	582.1	226.2	0.0	1.5
Waveform	134.3	104.5	587.4	290.5	1.9	**6.0**	2.9	731.9	374.7	1.0	1.6

Table 3: Experimentally measured evaluation times along with the parameters that theoretically determine the evaluation times of SG, GM, CM, and CGM methods on the data sets collected by Rätsch et al. (2001)

error rate, the evaluation of the CM method is 95 times faster than a classical SVM with Gaussian kernel. This speedup in the evaluation of the classifiers can be very useful for applications targeted at small computers with limited computational power.

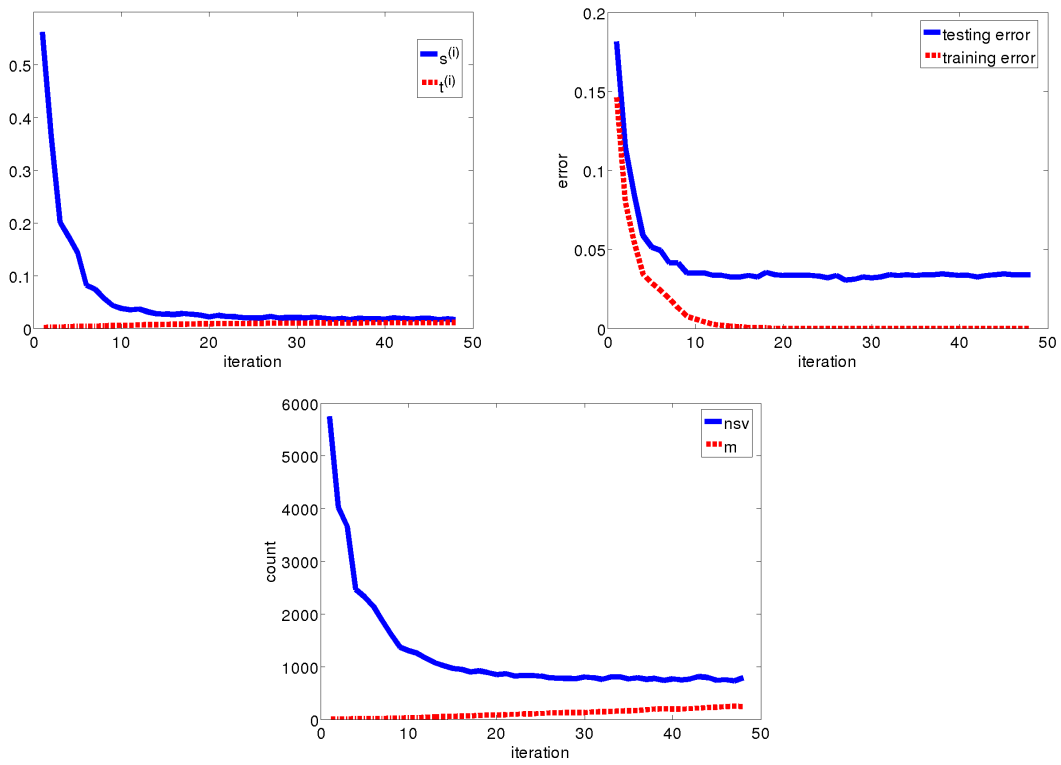


Figure 1: Evolution of the values m, nsv, $t^{(i)}$, $s^{(i)}$, training error, and test error during training of the proposed method on the USPS data set.

9.2 Experiments on the USPS Data Set

In this section, we show the applicability of the proposed method[16] to a real-world digit recognition problem. We consider the problem of classifying digits 0-4 against 5-9 on the USPS handwritten digit recognition data set as considered by Chapelle et al. (2002) for evaluating their kernel learning method. This data set consists of 7291 training examples and 2007 test examples of digit images of size 16×16. With polynomial kernel and 256 scaling factors, Chapelle et al. (2002) were able to get a test error rate of 9.0%. We trained the proposed method with the parameters $\varepsilon = 0.001$, and $\beta = 3.0$. After two hours of training, the algorithm produced a model with 244 cosine kernels and 790 support vectors. It took 1.3 of a second to test the model on the 2007 test samples and we obtained a test error rate of 3.4% which is significantly better than the 9.0% result reported by Chapelle et al. (2002). Figure 1 shows the evolution of the values m, nsv, $t^{(i)}$, $s^{(i)}$, training error, and test error during training of the USPS data set, where $s^{(i)}$ and $t^{(i)}$ are defined in Algorithm 1.

9.3 Experiments on the MNIST Data Set

While many algorithms for kernel learning consider the combination of a finite number of kernels, learning translation invariant kernels corresponds to combining an infinite number of kernels. In

16. From this section onward, the proposed method refers to the cosine mixture method.

this section, we compare the proposed algorithm with the DC method proposed by Argyriou et al. (2006) which is based on the theory developed by Micchelli and Pontil (2005). They considered the problem of finding an optimal kernel over the whole class of radial kernels which is equivalent to the problem of learning the best convex combination of Gaussian kernels with isotropic covariance matrices.

Argyriou et al. (2006) performed a series of experiments on the MNIST data set by using the first 500 training examples for training and the first 1000 test examples for evaluation. The MNIST data set contains 28×28 images of handwritten digits which are divided into 60,000 training and 10,000 test examples. In addition, the results reported by the DC method have been obtained by splitting each image into four sub-images which is a use of extra information. In the first experiment, we use the first $3,000$ training examples[17] for training both systems and evaluate them on the whole test set. As Argyriou et al. (2006), we consider the tasks of classifying digits 3 vs. 8, 4 vs. 7, and odds vs. evens. We downloaded the implementation of the DC method from *http://www.cs.ucl.ac.uk/staff/a.argyriou/code/dc* and chose the range $[75, 25000]$ for the parameter σ of the Gaussian kernel which is the largest range considered by Argyriou et al. (2006). The parameter μ of the DC method and the parameter β of the CM method were optimized by hand. The parameter ε was set to the value 0.001. The first three rows of Table 4 show the results of this experiment. It can be seen that the main benefit of the DC method is its short training time, while the CM method has superiority in terms of the evaluation time.

Another remarkable feature of the CM method is its applicability to large-scale problems. To illustrate this fact, we increased the size of the training set of the previous experiments from $3,000$ to $10,000$. We also increased the parameter ε from 0.001 to 0.01 to decrease the training time. The DC method could not handle this size of training samples and ran out of memory. The last three rows and columns of Table 4 show the results of the experiments with the CM method. Figure 2 depicts the evolution of the values m, nsv, $t^{(i)}$, $s^{(i)}$, training error, and test error during training of CM algorithm on the ods vs even task with $10,000$ training samples. Note that the model produced by the CM method on the larger training set is more accurate and faster-to-evaluate than the best model that the DC algorithm could produce. We think that the capabilities of the CM method and DC method are complementary. The DC method works with full-rank matrices, is not large-scale, converges fast, and its model takes more time to compute. On the other hand, the CM method works with low-rank matrices, is large-scale, converges slowly, and its model can be evaluated very fast. One open problem is that whether these methods can be combined in a way that the benefits of both methods are achieved.

9.4 Assessing the Effect of the Proposed Complexity Control Mechanism

In Section 8 we provided theoretical support for the necessity of controlling the complexity of the class of translation invariant kernels. Here we support this claim by experimenting on the *Heart* data set chosen from the benchmark produced by Rätsch et al. (2001). This data set contains 170 train patterns and 100 test patterns of dimension 13. The experiments of the previous section show that the proposed method was completely successful in obtaining a low test error rate on this data set. In addition, the mean error rate of the proposed method on the train set is 14.0% which is close to the mean error rate of 15.5% obtained on the test set. The left plot of Figure 3 illustrates the trajectories of the train and test errors of the proposed method during training on the *Heart* data set.

17. This is approximately the largest possible train-set size where the DC method did not ran out of memory.

Data Set		DC method			Cosine mixture		
Task	**#tr**	**error** (%)	**train** (*min*)	**test** (*sec*)	**error** (%)	**train** (*min*)	**test** (*sec*)
odd vs even	3000	3.1	11	60.1	3.2	37	10.1
3 vs 8	3000	0.7	9	24.8	0.8	192	2.6
4 vs 7	3000	0.4	16	26.1	0.5	122	2.5
odd vs even	10000	-	-	-	2.0	63	9.1
3 vs 8	10000	-	-	-	0.4	1133	5.3
4 vs 7	10000	-	-	-	0.2	649	3.1

Table 4: Test errors of the proposed method (CM) and the DC method on different tasks on the MNIST data set. The dash sign indicates running out of memory.

In another experiment we disabled our complexity control mechanism by setting $\beta = 10^{-6}$ and instead tried to control the capacity of the learning machine by adjusting the parameter C.[18] We optimized the parameter C using the 5-fold cross-validation method described in the previous section. After testing on all the 100 splits of the *Heart* data set we obtained a mean test error rate of 20.8% and a mean train error rate of 17.4%. The right plot of Figure 3 illustrates the trajectories of the train and test error rates of this experiment on the *Heart* data set. This experiment confirms the usefulness of controlling the complexity of the class of translation invariant kernels, as was claimed in Section 8.

10. Conclusions

In this paper we addressed the problem of learning a translation invariant kernel function for the task of binary classification with SVM. We proposed a mechanism for controlling the complexity of the class of translation invariant kernels which was found to be very useful in practice. The criterion proposed by Lanckriet et al. (2004) was modified to ensure the compactness of the parameter space of SVM and to give a probabilistic meaning to the regularization parameter of the 2-norm SVM. We then introduced a semi-infinite programming formulation of the problem. The proposed method can automatically learn the regularization parameter of the 2-norm SVM, as well. We have also shown that how other classes of kernels can be included in the learning process. To numerically solve the SIP problem on a computer, we introduced a large-scale algorithm which is applicable to problems with both huge number of training samples and large number of features. Since the optimal translation invariant kernel is complex-valued, we then introduced a method for applying the kernel trick to complex-valued kernels. It revealed that the optimal translation invariant kernel is a mixture of cosine kernels. An interesting feature of the proposed method is that there is a very fast way for evaluating the classifier at run-time. While an ordinary MKL algorithm with m kernels requires $O(m \times nsv \times n)$ steps for computing the classifier, the optimal classifier of the proposed method can be computed in $O(m \times n)$ steps.

In continuation of this work, we plan to extend it in several directions. First, we intend to generalize the proposed kernel learning method from binary classification to other learning problems, including regression, multiclass classification, clustering, and kernel PCA. Second, we will try to propose a novel large-scale algorithm that combines the benefits of the full-rank Gaussian and the

18. Setting $\beta = 0$ (exactly) allowed the global search algorithm to find points at infinity which caused problems.

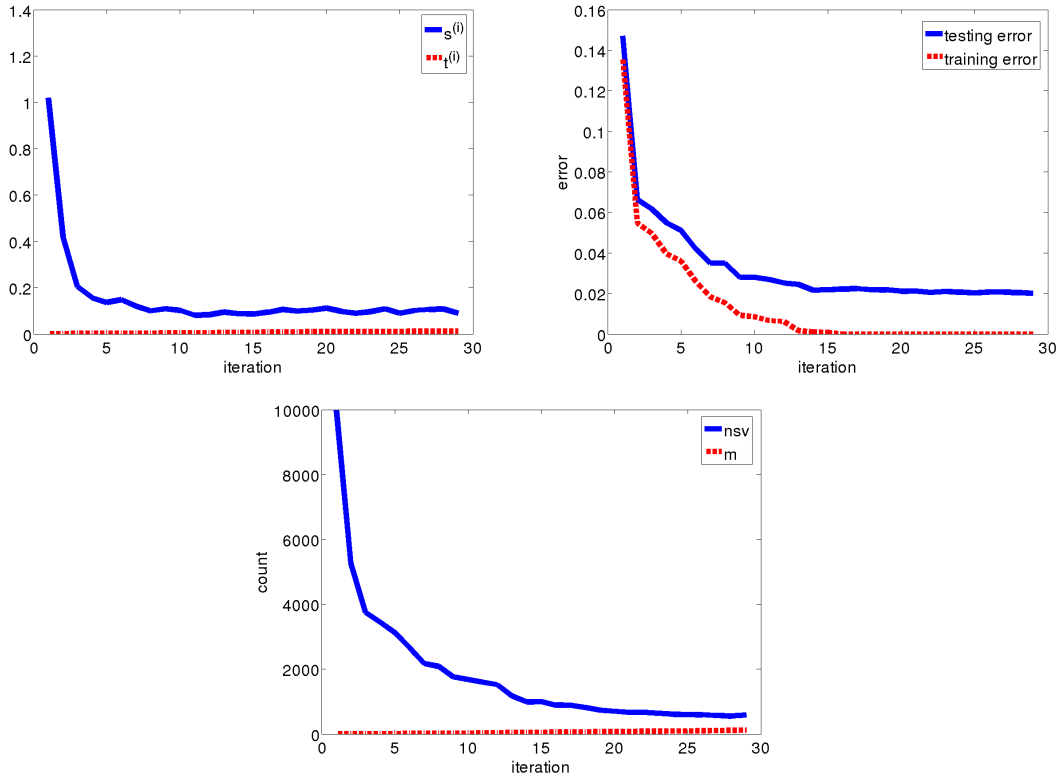


Figure 2: Evolution of the values $m, nsv, t^{(i)}, s^{(i)}$, training error, and testing error during training of the proposed method with the first 10,000 samples of the MNIST data set for the task of classifying odds vs. evens.

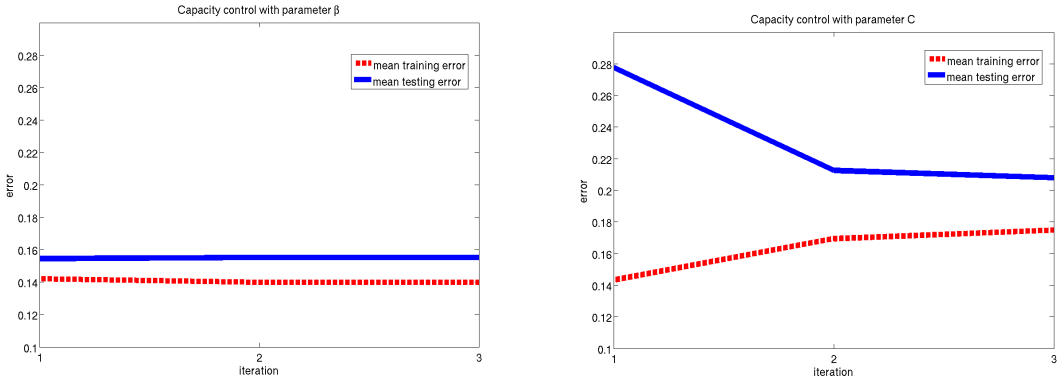


Figure 3: Comparison between the parameters β and C for controlling the capacity of the class of translation invariant kernels on the *Heart* data set.

low-rank cosine kernels. Third, we intend to investigate the applicability of the idea of Xu et al. (2008) about adding a regularization term that smoothes the fluctuating behaviour of SILP algorithms, to the proposed SIP algorithm. We hope that this study would greatly decrease the training time of the proposed method. Another direction is to support the complexity control mechanism of Section 2 by introducing upper bounds for the generalization error of the proposed method. Our long time plan is to investigate the use of other low-rank kernels and make it a competing popular technology.

Acknowledgments

The first author would like to thank professor Abdolhamid Riazi for allowing him to participate in his functional analysis class and helping him to get acquainted with the concepts of this graduate course. We also want to thank Stephen Boyd, Andreas Argoriou, and Peter Gehler for providing us with their valuable materials.

References

S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

E.D. Andersen and K.D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In T. Terlaky H. Frenk, K. Roos and S. Zhang, editors, *High Performance Optimization*, pages 197–232. Kluwer Academic Publishers, 2000.

A. Argyriou, C.A. Micchelli, and M. Pontil. Learning convex combinations of continuously parameterized basic kernels. In *Proceedings of the 18th Conference on Learning Theory*, volume 18, pages 338–352, 2005.

A. Argyriou, R. Hauser, C.A. Micchelli, and M. Pontil. A DC-programming algorithm for kernel selection. In *Proceedings of the* 23^{rd} *International Conference on Machine Learning*, volume 23, pages 338–352, Pittsburgh, PA, 2006.

F.R. Bach, G.R.G. Lanckriet, and M.I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the 21st International Conference on Machine Learning*, volume 21, pages 41–48, Banff, Canada, 2004. Omnipress.

S. Bochner. Monotone functions, Stieltjessche integrale und harmonische analyse. *Mathematische Annalen*, 108:378–410, 1933.

S. Bochner. *Harmonic Analysis and the Theory of Probability*. University of California Press, Los Angeles, California, 1955.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

D.J. Crisp and C.J.C. Burges. A geometric interpretation of ν-svm classifiers. In *Advances in Neural Information Processing Systems*, volume 12, pages 244–250, Cambridge, MA:, 1999. MIT Press.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

Nello Cristianini, Colin Campbell, and John Shawe-taylor. Dynamically adapting kernels in support vector machines. In *Advances in Neural Information Processing Systems 11*, pages 204–210. MIT Press, 1998.

F. Cucker and D.X. Zhou. *Learning Theory: An Approximation Theory Viewpoint*. Cambridge University Press, Cambridge, UK, 2007.

J. M. Danskin. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664, 1966.

P. V. Gehler and S. Nowozin. Infinite kernel learning. In *Proceedings of the NIPS 2008 Workshop on "Kernel Learning: Automatic Selection of Optimal Kernels"*, pages 1–4, 2008.

M.G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2:299–312, 2001.

F. Girosi, M. Jones, and T. Poggio. Priors, stabilizers and basis functions: from regularization to radial, tensor and additive splines. A.I. Memo 1430, Artificial Intelligence Labratory, Massachusetts Institute of Technology, June 1993.

T. Glasmachers and C. Igel. Gradient-based adaptation of general Gaussian kernels. *Neural Computation*, 17:2099–2105, 2005.

R. Hettich and K.O. Kortanek. Semi-infinite programming: theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods- Support Vector Learning*, pages 169–184. MIT Press, 1999.

E. Kreyszig. *Introductory Functional Analysis with Applications*. Wiley, New York, 1989.

G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, 45(3):503–528, 1989.

M.E. Mavroforakis and S. Theodorodis. A geometric approach to support vector machine (SVM) classification. *IEEE Transactions on Neural Networks*, 17(3):671–682, 2006.

C.A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

C.A. Micchelli, M. Pontil, Q. Wu, and D.X. Zhou. Error bounds for learning the kernel. Research Note RN/05/09, Department of Computer Science, University College London, June 2005b.

J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35 (151):773–782, 1980.

J. Nocedal and S.J. Wright. *Numerical Optimization, second edition*. Springer Science+Business Media, LLC, New York, USA, 2006.

C.S. Ong, A.J. Smola, and R.C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6:1043–1071, 2005.

J.C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods- Support Vector Learning*, pages 185–208. MIT Press, 1999.

A. Rakotomamonjy, F.R. Bach, S. Canu, and V. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

G. Rätsch, T. Onoda, and K.R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3): 287–320, 2001.

R. Reemtsen and S. Görner. Numerical methods for semi-infinite programming: A survey. In R. Reemtsen and Rückmann, editors, *Semi-infinite Programming*, pages 195–275. Kluwer Academic Publishers, 1998.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, New Jersey, 1970.

H.L. Royden. *Real Analysis, 3rd edition*. Macmillan Publishing Company, New York, 1988.

W. Rudin. *Functional Analysis, 2nd edition*. McGraw-Hill, New York, 1991.

W. Rudin. *Real & Complex Analysis, 3rd edition*. McGraw-Hill, New York, 1987.

I.J. Schoenberg. Metric spaces and completely monotone functions. *Annals of Mathematics*, 39(4): 811–841, 1938.

B. Schölkopf and A. Smola. *Learning with Kernels- Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, P. Knirsch, A. Smola, and C. Burges. Fast approximation of support vector kernel expansions, and an interpretation of clustering as approximation in feature spaces. In *DAGM Symposium Mustererkennung*. Springer Lecture Notes in Computer Science, 1998.

S. Sonnenburg, G. Rätsch, and C. Schafer. A general and efficient multiple kernel learning algorithm. In *Advances in Neural Information Processing Systems*, volume 18, pages 1275–1282, Cambridge MA, 2005. MIT Press.

S. Sonnenburg, G. Rätsch, C. Schafer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1567, 2006.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

V.N. Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999, 1999.

H. Xiong, M.N.S. Swamy, and M.O. Ahmad. Optimizing the kernel in the empirical feature space. *IEEE Transactions on Neural Networks*, 16(2):460–474, 2005.

Z. Xu, R. Jin, I. King, and M.R. Lyu. An extended level method for efficient multiple kernel learning. In *Advances in Neural Information Processing Systems*, volume 21, pages 1825–1832, British Columbia, Canada, 2008. MIT Press.

Y. Yiming and D.X. Zhou. Learnability of Gaussians with flexible variances. *Journal of Machine Learning Research*, 8:249–276, 2007.

Journal of Machine Learning Research 11 (2010) 1391-1423 Submitted 4/08; Revised 8/09; Published 4/10

Consistent Nonparametric Tests of Independence

Arthur Gretton* ARTHUR@TUEBINGEN.MPG.DE
MPI for Biological Cybernetics
Department Schölkopf
Spemannstraße 38
72076 Tübingen, Germany

László Györfi GYORFI@SZIT.BME.HU
Budapest University of Technology and Economics
H-1521 Stoczek u. 2
Budapest, Hungary.

Editor: John Shawe-Taylor

Abstract

Three simple and explicit procedures for testing the independence of two multi-dimensional random variables are described. Two of the associated test statistics (L_1, log-likelihood) are defined when the empirical distribution of the variables is restricted to finite partitions. A third test statistic is defined as a kernel-based independence measure. Two kinds of tests are provided. Distribution-free strong consistent tests are derived on the basis of large deviation bounds on the test statistics: these tests make almost surely no Type I or Type II error after a random sample size. Asymptotically α-level tests are obtained from the limiting distribution of the test statistics. For the latter tests, the Type I error converges to a fixed non-zero value α, and the Type II error drops to zero, for increasing sample size. All tests reject the null hypothesis of independence if the test statistics become large. The performance of the tests is evaluated experimentally on benchmark data.

Keywords: hypothesis test, independence, L1, log-likelihood, kernel methods, distribution-free consistent test

1. Introduction

Consider a sample of $\mathbb{R}^d \times \mathbb{R}^{d'}$-valued random vectors $(X_1,Y_1),\ldots,(X_n,Y_n)$ with independent and identically distributed (i.i.d.) pairs defined on the same probability space. The distribution of (X,Y) is denoted by ν, while μ_1 and μ_2 stand for the distributions of X and Y, respectively. We are interested in testing the null hypothesis that X and Y are independent,

$$\mathcal{H}_0 : \nu = \mu_1 \times \mu_2, \tag{1}$$

while making minimal assumptions regarding the distribution.

We consider two main approaches to independence testing. The first is to partition the underlying space, and to evaluate the test statistic on the resulting discrete empirical measures. Consistency of the test must then be verified as the partition is refined for increasing sample size. Previous multivariate hypothesis tests in this framework, using the L_1 divergence measure, include homogeneity tests (to determine whether two random variables have the same distribution), by Biau and Györfi

*. Also at Carnegie Mellon University, Pittsburgh, PA, USA.

(2005); and goodness-of-fit tests (for whether a random variable has a particular distribution), by Györfi and van der Meulen (1990); Beirlant et al. (1994). The log-likelihood has also been employed on discretised spaces as a statistic for goodness-of-fit testing, by Györfi and Vajda (2002). We provide generalizations of both the L_1 and log-likelihood based tests to the problem of testing independence, representing to our knowledge the first application of these techniques to independence testing.

We obtain two kinds of tests for each statistic: first, we derive *strong consistent* tests—meaning that both on $\mathcal{H}_0$ and on its complement the tests make a.s. no error after a random sample size[1]—based on large deviation bounds. While such tests are not common in the classical statistics literature, they are well suited to data analysis from streams, where we receive a sequence of observations rather than a sample of fixed size, and must return the best possible decision at each time using only current and past observations. Our strong consistent tests are *distribution-free*, meaning they require no conditions on the distribution being tested; and *universal*, meaning the test threshold holds independent of the distribution. Second, we obtain tests based on the asymptotic distribution of the L_1 and log-likelihood statistics, which assume only that ν is nonatomic. Subject to this assumption, the tests are *consistent*: for a given asymptotic error rate on $\mathcal{H}_0$, the probability of error on $\mathcal{H}_1$ drops to zero as the sample size increases. Moreover, the thresholds for the asymptotic tests are distribution-independent. We also present conjectures regarding the form taken by strong consistent and asymptotic tests based on the Pearson χ^2 statistic, using the goodness-of-fit results of Györfi and Vajda (2002) (further related test statistics include the power divergence family of Read and Cressie, 1988, although we do not study them here). We emphasize that our tests are explicit, easy to carry out, and require very few assumptions on the partition sequences.

Our second approach to independence testing is kernel-based. In this case, our test statistic has a number of different interpretations: as an L_2 distance between Parzen window estimates (Rosenblatt, 1975), as a smoothed difference between empirical characteristic functions (Feuerverger, 1993; Kankainen, 1995; Ushakov, 1999), or as the Hilbert-Schmidt norm of a cross-covariance operator mapping between functions of the random variables (Gretton et al., 2005a, 2008). Each test differs from the others regarding the conditions on the kernels: the Parzen window statistic requires the kernel bandwidth to decrease with increasing sample size, and has a different limiting distribution to the remaining two statistics; while the Hilbert-Schmidt approach uses a fixed bandwidth, and can be thought of as a generalization of the characteristic function-based test. We provide two new results: a strong consistent test of independence based on a tighter large deviation bound than that of Gretton et al. (2005a), and an empirical comparison of the limiting distributions of the kernel-based statistic for fixed and decreasing kernel bandwidth, as used in asymptotic tests.

Additional independence testing approaches also exist in the statistics literature. For $d = d' = 1$, an early nonparametric test for independence, due to Hoeffding (1948); Blum et al. (1961), is based

1. In other words, denoting by $\mathbf{P}_0$ (*resp*. $\mathbf{P}_1$) the probability under the null hypothesis (*resp*. under the alternative), we have

$$\mathbf{P}_0\{\text{rejecting } \mathcal{H}_0 \text{ for only finitely many } n\} = 1 \tag{2}$$

and

$$\mathbf{P}_1\{\text{accepting } \mathcal{H}_0 \text{ for only finitely many } n\} = 1. \tag{3}$$

This concept relates to the definition of discernability introduced by Dembo and Peres (1994): two ensembles $\mathcal{H}_0$ and $\mathcal{H}_1$ of probability measures on $\mathbb{R}^k$ are said to be discernible if there exists a sequence $f_n : (\mathbb{R}^k)^n \to \{0,1\}$ of Borel measurable functions achieving (2) and (3). Thus our test implies discernability of the set $\mathcal{H}_0$ in (1) and the set $\mathcal{H}_1$ of dependent random variables.

on the notion of differences between the joint distribution function and the product of the marginals. The associated independence test is consistent under appropriate assumptions. Two difficulties arise when using this statistic in a test, however. First, quantiles of the null distribution are difficult to estimate. Second, and more importantly, the quality of the empirical distribution function estimates becomes poor as the dimensionality of the spaces $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ increases, which limits the utility of the statistic in a multivariate setting. Further approaches to independence testing can be employed when particular assumptions are made on the form of the distributions, for instance that they should exhibit symmetry. We do not address these approaches in the present study.

The current work is built on an earlier presentation by Gretton and Györfi (2008). Compared with this earlier work, the present study contains more detailed proofs of the main theorems, proofs of secondary theorems omitted by Gretton and Györfi (2008) due to space constraints, additional experiments on higher dimensional benchmark data, and an experimental comparison with the bootstrap approach for the L_1 and log-likelihood based tests (a similar comparison for the kernel-based test was made by Gretton et al., 2008).

The paper is organized as follows. Section 2 describes the large deviation and limit distribution properties of the L_1-test statistic. The large deviation result is used to formulate a distribution-free strong consistent test of independence, which rejects the null hypothesis if the test statistic becomes large. The limit distribution is used in an asymptotically α-level test, which is consistent when the distribution is nonatomic. Both a distribution-free strong consistent test and an asymptotically α-level test are presented for the log-likelihood statistic in Section 3. Section 4 contains a review of kernel-based independence statistics, and describes the associated hypothesis tests for both the fixed-bandwidth and variable-bandwidth cases. Finally, a numerical comparison between the tests is given in Section 5.

2. L_1-based Statistic

Denote by ν_n, $\mu_{n,1}$ and $\mu_{n,2}$ the empirical measures associated with the samples $(X_1,Y_1),\dots,(X_n,Y_n)$, $X_1,\dots,X_n$, and $Y_1,\dots,Y_n$, respectively, so that

$$\begin{aligned}\nu_n(A\times B) &= n^{-1}\#\{i : (X_i,Y_i)\in A\times B, i=1,\dots,n\},\\ \mu_{n,1}(A) &= n^{-1}\#\{i : X_i\in A, i=1,\dots,n\}, \quad \text{and}\\ \mu_{n,2}(B) &= n^{-1}\#\{i : Y_i\in B, i=1,\dots,n\},\end{aligned}$$

for any Borel subsets A and B. Given the finite partitions $\mathcal{P}_n = \{A_{n,1},\dots,A_{n,m_n}\}$ of $\mathbb{R}^d$ and $Q_n = \{B_{n,1},\dots,B_{n,m_n'}\}$ of $\mathbb{R}^{d'}$, we define the L_1 test statistic comparing ν_n and $\mu_{n,1}\times\mu_{n,2}$ as

$$L_n(\nu_n,\mu_{n,1}\times\mu_{n,2}) = \sum_{A\in\mathcal{P}_n}\sum_{B\in Q_n}|\nu_n(A\times B)-\mu_{n,1}(A)\cdot\mu_{n,2}(B)|.$$

In the following two sections, we derive the large deviation and limit distribution properties of this L_1 statistic, and the associated independence tests.

2.1 Strongly Consistent Test

For testing a simple hypothesis versus a composite alternative, Györfi and van der Meulen (1990) introduced a related goodness of fit test statistic L_n defined as

$$L_n(\mu_{n,1},\mu_1) = \sum_{A\in\mathcal{P}_n}|\mu_{n,1}(A)-\mu_1(A)|.$$

Beirlant, Devroye, Györfi, and Vajda (2001), and Biau and Györfi (2005) proved that, for all $0 < \varepsilon$,

$$\mathbf{P}\{L_n(\mu_{n,1},\mu_1) > \varepsilon\} \leq 2^{m_n} e^{-n\varepsilon^2/2}. \tag{4}$$

We now describe a similar result for our L_1 independence statistic.

Theorem 1 *Under $\mathcal{H}_0$, for all $0 < \varepsilon_1$, $0 < \varepsilon_2$ and $0 < \varepsilon_3$,*

$$\mathbf{P}\{L_n(\nu_n,\mu_{n,1}\times\mu_{n,2}) > \varepsilon_1+\varepsilon_2+\varepsilon_3\} \leq 2^{m_n\cdot m'_n} e^{-n\varepsilon_1^2/2} + 2^{m_n} e^{-n\varepsilon_2^2/2} + 2^{m'_n} e^{-n\varepsilon_3^2/2}.$$

Proof We bound $L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$ according to

$$\begin{aligned}
L_n(\nu_n,\mu_{n,1}\times\mu_{n,2}) &= \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\nu_n(A\times B) - \mu_{n,1}(A)\cdot\mu_{n,2}(B)| \\
&\leq \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\nu_n(A\times B) - \nu(A\times B)| \\
&\quad + \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\nu(A\times B) - \mu_1(A)\cdot\mu_2(B)| \\
&\quad + \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\mu_1(A)\cdot\mu_2(B) - \mu_{n,1}(A)\cdot\mu_{n,2}(B)|.
\end{aligned}$$

Under the null hypothesis $\mathcal{H}_0$, we have that

$$\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\nu(A\times B) - \mu_1(A)\cdot\mu_2(B)| = 0.$$

Moreover

$$\begin{aligned}
&\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\mu_1(A)\cdot\mu_2(B) - \mu_{n,1}(A)\cdot\mu_{n,2}(B)| \\
\leq\ &\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\mu_1(A)\cdot\mu_2(B) - \mu_1(A)\cdot\mu_{n,2}(B)| \\
&+ \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} |\mu_1(A)\cdot\mu_{n,2}(B) - \mu_{n,1}(A)\cdot\mu_{n,2}(B)| \\
=\ &\sum_{B\in\mathcal{Q}_n} |\mu_2(B) - \mu_{n,2}(B)| + \sum_{A\in\mathcal{P}_n} |\mu_1(A) - \mu_{n,1}(A)| \\
=\ &L_n(\mu_{n,1},\mu_1) + L_n(\mu_{n,2},\mu_2).
\end{aligned}$$

Thus, (4) implies

$$\begin{aligned}
&\mathbf{P}\{L_n(\nu_n,\mu_{n,1}\times\mu_{n,2}) > \varepsilon_1+\varepsilon_2+\varepsilon_3\} \\
\leq\ &\mathbf{P}\{L_n(\nu_n,\nu) > \varepsilon_1\} + \mathbf{P}\{L_n(\mu_{n,1},\mu_1) > \varepsilon_2\} + \mathbf{P}\{L_n(\mu_{n,2},\mu_2) > \varepsilon_3\} \\
\leq\ &2^{m_n\cdot m'_n} e^{-n\varepsilon_1^2/2} + 2^{m_n} e^{-n\varepsilon_2^2/2} + 2^{m'_n} e^{-n\varepsilon_3^2/2}.
\end{aligned}$$

■

Theorem 1 yields a strong consistent test of independence, which rejects the null hypothesis if $L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$ becomes large. The test is distribution-free, that is, the probability distributions ν, μ_1 and μ_2 are completely arbitrary; and the threshold is universal, that is, it does not depend on the distribution.

Corollary 2 *Consider the test which rejects $\mathcal{H}_0$ when*

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > c_1 \left(\sqrt{\frac{m_n m'_n}{n}} + \sqrt{\frac{m_n}{n}} + \sqrt{\frac{m'_n}{n}} \right) \approx c_1 \sqrt{\frac{m_n m'_n}{n}},$$

where

$$c_1 > \sqrt{2 \ln 2} \approx 1.177. \tag{5}$$

Assume that conditions

$$\lim_{n \to \infty} \frac{m_n m'_n}{n} = 0, \tag{6}$$

and

$$\lim_{n \to \infty} \frac{m_n}{\ln n} = \infty, \qquad \lim_{n \to \infty} \frac{m'_n}{\ln n} = \infty, \tag{7}$$

are satisfied. Then under $\mathcal{H}_0$, the test makes a.s. no error after a random sample size. Moreover, if

$$\nu \neq \mu_1 \times \mu_2,$$

and for any sphere S centered at the origin,

$$\lim_{n \to \infty} \max_{A \in \mathcal{P}_n, A \cap S \neq 0} \operatorname{diam}(A) = 0 \tag{8}$$

and

$$\lim_{n \to \infty} \max_{B \in \mathcal{Q}_n, B \cap S \neq 0} \operatorname{diam}(B) = 0, \tag{9}$$

then after a random sample size the test makes a.s. no error.

Proof Under $\mathcal{H}_0$, we obtain from Theorem 1 a non-asymptotic bound for the tail of the distribution of $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$, namely

$$\begin{aligned} & \mathbf{P}\left\{ L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > c_1 \left(\sqrt{\frac{m_n m'_n}{n}} + \sqrt{\frac{m_n}{n}} + \sqrt{\frac{m'_n}{n}} \right) \right\} \\ \leq \;& 2^{m_n m'_n} e^{-c_1^2 m_n m'_n/2} + 2^{m_n} e^{-c_1^2 m_n/2} + 2^{m'_n} e^{-c_1^2 m'_n/2} \\ \leq \;& e^{-(c_1^2/2 - \ln 2) m_n m'_n} + e^{-(c_1^2/2 - \ln 2) m_n} + e^{-(c_1^2/2 - \ln 2) m'_n} \end{aligned}$$

as $n \to \infty$. Therefore the conditions (7) imply

$$\sum_{n=1}^{\infty} \mathbf{P}\left\{ L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > c_1 \left(\sqrt{\frac{m_n m'_n}{n}} + \sqrt{\frac{m_n}{n}} + \sqrt{\frac{m'_n}{n}} \right) \right\} < \infty,$$

and the proof under the null hypothesis is completed by the Borel-Cantelli lemma.

For the result under the alternative hypothesis, we first apply the triangle inequality

$$\begin{aligned} L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \quad \geq \quad & \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu(A \times B) - \mu_1(A) \cdot \mu_2(B)| \\ & - \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} |\nu_n(A \times B) - \nu(A \times B)| \\ & - \sum_{B \in \mathcal{Q}_n} |\mu_2(B) - \mu_{n,2}(B)| \\ & - \sum_{A \in \mathcal{P}_n} |\mu_1(A) - \mu_{n,1}(A)|. \end{aligned}$$

The condition in (6) implies the three last terms of the right hand side tend to 0 a.s. Moreover, using the technique from Barron, Györfi, and van der Meulen (1992) we can prove that by conditions (8) and (9),

$$\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}|\nu(A\times B)-\mu_1(A)\cdot\mu_2(B)|\to 2\sup_C|\nu(C)-\mu_1\times\mu_2(C)|>0$$

as $n\to\infty$, where the last supremum is taken over all Borel subsets C of $\mathbb{R}^d\times\mathbb{R}^{d'}$, and therefore

$$\liminf_{n\to\infty}L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})\geq 2\sup_C|\nu(C)-\mu_1\times\mu_2(C)|>0$$

a.s. ■

2.2 Asymptotic α-level Test

Beirlant, Györfi, and Lugosi (1994) proved, under conditions

$$\lim_{n\to\infty}m_n=\infty,\qquad\lim_{n\to\infty}\frac{m_n}{n}=0, \tag{10}$$

and

$$\lim_{n\to\infty}\max_{j=1,\ldots,m_n}\mu_1(A_{nj})=0, \tag{11}$$

that

$$\sqrt{n}\left(L_n(\mu_{n,1},\mu_1)-\mathbf{E}\{L_n(\mu_{n,1},\mu_1)\}\right)/\sigma\xrightarrow{\mathcal{D}}\mathcal{N}(0,1),$$

where $\xrightarrow{\mathcal{D}}$ indicates convergence in distribution and $\sigma^2=1-2/\pi$. The technique of Beirlant, Györfi, and Lugosi (1994) involves a Poisson representation of the empirical process in conjunction with Bartlett's idea of partial inversion for obtaining characteristic functions of conditional distributions (see Bartlett, 1938). We apply these techniques in Appendix A to derive an asymptotic result for $L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$.

Theorem 3 *Assume that conditions (6) and*

$$\lim_{n\to\infty}\max_{A\in\mathcal{P}_n}\mu_1(A)=0,\quad\lim_{n\to\infty}\max_{B\in\mathcal{Q}_n}\mu_2(B)=0, \tag{12}$$

are satisfied. Then, under $\mathcal{H}_0$, there exists a centering sequence $(C_n)_{n\geq 1}$ depending on ν such that

$$\sqrt{n}\left(L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})-C_n\right)/\sigma\xrightarrow{\mathcal{D}}\mathcal{N}(0,1),$$

where $\sigma^2=1-2/\pi$.

Theorem 3 yields the asymptotic null distribution of a consistent independence test, which rejects the null hypothesis if $L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$ becomes large. In contrast to Corollary 2, and because of condition (11), this new test is *not* distribution-free: the measures μ_1 and μ_2 have to be nonatomic.

Corollary 4 *Let $\alpha \in (0,1)$. Consider the test which rejects $\mathcal{H}_0$ when*

$$\begin{aligned} L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) &> c_2 \sqrt{\frac{m_n m_n'}{n}} + \frac{\sigma}{\sqrt{n}} \Phi^{-1}(1-\alpha) \\ &\approx c_2 \sqrt{\frac{m_n m_n'}{n}}, \end{aligned}$$

where

$$\sigma^2 = 1 - 2/\pi \quad \textit{and} \quad c_2 = \sqrt{2/\pi} \approx 0.798,$$

and Φ denotes the standard normal distribution function. Then, under the conditions of Theorem 3, the test has asymptotic significance level α. Moreover, under the additional conditions (8) and (9), the test is consistent.

Before proceeding to the proof, we examine how the above test differs from that in Corollary 2. In particular, comparing c_2 above with c_1 in (5), both tests behave identically with respect to $\sqrt{m_n m_n'/n}$ for large enough n, but c_2 is smaller.

Proof According to Theorem 3, under $\mathcal{H}_0$,

$$\mathbf{P}\{\sqrt{n}(L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) - C_n)/\sigma \le x\} \approx \Phi(x),$$

therefore the error probability with threshold x is

$$\alpha = 1 - \Phi(x).$$

Thus the α-level test rejects the null hypothesis if

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > C_n + \frac{\sigma}{\sqrt{n}} \Phi^{-1}(1-\alpha).$$

As C_n depends on the unknown distribution, we apply an upper bound

$$C_n \le \sqrt{2/\pi} \sqrt{\frac{m_n m_n'}{n}}$$

(see Equation (22) in Appendix A for the definition of C_n, and Equation (23) for the bound), so decreasing the error probability. ■

3. Log-likelihood Statistic

In the literature on goodness-of-fit testing the *I-divergence statistic*, *Kullback-Leibler divergence*, or *log-likelihood statistic*,

$$I_n(\mu_{n,1}, \mu_1) = \sum_{j=1}^{m_n} \mu_{n,1}(A_{n,j}) \log \frac{\mu_{n,1}(A_{n,j})}{\mu_1(A_{n,j})},$$

plays an important role. For testing independence, the corresponding log-likelihood test statistic is defined as

$$I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\nu_n(A \times B)}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}.$$

The large deviation and the limit distribution properties of $I_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ can be derived from the properties of

$$I_n(\nu_n, \nu) = \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\nu_n(A \times B)}{\nu(A \times B)}.$$

We have that under $\mathcal{H}_0$,

$$\begin{aligned}
& I_n(\nu_n, \nu) - I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \\
= & \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\nu_n(A \times B)}{\nu(A \times B)} \\
& - \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\nu_n(A \times B)}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)} \\
= & \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}{\nu(A \times B)} \\
= & \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \log \frac{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}{\mu_1(A) \cdot \mu_2(B)},
\end{aligned}$$

therefore

$$\begin{aligned}
& I_n(\nu_n, \nu) - I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \\
= & \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \nu_n(A \times B) \left(\log \frac{\mu_{n,1}(A)}{\mu_1(A)} + \log \frac{\mu_{n,2}(B)}{\mu_2(B)} \right) \\
= & \sum_{A \in \mathcal{P}_n} \mu_{n,1}(A) \log \frac{\mu_{n,1}(A)}{\mu_1(A)} + \sum_{B \in \mathcal{Q}_n} \mu_{n,2}(B) \log \frac{\mu_{n,2}(B)}{\mu_2(B)} \\
= & I_n(\mu_{n,1}, \mu_1) + I_n(\mu_{n,1}, \mu_1) \\
\geq & 0.
\end{aligned}$$

3.1 Strongly Consistent Test

We refer to Tusnády (1977) and Barron (1989) who first discussed the exponential character of the tails of I_n. Kallenberg (1985), and Quine and Robinson (1985) proved that, for all $\varepsilon > 0$,

$$\mathbf{P}\{I_n(\mu_{n,1}, \mu_1) > \varepsilon\} \leq \binom{n + m_n - 1}{m_n - 1} e^{-n\varepsilon} \leq e^{m_n \log(n+m_n) - n\varepsilon}.$$

Note that using an alternative bound due to Barron (1989, Equation 3.5), we obtain under (10) and (11) that

$$\mathbf{P}\{I_n(\mu_{n,1}, \mu_1) > \varepsilon\} = e^{-n(\varepsilon + o(1))}, \tag{13}$$

such that

$$\lim_{n \to \infty} \frac{1}{n} \log \mathbf{P}\{I_n(\mu_{n,1}, \mu_1) > \varepsilon\} = -\varepsilon.$$

A large deviation based test can be introduced such that the test rejects the independence if

$$I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \geq \frac{m_n m_n' (\log(n + m_n m_n') + 1)}{n}.$$

Under $\mathcal{H}_0$, we obtain a non-asymptotic bound for the tail of the distribution of $I_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$:

$$\begin{aligned} & \mathbf{P}\left\{ I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > \frac{m_n m'_n (\log(n + m_n m'_n) + 1)}{n} \right\} \\ \leq \; & \mathbf{P}\left\{ I_n(\nu_n, \nu) > \frac{m_n m'_n (\log(n + m_n m'_n) + 1)}{n} \right\} \\ \leq \; & e^{m_n m'_n \log(n + m_n m'_n) - n \frac{m_n m'_n (\log(n + m_n m'_n) + 1)}{n}} \\ = \; & e^{-m_n m'_n}. \end{aligned}$$

Therefore condition (7) implies

$$\sum_{n=1}^{\infty} \mathbf{P}\left\{ I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) > \frac{m_n m'_n (\log(n + m_n m'_n) + 1)}{n} \right\} < \infty,$$

and by the Borel-Cantelli lemma we have strong consistency under the null hypothesis.

Under the alternative hypothesis the proof of strong consistency follows from the inequality, also called Pinsker's inequality, which upper bounds the L_1 error in terms of I-divergence (cf. Csiszár, 1967; Kemperman, 1969; Kullback, 1967),

$$L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})^2 \leq 2 I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}).$$

Therefore,

$$\begin{aligned} \liminf_{n \to \infty} 2 I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) & \geq (\liminf_{n \to \infty} L_n(\nu_n, \mu_{n,1} \times \mu_{n,2}))^2 \\ & \geq 4 \sup_C |\nu(C) - \mu_1 \times \mu_2(C)|^2 > 0 \end{aligned}$$

a.s., where the supremum is taken over all Borel subsets C of $\mathbb{R}^d \times \mathbb{R}^{d'}$. In fact, under conditions (8), (9), and

$$I(\nu, \mu_1 \times \mu_2) < \infty,$$

one may get

$$\lim_{n \to \infty} I_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) = I(\nu, \mu_1 \times \mu_2) > 0$$

a.s. (see Barron et al., 1992). Note that due to the form of the universal test threshold, strong consistency under $\mathcal{H}_1$ requires the condition

$$\lim_{n \to \infty} \frac{m_n m'_n}{n} \log(n + m_n m'_m) = 0,$$

as compared to (6).

3.2 Asymptotic α-level Test

Concerning the limit distribution, Inglot et al. (1990), and Györfi and Vajda (2002) proved that under (10) and (11),

$$\frac{2n I_n(\mu_{n,1}, \mu_1) - m_n}{\sqrt{2m_n}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1). \tag{14}$$

This implies that for any real valued x, under the conditions (6) and (12),

$$\begin{aligned}\mathbf{P}\left\{\frac{2nI_n(\nu_n,\mu_{n,1}\times\mu_{n,2})-m_nm_n'}{\sqrt{2m_nm_n'}}\geq x\right\} &\leq \mathbf{P}\left\{\frac{2nI_n(\nu_n,\nu)-m_nm_n'}{\sqrt{2m_nm_n'}}\geq x\right\}\\ &\to 1-\Phi(x),\end{aligned}$$

which results in a test rejecting the independence if

$$\frac{2nI_n(\nu_n,\mu_{n,1}\times\mu_{n,2})-m_nm_n'}{\sqrt{2m_nm_n'}}\geq\Phi^{-1}(1-\alpha),$$

or equivalently

$$I_n(\nu_n,\mu_{n,1}\times\mu_{n,2})\geq\frac{\Phi^{-1}(1-\alpha)\sqrt{2m_nm_n'}+m_nm_n'}{2n}.$$

Note that unlike the L_1 case, the ratio of the strong consistent threshold to the asymptotic threshold increases for increasing n.

4. Kernel-based Statistic

We now present a second class of approaches to independence testing, based on a kernel statistic. We can derive this statistic in a number of ways. The most immediate interpretation, introduced by Rosenblatt (1975), defines the statistic as the L_2 distance between the joint density estimate and the product of marginal density estimates. Let K and K' be density functions (called kernels) defined on $\mathbb{R}^d$ and on $\mathbb{R}^{d'}$, respectively. For the bandwidth $h>0$, define

$$K_h(x)=\frac{1}{h^d}K\left(\frac{x}{h}\right)\quad\text{and}\quad K_h'(y)=\frac{1}{h^{d'}}K'\left(\frac{y}{h}\right).$$

The Rosenblatt-Parzen kernel density estimates of the density of (X,Y) and X are respectively

$$f_n(x,y)=\frac{1}{n}\sum_{i=1}^n K_h(x-X_i)K_h'(y-Y_i)\text{ and }f_{n,1}(x)=\frac{1}{n}\sum_{i=1}^n K_h(x-X_i),\tag{15}$$

with $f_{n,2}(y)$ defined by analogy. Rosenblatt (1975) introduced the kernel-based independence statistic

$$T_n=\int_{\mathbb{R}^d\times\mathbb{R}^{d'}}(f_n(x,y)-f_{n,1}(x)f_{n,2}(y))^2dxdy.$$

Alternatively, defining

$$L_h(x)=\int_{\mathbb{R}^d}K_h(u)K_h(x-u)du=\frac{1}{h^d}\int_{\mathbb{R}^d}K(u)K(x-u)du$$

and $L_h'(y)$ by analogy, we may write the kernel test statistic

$$\begin{aligned}T_n=&\tfrac{1}{n^2}\textstyle\sum_{i=1}^n\sum_{j=1}^n L_h(X_i-X_j)L_h'(Y_i-Y_j)\\&-\tfrac{2}{n^3}\textstyle\sum_{i=1}^n\left(\sum_{j=1}^n L_h(X_i-X_j)\right)\left(\sum_{j=1}^n L_h'(Y_i-Y_j)\right)\\&+\left(\tfrac{1}{n^2}\textstyle\sum_{i=1}^n\sum_{j=1}^n L_h(X_i-X_j)\right)\left(\tfrac{1}{n^2}\textstyle\sum_{i=1}^n\sum_{j=1}^n L_h'(Y_i-Y_j)\right).\end{aligned}\tag{16}$$

Note that at independence, the expected value of the statistic is not zero, but

$$\begin{aligned}\mathbf{E}\{T_n\} &= \frac{n-1}{n^2}\left(L_h(0)-\mathbf{E}\{L_h(X_1-X_2)\}\right)\left(L'_h(0)-\mathbf{E}\{L'_h(Y_1-Y_2)\}\right) \\ &\leq n^{-1}L_h(0)L'_h(0) = (nh^d h^{d'})^{-1}\|K\|^2\|K'\|^2\end{aligned} \tag{17}$$

A second interpretation of the above statistic is as a smoothed difference between the joint characteristic function and the product of the marginals (Feuerverger, 1993; Kankainen, 1995; Ushakov, 1999). The characteristic function and Rosenblatt-Parzen window statistics can be quite similar: in fact, for appropriate smoothing and kernel choices and *fixed* n, they may be identical (Kankainen, 1995, p. 54, demonstrates this for a Gaussian kernel). That said, a number of important differences exist between the characteristic function-based statistic and that of Rosenblatt (1975). Most crucially, the kernel bandwidth is kept fixed for the characteristic function-based test, rather than decreasing as n rises (a decreasing bandwidth is needed to ensure consistency of the kernel density estimates), resulting in very different forms for the null distribution; and there are more restrictive conditions on the Rosenblatt-Parzen test statistic (Rosenblatt, 1975, conditions a.1-a.4). These issues are discussed in detail by Feuerverger (1993, Section 5) and Kankainen (1995, Section 5.4).

A further generalization of the statistic is presented by Gretton et al. (2005a, 2008), in terms of covariances between feature mappings of the random variables to reproducing kernel Hilbert spaces (RKHSs). We now briefly review this interpretation, beginning with some necessary terminology and definitions. Let $\mathcal{F}$ be an RKHS, with the continuous feature mapping $\phi(x) \in \mathcal{F}$ for each $x \in \mathbb{R}^d$, such that the inner product between the features is given by the positive definite kernel function $L_h(x,x') := \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$. Likewise, let $\mathcal{G}$ be a second RKHS on $\mathbb{R}^{d'}$ with kernel $L'_h(\cdot,\cdot)$ and feature map $\psi(y)$. Following Baker (1973) and Fukumizu et al. (2004), the cross-covariance operator $C_\nu : \mathcal{G} \to \mathcal{F}$ for the measure ν is defined such that for all $f \in \mathcal{F}$ and $g \in \mathcal{G}$,

$$\langle f, C_\nu g \rangle_{\mathcal{F}} = \mathbf{E}\left([f(X)-\mathbf{E}(f(X))]\,[g(Y)-\mathbf{E}(g(Y))]\right).$$

The cross-covariance operator can be thought of as a generalisation of a cross-covariance matrix between the (potentially infinite dimensional) feature mappings $\phi(x)$ and $\psi(y)$.

To see how this operator may be used to test independence, we recall the following characterization of independence (see, e.g., Jacod and Protter, 2000, Theorem 10.1e):

Theorem 5 *The random variables X and Y are independent if and only if* $\mathrm{cov}(f(X),g(Y)) = 0$ *for any pair (f,g) of bounded, continuous functions.*

While the bounded continuous functions are too rich a class to permit the construction of a covariance-based test statistic on a sample, Fukumizu et al. (2008) and Sriperumbudur et al. (2008) show that when $\widetilde{\mathcal{F}}$ is the unit ball in a *characteristic*[2] RKHS $\mathcal{F}$, and $\widetilde{\mathcal{G}}$ the unit ball in a characteristic RKHS $\mathcal{G}$, then

$$\sup_{f\in\widetilde{\mathcal{F}},g\in\widetilde{\mathcal{G}}} \mathbf{E}\left([f(X)-\mathbf{E}(f(X))]\,[g(Y)-\mathbf{E}(g(Y))]\right) = 0 \qquad \Longleftrightarrow \qquad \nu = \mu_1 \times \mu_2.$$

2. The reader is referred to Fukumizu et al. (2008) and Sriperumbudur et al. (2008) for conditions under which an RKHS is characteristic. We note here that the Gaussian kernel on $\mathbb{R}^d$ has this property, and provide further discussion below.

In other words, the spectral norm of the covariance operator C_ν between characteristic RKHSs is zero only at independence. Rather than the maximum singular value, we may use the squared Hilbert-Schmidt norm (the sum of the squared singular values), which has a population expression

$$\begin{aligned} \mathrm{H}(\nu;\mathcal{F},\mathcal{G}) &= \mathbf{E}\{L_h(X_1-X_2)L'_h(Y_1-Y_2)\} - 2\mathbf{E}\left\{\mathbf{E}\{L_h(X_1-X_2)|X_1\}\mathbf{E}\{L_h(Y_1-Y_2)|Y_1\}\right\} \\ &\quad +\mathbf{E}\{L_h(X_1-X_2)\}\mathbf{E}\{L'_h(Y_1-Y_2)\} \end{aligned}$$

(see Gretton et al., 2005a, Lemma 1): we call this the Hilbert-Schmidt independence criterion (HSIC).

The test statistic in (16) is then interpreted as a biased empirical estimate of $\mathrm{H}(\nu;\mathcal{F},\mathcal{G})$. Clearly, when K_h and K'_h are continuous and square integrable densities, the induced kernels L_h and L'_h are continuous positive definite RKHS kernels. However, as long as L_h and L'_h are characteristic kernels, then $\mathrm{H}(\nu;\mathcal{F},\mathcal{G}) = 0$ iff X and Y independent. The Gaussian and Laplace kernels are characteristic on $\mathbb{R}^d$ (Fukumizu et al., 2008), and universal kernels (in the sense of Steinwart, 2001) are characteristic on compact domains (Gretton et al., 2005a, Theorem 6). Sriperumbudur et al. (2008) provide a simple necessary and sufficient condition for a bounded continuous translation invariant kernel to be characteristic on $\mathbb{R}^d$: the Fourier spectrum of the kernel must be supported on the entire domain. Note that characteristic kernels need not be inner products of square integrable probability density functions: an example is the kernel

$$L_h(x_1,x_2) = \exp(x_1^T x_2/h)$$

from Steinwart (2001, Section 3, Example 1), which is universal, hence characteristic on compact subsets of $\mathbb{R}^d$. Moreover, an appropriate choice of kernels allows testing of dependence in non-Euclidean settings, such as distributions on strings and graphs (Gretton et al., 2008).

Finally, while we have focused on a kernel dependence measure based on the covariance, alternative kernel dependence measures exist based on the canonical correlation. Dauxois and Nkiet (1998) propose the canonical correlation between variables in a spline-based RKHS as a statistic for an independence test: this dependence measure follows the suggestion of Rényi (1959), but with a more restrictive pair of function classes used to compute the correlation (rather than the set of all square integrable functions). The variables are assumed in this case to be univariate. Likewise, Bach and Jordan (2002) use the canonical correlation between RKHS feature mappings as a measure of dependence between pairs of random variables (although they do not address the problem of hypothesis testing). Bach and Jordan employ a different regularization strategy to Dauxois and Nkiet, however, which is a roughness penalty on the canonical correlates, rather than projection on a finite basis. For an appropriate rate of decay of the regularization with increasing sample size, the empirical estimate of the canonical correlation converges in probability (Leurgans et al., 1993; Fukumizu et al., 2007). Fukumizu et al. (2008) provide a consistent RKHS-based estimate of the mean-square contingency, which is also based on the canonical correlation. This final independence measure is asymptotically independent of the kernel choice. When used as a statistic in an independence test, the kernel contingency was found empirically to have power superior to the HSIC-based test.

4.1 Strongly Consistent Test

The empirical statistic T_n was previously shown by Gretton et al. (2005a) to converge in probability to its expectation with rate $1/\sqrt{n}$. Given $0 \le L_h(0)L'_h(0) \le 1$, the corresponding result is

$$\mathbf{P}(T_n - \mathbf{E}(T_n) \ge \varepsilon^2) \le 3e^{-0.24n\varepsilon^4},$$

which follows from the straightforward application of a bound by Hoeffding (1963, p. 25). We now provide a more refined bound which scales better with ε, and is thus tighter when the bandwidth h decreases.

We will obtain our results for the semi-statistic

$$\tilde{T}_n = \|f_n(\cdot,\cdot) - \mathbf{E}f_n(\cdot,\cdot)\|^2,$$

since under the null hypothesis,

$$\begin{aligned}
\sqrt{T_n} &= \|f_n(\cdot,\cdot) - f_{n,1}(\cdot)f_{n,2}(\cdot)\| \\
&\leq \|f_n(\cdot,\cdot) - \mathbf{E}f_n(\cdot,\cdot)\| + \|f_{n,1}(\cdot)f_{n,2}(\cdot) - \mathbf{E}f_{n,1}(\cdot)\mathbf{E}f_{n,2}(\cdot)\| \\
&\leq \sqrt{\tilde{T}_n} + \|f_{n,1}(\cdot)(f_{n,2}(\cdot) - \mathbf{E}f_{n,2}(\cdot))\| + \|(f_{n,1}(\cdot) - \mathbf{E}f_{n,1}(\cdot))\mathbf{E}f_{n,2}(\cdot)\| \\
&= \sqrt{\tilde{T}_n} + \|f_{n,1}(\cdot)\|\,\|f_{n,2}(\cdot) - \mathbf{E}f_{n,2}(\cdot)\| + \|f_{n,1}(\cdot) - \mathbf{E}f_{n,1}(\cdot)\|\,\|\mathbf{E}f_{n,2}(\cdot)\| \\
&\approx \sqrt{\tilde{T}_n}.
\end{aligned}$$

Theorem 6 *For any* $\varepsilon > 0$,

$$\mathbf{P}\left\{\tilde{T}_n \geq \left(\varepsilon + \mathbf{E}\left\{\sqrt{\tilde{T}_n}\right\}\right)^2\right\} \leq e^{-n\varepsilon^2 \big/ (2L_h(0)L'_h(0))}.$$

Proof We apply the McDiarmid inequality (cf. McDiarmid, 1989): Let $Z_1, \ldots, Z_n$ be independent random variables taking values in a set A and assume that $f : A^n \to \mathbb{R}$ satisfies

$$\sup_{\substack{z_1,\ldots,z_n,\\ z'_i \in A}} |f(z_1,\ldots,z_n) - f(z_1,\ldots,z_{i-1},z'_i,z_{i+1},\ldots,z_n)| \leq c_i,\ 1 \leq i \leq n.$$

Then, for all $\varepsilon > 0$,

$$\mathbf{P}\{f(Z_1,\ldots,Z_n) - \mathbf{E}f(Z_1,\ldots,Z_n) \geq \varepsilon\} \leq e^{-2\varepsilon^2 \big/ \sum_{i=1}^n c_i^2}.$$

Because of

$$\begin{aligned}
\sqrt{\tilde{T}_n} &= \|f_n(\cdot,\cdot) - \mathbf{E}f_n(\cdot,\cdot)\| \\
&= \|\frac{1}{n}\sum_{i=1}^n K_h(\cdot - X_i)K'_h(\cdot - Y_i) - \mathbf{E}f_n(\cdot,\cdot)\| \\
&\leq \|\frac{1}{n}K_h(\cdot - X_1)K'_h(\cdot - Y_1)\| + \|\frac{1}{n}\sum_{i=2}^n K_h(\cdot - X_i)K'_h(\cdot - Y_i) - \mathbf{E}f_n(\cdot,\cdot)\|
\end{aligned}$$

we can apply McDiarmid inequality with

$$\frac{2}{n}\|K_h(\cdot - X_1)K'_h(\cdot - Y_1)\| = \frac{2}{n}\sqrt{L_h(0)L'_h(0)} =: c_i = c_1,$$

where we note that the c_i are independent of i, and can be replaced by a single c_1. Thus,

$$\begin{aligned}
\mathbf{P}\left\{\sqrt{\tilde{T}_n} - \mathbf{E}\left\{\sqrt{\tilde{T}_n}\right\} \geq \varepsilon\right\} &\leq e^{-2\varepsilon^2 \big/ \sum_{i=1}^n c_i^2} \\
&= e^{-2\varepsilon^2 \big/ (nc_1^2)} \\
&\leq e^{-n\varepsilon^2 \big/ (2L_h(0)L'_h(0))}.
\end{aligned}$$

This implies

$$\mathbf{P}\left\{\tilde{T}_n \geq \left(\varepsilon + \mathbf{E}\left\{\sqrt{\tilde{T}_n}\right\}\right)^2\right\} \leq e^{-n\varepsilon^2/(2L_h(0)L_h'(0))}.$$

■

From these inequalities we can derive a test of independence. Choose ε such that

$$n\varepsilon^2/(2L_h(0)L_h'(0)) = 2\ln n.$$

Because of

$$\mathbf{E}\{\tilde{T}_n\} \approx \mathbf{E}\{T_n\} \leq \frac{L_h(0)L_h'(0)}{n},$$

we choose the threshold

$$\left(\sqrt{\frac{L_h(0)L_h'(0)4\ln n}{n}} + \sqrt{\frac{L_h(0)L_h'(0)}{n}}\right)^2 = \frac{L_h(0)L_h'(0)}{n}(\sqrt{4\ln n}+1)^2,$$

that is, we reject the hypothesis of independence if

$$T_n > \frac{\|K\|^2\|K'\|^2}{nh^d h^{d'}}(\sqrt{4\ln n}+1)^2.$$

It follows from

$$\begin{aligned}
& \mathbf{P}\left\{T_n \geq \frac{L_h(0)L_h'(0)}{n}(\sqrt{4\ln n}+1)^2\right\} \\
\approx\ & \mathbf{P}\left\{\tilde{T}_n \geq \left(\sqrt{\frac{L_h(0)L_h'(0)4\ln n}{n}} + \sqrt{\frac{L_h(0)L_h'(0)}{n}}\right)^2\right\} \\
\leq\ & \mathbf{P}\left\{\tilde{T}_n \geq \left(\sqrt{\frac{L_h(0)L_h'(0)4\ln n}{n}} + \sqrt{\mathbf{E}\{\tilde{T}_n\}}\right)^2\right\} \\
\leq\ & \mathbf{P}\left\{\tilde{T}_n \geq \left(\sqrt{\frac{L_h(0)L_h'(0)4\ln n}{n}} + \mathbf{E}\left\{\sqrt{\tilde{T}_n}\right\}\right)^2\right\} \\
\leq\ & e^{-2\ln n}
\end{aligned}$$

that this test of independence is strongly consistent.

Under the alternative hypothesis, there are two cases:

- If $h \to 0$ and the density f exists and is square integrable, then

$$T_n \to \|f - f_1 f_2\|^2 > 0$$

a.s. The strong consistency is not distribution-free, since ν must have a square integrable density.

- If h is fixed, the strong law of large numbers implies

$$\begin{aligned} T_n &\rightarrow \mathbf{E}\{L_h(X_1-X_2)L'_h(Y_1-Y_2)\} - 2\mathbf{E}\{\mathbf{E}\{L_h(X_1-X_2)|X_1\}\mathbf{E}\{L_h(Y_1-Y_2)|Y_1\}\} \\ &\quad +\mathbf{E}\{L_h(X_1-X_2)\}\mathbf{E}\{L'_h(Y_1-Y_2)\} \\ &=: \mathrm{H}(\nu;\mathcal{F},\mathcal{G}) \end{aligned}$$

  If K_h and K'_h are continuous and square integrable densities, the induced kernels L_h and L'_h are continuous positive definite kernels: $\mathrm{H}(\nu;\mathcal{F},\mathcal{G})$ is then the squared Hilbert-Schmidt norm of the covariance operator for ν. We may replace L_h and L'_h with any *characteristic kernels* (in the sense of Fukumizu et al., 2008; Sriperumbudur et al., 2008), however, and retain the property $\mathrm{H}(\nu;\mathcal{F},\mathcal{G}) = 0$ iff X and Y independent. In this case, the strong consistency is distribution-free.

4.2 Approximately α-level Tests

We now describe the asymptotic limit distribution of the test statistic T_n in (16). We address two cases: first, when the kernel bandwidth decreases, and second, when it remains fixed.

Let us consider the case where $K_h(x)$ and $K'_h(y)$ are intended to be used in a Rosenblatt-Parzen density estimator, as in (15). The corresponding density estimates in T_n are mean square consistent if $h = h_n$ such that

$$h_n \rightarrow 0 \quad \text{and} \quad nh_n^d h_n^{d'} \rightarrow \infty. \tag{18}$$

Based on the results of Hall (1984), Cotterill and Csörgő (1985) and Beirlant and Mason (1995), we expect that, under these consistency conditions,

$$\frac{T_n - \mathbf{E}\{T_n\}}{\sqrt{\mathrm{var}(T_n)}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1).$$

We next calculate $\mathrm{var}(T_n) \approx \mathrm{var}(\tilde{T}_n)$. Under the null hypothesis,

$$\begin{aligned} \tilde{T}_n &= \|f_n(\cdot,\cdot) - \mathbf{E}f_n(\cdot,\cdot)\|^2 \\ &= \left\| \frac{1}{n}\sum_{i=1}^n (K_h(\cdot - X_i)K'_h(\cdot - Y_i) - \mathbf{E}\{K_h(\cdot - X)K'_h(\cdot - Y)\}) \right\|^2 \\ &= \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n \Big((K_h(\cdot - X_i)K'_h(\cdot - Y_i) - \mathbf{E}\{K_h(\cdot - X)K'_h(\cdot - Y)\}) \times \\ &\qquad (K_h(\cdot - X_j)K'_h(\cdot - Y_j) - \mathbf{E}\{K_h(\cdot - X)K'_h(\cdot - Y)\}) \Big) \\ &=: \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n M_h(X_i,Y_i,X_j,Y_j), \end{aligned}$$

and therefore

$$\mathrm{var}(\tilde{T}_n) = \frac{1}{n^4}\sum_{i=1}^n\sum_{j=1}^n\sum_{i'=1}^n\sum_{j'=1}^n \mathrm{cov}(M_h(X_i,Y_i,X_j,Y_j), M_h(X_{i'},Y_{i'},X_{j'},Y_{j'})).$$

One can check that

$$\mathrm{cov}(M_h(X_i,Y_i,X_j,Y_j), M_h(X_{i'},Y_{i'},X_{j'},Y_{j'})) = 0$$

unless $(i,j) = (i',j')$ or $(i,j) = (j',i')$. Thus,

$$\begin{aligned}\mathrm{var}(\tilde{T}_n) &= \frac{1}{n^4}\left(n\mathrm{var}(M_h(X_1,Y_1,X_1,Y_1)) + 2n(n-1)\mathrm{var}(M_h(X_1,Y_1,X_2,Y_2))\right)\\ &\approx \frac{2}{n^2}\mathrm{var}(M_h(X_1,Y_1,X_2,Y_2)).\end{aligned}$$

If $h \to 0$ then

$$\frac{2}{n^2}\mathrm{var}(M_h(X_1,Y_1,X_2,Y_2)) \approx \frac{2\|f\|^2}{n^2 h^d h^{d'}}, \tag{19}$$

therefore a possible form for the asymptotic normal distribution is

$$nh^{d/2}h^{d'/2}(T_n - \mathbf{E}\{T_n\})/\sigma \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

where

$$\sigma^2 = 2\|f\|^2.$$

Thus the asymptotic α-level test rejects the null hypothesis if

$$T_n > \mathbf{E}\{T_n\} + \frac{\sigma}{nh^{d/2}h^{d'/2}}\Phi^{-1}(1-\alpha),$$

where $\mathbf{E}\{T_n\}$ may be replaced by its upper bound,

$$L_h(0)L'_h(0)/n = \|K\|^2\|K'\|^2/(nh^d h^{d'}).$$

The only problem left is that the threshold is not distribution-free: σ depends on the unknown f. The simplest distribution-free bound for the variance,

$$\sigma^2 \leq \frac{\|K\|^4\|K'\|^4}{n^2 h^{2d} h^{2d'}}$$

is unsatisfactory since its performance as a function of h is worse than the result (19). An improved distribution-free bound on the variance (for both fixed and decreasing h) is a topic for future research: we give an empirical estimate below (Equation 20) for use in asymptotic hypothesis tests.

We now consider the case of fixed h. Following Feuerverger (1993); Serfling (1980), the distribution of T_n under $\mathcal{H}_0$ is

$$nT_n \xrightarrow{D} \sum_{l=1}^{\infty} \lambda_l z_l^2,$$

where $z_l \sim \mathcal{N}(0,1)$ i.i.d., and λ_l are the solutions to an eigenvalue problem depending on the unknown distribution of X and Y (see Gretton et al., 2008, Theorem 2 for details).

A difficulty in using the statistic (16) in a hypothesis test therefore arises due to the form of the null distribution of the statistic, which is a function of the unknown distribution over X and Y, whether or not h is fixed. In the case of h decreasing according to (18), we may use an empirical estimate of the variance of T_n under $\mathcal{H}_0$ due to Gretton et al. (2008, Theorem 4). Denoting by $\odot$ the entrywise matrix product and $A^{\cdot 2}$ the entrywise matrix power,

$$\mathrm{var}(T_n) = \mathbf{1}^\top\left(\mathbf{B} - \mathrm{diag}(\mathbf{B})\right)\mathbf{1}, \tag{20}$$

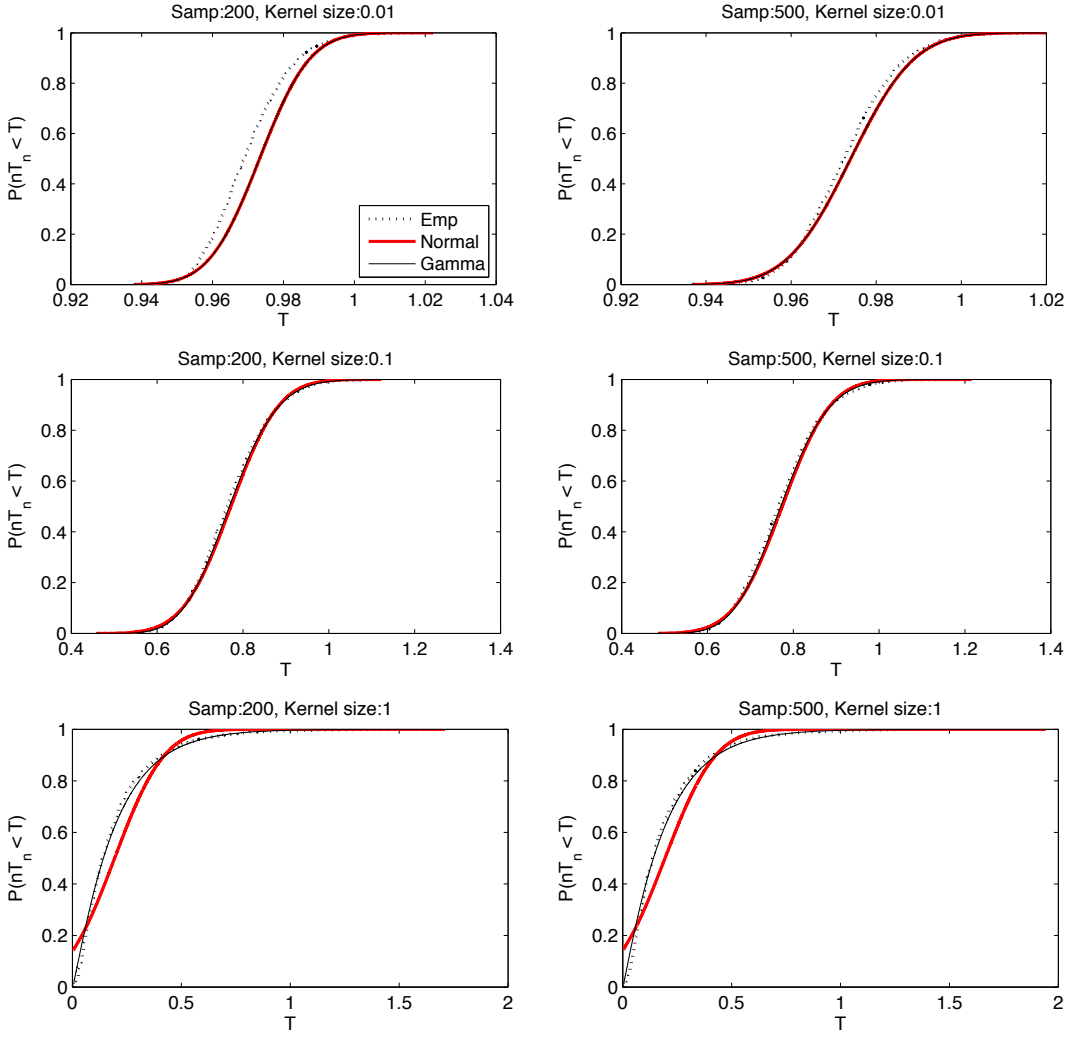


Figure 1: Simulated cumulative distribution function of T_n (*Emp*) under $\mathcal{H}_0$ for $n = 200$ (left column) and $n = 500$ (right column), compared with the two-parameter Gamma distribution (*Gamma*) and the Normal distribution (*Normal*). The empirical CDF was obtained empirically using 5000 independent draws of T_n. Both the parametric approximations are fit using the mean and variance in Equations (17) and (20). "Samp" is the number n of samples, and the bandwidth is h.

where

$$\mathbf{B} = \left((\mathbf{HLH}) \odot \left(\mathbf{HL}'\mathbf{H}\right)\right)^{\cdot 2},$$

$\mathbf{L}$ is a matrix with entries $L_h(X_i - X_j)$, $\mathbf{L}'$ is a matrix with entries $L'_h(Y_i - Y_j)$, $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\top$ is a centering matrix, and $\mathbf{1}$ an $n \times 1$ vector of ones.

Two approaches have been proposed in the case of fixed h to obtain appropriate quantiles of the null distribution for hypothesis testing: repeated shuffling of the sample (Feuerverger, 1993), and

approximation by a two-parameter Gamma density (Kankainen, 1995),

$$nT_n \sim \frac{x^{\alpha-1}e^{-x/\beta}}{\beta^\alpha \Gamma(\alpha)}$$
$$\text{where} \quad \alpha = \frac{(\mathbf{E}\{T_n\})^2}{\text{var}(T_n)}, \quad \beta = \frac{n\text{var}(T_n)}{\mathbf{E}\{T_n\}},$$

and $\mathbf{E}\{T_n\}$ is given in (17). This Gamma approximation was found by Gretton et al. (2008) to perform identically on the Section 5 benchmark data to the more computationally expensive approach of Feuerverger (1993). We emphasize, however, that this approximation is a heuristic: no guarantees are provided regarding the asymptotic performance of this approximation in terms of Type II error, nor is it established under what conditions the approximation fails.

We end this section with an empirical comparison between the Normal and two-parameter Gamma null distribution approximations, and the null CDF generated by repeated independent samples of T_n. We chose X and Y to be independent and univariate, with X having a uniform distribution and Y being a symmetric bimodal mixture of Gaussians. Both variables had zero mean and unit standard deviation. Results are plotted in Figure 1.

We observe that as the bandwidth increases, the Gamma approximation of T_n becomes more accurate (although it is always good for large quantiles, which is the region most important to a hypothesis test). The Normal approximation is very close to the Gamma approximation for small bandwidths, but is less accurate (with respect to both the Gamma distribution and the simulated CDF) for larger bandwidths. Finally, for the smallest bandwidth ($h = 0.01$), both approximate null distributions become more accurate for increasing n (for larger kernel sizes, the effect is too small to see on the plots). We will return to these points in the next section when analysing our experimental results.

5. Numerical Results

In comparing the independence tests, we made use of the multidimensional benchmark data proposed by Gretton et al. (2008). We tested the independence in two, four, and six dimensions (i.e., $d \in 1,2,3$ and $d = d'$). The data were constructed as follows. First, we generated n samples of two independent univariate random variables, each drawn at random from the ICA benchmark densities of Bach and Jordan (2002, Figure 5): these included super-Gaussian, sub-Gaussian, multimodal, and unimodal distributions, with the common property of zero mean and unit variance. The densities are described in Table 5, as reproduced from Gretton et al. (2005b, Table 3). Second, we mixed these random variables using a rotation matrix parametrised by an angle θ, varying from 0 to $\pi/4$ (a zero angle meant the data were independent, while dependence became easier to detect as the angle increased to $\pi/4$: see the two plots in Figure 2). Third, in the cases $d = 2$ and $d = 3$, independent Gaussian noise of zero mean and unit variance was used to fill the remaining dimensions, and the resulting vectors were multiplied by independent random two- or three-dimensional orthogonal matrices, to obtain random vectors X and Y dependent across all observed dimensions. We emphasise that classical approaches (such as Spearman's ρ or Kendall's τ) are unable to find this dependence, since the variables are uncorrelated; nor can we recover the subspace in which the variables are dependent using PCA, since this subspace has the same second order properties as the noise. We investigated sample sizes $n = 128, 512, 1024$, and 2048.

Label	Definition	Kurtosis
a	Student's t distribution, 3 DOF	∞
b	Double exponential	3.00
c	Uniform	-1.20
d	Students's t distribution, 5 DOF	6.00
e	Exponential	6.00
f	Mixture, 2 double exponentials	-1.70
g	Symmetric mixture 2 Gauss., multimodal	-1.85
h	Symmetric mixture 2 Gauss., transitional	-0.75
i	Symmetric mixture 2 Gauss., unimodal	-0.50
j	Asymm. mixture 2 Gauss., multimodal	-0.57
k	Asymm. mixture 2 Gauss., transitional	-0.29
l	Asymm. mixture 2 Gauss., unimodal	-0.20
m	Symmetric mixture 4 Gauss., multimodal	-0.91
n	Symmetric mixture 4 Gauss., transitional	-0.34
o	Symmetric mixture 4 Gauss., unimodal	-0.40
p	Asymm. mixture 4 Gauss., multimodal	-0.67
q	Asymm. mixture 4 Gauss., transitional	-0.59
r	Asymm. mixture 4 Gauss., unimodal	-0.82

Table 1: Labels of distributions used in the independence test benchmarks, and their respective kurtoses. All distributions have zero mean and unit variance.

We compared three different asymptotic independence testing approaches based on space partitioning: the L_1 test, denoted *L1*; the log likelihood test *Like*; and a third test, *Pears*, based on a conjecture regarding the asymptotic distribution of the Pearson χ^2 statistic

$$\chi^2_n(\nu_n, \mu_{n,1} \times \mu_{n,2}) \quad = \quad \sum_{A \in \mathcal{P}_n} \sum_{B \in Q_n} \frac{(\nu_n(A \times B) - \mu_{n,1}(A) \cdot \mu_{n,2}(B))^2}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}$$

(see Appendix B for details, and for a further conjecture regarding a strongly consistent test for the χ^2_n statistic). The number of discretisations per dimension was set at $m_n = m'_n = 4$, besides in the $n = 128, d = 2$ case and the $d = 3$ cases, where it was set at $m_n = m'_n = 3$: for the latter values of n and d, there were too few samples per bin when a greater number of partitions were used, causing poor performance. We divided our spaces $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ into roughly equiprobable bins. Further increases in the number of partitions per dimension, where sufficient samples were present to justify this (i.e., the $n = 512, d = 1$ case), resulted only in very minor shifts in performance.

We compared the partitioning approaches with the kernel approach from Section 4, using both the Gamma *Ker(g)* and Normal *Ker(n)* approximations to the null distribution. Our kernels were Gaussian for both X and Y, with bandwidths set to the median distance between samples of the respective variables. Note that a more sophisticated but computationally costly approach to bandwidth selection is described by Fukumizu et al. (2008), which involves matching the closed-form expression for the variance of T_n in (20) with an estimate obtained by data shuffling.

Results are plotted in Figure 3 (average over 500 independent generations of the data). The y-intercept on these plots corresponds to the acceptance rate of $\mathcal{H}_0$ at independence, or $1 - (\text{Type I error})$,

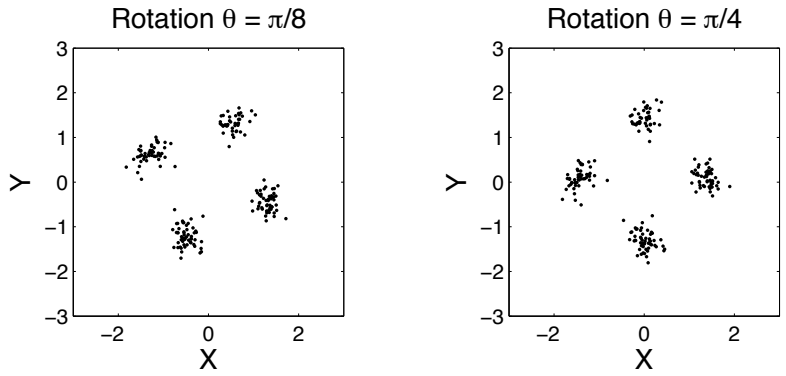


Figure 2: Example data set for $d = d' = 1, n = 200$, and rotation angles $\theta = \pi/8$ (left) and $\theta = \pi/4$ (right). In this case, both sources are mixtures of two Gaussians (source *(g)* in Gretton et al., 2005b, Table 3).

and should be close to the design parameter of $1 - \alpha = 0.95$. Elsewhere, the plots indicate acceptance of $\mathcal{H}_0$ where the underlying variables are dependent, that is, the Type II error.

As expected, we observe dependence becomes easier to detect as θ increases from 0 to $\pi/4$, when n increases, and when d decreases. Although no tests are reliable for small θ, several tests do well as θ approaches $\pi/4$ (besides the case of $n = 128,\ d = 2$). The L_1 test has a lower Type II error than the χ^2 test when the number of samples per partition is small ($n = 128, d = 1$, $n = 128, d = 2$, and $n = 1024, d = 3$), but this advantage is lessened for larger numbers of samples per partition. The log-likelihood test generally has the lowest Type II error of the three partition-based tests, however it gives a Type I error larger than the design parameter of 0.05 when the number of samples per bin is insufficient: this problem is severe in the case $n = 1024$ and $d = 3$, but can also be observed at $n = 2048, d = 3$ (for larger sample sizes $n = 3072, d = 3$ and $n = 4096, d = 3$, the Type I error of the log-likelihood test was at or below the design value). This suggests the log-likelihood test is more susceptible to bias for small numbers of samples per bin than the L_1 and χ^2 tests. In the remaining cases, performance of the log-likelihood test and the L_1 test is comparable, besides in the case $n = 512, d = 2$, where the log-likelihood test has an advantage.

The superior performance of the log-likelihood test compared with the χ^2 test (in the cases $d = 1$ and $d = 2$) might arise due to the different convergence properties of the two test statistics. In particular, we note the superior convergence behaviour of the goodness-of-fit statistic for the log likelihood (Equation 13), as compared with the χ^2 statistic (Equation 24 in Appendix B), in terms of the dependence of the latter on the number m_n of partitions used. By analogy, we anticipate the log-likelihood independence statistic $I_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$ will also converge faster than the Pearson χ^2 independence statistic $\chi^2_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$, and thus provide better test performance. A more formal discussion of this behaviour is a topic for future research.

In all cases, the kernel-based test has the lowest Type II error.[3] That said, one should bear in mind the kernel test thresholds require $\mathbf{E}\{T_n\}$ and $\mathrm{var}(T_n)$, which are unknown and must be estimated from the data using Equations (17) and (20), respectively. In other words, unlike the L_1

3. Aside from $n = 1024$ and $d = 3$, where the log-likelihood has a lower Type II error: we disregard this result since it is due to the log-likelihood test being affected by bias, as discussed above.

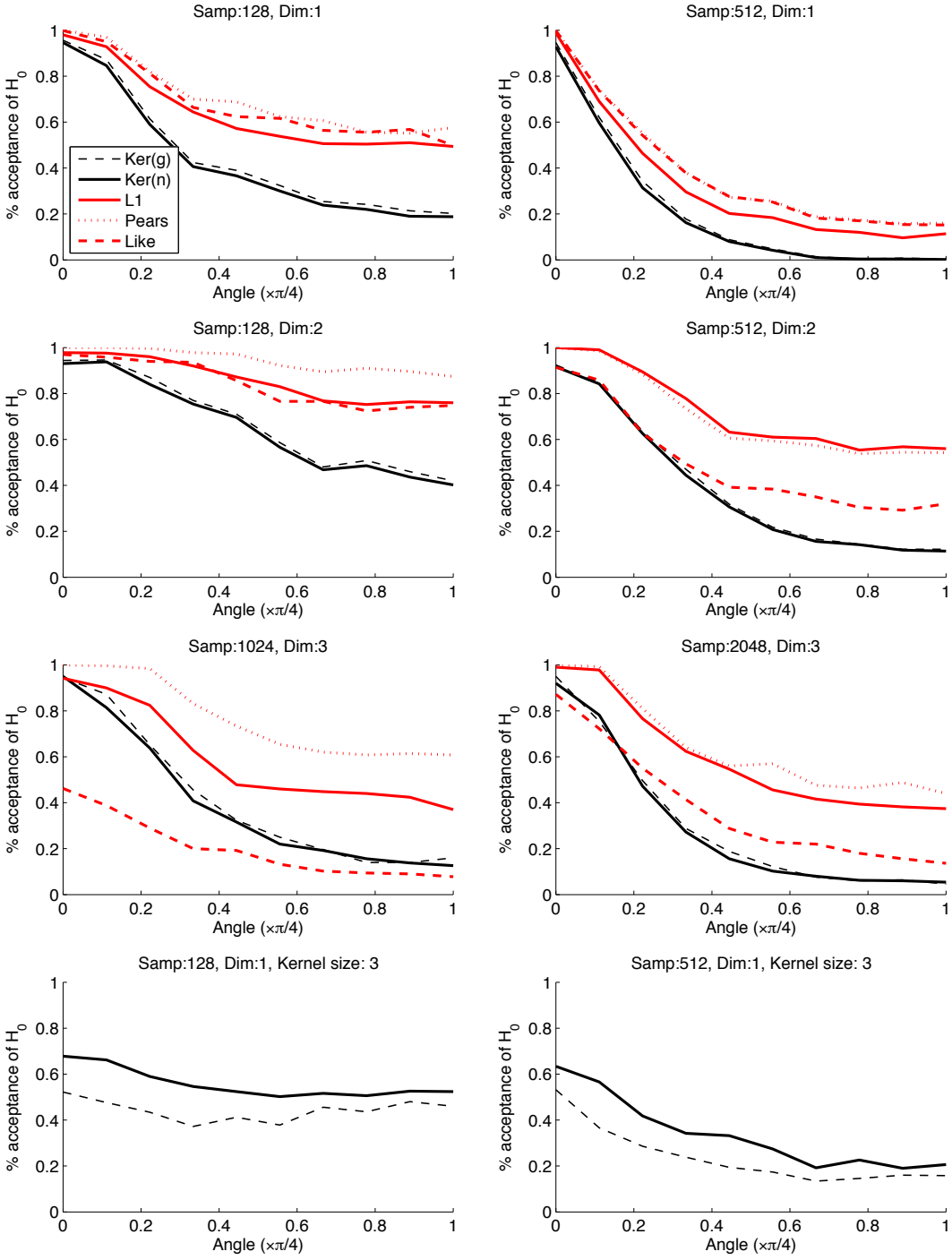


Figure 3: Rate of acceptance of $\mathcal{H}_0$ for the *Ker(g)*, *Ker(n)*, *L1*, *Pears*, and *Like* tests. "Samp" is the number n of samples, and "dim" is the dimension $d = d'$ of x and y. In the final row, the performance of the *Ker(g)* and *Ker(n)* tests is plotted for a large bandwidth $h = 3$, and $\tilde{\alpha} = 0.5$, to illustrate the difference between the Normal and two-parameter Gamma approximations to the null distribution.

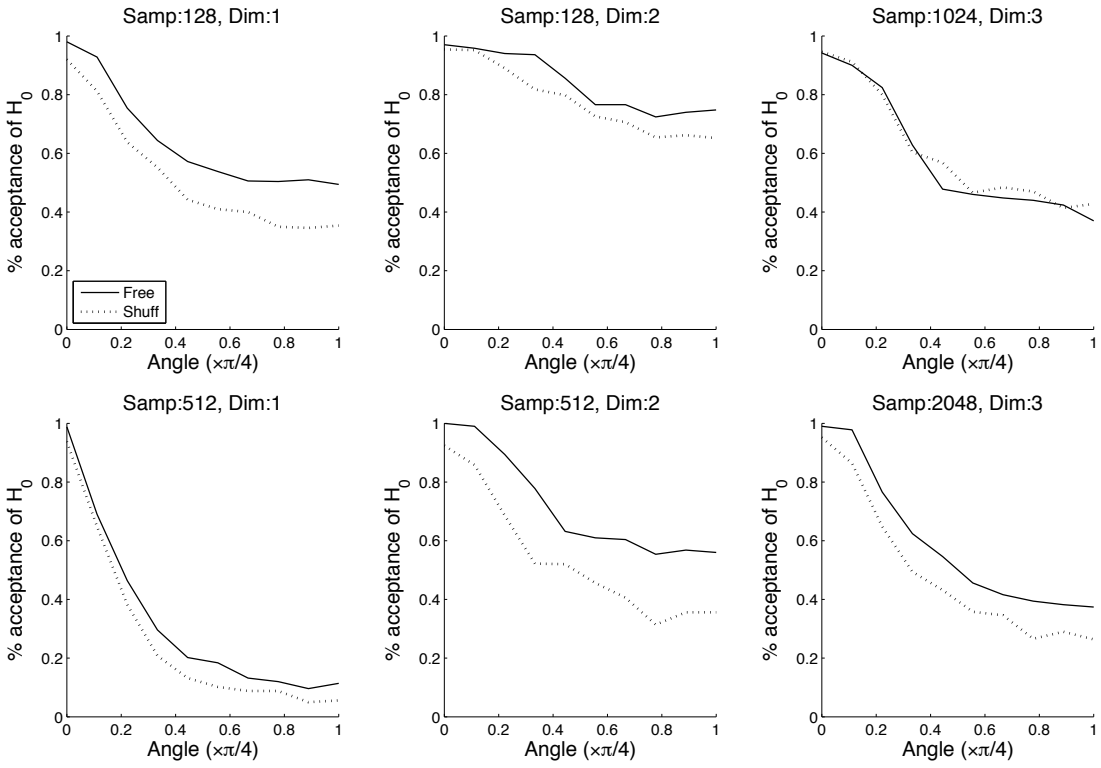


Figure 4: Rate of acceptance of $\mathcal{H}_0$ for the distribution-free (*Free*) and shuffling-based (*Shuff*) null distribution quantiles, using the L1 test statistic. "Samp" is the number n of samples, and "dim" is the dimension $d = d'$ of x and y.

and log likelihood tests, the kernel test thresholds in our experiments are themselves finite sample estimates (which we have not attempted to account for, and which could impact on test performance). Moreover, the Gamma approximation to the null distribution is simply a heuristic, with no asymptotic guarantees.

It is of interest to further investigate the null distribution approximation strategies for the kernel tests, and in particular to determine the effect on test performance of the observations made in Figure 1. Since the median distance between sample points was small enough in our previous experiments for the Normal and Gamma estimates to be very similar, we used an artificially high kernel bandwidth $h = 3$. In addition, we employed a much lower $\tilde{\alpha} = 0.5$, since this provided a more visible performance difference. The final row of Figure 3 shows the resulting test performance. We recall from Figure 1 that for large kernel sizes and $\tilde{\alpha} = 0.5$, the Gaussian approximation returns a larger threshold than the true CDF would require, and thus the Normal distribution has a lower Type I error (the error for very small values of α is in the opposite direction, but had a less pronounced effect in our experiments). The large bandwidth required to observe this behaviour results in a substantial performance penalty on the Type II error, however, and would not be used in practice.

An alternative approach to obtaining null distribution quantiles for test thresholds is via a shuffling procedure: the ordering of the $Y_1, \ldots, Y_n$ sample is permuted repeatedly while that of $X_1, \ldots, X_n$ sample is kept fixed, and the $1 - \alpha$ quantile is obtained from the resulting estimated cumulative distribution function of the test statistic. Again, we emphasize that unlike the asymptotic L1 and

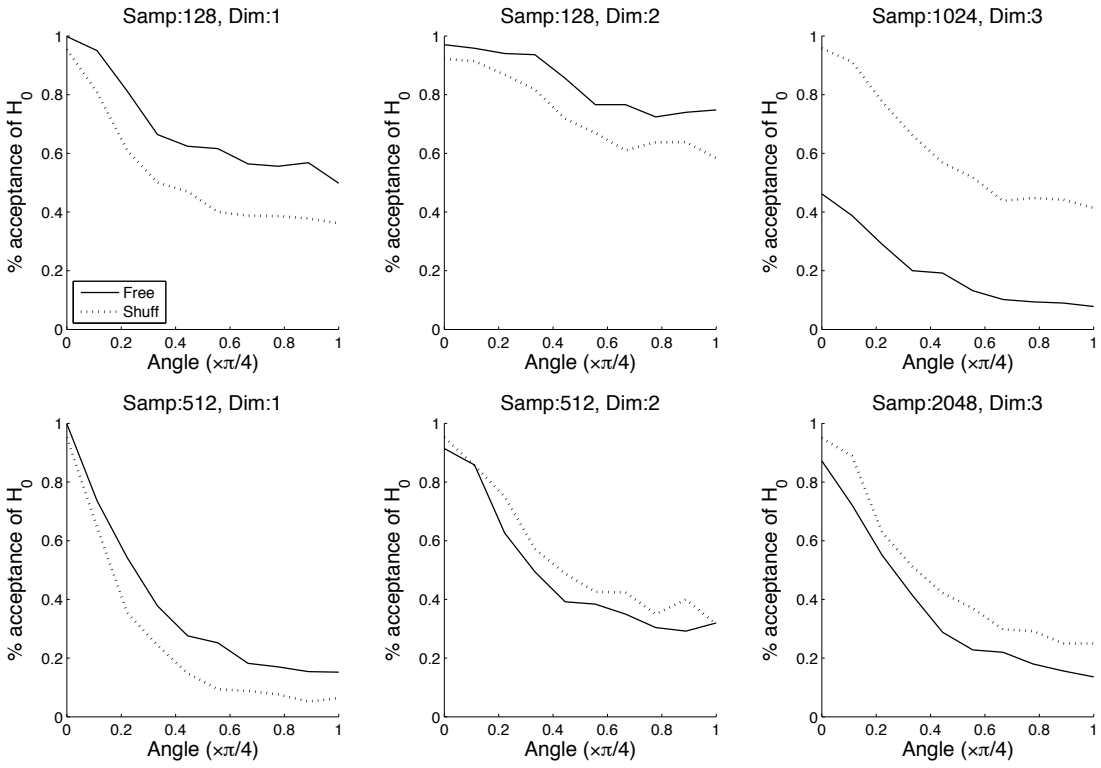


Figure 5: Rate of acceptance of $\mathcal{H}_0$ for the distribution-free (*Free*) and shuffling-based (*Shuff*) null distribution quantiles, using the log-likelihood test statistic. "Samp" is the number n of samples, and "dim" is the dimension $d = d'$ of x and y.

log-likelihood tests we have proposed, the resulting test threshold is an empirical estimate, and the convergence behaviour of this estimate is not accounted for. In our final experiments, we compared the performance of our asymptotic tests for *L1* and *Like* with this shuffling approach, for the same data as in our Figure 3 experiments.[4] We used $p = 200$ permutations in obtaining the approximation to the null distribution. Results for the *L1* case are plotted in Figure 4, and those for the *Like* case in Figure 5.

In the case of the *L1* statistic, we observe the distribution-free approach is conservative in terms of the Type I error, generally setting it slightly lower than the target value. The shuffling approach returns a lower Type II error, however it is notable that the performance difference is not particularly large with respect to our distribution-free threshold, and that apart from an offset, the error as a function of angle takes the same form. We should further bear in mind that the shuffling approach has a substantially greater computational cost (p times the cost of the distribution-free test). In the case of the *Like* statistic, we observe similar behaviour to *L1* in the cases $d = 1$ and $d = 2$. In the $d = 3$ case, however, the *Like* test gives too large a Type I error, and thus the Type II performance of the two approaches cannot be compared (although for $n = 2048$, the *Like* test is observed to approach the asymptotic regime, and the Type I performance is closer to the target value).

4. This comparison was made for the kernel statistic on these data by Gretton et al. (2008), and no performance difference was found.

6. Conclusion

We have described distribution-free strong consistent tests of independence, and asymptotically α-level tests, based on three statistics: the L_1 distance, the log-likelihood, and a kernel-based distance. The asymptotic L_1 and log-likelihood tests require that the distributions be non-atomic, but make no assumptions apart from this: in particular, the test thresholds are *not* functions of the distribution. The kernel statistic is interpretable as either an L_2 distance between kernel density estimates (if the kernel bandwidth shrinks for increasing sample size), or as the Hilbert-Schmidt norm of a covariance operator between reproducing kernel Hilbert spaces (if the kernel bandwidth is fixed). We have provided a novel strong consistent test for the kernel statistic, as well as reviewing two asymptotically α-level tests (for both fixed and shrinking kernel bandwidth). Unlike the L_1 and log-likelihood tests, the thresholds for the kernel asymptotic tests are distribution dependent. We also gave conjectures regarding the strong consistent test and asymptotically α-level test for the Pearson χ^2 distance.

Our experiments showed the asymptotic tests to be capable of detecting dependence for both univariate and multi-dimensional variables (of up to three dimensions each), for variables having no linear correlation. The kernel tests had lower Type II error than the L_1 and log-likelihood tests for a given Type I error, however we should bear in mind that the kernel test thresholds were finite sample estimates, and the resulting convergence issues have not been addressed. The log-likelihood test appeared to suffer more from bias than the L_1 test, in cases where there were few samples per partition (this effect was most visible in high dimensions).

This study raises a number of questions for future research. First, the χ^2 tests remain conjectures, and proofs should be established. Second, there is as yet no distribution-free asymptotic threshold for the kernel test, which could be based on a tighter bound on the variance of the test statistic under the null distribution. Third, the asymptotic distribution of the kernel statistic with fixed bandwidth is presently a heuristic: it would therefore be of interest to replace this with a null distribution estimate having appropriate convergence guarantees.

Acknowledgments

This work was supported in part by the Computer and Automation Research Institute of the Hungarian Academy of Sciences, and by the IST Program of the EC, under the FP7 Network of Excellence, ICT-216886-NOE. Part of this work was completed while Arthur Gretton was a project scientist at CMU, under grants DARPA IPTO FA8750-09-1-0141, ONR MURI N000140710747, and NSF NeTS-NOSS CNS-0625518.

Appendix A. Proof of Theorem 3

The main difficulty in proving Theorem 3 is that it states the asymptotic normality of $L_n(\nu_n, \mu_{n,1} \times \mu_{n,2})$, which is a sum of *dependent* random variables. To overcome this problem, we use a "Poissonization" argument originating from the fact that an empirical process is equal in distribution to the conditional distribution of a Poisson process given the sample size (for more on Poissonization techniques, we refer the reader to Beirlant, Györfi, and Lugosi, 1994).

We begin by introducing the necessary terminology. For each $n \geq 1$, denote by N_n a Poisson(n) random variable, defined on the same probability space as the sequences $(X_i)_{i\geq 1}$ and $(Y_i)_{i\geq 1}$, and

independent of these sequences. Denote by ν_{N_n}, $\mu_{N_n,1}$ and $\mu_{N_n,2}$ the Poissonized version of the empirical measures associated with the samples $\{(X_i,Y_i)\}$, $\{X_i\}$ and $\{Y_i\}$, respectively, so that

$$\nu_{N_n}(A\times B)=\frac{\#\{i:(X_i,Y_i)\in A\times B, i=1,\ldots,N_n\}}{n},$$

$$\mu_{N_n,1}(A)=\frac{\#\{i:X_i\in A, i=1,\ldots,N_n\}}{n},$$

and

$$\mu_{N_n,2}(B)=\frac{\#\{i:Y_i\in B, i=1,\ldots,N_n\}}{n}$$

for any Borel subsets A and B. The Poissonized version $\tilde{L}_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$ of $L_n(\nu_n,\mu_{n,1}\times\mu_{n,2})$ is then

$$\tilde{L}_n(\nu_n,\mu_{n,1}\times\mu_{n,2})=\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}|\nu_{N_n}(A\times B)-\mu_{N_n,1}(A)\cdot\mu_{N_n,2}(B)|.$$

Clearly,

$$n\nu_{N_n}(A\times B)=\#\{i:(X_i,Y_i)\in A\times B, i=1,\ldots,N_n\},$$

$$n\mu_{N_n,1}(A)=\#\{i:X_i\in A, i=1,\ldots,N_n\},$$

and

$$n\mu_{N_n,2}(B)=\#\{i:Y_i\in B, i=1,\ldots,N_n\}$$

are Poisson random variables.

Key to the proof of Theorem 3 is the following property, which is a slight extension of the proposition of Beirlant, Györfi, and Lugosi (1994, p. 311).

Proposition 7 *Let g_{njk} $(n\geq 1,\ j=1,\ldots,m_n, k=1,\ldots,m_n')$ be real measurable functions, and let*

$$M_n:=\sum_{j=1}^{m_n}\sum_{k=1}^{m_n'}g_{njk}\left(\nu_{N_n}(A_{nj}\times B_{nk})-\mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})\right).$$

Assume that, under the null hypothesis,

$$\mathbf{E}\{g_{njk}\left(\nu_{N_n}(A_{nj}\times B_{nk})-\mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})\right)\}=0,$$

and that

$$\left(M_n,\frac{N_n-n}{\sqrt{n}}\right)\xrightarrow{\mathcal{D}}\mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}\sigma^2&0\\0&1\end{bmatrix}\right) \tag{21}$$

as $n\to\infty$, where σ is a positive constant and $\mathcal{N}(\mathbf{m},\mathbf{C})$ is a normally distributed random variable with mean $\mathbf{m}$ and covariance matrix $\mathbf{C}$. Then

$$\frac{1}{\sigma}\sum_{j=1}^{m_n}\sum_{k=1}^{m_n'}g_{njk}\left(\nu_n(A_{nj}\times B_{nk})-\mu_{n,1}(A_{nj})\mu_{n,2}(B_{nk})\right)\xrightarrow{\mathcal{D}}\mathcal{N}(0,1).$$

Proof The proof is in sketch form, along the lines of Biau and Györfi (2005). Define the two characteristic functions

$$\Phi_n(t,v) := \mathbf{E}\left\{\exp\left(\imath t M_n + \imath v \frac{N_n - n}{\sqrt{n}}\right)\right\}$$

and

$$\Psi_n(t) := \mathbf{E}\left\{\exp\left(\imath t \sum_{j=1}^{m_n}\sum_{k=1}^{m'_n} g_{njk}\left(\nu_n(A_{nj}\times B_{nk}) - \mu_{n,1}(A_{nj})\mu_{n,2}(B_{nk})\right)\right)\right\}.$$

We begin with the result

$$\mathbf{E}\left\{\exp\left(\imath t M_n + \imath u N_n\right)\right\} = \sum_{l=0}^{\infty} \mathbf{E}\{\exp(\imath t M_n)|N_n = l\} e^{\imath u l} p_n(l),$$

where $p_n(l)$ is the probability distribution of the Poisson(n) random variable N_n,

$$p_n(l) = \mathbf{P}\{N_n = l\} = e^{-n} n^l / l!,$$

and

$$\Psi_n(t) = \mathbf{E}\{\exp(\imath t M_n)|N_n = n\}.$$

Taking the inverse Fourier transform,

$$\mathbf{E}\{\exp(\imath t M_n)|N_n = n\} = \frac{1}{2\pi p_n(n)} \int_{-\pi}^{\pi} e^{-\imath u n} \mathbf{E}\left\{\exp\left(\imath t M_n + \imath u N_n\right)\right\} du.$$

We now replace $n!$ with the Stirling approximation to obtain

$$2\pi p_n(n) = \frac{2\pi e^{-n} n^n}{n!} \approx \sqrt{\frac{2\pi}{n}} \quad \text{as} \quad n \to \infty.$$

Then, substituting $v = u\sqrt{n}$, we get

$$\Psi_n(t) = \frac{1}{\sqrt{2\pi}}(1 + o(1)) \int_{-\pi\sqrt{n}}^{\pi\sqrt{n}} \Phi_n(t,v) dv.$$

By assumption,

$$\Phi_n(t,v) \to e^{-t^2\sigma^2/2} e^{-v^2/2}$$

as $n \to \infty$. The result follows from Rao (1973, p. 136). ■

We now use Proposition 7 to prove

$$\frac{\sqrt{n}}{\sigma}\left(L_n(\nu_n, \mu_{n,1}\times\mu_{n,2}) - \mathbf{E}\{\tilde{L}_n(\nu_n, \mu_{n,1}\times\mu_{n,2})\}\right) \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

where we recall $\sigma^2 = 1 - 2/\pi$. This provides the result in Theorem 3 with the centering constant

$$C_n = \mathbf{E}\{\tilde{L}_n(\nu_n, \mu_{n,1}\times\mu_{n,2})\} = \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n} \mathbf{E}\{|\nu_{N_n}(A\times B) - \mu_{N_n,1}(A)\cdot\mu_{N_n,2}(B)|\}. \tag{22}$$

To apply Proposition 7, we must prove assumption (21) holds. Define

$$g_{njk}(x) = \sqrt{n}\left(|x| - \mathbf{E}\left|\nu_{N_n}(A_{nj}\times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})\right|\right).$$

Let

$$\begin{aligned} S_n &:= t\sqrt{n}\sum_{j=1}^{m_n}\sum_{k=1}^{m_n'}\Big(\left|\nu_{N_n}(A_{nj}\times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})\right| \\ &\quad -\mathbf{E}\left|\nu_{N_n}(A_{nj}\times B_{nk}) - \mu_{N_n,1}(A_{nj})\mu_{N_n,2}(B_{nk})\right|\Big) \\ &\quad +v\sqrt{n}\left(\frac{N_n}{n}-1\right). \end{aligned}$$

Our goal is to prove the assumption in (21) holds. In particular, we require the variance of the Poissonized statistic S_n. After this variance is calculated, the asymptotic normality in (21) can be proved by verifying the Lyapunov conditions as in Beirlant, Györfi, and Lugosi (1994). From the definitions of ν_{N_n}, μ_1, and μ_2, we have

$$\frac{N_n}{n} - 1 = \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\nu_{N_n}(A\times B) - \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mu_1(A)\mu_2(B),$$

and thus the variance of S_n is

$$\begin{aligned} \operatorname{var}(S_n) &= t^2 n\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\operatorname{var}\left|\nu_{N_n}(A\times B) - \mu_{N_n,1}(A)\mu_{N_n,2}(B)\right| \\ &+ 2tvn\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathbf{E}\big\{\left|\nu_{N_n}(A\times B) - \mu_{N_n,1}(A)\mu_{N_n,2}(B)\right| \\ &\quad \cdot\left(\nu_{N_n}(A\times B) - \mu_1(A)\mu_2(B)\right)\big\} \\ &+ v^2. \end{aligned}$$

One can check that there exist standard normal random variables $Z_{A\times B}$, Z_A, and Z_B such that

$$\nu_{N_n}(A\times B) \overset{\mathcal{D}}{\approx} Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}} + \mu_1(A)\mu_2(B),$$

$$\mu_{N_n,1}(A) \overset{\mathcal{D}}{\approx} Z_A\sqrt{\frac{\mu_1(A)}{n}} + \mu_1(A),$$

and

$$\mu_{N_n,2}(B) \overset{\mathcal{D}}{\approx} Z_B\sqrt{\frac{\mu_2(B)}{n}} + \mu_2(B),$$

which implies

$$\begin{aligned}
& \nu_{N_n}(A\times B)-\mu_{N_n,1}(A)\mu_{N_n,2}(B) \\
\overset{\mathcal{D}}{\approx} \; & Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}+\mu_1(A)\mu_2(B) \\
& -\left(Z_A\sqrt{\frac{\mu_1(A)}{n}}+\mu_1(A)\right)\left(Z_B\sqrt{\frac{\mu_2(B)}{n}}+\mu_2(B)\right) \\
= \; & \sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}\left(Z_{A\times B}-Z_AZ_B\frac{1}{\sqrt{n}}-Z_A\sqrt{\mu_2(B)}-Z_B\sqrt{\mu_1(A)}\right) \\
\approx \; & Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}.
\end{aligned}$$

Thus,

$$\begin{aligned}
& \mathrm{var}(S_n) \\
\approx \; & t^2 n\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathrm{var}\left|Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}\right| \\
+ \; & 2tvn\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathbf{E}\left\{\left|Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}\right|\cdot\left(Z_{A\times B}\sqrt{\frac{\mu_1(A)\mu_2(B)}{n}}\right)\right\} \\
+ \; & v^2 \\
= \; & t^2\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathrm{var}|Z_{A\times B}|\mu_1(A)\mu_2(B) \\
+ \; & 2tv\sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathbf{E}\{|Z_{A\times B}|Z_{A\times B}\}\mu_1(A)\mu_2(B) \\
+ \; & v^2 \\
= \; & t^2(1-2/\pi)+v^2.
\end{aligned}$$

Finally, we use the variable $Z_{A\times B}$ in defining a distribution-free upper bound on C_n, which we use in our asymptotically α-level independence test,

$$\begin{aligned}
C_n &= \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathbf{E}\{|\nu_{N_n}(A\times B)-\mu_{N_n,1}(A)\cdot\mu_{N_n,2}(B)|\} \\
&\approx \sum_{A\in\mathcal{P}_n}\sum_{B\in\mathcal{Q}_n}\mathbf{E}\{|Z_{A\times B}|\}\sqrt{\mu_1(A)\mu_2(B)/n} \\
&\leq \sqrt{2/\pi}\sqrt{\frac{m_nm_n'}{n}} \qquad (23)
\end{aligned}$$

Appendix B. Conjectured Large Sample Properties of the Pearson χ^2 Statistic

For a real parameter λ, the *power divergence statistic* is defined as

$$D_{n,\lambda}(\mu_{n,1},\mu_1)=\frac{2}{\lambda(\lambda+1)}\sum_{j=1}^{m_n}\mu_{n,1}(A_{n,j})\left[\left(\frac{\mu_{n,1}(A_{n,j})}{\mu_1(A_{n,j})}\right)^{\lambda}-1\right]$$

provided $\lambda \neq 0$ and $\lambda \neq 1$ (cf. Read and Cressie, 1988). One can check that

$$\lim_{\lambda \to 0} D_{n,\lambda}(\mu_{n,1}, \mu_1) = I_n(\mu_{n,1}, \mu_1).$$

For $\lambda = 1$, we have the Pearson χ^2 statistic:

$$\chi_n^2(\mu_{n,1}, \mu_1) = D_{n,1}(\mu_{n,1}, \mu_1) = \sum_{j=1}^{m_n} \frac{(\mu_{n,1}(A_{n,j})) - \mu_1(A_{n,j}))^2}{\mu_1(A_{n,j})}.$$

For testing independence, we employ the Pearson χ^2 test statistic

$$\chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) \quad = \quad \sum_{A \in \mathcal{P}_n} \sum_{B \in \mathcal{Q}_n} \frac{(\nu_n(A \times B) - \mu_{n,1}(A) \cdot \mu_{n,2}(B))^2}{\mu_{n,1}(A) \cdot \mu_{n,2}(B)}.$$

B.1 Strongly Consistent Test

Quine and Robinson (1985) proved that, for all $\varepsilon > 0$,

$$\mathbf{P}\{\chi_n^2(\mu_{n,1}, \mu_1) > \varepsilon\} \leq \binom{n + m_n - 1}{m_n - 1} e^{-\frac{n \log m_n}{2\sqrt{m_n}} \sqrt{\varepsilon}} \leq e^{m_n \log(n+m_n) - \frac{n \log m_n}{2\sqrt{m_n}} \sqrt{\varepsilon}}. \tag{24}$$

A large deviation-based test can be introduced that rejects independence if

$$\chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) \geq \left(\frac{2(m_n m_n')^{3/2} (\log(n + m_n m_n') + 1)}{n \log(m_n m_n')} \right)^2.$$

Under $\mathcal{H}_0$, we conjecture a non-asymptotic bound for the tail of the distribution of $\chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2})$,

$$\begin{aligned} & \mathbf{P}\left\{ \chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) > \left(\frac{2(m_n m_n')^{3/2} (\log(n + m_n m_n') + 1)}{n \log(m_n m_n')} \right)^2 \right\} \\ \leq \; & e^{m_n m_n' \log(n + m_n m_n') - \frac{n \log(m_n m_n')}{2\sqrt{m_n m_n'}} \frac{2(m_n m_n')^{3/2} (\log(n + m_n m_n') + 1)}{n \log(m_n m_n')}} \\ = \; & e^{-m_n m_n'}. \end{aligned}$$

Therefore the conditions (7) imply

$$\sum_{n=1}^{\infty} \mathbf{P}\left\{ \chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2}) > \left(\frac{2(m_n m_n')^{3/2} (\log(n + m_n m_n') + 1)}{n \log(m_n m_n')} \right)^2 \right\} < \infty,$$

and by the Borel-Cantelli lemma we have strong consistency under the null hypothesis.

Under the alternative hypothesis the proof strong consistency follows from the proof for the information divergence since

$$I_n(\nu_n, \mu_{n,1} \times \mu_{n,2})/2 \leq \chi_n^2(\nu_n, \mu_{n,1} \times \mu_{n,2})$$

(cf. Györfi et al., 1998).

B.2 Asymptotic α-level Test

Morris (1975), Inglot et al. (1990), and Györfi and Vajda (2002) proved that under (10) and (11),

$$\frac{n\chi_n^2(\mu_{n,1},\mu_1)-m_n}{\sqrt{2m_n}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

which is the same asymptotic normality result as for $2I_n(\mu_{n,1},\mu_1)$ (see Equation (14) in Section 3.2). We conjecture that under the conditions (6) and (12),

$$\frac{n\chi_n^2(\nu_n,\mu_{n,1}\times\mu_{n,2})-m_nm_n'}{\sqrt{2m_nm_n'}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1).$$

Thus, as for the log-likelihood statistic, the hypothesis of independence is rejected if

$$\chi_n^2(\nu_n,\mu_{n,1}\times\mu_{n,2}) \geq \frac{\Phi^{-1}(1-\alpha)\sqrt{2m_nm_n'}+m_nm_n'}{n}.$$

References

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

C. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

A. R. Barron. Uniformly powerful goodness of fit tests. *The Annals of Statistics*, 17:107–124, 1989.

A. R. Barron, L. Györfi, and E. C. van der Meulen. Distribution estimation consistent in total variation and in two types of information divergence. *IEEE Transactions on Information Theory*, 38:1437–1454, 1992.

M. S. Bartlett. The characteristic function of a conditional statistic. *Journal of the London Mathematical Society*, 13:62–67, 1938.

J. Beirlant and D. M. Mason. On the asymptotic normality of l_p-norms of empirical functionals. *Mathematical Methods of Statistics*, 4:1–19, 1995.

J. Beirlant, L. Györfi, and G. Lugosi. On the asymptotic normality of the l_1- and l_2-errors in histogram density estimation. *Canadian Journal of Statistics*, 22:309–318, 1994.

J. Beirlant, L. Devroye, L. Györfi, and I. Vajda. Large deviations of divergence measures on partitions. *Journal of Statistical Planning and Inference*, 93:1–16, 2001.

G. Biau and L. Györfi. On the asymptotic properties of a nonparametric l_1-test statistic of homogeneity. *IEEE Transactions on Information Theory*, 51:3965–3973, 2005.

J. R. Blum, J. Kiefer, and M. Rosenblatt. Distribution free tests of independence based on the sample distribution function. *The Annals of Mathematical Statistics*, 32:485–498, 1961.

D. S. Cotterill and M. Csörgő. On the limiting distribution of and critical values for the Hoeffding, Blum, Kiefer, Rosenblatt independence criterion. *Statistics and Decisions*, 3:1–48, 1985.

I. Csiszár. Information-type measures of divergence of probability distributions and indirect observations. *Studia Scientiarum Mathematicarum Hungarica*, 2:299–318, 1967.

J. Dauxois and G. M. Nkiet. Nonlinear canonical analysis and independence tests. *The Annals of Statistics*, 26(4):1254–1278, 1998.

A. Dembo and Y. Peres. A topological criterion for hypothesis testing. *The Annals of Statistics*, 22: 106–117, 1994.

A. Feuerverger. A consistent test for bivariate dependence. *International Statistical Review*, 61(3): 419–433, 1993.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

K. Fukumizu, F. Bach, and A. Gretton. Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, 8:361–383, 2007.

K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems 20*, pages 489–496, Cambridge, MA, 2008. MIT Press.

A. Gretton and L. Györfi. Nonparametric independence tests: Space partitioning and kernel approaches. In *Algorithmic Learning Theory: 19th International Conference*, pages 183–198, Berlin, 2008. Springer.

A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Algorithmic Learning Theory: 16th International Conference*, pages 63–78, Berlin, 2005a. Springer.

A. Gretton, R. Herbrich, A. Smola, O. Bousquet, and B. Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:2075–2129, 2005b.

A. Gretton, K. Fukumizu, C.-H. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pages 585–592, Cambridge, MA, 2008. MIT Press.

L. Györfi and I. Vajda. Asymptotic distributions for goodness of fit statistics in a sequence of multinomial models. *Statistics and Probability Letters*, 56:57–67, 2002.

L. Györfi and E. C. van der Meulen. A consistent goodness of fit test based on the total variation distance. In G. Roussas, editor, *Nonparametric Functional Estimation and Related Topics*, pages 631–645. Kluwer, Dordrecht, 1990.

L. Györfi, F. Liese, I. Vajda, and E. C. van der Meulen. Distribution estimates consistent in χ^2-divergence. *Statistics*, 32:31–57, 1998.

P. Hall. Central limit theorem for integrated square error of multivariate nonparametric density estimators. *Journal of Multivariate Analysis*, 14:1–16, 1984.

W. Hoeffding. A nonparametric test for independence. *The Annals of Mathematical Statistics*, 19 (4):546–557, 1948.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

T. Inglot, T. Jurlewitz, and T. Ledwina. Asymptotics for multinomial goodness of fit tests for a simple hypothesis. *Theory of Probability and Its Applications*, 35:797–803, 1990.

J. Jacod and P. Protter. *Probability Essentials*. Springer, New York, 2000.

W. C. M. Kallenberg. On moderate and large deviations in multinomial distributions. *The Annals of Statistics*, 13:1554–1580, 1985.

A. Kankainen. *Consistent Testing of Total Independence Based on the Empirical Characteristic Function*. PhD thesis, University of Jyväskylä, 1995.

J. H. B. Kemperman. An optimum rate of transmitting information. *The Annals of Mathematical Statistics*, 40:2156–2177, 1969.

S. Kullback. A lower bound for discrimination in terms of variation. *IEEE Transactions on Information Theory*, 13:126–127, 1967.

S. E. Leurgans, R. A. Moyeed, and B. W. Silverman. Canonical correlation analysis when the data are curves. *Journal of the Royal Statistical Society, Series B (Methodological)*, 55(3):725–740, 1993.

C. McDiarmid. On the method of bounded differences. In *Survey in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

C. Morris. Central limit theorems for multinomial sums. *The Annals of Statistics*, 3:165–188, 1975.

M.P. Quine and J. Robinson. Efficiencies of chi-square and likelihood ratio goodness-of-fit tests. *The Annals of Statistics*, 13:727–742, 1985.

C. R. Rao. *Statistical Inference and its Applications*. Wiley, New York, second edition, 1973.

T. Read and N. Cressie. *Goodness-Of-Fit Statistics for Discrete Multivariate Analysis*. Springer-Verlag, New York, 1988.

A. Rényi. On measures of dependence. *Acta Mathematica Academiae Scientiarum Hungaricae*, 10: 441–451, 1959.

M. Rosenblatt. A quadratic measure of deviation of two-dimensional density estimates and a test of independence. *The Annals of Statistics*, 3(1):1–14, 1975.

R. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Injective hilbert space embeddings of probability measures. In *Proceedings of the 21st Annual Conference on Learning Theory*, pages 111–122, 2008.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

G. Tusnády. On asymptotically optimal tests. *The Annals of Statistics*, 5:385–393, 1977.

N. Ushakov. *Selected Topics in Characteristic Functions*. Modern Probability and Statistics. Walter de Gruyter, Berlin, 1999.

Journal of Machine Learning Research 11 (2010) 1425-1470 Submitted 4/09; Revised 12/09; Published 4/10

Characterization, Stability and Convergence of Hierarchical Clustering Methods

Gunnar Carlsson GUNNAR@MATH.STANFORD.EDU
Facundo Mémoli* MEMOLI@MATH.STANFORD.EDU
Department of Mathematics
Stanford University
Stanford, CA 94305

Editor: Ulrike von Luxburg

Abstract

We study hierarchical clustering schemes under an axiomatic view. We show that within this framework, one can prove a theorem analogous to one of Kleinberg (2002), in which one obtains an existence and uniqueness theorem instead of a non-existence result. We explore further properties of this unique scheme: stability and convergence are established. We represent dendrograms as ultrametric spaces and use tools from metric geometry, namely the Gromov-Hausdorff distance, to quantify the degree to which perturbations in the input metric space affect the result of hierarchical methods.

Keywords: clustering, hierarchical clustering, stability of clustering, Gromov-Hausdorff distance

1. Introduction

Clustering techniques play a very central role in various parts of data analysis. They can give important clues to the structure of data sets, and therefore suggest results and hypotheses in the underlying science. Many of the interesting methods of clustering available have been applied to good effect in dealing with various data sets of interest. However, despite being one of the most commonly used tools for unsupervised exploratory data analysis, and despite its extensive literature, very little is known about the theoretical foundations of clustering methods. These points have been recently made prominent by von Luxburg and Ben-David (2005); Ben-David et al. (2006).

The general question of which methods are "best", or most appropriate for a particular problem, or how significant a particular clustering is has not been addressed too frequently. This lack of theoretical guarantees can be attributed to the fact that many methods involve particular choices to be made at the outset, for example how many clusters there should be, or the value of a particular thresholding parameter. In addition, some methods depend on artifacts in the data, such as the particular order in which the observations are listed.

In Kleinberg (2002), Kleinberg proves a very interesting impossibility result for the problem of even defining a clustering scheme with some rather mild invariance properties. He also points out that his results shed light on the trade-offs one has to make in choosing clustering algorithms.

Standard clustering methods take as input a finite metric space (X,d) and output a partition of X. Let $\mathcal{P}(X)$ denote the set of all possible partitions of the set X. Kleinberg (2002) discussed

*. Corresponding author.

this situation in an axiomatic way and identified a set of reasonable properties of standard clustering schemes, namely, scale invariance, richness and consistency. Fix a standard clustering method f and a metric space (X,d) and let $f(X,d) = \Pi \in \mathcal{P}(X)$. Kleinberg identified the following desirable properties of a clustering scheme:

- **Scale Invariance**: For all $\alpha > 0$, $f(X, \alpha \cdot d) = \Pi$.
- **Richness**: Fix any finite set X. Then for all $\Pi \in \mathcal{P}(X)$, *there exists* d_Π, a metric on X s.t. $f(X,d_\Pi) = \Pi$.
- **Consistency**: Let $\Pi = \{B_1, \ldots, B_\ell\}$. Let $\widehat{d}$ be any metric on X s.t.
  1. for all $x, x' \in B_\alpha$, $\widehat{d}(x,x') \leqslant d(x,x')$ and
  2. for all $x \in B_\alpha, x' \in B_{\alpha'}$, $\alpha \neq \alpha'$, $\widehat{d}(x,x') \geqslant d(x,x')$.

  Then, $f(X,\widehat{d}) = \Pi$.

He then proved, in the same spirit of Arrow's impossibility theorem, that no clustering scheme satisfying these conditions simultaneously can exist.

Theorem 1 (Kleinberg, 2002) *There exists no clustering algorithm that satisfies scale invariance, richness and consistency.*

Then, in particular, Kleinberg's axioms rule out single, average and complete linkage (standard) clustering. Clusters in any of these three methods can be obtained by first constructing a hierachical decomposition of space (such as those provided by hierarchical clustering methods) and then selecting the partition that arises at a given, fixed, threshold.

A natural question is whether Kleinberg's impossibility results still holds when one admits clustering schemes that do not try to return a fixed partition of a space, but are allowed to return a *hierarchical decomposition*.

Furthermore, data sets can exhibit multiscale structure and this can render standard clustering algorithms inapplicable in certain situations, see Figure 1. This further motivates the use of **Hierarchical clustering methods.** Hierarchical methods take as input a finite metric space (X,d) and output a hierarchical family of partitions of X.

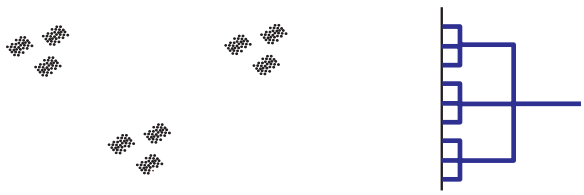


Figure 1: Data set with multiscale structure. Any standard clustering algorithm will fail to capture the structure of the data.

These hierarchical families of partitions that constitute the output of hierarchical methods receive the name of *dendrograms*. Dendrograms come in two versions: *proximity* and *threshold* dendrograms. These two types of dendrograms differ in whether they retain some proximity information about the underlying clusters that they represent or not: proximity dendrograms do retain

such information whereas threshold dendrograms do not. Practicioners of statistical data analysis seem to work almost exclusively with proximity dendrograms. For this reason we opt to carry out our analysis under the model that hierarchical methods take as input a finite metric space X and output a proximity dendrogram over X, see Remark 3.

We remind the reader that we are using the term standard clustering methods to refer to procedures that take a finite metric space as input and output a fixed single partition of the metric space.

In a similar spirit to Kleinberg's theorem, we prove in Theorem 18 that in the context of hierarchical methods, one obtains *uniqueness* instead of non-existence. We emphasize that our result can be interpreted as a *relaxation* of the theorem proved by Kleinberg, in the sense that allowing clustering schemes that output a nested family of partitions in the form of a proximity dendrogram, instead of a fixed partition, removes the obstruction to existence. The unique HC method characterized by our theorem turns out to be single linkage hierarchical clustering.

We stress the fact that our result assumes that outputs of hierarchical methods are proximity dendrograms, whereas Kleinberg's Theorem applies to flat/standard clustering, a situation in which the output contains no proximity information between clusters.

In order to state and prove our results we make use of the well known **equivalent representation** of dendrograms, the output of HC methods, using *ultrametrics*. This already appears in the book of Hartigan and others, see Hartigan (1985), Jain and Dubes (1988, §3.2.3) and references therein.

In recent years, the theme of studying the properties of metrics with prescribed generalized curvature properties has been studied intensively. In particular, the work of Gromov (1987) has been seminal, and many interesting results have been proved concerning objects other than metric spaces, such as finitely generated groups, depending on these methods. The curvature conditions can be formulated in terms of properties of triangles within the metric spaces, and the most extreme of these properties is that embodied in ultrametric spaces. A second idea of Gromov's is to make the collection of all metric spaces into its own metric space, and the resulting metric gives a very useful and natural way to distinguish between metric spaces (Gromov, 2007). This metric is known as the Gromov-Hausdorff distance and its restriction to the subclass of ultrametric spaces is therefore a very natural object to study.

1.1 Stability

Stability of some kind is clearly a desirable property of clustering methods and, therefore, a point of interest is studying whether results obtained by a given clustering algorithm are *stable* to perturbations in the input data. Since input data are modelled as finite metric spaces, and the output of hierarchical methods can be regarded as finite ultrametric spaces, the Gromov-Hausdorff distance provides a natural tool for studying *variability* or *perturbation* of the inputs and outputs of hierarchical clustering methods.

After observing in §3.6 that average and complete linkage clustering are not stable in the metric sense alluded to above, we prove in Proposition 26 that single linkage does enjoy a kind of stability:

Proposition 2 *Let (X,d_X) and (Y,d_Y) be two finite metric spaces and let (X,u_X) and (Y,u_X) be the two (finite metric ultrametric spaces) corresponding outputs yielded by single linkage HC. Then,*

$$d_{\mathcal{GH}}\big((X,u_X),(Y,u_Y)\big) \leqslant d_{\mathcal{GH}}\big((X,d_X),(Y,d_Y)\big).$$

Here, $d_{\mathcal{GH}}$ stands for the Gromov-Hausdorff distance.

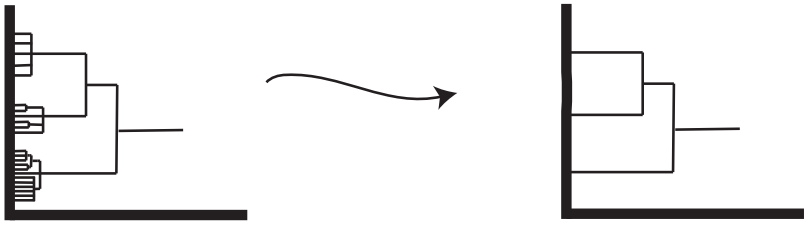


Figure 2: Convergence of dendrograms. We formalize this concept by equivalently representing dendrogram as ultrametrics and then computing the Gromov-Hausdorff distance between the resulting metrics. We prove in Theorem 30 that by taking increasingly many i.i.d. samples from a given probability distribution μ on a metric space,then with probability 1 one recovers a multiscale representation of the supprt of μ.

This result is very important for the convergence theorems which we prove in the later parts of the paper. These results describe in a very precise way the fact that for compact metric spaces X, the results of clustering the finite subsets of X yields a collection of dendrograms which ultimately converge to the dendrogram for X. In order for this to happen, one needs the metric on the ultrametric spaces as well as the behavior of the clustering construction on the Gromov-Hausdorff distance, which is what Proposition 2 does. The issue of stability is further explored in §5.

1.2 Probabilistic Convergence

Finally, in Theorem 30 we also prove that for random i.i.d. observations $\mathbb{X}_n = \{x_1, \ldots, x_n\}$ with probability distribution μ compactly supported in a metric space (X, d), the result $(\mathbb{X}_n, u_{\mathbb{X}_n})$ of applying single linkage clustering to $(\mathbb{X}_n, d)$ **converges almost surely** in the Gromov-Hausdorff sense to an ultrametric space that recovers the multiscale structure of the *support* of μ, see Figure 20. This can be interpreted as a refinement of a previous observation (Hartigan, 1985) that SLHC is insensitive to the distribution of mass of μ in its support.

1.3 Organization of the Paper

This paper is organized as follows: §A provides a list of all the notation defined and used throughout the paper; §2 introduces the terminology and basic concepts that we use in our paper; §3.2 reviews hierarchical clustering methods in general; §3.3 discusses the representation of dendrograms as ultrametric spaces and establishes the equivalence of both repersentations; and §3.5 delves into the issue of constructing a notion of distance between dendrograms which is based in the equivalence of dendrograms and ultrametrics; §3.6 comments on issues pertaining to the theoretical properties of HC methods. In §4 we present our characterization result, Theorem 18, for SL in a spirit similar to the axiomatic treatment of Kleinberg. We delve into the stability and convergence questions of SL in §5, where we introduce all the necessary concepts from Metric Geometry. Proposition 26 and Theorem 28 contain our results for the deterministic case. In §5.3 we prove a probabilistic convergence result Theorem 30 that hinges on a general sampling theorem for measure metric spaces, Theorem 34. Finally, we conclude the paper with a discussion on future directions.

For clarity of exposition, we have chosen to move most of the proofs in this paper to an appendix. The ones which remain in the main text are intended to provide intuition which would not otherwise be there.

2. Background and Notation

A **metric space** is a pair (X,d) where X is a set and $d : X \times X \to \mathbb{R}^+$ satisfies

1. For all $x,x' \in X$, $d(x',x) = d(x,x') \geqslant 0$ and $d(x,x') = 0$ if and only if $x = x'$.
2. For all $x,x',x'' \in X$, $d(x,x'') \leqslant d(x,x') + d(x',x'')$.

A metric space (X,u) is an **ultrametric space** if and only if for all $x,x',x'' \in X$,

$$\max(u(x,x'),u(x',x'')) \geqslant u(x,x''). \tag{1}$$

Ultrametric spaces are therefore metric spaces which satisfy a stronger type of triangle inequality. It is interesting to observe that this ultrametric triangle inequality (1) implies that all triangles are *isosceles*.[1]

Notice that by iterating the ultrametric property one obtains that if $x_1,x_2,\ldots,x_k$ is any set of k points in X, then

$$\max\left(u(x_1,x_2),u(x_2,x_3),\ldots,u(x_{k-1},x_k)\right) \geqslant u(x_1,x_k).$$

For a fixed finite set X, we let $\mathcal{U}(X)$ denote the collection of all ultrametrics on X. For $n \in \mathbb{N}$ let $\mathcal{X}_n$ (resp. $\mathcal{U}_n$) denote the collection of all metric spaces (resp. ultra-metric spaces) with n points. Let $\mathcal{X} = \bigsqcup_{n\geqslant 1} \mathcal{X}_n$ denote the collection of all finite metric spaces and $\mathcal{U} = \bigsqcup_{n\geqslant 1} \mathcal{U}_n$ all finite ultrametric spaces. For $(X,d) \in \mathcal{X}$ let

$$\mathrm{sep}(X,d) := \min_{x\neq x'} d(x,x') \quad \text{and} \quad \mathrm{diam}\,(X,d) := \max_{x,x'} d(x,x')$$

be the *separation* and the *diameter* of X, respectively.

We now recall the definition of an **equivalence relation**. Given a set A, a *binary relation* is a subset $S \subset A \times A$. One says that a and a' are *related* and writes $a \sim a'$ whenever $(a,a') \in S$. S is called an *equivalence relation* if and only if for all $a,b,c \in A$, all the following hold true:

- Reflexivity: $a \sim a$.
- Symmetry: if $a \sim b$ then $b \sim a$.
- Transitivity: if $a \sim b$ and $b \sim c$ then $a \sim c$.

The *equivalence class* of a under $\sim$, denoted $[a]$, is defined as all those a' which are related to a: $[a] = \{a' \in A, \text{s.t. } a' \sim a\}$. Finally, the *quotient space* $A\backslash\sim$ is the collection of all equivalence classes: $A\backslash\sim := \{[a], a \in A\}$.

We now construct our first example which will be crucial in our presentation.

Example 1 (r-equivalence) *Given a finite metric space (X,d) and $r \geqslant 0$ we say that points $x,x' \in X$ are* **r-equivalent** *(denoted $x \sim_r x'$) if and only if there exists points $x_0,x_1,\ldots,x_t \in X$ with $x_0 = x$,*

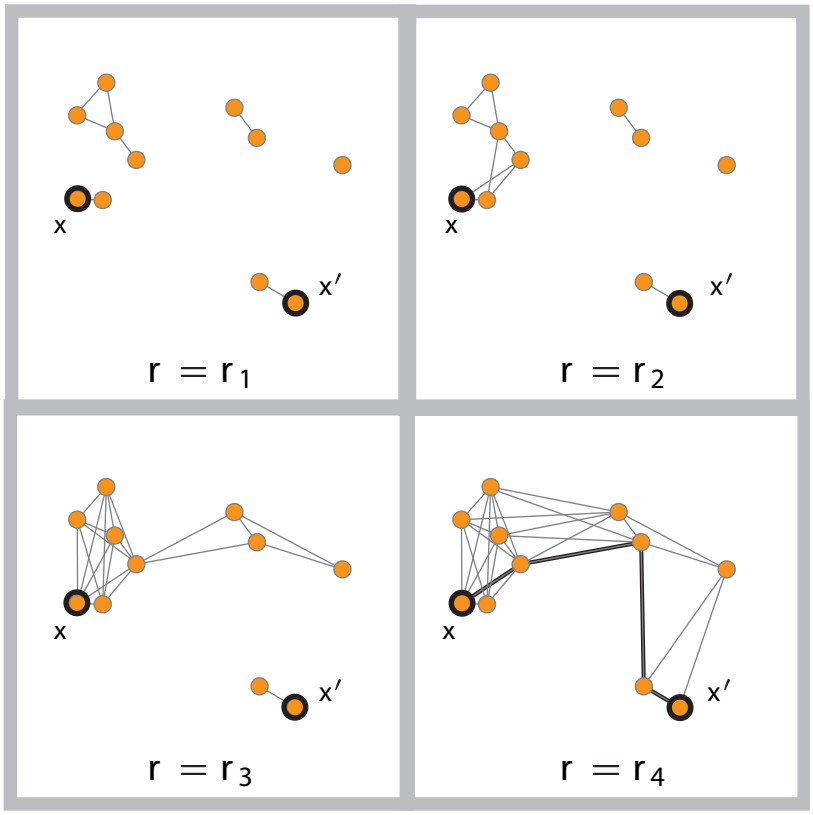


Figure 3: Illustration of the equivalence relation $\sim_r$. A finite metric space X is specified by the points in orange which are endowed with the Euclidean distance. This construction can be understood as allowing the creation of edges joining two points whenever the distance between them does not exceed r. Then, two points x and x' in black are deemed r-equivalent if one can find a sequence of edges on the resulting graph connecting x to x'. From left to right and top to bottom we show the resulting graph one obtains for 4 increasing values of r. The points x and x' are not r-equivalent when $r = r_1, r_2$ or r_3, but they are r_4-equivalent.

$x_t = x'$ *and* $d(x_i, x_{i+1}) \leqslant r$ *for* $i = 0, \ldots, t-1$. *It is easy to see that* $\sim_r$ *is indeed an equivalence relation on* X.

This definition embodies the simple idea of partitioning a finite metric space into path connected components, where the granularity of this partitioning is specified by the parameter $r \geqslant 0$, *see Figure 1.*

1. Indeed, assume that all sides a, b, c of a triangle in a given ultrametric space are different. Then, without loss of generality $a > b > c$. But then, $a > \max(a, b)$ which violates (1). Hence, there must be at least two equal sides in every triangle in an ultrametric space.

For a finite set X, and a symmetric function $W : X \times X \to \mathbb{R}^+$ let $\mathcal{L}(W)$ denote the *maximal metric* on X less than of equal to W (Bridson and Haefliger, 1999), that is,

$$\mathcal{L}(W)(x,x') = \min\left\{ \sum_{i=0}^{m-1} W(x_i,x_{i+1}) \Big| x = x_0,\ldots,x_m = x \right\}$$

for $x,x' \in X$.

For a finite set X, we let $\mathcal{C}(X)$ denote the collection of all non-empty subsets of X. By $\mathcal{P}(X)$ we denote the set of all partitions of X. For a given partition $\Pi \in \mathcal{P}(X)$ we refer to each $\mathcal{B} \in \Pi$ as a *block* of Π. For partitions $\Pi, \Pi' \in \mathcal{P}(X)$, we say that Π is *coarser* than Π', or equivalently that Π' is a *refinement* of Π, if for every block $\mathcal{B}' \in \Pi'$ there exists a block $\mathcal{B} \in \Pi$ s.t. $\mathcal{B}' \subset \mathcal{B}$.

For $k \in \mathbb{N}$ and $r > 0$ let $S^{k-1}(r) \subset \mathbb{R}^k$ denote the $(k-1)$ dimensional sphere with radius r. By $((a))$ we will denote a matrix of elements a_{ij}.

3. Hierarchical Clustering: Formulation

In this section we formaly define hierarchical clustering methods as maps that assign a dendrogram to a finite metric space. First, in §3.1 formalize the standard concept of dendrogram; then, in §3.2 we present a formal treatment of HC methods which emphasizes the need for a formulation that is insensitive to arbitrary choices such as the labels given to the points in the data set. Finally, in §3.3 we prove that the collection of all dendrograms over a finite set is in a one to one correspondence with the collection of all ultrametrics on this set. We then redefine HC methods as maps from the collection of finite metric spaces to the collection all finite ultrametric spaces. This change in perspective permits a natural formulation and study of the *stability* and *convergence* issues in later sections of the paper. In particular, in §3.5, we discuss the construction of notions of *distance between dendrograms* by appealing to the ultrametric representation. These notions are instrumental for the arguments in §5.

Finally, in §3.6, we disgress on some critiques to the classical HC methods. The situation with HC methods is seemingly paradoxical in that SL is the one that seems to enjoys the best theoretical properties while CL and AL, despite exhibiting some undesirable behaviour, are the usual choices of practicioners.

3.1 Dendrograms

A **dendrogram** over a finite set X is defined to be nested family of partitions, usually represented graphically as a rooted tree. Dendrograms are meant to represent a hierarchical decompositions of the underlying set X, such as those that are produced by hierarchical clustering algorithms, and therefore the nested family of partitions provided must satisfy certain conditions. We formally describe dendrograms as pairs (X,θ), where X is a finite set and $\theta : [0,\infty) \to \mathcal{P}(X)$. The parameter of θ usually represents a certain notion of *scale* and it is reflected in the height of the different levels, see Figure 3.1. We require that θ satisfies:

1. $\theta(0) = \{\{x_1\},\ldots,\{x_n\}\}$. This condition means that the initial decomposition of space is the finest possible: the space itself.

2. There exists t_0 s.t. $\theta(t)$ is the *single block partition* for all $t \geqslant t_0$. This condition encondes the fact that for large enough t, the partition of the space becomes trivial.

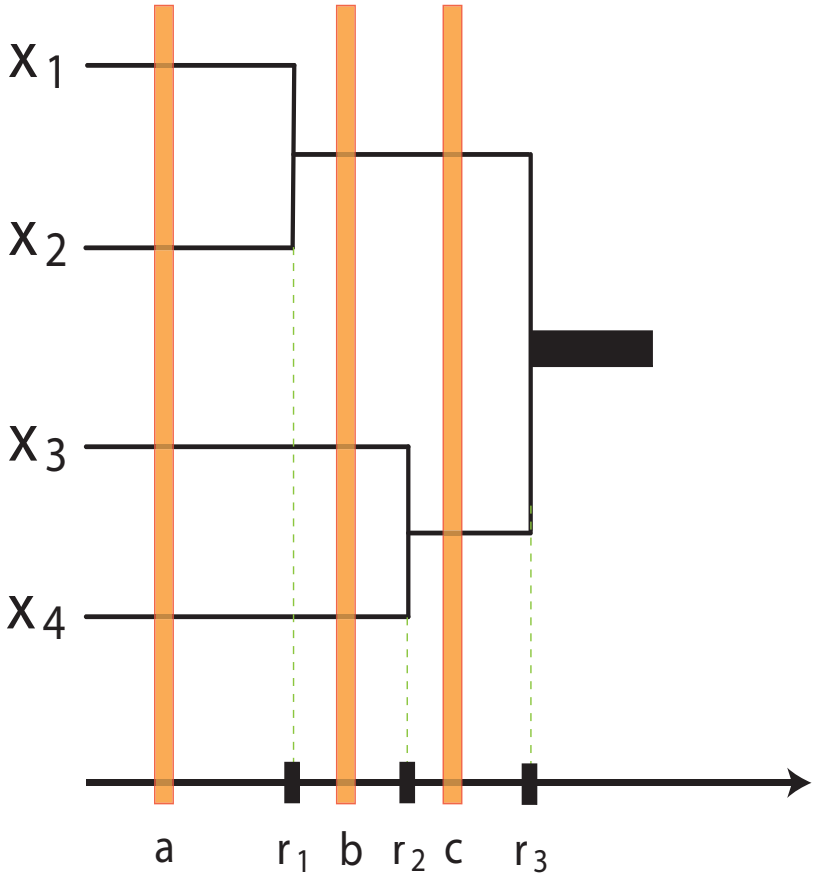


Figure 4: A graphical representation of a dendrogram over the set $X = \{x_1, x_2, x_3, x_4\}$. Let θ denote the dendrogram. Notice for example that $\theta(a) = \big\{\{x_1\},\{x_2\},\{x_3\},\{x_4\}\big\}$; $\theta(b) = \big\{\{x_1,x_2\},\{x_3\},\{x_4\}\big\}$; $\theta(c) = \big\{\{x_1,x_2\},\{x_3,x_4\}\big\}$; and $\theta(t) = \big\{x_1,x_2,x_3,x_4\big\}$ for any $t \geqslant r_3$.

3. If $r \leqslant s$ then $\theta(r)$ *refines* $\theta(s)$. This condition ensures that the family of partitions provided by the dendrogram is indeed nested.

4. For all r there exists $\varepsilon > 0$ s.t. $\theta(r) = \theta(t)$ for $t \in [r, r+\varepsilon]$. (technical condition)

Let $\mathcal{D}(X)$ denote the collection of all possible dendrograms over a given finite set X. When understood from context, we will omit the first component of a dendrogram $(X,\theta) \in \mathcal{D}(X)$ and refer to θ as a dendrogram over X.

Remark 3 (About our definition of dendrogram) *Our definition coincides with what Jain and Dubes call* ***proximity dendrograms*** *in Jain and Dubes (1988, §3.2). We stress that we view the parameter t in our definition as part of the information about the hierarchical clustering. Jain and Dubes also discuss a simpler version of dendrograms, which they call* threshold dendrograms, *which retain merely the order in which succesive partitions are created. These of course can be viewed as functions from $\mathbb{N}$ into $\mathcal{P}(X)$ satisfying the constraints (1), (2) and (3) above, instead of having the domain $[0,\infty)$.*

It seems that proximity dendrograms are the type of dendrograms that are most often employed by practicioners and statisticians, see for example the dendrograms provided by the statistical software R[2] *and by Matlab's statistics toolbox,*[3] *whereas threshold dendrograms are more popular in the Machine Learning and Computer Science communities.*

Usually, Hierarchical Clustering methods are defined as those maps that to each finite metric space (X,d) assign a dendrogram over X.

Using the definitions above we now construct our first example.

Example 2 *For each finite metric space (X,d) let $(X,\theta^*) \in \mathcal{D}(X)$ be given by $\theta^*(r) = X\setminus\sim_r$. In other words, for each $r \geqslant 0$, $\theta^*(r)$ returns the partition of X into $\sim_r$-equivalence classes. Recall (Example 1) that two points x amd x' are $\sim_r$ equivalent if and only if one can find a sequence of points $x_0, x_1, \ldots, x_k$ s.t. the first of them is x and the last one is x' and all the hops are smaller than r: $\max_i d_X(x_i, x_{i+1}) \leqslant r$. We will see below that this definition coincides with single linkage hierarchical clustering. See Figure 2 for an illustration of this concept.*

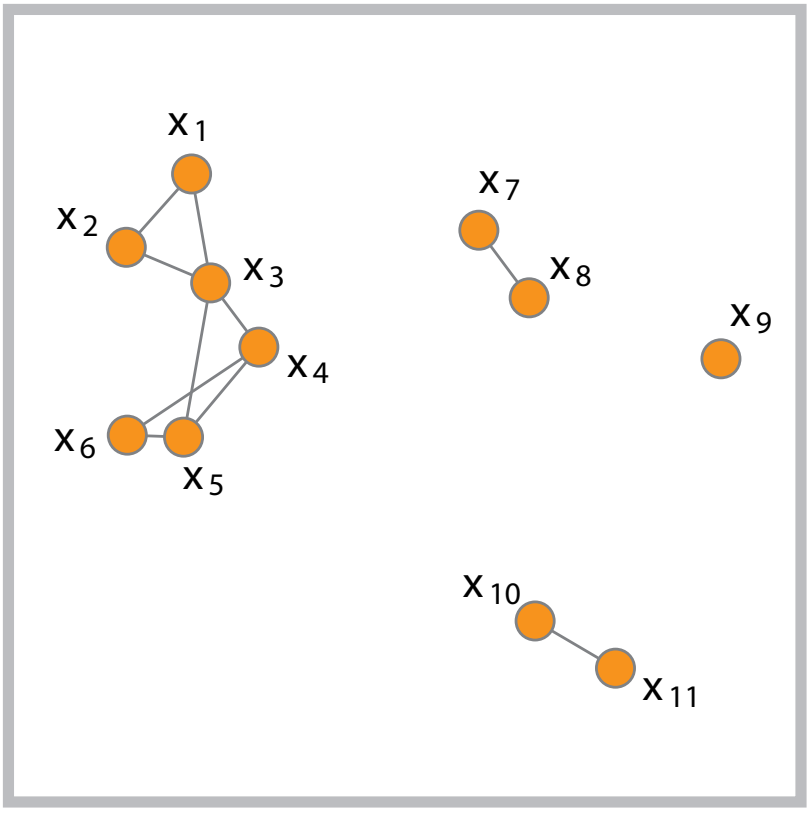


Figure 5: For the same finite metric space X of Example 1 and the value $r = r_2$, $X\setminus\sim_{r_2} = \big\{\{x_1,x_2,x_3,x_4,x_5,x_6\},\{x_7,x_8\},\{x_9\},\{x_{10},x_{11}\}\big\}$, that is, $\sim_{r_2}$ splits X into four path connected components.

2. Available at `http://www.r-project.org/`.
3. Available at `http://www.mathworks.com/products/statistics/`.

In order to build up intuition about our definitions, we prove that (X,θ^*) is indeed a dendrogram. Since X is a metric space, $x \sim_0 x'$ if and only if $x = x'$. Thus condition (1) above is satisfied. Clearly, for $t \geqslant diam(X,d)$, $x \sim_t x'$ for all x,x', and thus condition (2) holds. Fix $0 \leqslant r \leqslant s$ and let $\mathcal{B}$ be a maximal connected component of $\theta^*(r)$ and let $x,x' \in \mathcal{B}$. Then, by definition of $\theta^*(r)$, $x \sim_r x'$. But it follows from the definition of $\sim_r$ that if $x \sim_r x'$, then $x \sim_s x'$ for all $s \geqslant r$. Hence, x,x' are in the same block of $\theta^*(s)$ and condition (3) holds. Condition (4) holds since clearly θ^* is right continuous, has finitely many discontinuity points, and is piecewise constant.

We now need to discuss a formal description of agglomerative HC methods.

3.2 A General Description of Agglomerative Hierarchical Clustering Methods

In this section we give a description of agglomerative HC methods that is suitable for our theoretical analyses. Standard algorithmic descriptions of HC methods typically make the assumption that in the merging process there are only two points at minimal linkage value of eachother. For example, the formulation of Lance and Williams (1967) does not specifically explain how to deal with the case when more than two points are candidates for merging. In practice one could argue that if at a certain stage, say, three points are at minimal linkage value of eachother, then one could proceed to merge them two at a time, according to some predefined rule that depends on the indices of the points.

Whereas this *tie breaking* strategy seems reasonable from a computational point of view, it invariably leads to dendrograms that depend on the ordering of the points. This is no doubt an undesirable feature that can be translated into, for example, that the results of the clustering methods depend on the order in which the data samples were obtained. Single linkage HC is exempted from this problem however, because of the fact that at each stage only *minimal distances* are taken into account. In contrast, complete and average linkage will produce results that do not behave well under reordering of the points.

The problems arising from ad hoc tie breaking are often not even mentioned in books on clustering. A notable exception is the book Jain and Dubes (1988), especially Section §3.2.6, where the reader can find a careful exposition of these issues.

Below, we formulate HC methods in a way that is independent of these extraneous features. In order to do so , we need to have some kind of **invariance** in the formulation. More precisely, let (X,d_X) be the input metric space, where we assume that $X = \{1,\ldots,n\}$ consists of exactly n points. Write (X,θ_X) is the output dendrogram of a given HC method applied to (X,d_X). Let π be a permutation of the indices $\{1,2,\ldots,n\}$, and (Y,d_Y) be the metric space with points $\{1,\ldots,n\}$ and permuted metric: $d_Y(i,j) := d_X(\pi_i,\pi_j)$ for all $i,j \in \{1,\ldots,n\}$; further, denote by (Y,θ_Y) the output dendrogram of the same HC method applied on (Y,d_Y). Then, we require that for all permutations π, the result of computing the dendrogram first and then permuting the result **is the same** as the result of first permuting the input distance matrix and then computing the output dendrogram:

$$\pi \circ \theta_X(t) = \theta_Y(t), \text{ for all } t \geqslant 0. \tag{2}$$

Formally, the action of a permutation π over a partition (such as $\theta_X(t)$) above must be understood in the following sense: if $P = \{\mathcal{B}_1,\ldots,\mathcal{B}_r\}$ is a partition of $\{1,2,\ldots,n\}$, then $\pi \circ P$ is the partition with blocks $\{\pi \circ \mathcal{B}_i,\, 1 \leqslant i \leqslant r\}$, where in turn $\pi \circ \mathcal{B}_i$ consists of all those indices π_j for $j \in \mathcal{B}_i$.

We elaborate on this in the next example. We first recall the usual definition of CLHC, and then construct a simple metric space consisting of five points where this usual formulation of CL fails to exhibit invariance to permutations.

3.2.1 THE STANDARD FORMULATION OF COMPLETE LINKAGE HC

We assume $(X,((d)))$ is a given finite metric space. In this example, we use the formulas for CL but the structure of the iterative procedure in this example is common to all HC methods (Jain and Dubes, 1988, Chapter 3). Let θ be the dendrogram to be constructed in this example.

1. Set $X_0 = X$ and $D_0 = ((d))$ and set $\theta(0)$ to be the partition of X into singletons.

2. Search the matrix D_0 for the smallest non-zero value, that is, find $\delta_0 = \mathrm{sep}(X_0)$, and find all pairs of points $\{(x_{i_1},x_{j_1}),(x_{i_2},x_{j_2})\ldots,(x_{i_k},x_{j_k})\}$ at distance δ_0 from eachother, that is, $d(x_{i_\alpha},x_{j_\alpha}) = \delta_0$ for all $\alpha = 1,2,\ldots,k$, where one orders the indices s.t. $i_1 < i_2 < \ldots < i_k$.

3. Merge the first pair of elements in that list, (x_{i_1},x_{j_1}), into a single group. The procedure now removes (x_{i_1},x_{j_1}) from the initial set of points and adds a point c to represent the cluster formed by both: define $X_1 = \big(X_0\backslash\{x_{i_1},x_{j_1}\}\big)\cup\{c\}$. Define the dissimilarity matrix D_1 on $X_1\times X_1$ by $D_1(a,b) = D_0(a,b)$ for all $a,b\neq c$ and $D_1(a,c) = D_1(c,a) = \max\big(D_0(x_{i_1},a),D_0(x_{j_1},a)\big)$ (this step is the only one that depends on the choice corresponding to CL). Finally, set

$$\theta(\delta) = \{x_{i_1},x_{j_1}\}\cup\bigcup_{i\neq i_1,j_1}\{x_i\}.$$

4. The construction of the dendrogram θ is completed by repeating the previous steps until all points have been merged into a single cluster.

Example 3 (about the standard formulation of complete linkage) *The crux of the problem lies in step 3 of the procedure outlined above. The choice to merge just the first pair of points in the list causes the procedure to not behave well under relabeling of the points in the sense of (2).*

An explicit example is the following: consider the metric space $\big(\{1,2,3,4,5\},((d))\big)$ *with five points and distance matrix*

$$((d)) = \begin{array}{c} \\ 1\\2\\3\\4\\5\end{array}\!\!\begin{array}{c}\begin{array}{ccccc}1&2&3&4&5\end{array}\\\begin{pmatrix}0&1&2&5&5\\1&0&3&6&6\\2&3&0&3&7\\5&6&3&0&4\\6&7&4&6&0.\end{pmatrix}\end{array}$$

This metric space arises from considering the graph metric on the graph depicted in Figure 6. Under CLHC (as defined in §3.2.1), and under the action of all possible permutations of the labels of its 5 points, this metric space produces 3 different ***non-equivalent*** *dendrograms, see Figure 7. This is an undesirable feature, as discussed at length in Jain and Dubes (1988, Chapter 3).*

We now re-define general HC methods in a way that they satisfy (2).

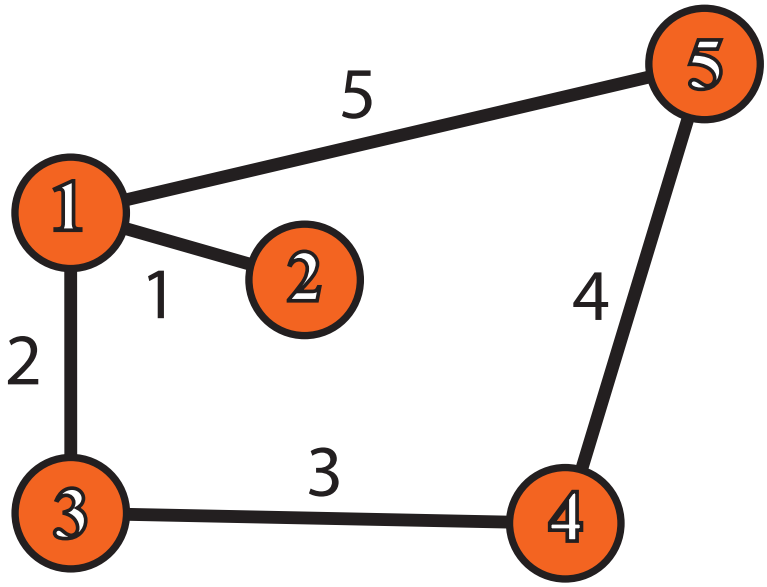


Figure 6: A finite metric space that under permutations leads to different outputs of the usual CL HC algorithm, see text for details. The metric is defined by the graph distance on the weighted graph shown.

3.2.2 THE PERMUTATION INVARIANT FORMULATION

Here we consider the family of Agglomerative Hierarchical clustering techniques (Jain and Dubes, 1988, Chapter 3). We define these by the recursive procedure described next. The main difference with §3.2.1 lies that in Step 3 we will allow for more than just two points into the same cluster and also, it could happen, for example, that four points A,B,C,D merge into two different clusters $\{A,B\}$ and $\{C,D\}$ at the same time.

Let the finite metric space (X,d) be given where $X = \{x_1,\ldots,x_n\}$ and let L denote a family of *linkage functions* on X:

$$L := \{\ell : \mathcal{C}(X)\times\mathcal{C}(X) \to \mathbb{R}^+\}$$

with the property all that $\ell \in L$ are bounded non-negative functions. These functions assign a non-negative value to each pair of non-empty subsets of X, and provide a certain measure of *distance* between two clusters. Let $\mathcal{B},\mathcal{B}' \in \mathcal{C}(X)$, then, some possible standard choices for ℓ are:

- *Single linkage*: $\ell^{\mathrm{SL}}(\mathcal{B},\mathcal{B}') = \min_{x\in\mathcal{B}} \min_{x'\in\mathcal{B}'} d(x,x')$;
- *Complete linkage*: $\ell^{\mathrm{CL}}(\mathcal{B},\mathcal{B}') = \max_{x\in\mathcal{B}} \max_{x'\in\mathcal{B}'} d(x,x')$; and
- *Average linkage*: $\ell^{\mathrm{AL}}(\mathcal{B},\mathcal{B}') = \frac{\sum_{x\in\mathcal{B}}\sum_{x'\in\mathcal{B}'} d(x,x')}{\#\mathcal{B}\cdot\#\mathcal{B}'}$.
- *Hausdorff linkage*: $\ell^{\mathrm{HL}}(\mathcal{B},\mathcal{B}') = d_{\mathcal{H}}(\mathcal{B},\mathcal{B}')$.[4]

The permutation invariant formulation is as follows:

4. The Hausdorff distance is defined in Definition 21.

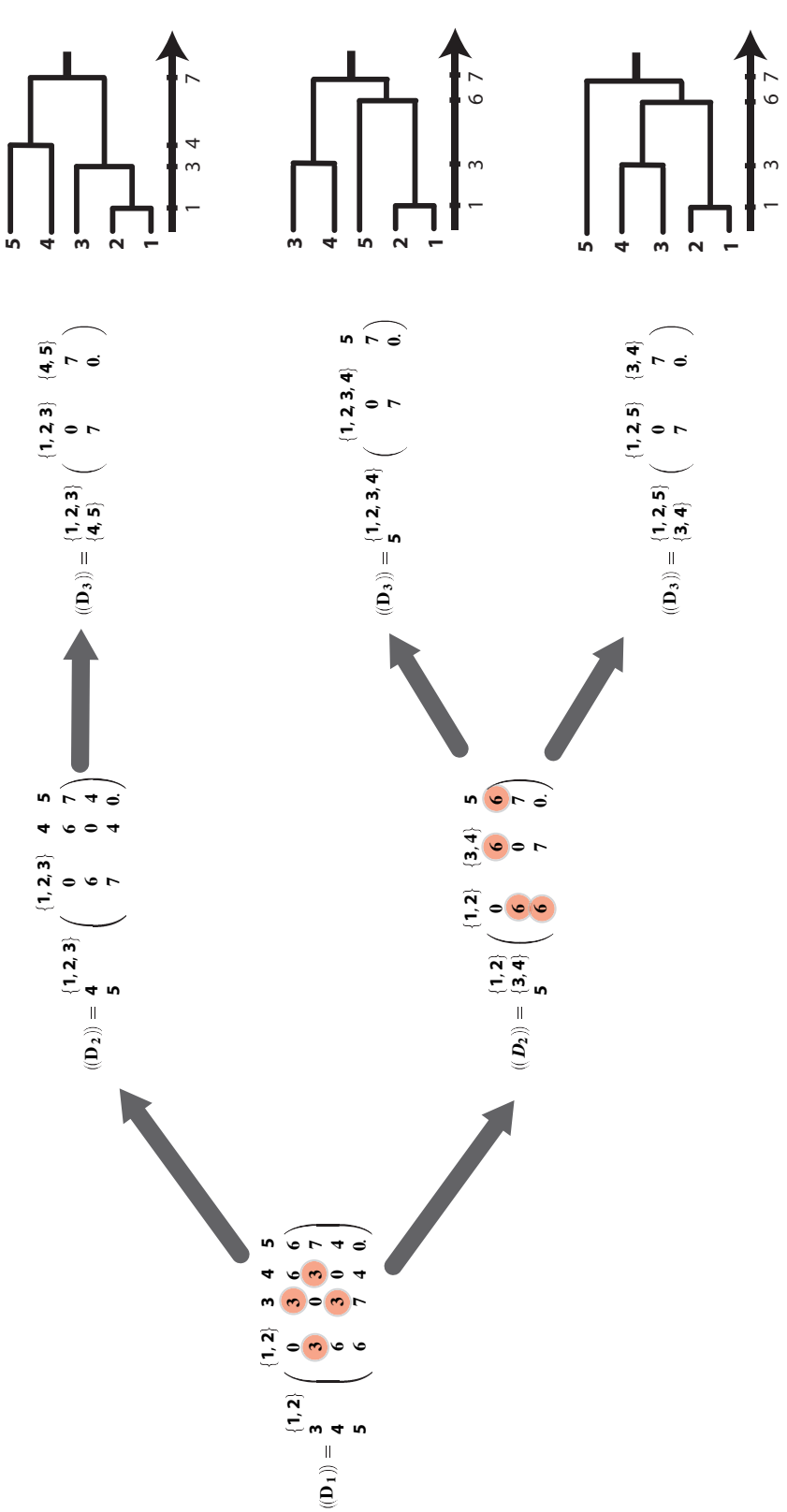


Figure 7: The five point metric space of Example 3 leads to 3 non-equivalent CL-dendrograms. Here we show the agglomerative procedure corresponding to the usual order dependent definition of CL HC, see text in Example 3 for details. Notice that in D_1, there are two pairs at minimal linkage value (this value is 3, in red circles): $\big(\{1,2\},3\big)$ on one hand, and $\big(3,4\big)$ on the other. The top row shows the agglomeration process that follows after merging $\{1,2\}$ with 3. The bottom row shows that when one instead chooses to merge 3 and 4 instead, then at the next step there is again ambiguity (represented again by the red circles). The three dendrograms shown on the right are the three possible outputs one finds when choosing different orders for the agglomeration processes. Clearly, these three dendrograms are not equivalent under permutations of the labels of their base points.

1. Fix $\ell \in L$. For each $R > 0$ consider the equivalence relation $\sim_{\ell,R}$ on blocks of a partition $\Pi \in \mathcal{P}(X)$, given by $\mathcal{B} \sim_{\ell,R} \mathcal{B}'$ if and only if there is a sequence of blocks $\mathcal{B} = \mathcal{B}_1, \ldots, \mathcal{B}_s = \mathcal{B}'$ in Π with $\ell(\mathcal{B}_k, \mathcal{B}_{k+1}) \leqslant R$ for $k = 1, \ldots, s-1$.

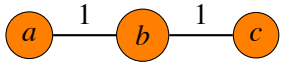
2. Consider the sequences $R_1, R_2, \ldots \in [0, \infty)$ and $\Theta_1, \Theta_2, \ldots \in \mathcal{P}(X)$ given by $\Theta_1 := \{x_1, \ldots, x_n\}$, and recursively for $i \geqslant 1$ by $\Theta_{i+1} = \Theta_i / \sim_{\ell,R_i}$ where

$$R_i := \min\{\ell(\mathcal{B}, \mathcal{B}'); \mathcal{B}, \mathcal{B}' \in \Theta_i, \mathcal{B} \neq \mathcal{B}'\}.$$

   Note that this process necessarily ends in finitely many steps. This construction reflects the fact that at step i one agglomerates those clusters at distance $\leqslant R_i$ from eachother (as measured by the linkage function ℓ). More than two clusters could be merged at any given step.

3. Finally, we define $\theta^\ell : [0, \infty) \to \mathcal{P}(X)$ by $r \mapsto \theta^\ell(r) := \Theta_{i(r)}$ where $i(r) := \max\{i | R_i \leqslant r\}$.

Remark 4 (About our definition of HC methods) *Note that, unlike the usual definition of agglomerative hierarchical clustering §3.2.1 (Jain and Dubes, 1988, §3.2), at each step of the inductive definition we allow for more than two clusters to be merged. Of course, the standard formulation can be recovered if one assumes that at each step i of the algorithm, there exist* only *two blocks $\mathcal{B}$ and $\mathcal{B}'$ in Θ_i s.t. $R_i = \ell(\mathcal{B}, \mathcal{B}')$. Then, at each step, only two blocks will be merged.*

Example 4 *Note for example that for the five point metric space in Example 3, the result of applying CL (according to the permutation invariant formulation) is the dendrogram in Figure 8 (a). It also follows, for example, that when applied to the metric space* $L_3 := \left(\{a,b,c\}, \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} \right)$, *which can be represented by three points on a line:* 

, *SL, AL and CL all yield the same dendrogram, which is shown in Figure 8 (b).*

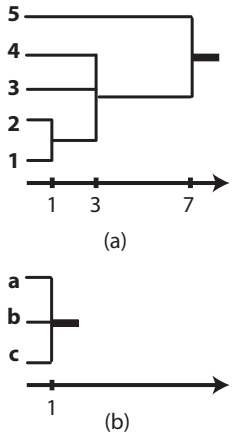


Figure 8: (a) shows the result of applying the permutation invariant formulation of CL to the five point metric space of Example 3 (see also Figure 6). (b) shows the dendrogram that one obtains as output of (the permutation invariant formulation of) SL, AL and CL applied to the metric space L_3.

Proposition 5 *We have the following properties of the construction above:*

- *For $i = 1, 2, \ldots$, Θ_{i+1} is coarser than Θ_i and*
- $R_{i+1} \geqslant R_i$.
- θ^ℓ *is a dendrogram over* X.

Proof The only non trivial claim is that $R_{i+1} \geqslant R_i$, which can be proved by induction on i. ■

Remark 6 *From this point forward, all references to SL, AL, and CL clustering will be to the permutation invariant formulation, in which more than two clusters can be merged at a given step.*

The following result is clear, and we omit its proof.

Proposition 7 *The above construction of hierarchical clustering algorithms (including SL, AL, and CL) yields algorithms which are permutation invariant.*

A simplification for SL HC. In the particular case of SL, there is an alternative formulation that uses the equivalence relation introduced in Example 1 and its associated dendrogram (Example 2). The proof of the following Proposition is deferred to the appendix.

Proposition 8 *Let (X,d) be a finite metric space and θ^{SL} be the dendrogram over X obtained by the single linkage agglomerative procedure described above, and let θ^* be the dendrogram over X constructed in Example 2. Then, $\theta^{SL}(r) = \theta^*(r)$ for all $r \geqslant 0$.*

3.3 Dendrograms as Ultrametric Spaces

The representation of dendrograms as ultrametrics is well known and it appears in the book by Jardine and Sibson (1971), it has already been used in the work of Hartigan (1985), and is touched upon in the classical reference of Jain and Dubes (1988, §3.2.3).

We now present the main ideas regarding this change in perspective which we will adopt for all subsequent considerations. The formulation of the output of hierarchical clustering algorithms as ultrametric spaces is powerful when one is proving stability results, as well as results about the approximation of the dendrograms of metric spaces by their finite subspaces. This is so because of the fact that once a dendrogram is regarded as a metric space, the Gromov-Hausdorff metric provides a very natural notion of distance on the output, in which the right kind of stability results are easily formulated. We state these theorems in §5.

The main result in this section is that dendrograms and ultrametrics are **equivalent**.

Theorem 9 *Given a finite set X, there is a bijection $\Psi : \mathcal{D}(X) \to \mathcal{U}(X)$ between the collection $\mathcal{D}(X)$ of all dendrograms over X and the collection $\mathcal{U}(X)$ of all ultrametrics over X such that for any dendrogram $\theta \in \mathcal{D}(X)$ the ultrametric $\Psi(\theta)$ over X generates the same hierarchical decomposition as θ, that is,*

$$(*) \quad \text{for each } r \geqslant 0,\ x, x' \in \mathcal{B} \in \theta(r) \iff \Psi(\theta)(x,x') \leqslant r.$$

Furthermore, this bijection is given by

$$\Psi(\theta)(x,x') = \min\{r \geqslant 0 \,|\, x, x' \text{ belong to the same block of } \theta(r)\}.$$

In order to establish the above theorem, we first construct certain natural mappings from $\mathcal{D}(X)$ to $\mathcal{U}(X)$ and from $\mathcal{U}(X)$ to $\mathcal{D}(X)$, and we then prove they are inverses of eachother and satisfy $(*)$.

3.3.1 FROM DENDROGRAMS TO ULTRAMETRICS

Let X be a finite set and $\theta : [0,\infty) \to \mathcal{P}(X)$ a dendrogram over X. Consider the symmetric map $u_\theta : X \times X \to \mathbb{R}^+$ given by

$$(x,x') \mapsto \min\{r \geqslant 0 \,|\, x,x' \text{belong to the same block of } \theta(r)\}. \tag{3}$$

See Figure 9 for an illustration of this definition. Note that condition (4) in the definition of dendrograms guarantees that u_θ is well defined. It is easy to see that u_θ defines an ultrametric on X:

Lemma 10 *Let X be a finite set and $(X,\theta) \in \mathcal{D}(X)$. Then $u_\theta : X \times X \to \mathbb{R}^+$ defined in (3) is an ultrametric.*

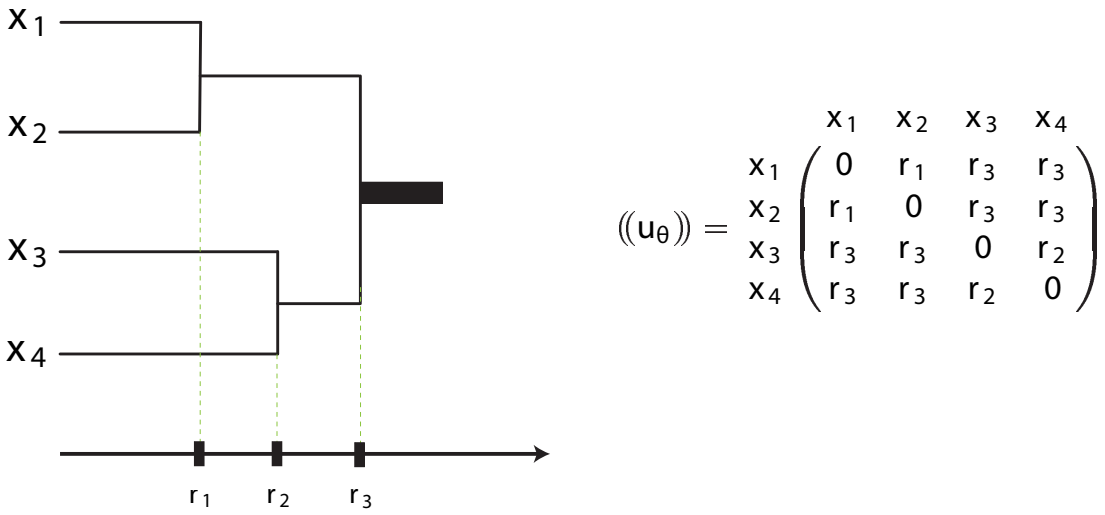


Figure 9: A graphical representation of a dendrogram θ over $X = \{x_1,x_2,x_3,x_3\}$ and the ultrametric u_θ. Notice for example, that according to (3), $u_\theta(x_1,x_2) = r_1$ since r_1 is the first value of the (scale) parameter for which x_1 and x_2 are merged into the same cluster. Similarly, since x_1 and x_3 are merged into the same cluster for the first time when the parameter equals r_3, then $u_\theta(x_1,x_3) = r_3$.

3.3.2 FROM ULTRAMETRICS TO DENDROGRAMS

Conversely, given an ultrametric $u : X \times X \to \mathbb{R}^+$, its associated dendrogram

$$\theta^u : [0,\infty) \to \mathcal{P}(X)$$

can be obtained as follows: for each $r \geqslant 0$ let $\theta^u(r)$ be the collection of equivalence classes of X under the relation $x \sim x'$ if and only if $u(x,x') \leqslant r$. That this defines an equivalence relation follows immediately from the fact that u is an ultrametric. Indeed, assume that $x \sim x'$ and

$x' \sim x''$ for some $r \geqslant 0$. Then, $u(x,x') \leqslant r$ and $u(x',x'') \leqslant r$. Now, by the ultrametric property, $\max\big(u(x,x'),u(x',x'')\big) \geqslant u(x,x'')$ and hence $u(x,x'') \leqslant r$ as well. We conclude that $x \sim x''$ thus establishing the transitivity of $\sim$.

Example 5 *Consider the ultrametric u on* $X = \{x_1,x_2,\ldots,x_6\}$ *given by*

$$((u)) = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{array} \\ \begin{pmatrix} 0 & 2 & 2 & 5 & 6 & 6 \\ 2 & 0 & 2 & 5 & 6 & 6 \\ 2 & 2 & 0 & 5 & 6 & 6 \\ 5 & 5 & 5 & 0 & 6 & 6 \\ 6 & 6 & 6 & 6 & 0 & 4 \\ 6 & 6 & 6 & 6 & 4 & 0. \end{pmatrix} \end{array}$$

Then, for example $\theta^u(0) = \big\{\{x_1\},\{x_2\},\{x_3\},\{x_4\},\{x_5\},\{x_6\}\big\}$, $\theta^u(3) = \big\{\{x_1,x_2,x_3\},\{x_4\},\{x_5\},\{x_6\}\big\}$, $\theta^u(4.5) = \big\{\{x_1,x_2,x_3\},\{x_4\},\{x_5,x_6\}\big\}$, $\theta^u(5.5) = \big\{\{x_1,x_2,x_3,x_4\},\{x_5,x_6\}\big\}$ *and* $\theta^u(7) = \big\{x_1,x_2,x_3,x_4,x_5,x_6\big\}$. *A graphical representation of the dendrogram* θ^u *is given in Figure 10.*

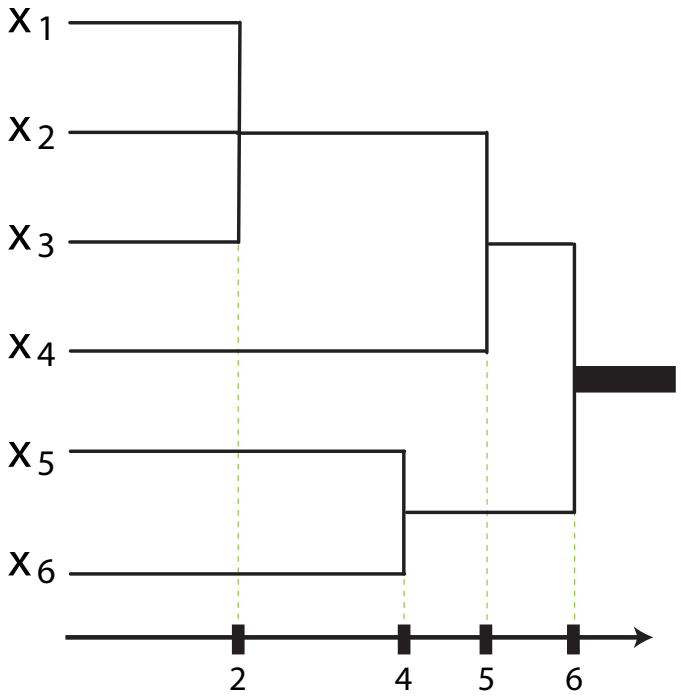


Figure 10: A graphical representation of the dendrogram θ^u of Example 5, see the text for details.

3.3.3 THE CONCLUSION OF THE PROOF OF THEOREM 9.

It is easy to check that (1) given any dendrogram θ on X, $\theta^{u_\theta} = \theta$ and (2) given any ultrametric u on X, $u_{\theta^u} = u$. Now, let $\Psi : \mathcal{D}(X) \to \mathcal{U}(X)$ be defined by $\theta \mapsto \Psi(\theta) := u_\theta$. By construction we see that

$\Psi : \mathcal{D}(X) \to \mathcal{U}(X)$ is a bijection and that Ψ^{-1} is given by $u \mapsto \theta^u$. From (3) we see that Ψ satisfies $(*)$. Hence, we obtain Theorem 9.

From now, whenever given a dendrogram θ_X over a set X, we will be using the notation $\Psi(\theta_X)$ for the ultrametric associated to X given by Theorem 9. In a similar manner, given an ultrametric u on X, $\Psi^{-1}(u)$ will denote the dendrogram over X given by Theorem 9.

3.4 Reformulation of Hierarchical Clustering using Ultrametrics

In the sequel, appealing to Theorem 9 which states the **equivalence between ultrametrics and dendrograms**, we represent dendrograms as *ultrametric* spaces. Then, any hierarchical clustering method can be regarded as a map from finite metric spaces into finite ultrametric spaces. This motivates the following definition:

Definition 11 *A* hierarchical clustering method *is defined to be a map*

$$\mathfrak{T} : \mathcal{X} \to \mathcal{U} \quad s.t. \quad \mathcal{X}_n \ni (X,d) \mapsto (X,u) \in \mathcal{U}_n,\ n \in \mathbb{N}.$$

Example 6 *For a given finite metric space (X,d) consider the HC method $\mathfrak{T}^{\mathrm{SL}}$ given by $\mathfrak{T}^{\mathrm{SL}}(X,d) = (X, \Psi(\theta^{\mathrm{SL}}))$, where θ^{SL} is the single linkage dendrogram over X defined in §3.2. Similarly, we define $\mathfrak{T}^{\mathrm{CL}}$ and $\mathfrak{T}^{\mathrm{AL}}$.*

Example 7 (maximal sub-dominant ultrametric) *There is a canonical construction: Let $\mathfrak{T}^* : \mathcal{X} \to \mathcal{U}$ be given by $(X,d) \mapsto (X,u^*)$ where*

$$u^*(x,x') := \min \left\{ \max_{i=0,\ldots,k-1} d(x_i, x_{i+1}),\ s.t.\ x = x_0, \ldots, x_k = x' \right\}. \tag{4}$$

We remark that the minimum above is taken over $k \in \mathbb{N}$ and all $k+1$-tuples of points $x_0, x_1, \ldots, x_k$ in X s.t. $x_0 = x$ and $x_k = x'$. Notice that for all $x, x' \in X$, $u^*(x,x') \leqslant d(x,x')$.

*This construction is sometimes known as the **maximal sub-dominant ultrametric** and it has the property that if $u \leqslant d$ is any other ultrametric on X, then $u \leqslant u^*$. The Lemma below proves that this canonical construction is equivalent to the ultrametric induced by the equivalence relation in Example 1.*

Lemma 12 *For $(X,d) \in \mathcal{X}$ write $\mathfrak{T}^*(X,d) = (X,u^*)$ and let $(X,\theta^*) \in \mathcal{D}(X)$ be the dendrogram arising from the construction in Example 2. Then, $u^* = \Psi(\theta^*)$.*

Remark 13 *Notice that another way of stating the Lemma above is that $x \sim_r x'$ if and only if $u^*(x,x') \leqslant r$.*

It turns out that T* yields **exactly** *single linkage clustering* as defined in §3.2.

Corollary 14 *One has that $\mathfrak{T}^{SL} = \mathfrak{T}^*$.*

Equivalently, for any finite metric space X, the single linkage dendrogram θ^{SL} on X agrees with $\Psi^{-1}(u^*)$.

Proof The proof follows easily from Proposition 8 and Lemma 12. ■

We emphasize that, as it follows from Corollary 14, $\mathfrak{T}^*$ produces ultrametric outputs which are exactly those corresponding to SLHC. We will use this fact strongly in the sequel.

3.4.1 INTERPRETATION OF THE ULTRAMETRIC

For a HC method $\mathfrak{T}$ and $(X,d) \in \mathcal{X}$, let $\mathfrak{T}(X,d) = (X,u)$. The intuition that arises from (3) is that for two points $x,x' \in X$, $u(x,x')$ measures the minimal **effort** method $\mathfrak{T}$ makes in order to join x to x' into the same cluster.

We note in particular that a desirable property of a HC algorithm should be that upon shrinking some of the distances in the input metric space, the corresponding "efforts" also decrease. This property is exactly verified by $\mathfrak{T}^*$. Indeed, let X be a finite set and d_1 and d_2 two metrics on X s.t. $d_1 \geqslant d_2$. Write $\mathfrak{T}^*(X,d_1) = (X,u_1^*)$ and $\mathfrak{T}^*(X,d_2) = (X,u_2^*)$. Then, it follows immediately from Equation (4) that $u_1^* \geqslant u_2^*$ (compare with Kleinberg's *consistency* property, pp 1425).

Observe that CL and AL HC *fail to satisfy this property*. An example is provided in Figure 19.

We see in Theorem 18 that a condition of this type, together with two more natural normalizing conditions, *completely characterizes SLHC*.

3.5 Comparing results of Hierarchical Clustering Methods

One of the goals of this paper is to study the stability of clustering methods to perturbations in the input metric space. In order to do so one needs to define certain suitable notions of distance between dendrograms. We choose to do this by appealing to the ultrametric representation of dendrograms, which provides a natural way of defining a distance between hierarchical clusterings. We now delve into the construction.

Consider first the simple case of two different dendrograms (X,α) and (X,β) over the same fixed finite set X. In this case, as a tentative measure of dissimilarity between the dendrograms we look at the maximal difference between the associated ultrametrics given by Theorem 9: $u_\alpha = \Psi(\alpha)$ and $u_\beta = \Psi(\beta)$: $\max_{x,x' \in X} |u_\alpha(x,x') - u_\beta(x,x')|$. There is a natural interpretation of the condition that $\max_{x,x' \in X} |u_\alpha(x,x') - u_\beta(x,x')| \leqslant \varepsilon$: if we look at the graphical representation of the dendrograms α and β, then the transition horizontal lines in Figure 11 have to occur within ε of eachother.[5] This is easy to see by recalling that by (3),

$$u_\alpha(x,x') = \min\{r \geqslant 0 \,|\, x,x' \text{belong to the same block of } \alpha(r)\}$$

and

$$u_\beta(x,x') = \min\{r \geqslant 0 \,|\, x,x' \text{belong to the same block of } \beta(r)\}.$$

For the example in Figure 11, we then obtain that $\max_i |r_i - r_i'| \leqslant \varepsilon$, which is not surprising since $r_2 = u_\alpha(x_1,x_2)$, $r_2' = u_\beta(x_1,x_2)$, etc.

5. These lines represent values of the scale parameter for which there is a merging of blocks of the partitions encoded by the dendrograms.

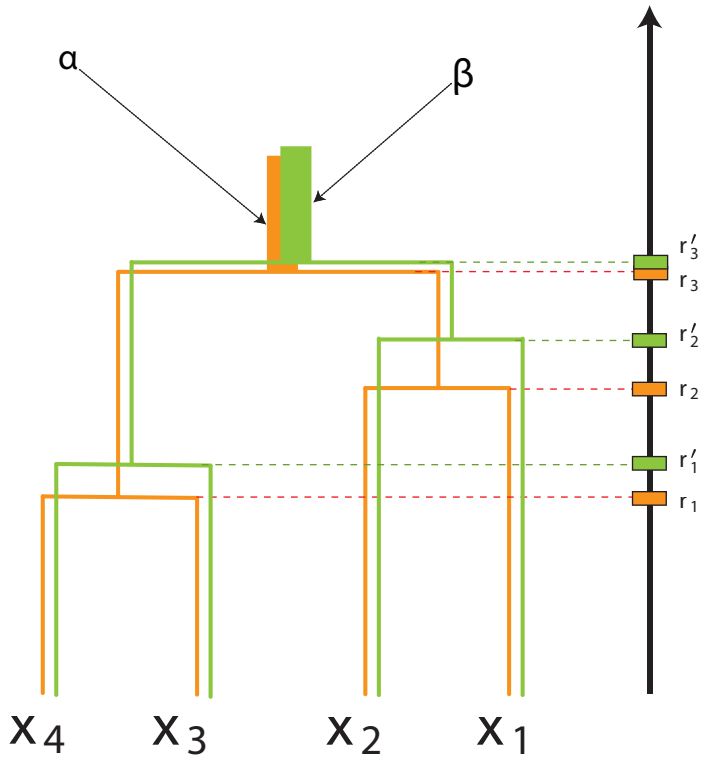


Figure 11: Two different dendrograms (X,α) and (X,β) over the same underlying set $X = \{x_1,x_2,x_3,x_4\}$. The condition that $\|u_\alpha - u_\beta\|_{L^\infty(X\times X)} \leqslant \varepsilon$ is equivalent to the horizontal dotted lines corresponding to r_i and r'_i $(i = 1,2,3)$ being within ε of eachother.

Now, in a slightly more general situation we may be faced with the task of comparing two different dendrograms α and β without knowing (or caring about) the exact labels of the points. In this case, a natural solution is to look at the minimum of the maximum difference of the corresponding ultrametrics under all possible permutations, namely:

$$\min_{\pi\in P_n} \max_{x,x'\in X} |u_\alpha(x,x') - u_\beta(\pi(x),\pi(x'))|, \tag{5}$$

where n is the cardinality of X and P_n is the collection of all permutations of n elements.

The most general case arises when we do not know whether the dendrograms come from the same underlying set or not. This situation may arise, for example, when comparing the results of clustering two different samples, of possibly different sizes, coming from the same data set. One may want to be able to compare two such clusterings as a way to ascertain whether the sample size is sufficient for capturing the structure of the underlying data set.

Assume then that we are given (X_1,α) and (X_2,β), two different dendrograms, defined possibly over two *different sets* X_1 and X_2 of different cardinality. This potential difference in cardinality in the two sets forces us to consider transformations other than mere permutations. A natural solution, which can be interpreted as a *relaxation* of the permutation based distance (5) discussed above, is to consider maps $f : X_1 \to X_2$ and $g : X_2 \to X_1$ and look at their *distortions*:

$$\operatorname{dis}(f) := \max_{x,x' \in X_1} |u_\alpha(x,x') - u_\beta(f(x),f(x'))|,$$

$$\operatorname{dis}(g) := \max_{x,x' \in X_2} |u_\alpha(g(x),g(x')) - u_\beta(x,x)|.$$

The next natural step would be to optimize over the choice of f and g, for example by minimizing the maximum of the two distortions:

$$\min_{f,g} \max\big(\operatorname{dis}(f), \operatorname{dis}(g)\big).$$

This construction is depicted in Figure 12. Roughly speaking, this idea leads to the Gromov-Hausdorff distance. The difference lies in the fact that in standard definition of the Gromov-Hausdorff distance, one also considers a term that measures the degree to which f and g are inverses of eachother. Being more precise, given the maps f and g, this term, called the *joint distortion* of f and g is given by

$$\operatorname{dis}(f,g) := \max_{x \in X, y \in Y} \big|u_\alpha(x,g(y)) - u_\beta(y,f(x))\big|.$$

One defines the Gromov-Hausdorff distance between (X_1,u_α) and (X_2,u_β) by

$$d_{\mathcal{GH}}\big(X_1,X_2\big) := \frac{1}{2} \min_{f,g} \max\big(\operatorname{dis}(f), \operatorname{dis}(g), \operatorname{dis}(f,g)\big).^{6} \tag{6}$$

We now see exactly how the inclusion of the new term enforces f and g to be approximate inverses of eachother. Assume that for some $\varepsilon > 0$ $d_{\mathcal{GH}}\big(X_1,X_2\big) < \varepsilon$, then, in particular, there exist maps f and g such that $\big|u_\alpha(x,g(y)) - u_\beta(y,f(x))\big| \leqslant 2\varepsilon$ for all $x \in X_1$ and $y \in X_2$. Choosing $y = f(x)$, in particular, we obtain that $u_\alpha(x,g(f(x))) \leqslant 2\varepsilon$ for all $x \in X_1$. Similarly one obtains that $u_\beta(y,f(g(y))) \leqslant 2\varepsilon$ for all $y \in X_2$. These two inequalities measure the degree to which $f \circ g$ and $g \circ f$ differ from the identities, and thus, measure the degree to which f and g fail to be inverses of eachother. This is a useful feature when one considers *convergence issues* such as we do in §5.

3.5.1 INTERPRETATION OF THE GROMOV-HAUSDORFF DISTANCE IN TERMS OF DENDROGRAMS

Assume that $d_{\mathcal{GH}}\big((X_1,u_\alpha),(X_2,u_\beta)\big) \leqslant \frac{\eta}{2}$ for some $\eta \geqslant 0$. Then there exist maps $f : X \to Y$ and $g : Y \to X$ such that the following conditions hold (see Figure 13):

- If x,x' fall in the same block of $\alpha(t)$ then $f(x), f(x')$ belong to the same block of $\beta(t')$ for all $t' \geqslant t + \eta$.
- If y,y' fall in the same block of $\beta(t)$ then $g(y), g(y')$ belong to the same block of $\alpha(t')$ for all $t' \geqslant t + \eta$.

For the next section we do not need to make use of the full generality in these considerations: there we only compare dendrograms defined over the same underlying set. A more detailed use and additional material about the Gromov-Hausdorff ideas is given in §5.

We finish this section with a precise result regarding the stability of dendrograms arising from SLHC.

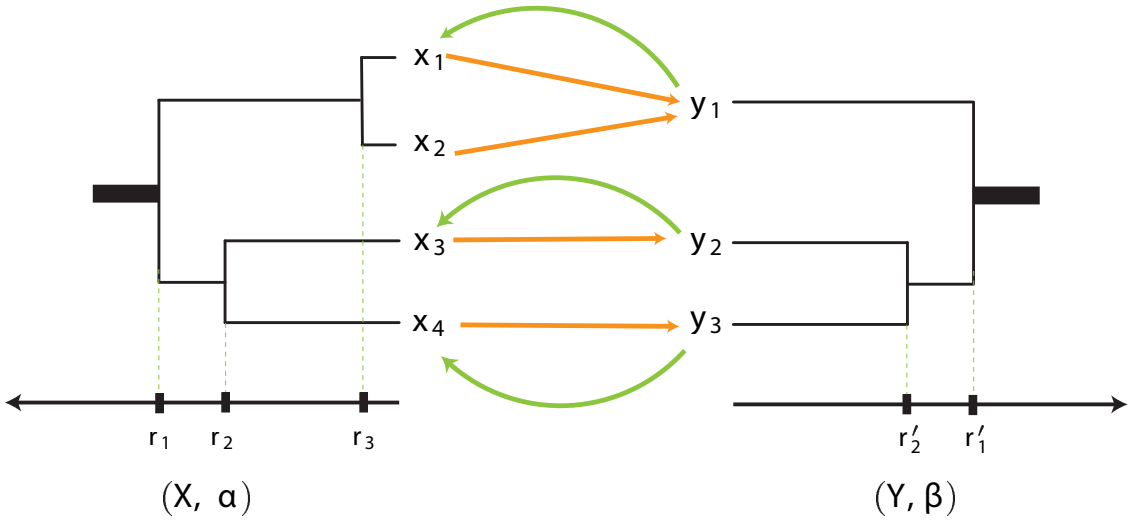


Figure 12: In this example, two different dendrograms, (X,α) and (Y,β), are given. The (straight) arrows pointing from left to right show a map $f : X \to Y$, and the (curved) arrows pointing from right to left show the map $g : Y \to X$. With simple explicit computations one sees that these choices of maps f and g incur distortions $\mathrm{dis}(f) = \mathrm{dis}(f,g) = \max\left(r_3, |r_1 - r_1'|, |r_2 - r_2'|\right)$ and $\mathrm{dis}(g) = \max\left(|r_1 - r_1'|, |r_2 - r_2'|\right)$, respectively. Hence, we see that $d_{\mathcal{GH}}\big((X,\Psi(\alpha)),(Y,\Psi(\beta))\big) \leqslant \frac{1}{2}\max\left(r_3, |r_1 - r_1'|, |r_2 - r_2'|\right)$.

The following Lemma deals with the situation when we have a fixed finite set P and two different metrics on P and then we compute the result of applying $\mathfrak{T}^*$ each of these metrics. This lemma is a particular case of our main stability result, Proposition 26 in §5. In the interest of clarity, we prove it here to provide some intuition about the techniques.

Lemma 15 *Let P be a fixed finite set and let d_1, d_2 be two metrics on P. Write $\mathfrak{T}^*(P,d_i) = (P,u_i)$, $i = 1,2$. Then,*

$$\max_{p,q\in P} |u_1(p,q) - u_2(p,q)| \leqslant \max_{p,q\in P} |d_1(p,q) - d_2(p,q)|.$$

Proof Let $\eta = \max_{p,q\in P} |d_1(p,q) - d_2(p,q)|$. Let $p_0,\ldots,p_k \in P$ be s.t. $p_0 = p$, $p_k = q$ and $\max_i d_1(p_i,p_{i+1}) = u_1(p,q)$. Then, by definition of u_2 (which is the minimum over all chains of the maximal hop measured with metric d_2) and the fact that $d_2 \leqslant d_1 + \eta$:

$$u_2(p,q) \leqslant \max_i d_2(p_i,p_{i+1}) \leqslant \max_i (\eta + d_1(p_i,p_{i+1})) = \eta + u_1(p,q).$$

Similarly, $u_1(p,q) \leqslant \eta + u_2(p,q)$, and hence $|u_1(p,q) - u_2(p,q)| \leqslant \eta$. The claim follows since $p,q \in P$ are arbitrary. ■

6. The factor $\frac{1}{2}$ is of course inmaterial but kept here for coherence with the standard definition.

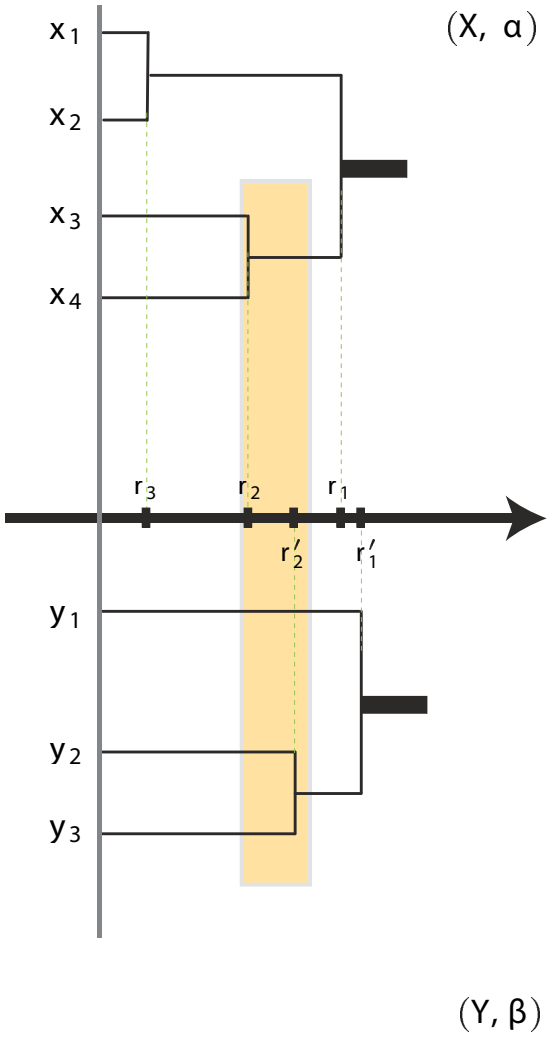


Figure 13: These are the same dendrograms as in Figure 12. Let $r_1 = \frac{5}{3}, r_2 = 1, r_3 = \frac{1}{3}, r'_1 = \frac{11}{6}$ and $r'_2 = \frac{4}{3}$. For the maps f and g s.t. $f(x_1) = f(x_2) = y_1$, $f(x_3) = y_2$, $f(x_4) = y_3$, $g(y_1) = x_1$, $g(y_2) = x_3$ and $g(y_3) = x_4$, using the formulas computed in Figure 12 we see that $\mathrm{dis}(f) = \mathrm{dis}(g) = \mathrm{dis}(f,g) = \frac{1}{3}$ and hence $d_{\mathcal{GH}}\big((X,\Psi(\alpha)),(Y,\Psi(\beta))\big) \leqslant \frac{1}{6}$. Now notice for instance that x_3 and x_4 fall in the same block of $\alpha(r_2) = \alpha(1)$ and that $y_2 = f(x_3)$ and $y_3 = f(x_4)$ fall in the same block of $\beta(t')$ for all $t' \geqslant r_2 + 2 \cdot \frac{1}{6} = 1 + \frac{1}{3} = \frac{4}{3} = r'_2$.

3.6 Some Remarks about Hierarchical Clustering Methods

Practitioners of clustering often prefer AL and CL to SL because it is perceived that the former two methods tend to produce clusters which are more coherent conceptually, and which are in a non-technical sense viewed as more compact. In fact, SL exhibits the so called **chaining effect** which makes it more likely to produce clusterings which separate items which conceptually should be together. We view these observations as evidence for the idea that good clustering schemes need to take some notion of density into account, rather than straightforward geometric information alone.

One can loosely argue that given the actual definition of the linkage functions used by AL and CL, these two methods do enjoy some sort of sensitivity to density. Unfortunately, AL and CL are **unstable**, and in particular, **discontinuous** in a very precise sense (see Remark 16 below), whereas SL enjoys all the nice theoretical properties that the other two methods lack.

In this section we review this seemingly paradoxical situation.

For each $n \in \mathbb{N}$ let L_n be a metric space with n points $P = \{p_1, \ldots, p_n\}$ and metric $d_{L_n}(p_i, p_j) = |i-j|$, $i, j \in \{1, \ldots, n\}$. Similarly, let Δ_n be the metric space with the same underlying set and metric $d_{\Delta_n}(p_i, p_j) = 1$, $i, j \in \{1, \ldots, n\}$, $i \neq j$. Clearly, the metric space L_n is isometric to points equally spaced on a line in Euclidean space whereas (s.t. two adjacent points are at distance 1 from eachother) Δ_n is isometric to the $(n-1)$-unit-simplex as a subset of $\mathbb{R}^{n-1}$.

Clearly, the outputs of Single Linkage HC applied to both L_n and Δ_n coincide for all $n \in \mathbb{N}$:

$$\mathfrak{T}^*(P, d_{L_n}) = \mathfrak{T}^*(P, d_{\Delta_n}) = \big(P, ((\gamma))\big) \tag{7}$$

where $\gamma_{ij} = 0$ if $i = j$ and $\gamma_{ij} = 1$ if $i \neq j$, for all $n \in \mathbb{N}$, see Figure 14.

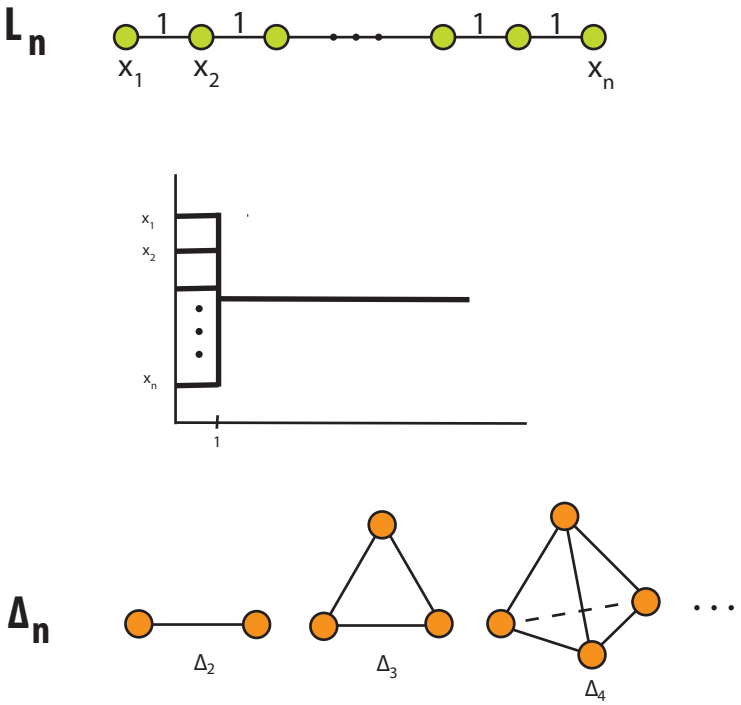


Figure 14: The metric spaces L_n and Δ_n both have n points. Single linkage HC applied to either of them yields the dendrogram in the center.

By appealing to the Euclidean realizations of L_n and Δ_n, one can define *perturbed* versions of these two metric spaces. Indeed, fix $\varepsilon > 0$ and let $\{a_1, \ldots, a_n\} \subset [0, \varepsilon/2]$ and $\{b_1, \ldots, b_n\} \subset S^{n-1}(\varepsilon/2)$. Define L_n^ε to be the metric space with underlying set P and metric $d_{L_n^\varepsilon}(p_i, p_j) = |i-j+$

$a_i - a_j|$. Similarly, define Δ_n^ε to be the metric space with underlying set P and metric $d_{\Delta_n^\varepsilon}(p_i,p_j) = \|s_i - s_j + b_i - b_j\|$.

Notice that by construction,

$$\max_{i,j} \left| d_{L_n}(p_i,p_j) - d_{L_n^\varepsilon}(p_i,p_j) \right| \leqslant \varepsilon \tag{8}$$

and

$$\max_{i,j} \left| d_{\Delta_n}(p_i,p_j) - d_{\Delta_N^\varepsilon}(p_i,p_j) \right| \leqslant \varepsilon. \tag{9}$$

We thus say that the spaces $(P, d_{L_n^\varepsilon})$ and $(P, d_{\Delta_n^\varepsilon})$ are perturbed versions of (P, d_{L_n}) and (P, d_{Δ_n}), respectively.

Remark 16 (About a critique to SL) *Single linkage is generally regarded as a poor choice in practical applications. The reason for this is the so called* chaining *effect observed experimentally, which is central to the criticism to SL made in Lance and Williams (1967) (see also the discussion in Wishart, 1969, pp. 296). The following two observations are important:*

(O1) *It is generally argued that since $(P, d_{L_n^\varepsilon})$ corresponds to points on the vicinity of a line, whereas $(P, d_{\Delta_n^\varepsilon})$ corresponds to points in the close vicinity of a $(n-1)$-simplex, then the cluster formed by points on the latter metric space is more* compact *or* denser *than the one formed by the former, and thus more meaningful.*

(O2) *The outputs of SL to the spaces $(P, d_{L_n^\varepsilon})$ and $(P, d_{\Delta_n^\varepsilon})$ are very similar and this similarity is of order ε.*

Indeed, if we write $\mathfrak{T}^*(P, d_{L_n^\varepsilon}) = (P, u_{L_n^\varepsilon})$ and $\mathfrak{T}^*(P, d_{\Delta_n^\varepsilon}) = (P, u_{\Delta_n^\varepsilon})$, then, by the triangle inequality for the L^∞ norm,

$$\begin{aligned} \|u_{L_n^\varepsilon} - u_{\Delta_n^\varepsilon}\|_{L^\infty(P\times P)} &\leqslant \|u_{L_n^\varepsilon} - u_{L_n^0}\|_{L^\infty(P\times P)} \\ &+ \|u_{L_n^0} - u_{\Delta_n^0}\|_{L^\infty(P\times P)} \\ &+ \|u_{\Delta_n^0} - u_{\Delta_n^\varepsilon}\|_{L^\infty(P\times P)}. \end{aligned} \tag{10}$$

As we pointed out in (7) at the beginning of Section §3.6,

$$u_{L_n^0} = u_{L_n} = ((\gamma)) = u_{\Delta_n} = u_{\Delta_n^0},$$

thus, (10) simplifies into:

$$\begin{aligned} \|u_{L_n^\varepsilon} - u_{\Delta_n^\varepsilon}\|_{L^\infty(P\times P)} &\leqslant \|u_{L_n^\varepsilon} - u_{L_n^0}\|_{L^\infty(P\times P)} \\ &+ \|u_{\Delta_n^\varepsilon} - u_{\Delta_n^0}\|_{L^\infty(P\times P)} \\ &\quad \textit{(and by Lemma 15:)} \\ &\leqslant \|d_{L_n^\varepsilon} - d_{L_n^0}\|_{L^\infty(P\times P)} \\ &+ \|d_{\Delta_n^\varepsilon} - d_{\Delta_n^0}\|_{L^\infty(P\times P)}. \end{aligned} \tag{11}$$

Hence, by (11) and the construction of $d_{L_n^\varepsilon}$ and $d_{\Delta_n^\varepsilon}$ (Equations (8) and (9)), we conclude that

$$\|u_{L_n^\varepsilon} - u_{\Delta_n^\varepsilon}\|_{L^\infty(P\times P)} \leqslant 2\varepsilon.$$

This means that for any small perturbations of L_n and Δ_n, the output of SL to these perturbations are at a small distance from eachother, as we claimed.

*When put together, observations **(O1)** and **(O2)** suggest that SL is unable to pick denser associations of data, such as cliques, over sparser ones, such as linear structures. This feature is undesirable in practical applications where often times one would like to regard clusters as modes of an underlying distribution (Wishart, 1969; Hartigan, 1981).*

It is then the case that in practical applications, CL and especially AL are preferred over SL. These two methods have the property that they indeed somehow favor the association of compact subsets of points. For CL this can be explained easily using the concept of maximal clique *(maximally connected sub-graphs of a given graph) (Jain and Dubes, 1988, Section 3.2.1). Let d_k be the diameter of the cluster created in step k of CL clustering and define a graph $G(k)$ as the graph that links all data points with a distance of at most d_k. Then the clusters after step k are the maximal cliques of $G(k)$. This observation reinforces the perception that CL yields clusters that are dense as measured by the presence of cliques. The sensitivity of AL to density has been discussed by Hartigan in Hartigan (1985, Section 3) and is basically due to the averaging performed in the definition of its linkage function.*

*A more principled way of taking density into account, that does not depend on ad hoc constructions which destroy the stability property, would be to explicitly build the density into the method. In Carlsson and Mémoli (2009) we study **multiparameter clustering methods**, which are similar to HC methods but we* track *connected components in a multiparameter landscape. We also study the classification and stability properties of multiparameter clustering methods.*

Remark 17 (Instability of CL and AL) *It turns out that CL and AL, despite not exhibiting the undesirable feature of the chaining effect, and despite being regarded as more sensitive to density, are **unstable** in a precise sense. Consider for example CL and let $n = 3$. In the construction of (P, d_L^ε) above let $a_1 = a_2 = 0$ and $a_3 = \varepsilon$, then*

$$((d_L)) = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{ccc} p_1 & p_2 & p_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} \end{array} \quad \textit{and} \quad ((d_L^\varepsilon)) = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{ccc} p_1 & p_2 & p_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 2+\varepsilon \\ 1 & 0 & 1+\varepsilon \\ 2+\varepsilon & 1+\varepsilon & 0 \end{pmatrix} \end{array}.$$

Write $\mathfrak{T}^{CL}(P, d_L) = (P, u_L)$ and $\mathfrak{T}^{CL}(P, d_L^\varepsilon) = (P, u_L^\varepsilon)$. Clearly,

$$((u_L)) = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{ccc} p_1 & p_2 & p_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{array} \quad \textit{and} \quad ((u_L^\varepsilon)) = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{ccc} p_1 & p_2 & p_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 2+\varepsilon \\ 1 & 0 & 2+\varepsilon \\ 2+\varepsilon & 2+\varepsilon & 0 \end{pmatrix} \end{array}.$$

Notice that despite $\max_{i,j} |d_L(p_i, p_j) - d_L^\varepsilon(p_i, p_j)| = \varepsilon$, $\max_{i,j} |u_L(p_i, p_j) - u_L^\varepsilon(p_i, p_j)| = 1 + \varepsilon > 1$ for all $\varepsilon > 0$. *We thus conclude that CL is not stable under small perturbations of the metric. Note that in particular, it follows that CL is **not continuous**. The same construction can be adapted for AL. See Figure 15.*

4. A Characterization Theorem for SL Hierarchical Clustering

In this section we obtain a **characterization** of SL hierarchical clustering in terms of some simple axioms. The main axiom, (II) below, says that the clustering scheme has a prescribed behavior

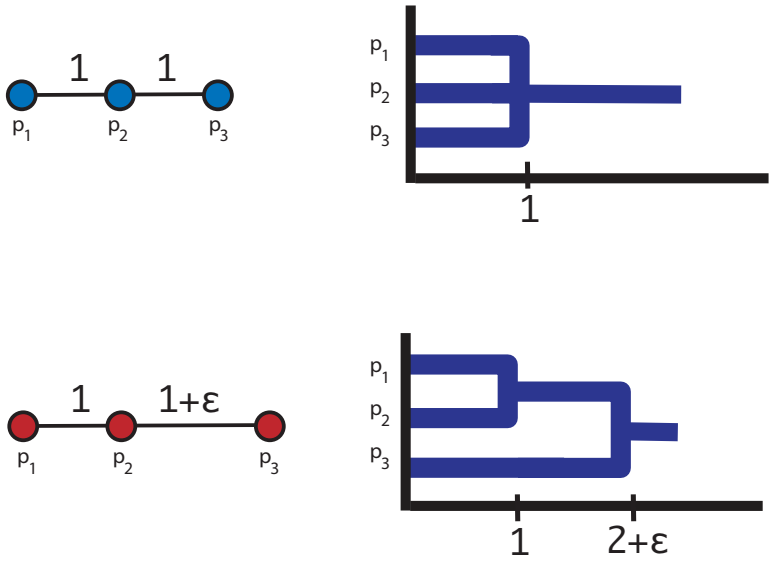


Figure 15: Complete Linkage is not stable to small perturbations in the metric. On the left we show two metric spaces that are metrically very similar. To the right of each of them we show their CL dendrogram outputs. Regardless of $\varepsilon > 0$, the two outputs are always very dissimilar. We make the notion of *similarity* between dendrogram precise in §5 by interpreting dendrograms as ultrametric spaces and by computing the Gromov-Hausdorff distance between these ultrametric spaces.

under distance non-increasing maps of metric space. The behavior is that the map of metric spaces should induce a map of clusters, that is, that if two points in the domain space belong to the same cluster, then so do their images in the clustering of the image metric space. This notion, referred to as **functoriality** in the mathematics literature, appears to us to be a very natural one, and it is closely related to Kleinberg's consistency property (cf. pp. 1425) for ordinary clustering methods; see Remark 19 for an interpretation of our axioms.

Theorem 18 *Let $\mathfrak{T}$ be a hierarchical clustering method s.t.*

(I) $\mathfrak{T}\left(\{p,q\},\left(\begin{smallmatrix}0&\delta\\ \delta&0\end{smallmatrix}\right)\right)=\left(\{p,q\},\left(\begin{smallmatrix}0&\delta\\ \delta&0\end{smallmatrix}\right)\right)$ *for all* $\delta>0$.

(II) *Whenever $X,Y\in\mathcal{X}$ and $\phi:X\to Y$ are such that $d_X(x,x')\geqslant d_Y(\phi(x),\phi(x'))$ for all $x,x'\in X$, then*

$$u_X(x,x')\geqslant u_Y(\phi(x),\phi(x'))$$

also holds for all $x,x'\in X$, where $\mathfrak{T}(X,d_X)=(X,u_X)$ and $\mathfrak{T}(Y,d_Y)=(Y,u_Y)$. prop

(III) *For all $(X,d)\in\mathcal{X}$,*

$$u(x,x')\geqslant sep(X,d)\text{ for all }x\neq x'\in X$$

where $\mathfrak{T}(X,d)=(X,u)$.

Then $\mathfrak{T}=\mathfrak{T}^*$, that is, $\mathfrak{T}$ is exactly single linkage hierarchical clustering.

Remark 19 (Interpretation of the conditions) *Let $(X,d) \in \mathcal{X}$ and write $\mathfrak{T}(X,d) = (X,u)$. The intuition is that $u(x,x')$ measures the* effort *method $\mathfrak{T}$ makes in order to join x to x' into the same cluster.*

Condition (I) is clear, the two-point metric space contains only one degree of freedom which has to determine unambiguously the behavior of any clustering method $\mathfrak{T}$. In terms of dendrograms, this means that the two point metric space $\left(\{A,B\}, \left(\begin{smallmatrix} 0 & \delta \\ \delta & 0 \end{smallmatrix}\right)\right)$ must be mapped to the dendrogram where A and B are merged at parameter value δ, see Figure 16.

Condition (II) is crucial and roughly says that whenever one shrinks some distances (even to zero) to obtain a new (pseudo) metric space, then the corresponding efforts in this new space have to be smaller than the efforts in the original metric space. This is consistent with the notion that reducing the distance between two points (without increasing all other distances) makes them more likely to belong to the same cluster.

Let $\theta_X = \Psi^{-1}(u_X)$ and $\theta_Y = \Psi^{-1}(u_Y)$ be the dendrograms associated to u_X and u_Y. In terms of dendrograms, this means that if two points $x,x' \in X$ are in the same block of $\theta_X(t)$ for some $t > 0$, then $\phi(x)$ and $\phi(x')$ must be *in the same block of $\theta_Y(t)$. see Figure 17.*

Condition (III) expresses the fact that in order to join two points $x,x' \in X$, any clustering method $\mathfrak{T}$ has to make an effort of at least the separation $sep(X,d)$ of the metric space. In terms of dendrograms, this means that $\theta_X(t)$ has to equal the partition of X into singletons for all $0 \leqslant t < sep(X,d)$. See Figure 18.

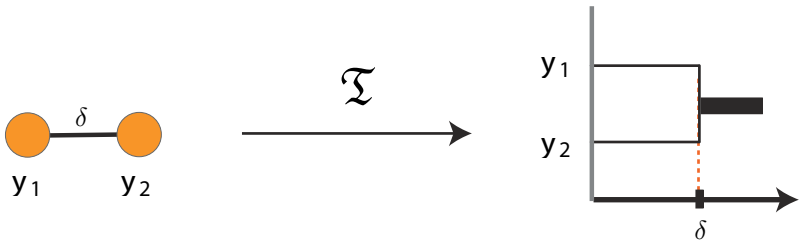


Figure 16: **Interpretation of Condition I**: For all $\delta > 0$ the two point metric space on the left must be mapped by $\mathfrak{T}$ into the dendrogram on the right.

Remark 20 *It is interesting to point out why complete linkage and average linkage hierarchical clustering, as defined in §3.2.2, fail to satisfy the conditions in Theorem 18. It is easy to see that conditions (I) and (III) are always satisfied by CL and AL.*

Consider the metric spaces $X = \{A,B,C\}$ with metric given by the edge lengths $\{4,3,5\}$ and $Y = (A',B',C')$ with metric given by the edge lengths $\{4,3,2\}$, as given in Figure 19. Obviously, the map ϕ from X to Y with $\phi(A) = A'$, $\phi(B) = B'$ and $\phi(C) = C'$ is s.t.

$$d_Y(\phi(x),\phi(x')) \leqslant d_X(x,x') \quad \text{for all } x,x' \in \{A,B,C\}.$$

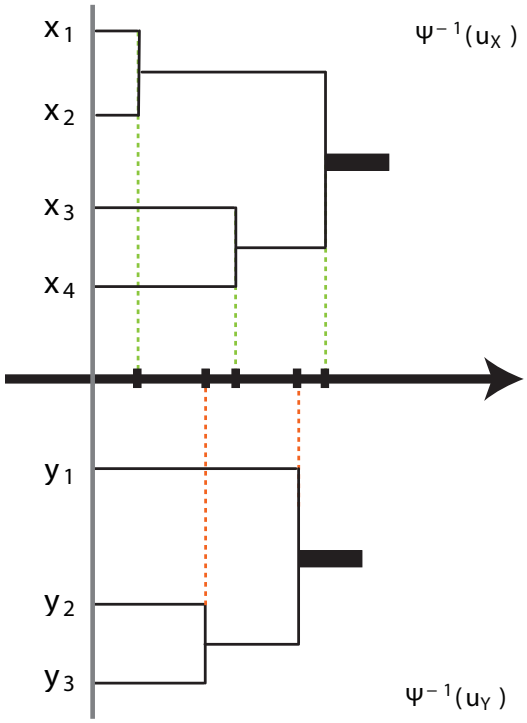


Figure 17: **Interpretation of Condition II**: Assume that $\phi : X \to Y$ is a distance non-increasing map such that $\phi(x_1) = \phi(x_2) = y_1$, $\phi(x_3) = y_2$ and $\phi(x_4) = y_3$. Then, Condition (II) requires that if $x, x' \in X$ are merged into the same cluster of $\Psi^{-1}(u_X)$ at parameter value t, then $\phi(x)$ and $\phi(x')$ must merge into the same cluster of $\Psi^{-1}(u_Y)$ for some parameter value $\leqslant t$. In the Figure, this translates into the condition that vertical dotted lines corresponding to mergings of pairs of points in X should happen at parameter values greater than or equal than the parameter values for which correponding points in Y (via ϕ) are merged into the same cluster. For example, $\phi(x_1), \phi(x_2)$ merge into the same cluster at parameter value 0. The condition is clearly verified for this pair since by definition of ϕ, $\phi(x_1) = \phi(x_2) = y_1$. Take now x_3 and x_4: clearly the vertical line that shows the parameter value for which they merge is to the right of the vertical line showing the parameter value for which $y_2 = \phi(x_3)$ and $y_3 = \phi(x_4)$ merge.

It is easy to check that

$$((u_X)) = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & 5 & 3 \\ 5 & 0 & 5 \\ 3 & 5 & 0 \end{pmatrix} \end{array} \quad \textit{and} \quad ((u_Y)) = \begin{array}{c} \\ A' \\ B' \\ C' \end{array} \begin{array}{c} \begin{array}{ccc} A' & B' & C' \end{array} \\ \begin{pmatrix} 0 & 2 & 4 \\ 2 & 0 & 4 \\ 4 & 4 & 0 \end{pmatrix} \end{array}.$$

Note that for example $3 = u_X(A,C) < u_Y(\phi(A),\phi(C)) = u_Y(A',C') = 4$ *thus violating property (II). The same construction yields a counter-example for average linkage.*

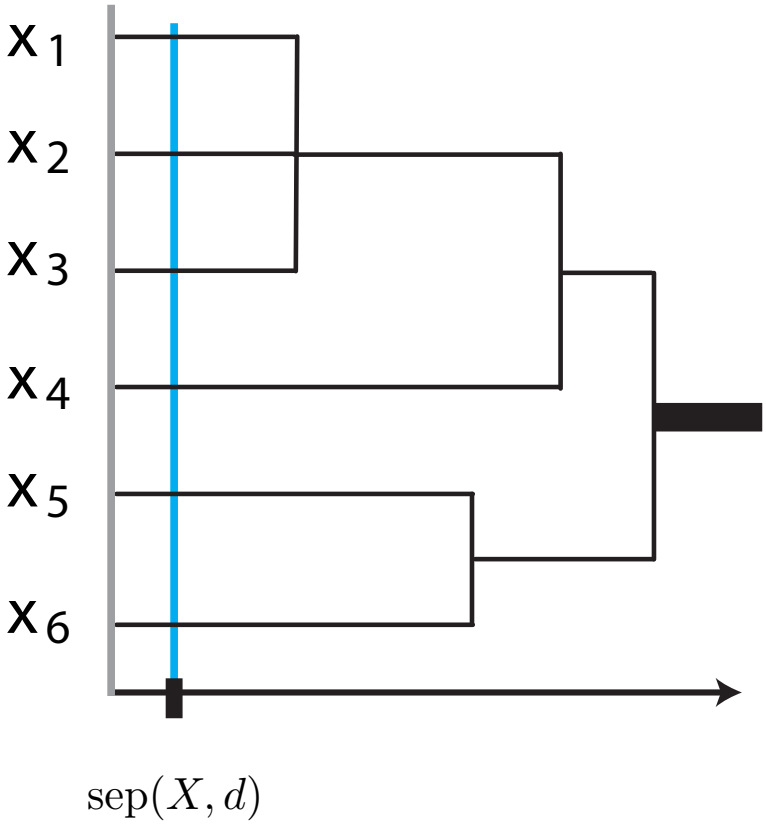


Figure 18: **Interpretation of Condition III**: The vertical line at parameter value $t = \mathrm{sep}(X,d)$ must intersect the horizontal lines of the dendrogram before any two points are merged.

5. Metric Stability and Convergence of $\mathfrak{T}^*$

The Proposition and Theorem below assert the metric stability and consistency/convergence of the method $\mathfrak{T}^*$ (i.e., of SLHC, by virtue of Proposition 14. We use the notion of Gromov-Hausdorff distance between metric spaces (Burago et al., 2001). This notion of distance permits regarding the collection of all compact metric spaces as a metric space in itself.

This seemingly abstract construction is in fact very useful. Finite metric spaces are by now ubiquitous in virtually all areas of data analysis, and the idea of assigning a metric to the collection of all of them is in fact quite an old one. For Euclidean metric spaces, for example, the idea of constructing a metric was used by Kendall et al. (1999) and Bookstein et al. (1985) in constructing a *statistical shape theory*, motivated by the ideas about form of biological organisms developed by D'Arcy Thompson.

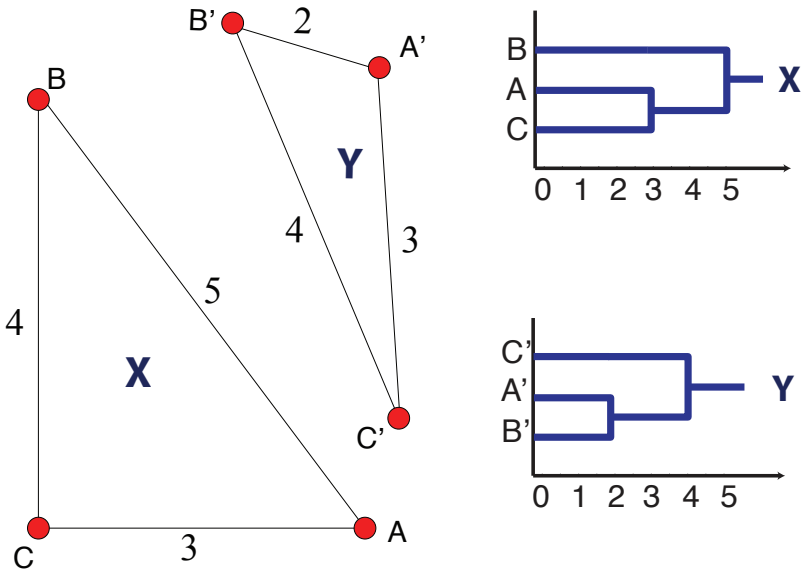


Figure 19: An example that shows why complete linkage fails to satisfy condition (2) of Theorem 18.

5.1 The Gromov-Hausdorff Distance and Examples

Definition 21 *Let (Z,d_Z) be a compact metric space. The Hausdorff distance between any two compact subsets A,B of Z is defined by*

$$d_{\mathcal{H}}^Z(A,B) := \max\left(\max_{a\in A}\min_{b\in B} d_Z(a,b), \max_{b\in B}\min_{a\in A} d_Z(a,b)\right).$$

Remark 22 *Let $\mathbb{Z} = \{z_1,\ldots,z_n\} \subset Z$. Then, $d_{\mathcal{H}}^Z(\mathbb{Z},Z) \leqslant \delta$ for some $\delta \geqslant 0$ if and only if $Z \subset \bigcup_{i=1}^n \overline{B(z_i,\delta)}$. In other words, $d_{\mathcal{H}}^Z(\mathbb{Z},Z)$ describes the minimal δ s.t. $\mathbb{Z}$ is a δ-net for Z and therefore measures how well $\mathbb{Z}$ covers Z.*

The Gromov-Hausdorff distance $d_{\mathcal{GH}}(X,Y)$ between compact metric spaces (X,d_X) and (Y,d_Y) was orignally defined to be the infimal $\varepsilon > 0$ s.t. there exists a metric d on $X \bigsqcup Y$ with $d_{|X\times X} = d_X$ and $d_{|Y\times Y} = d_Y$ for which the Hausdorff distance between X and Y (as subsets of $(X\bigsqcup Y,d)$) is less than ε (Gromov, 1987). There is, however, an alternative expression for the GH distance that is better suited for our purposes which we now recall.

Definition 23 (Correspondence) *For sets A and B, a subset $R \subset A\times B$ is a* correspondence *(between A and B) if and and only if*

- $\forall\, a\in A$, *there exists* $b\in B$ *s.t.* $(a,b)\in R$
- $\forall\, b\in B$, *there exists* $a\in X$ *s.t.* $(a,b)\in R$

Let $\mathcal{R}(A,B)$ denote the set of all possible correspondences between A and B.

We now give several examples to illustrate this definition.

Example 8 *Let $A=\{a_1,a_2\}$ and $B=\{b_1,b_2,b_3\}$. In this case, $R_1=\{(a_1,b_1),(a_2,b_2),(a_1,b_3)\}$ is a correspondence but $R_2=\{(a_1,b_1),(a_2,b_2)\}$ is not.*

Example 9 *Let A and B be finite s.t. $\#A=\#B=n$. In this case, if π is any permutation matrix of size n, then $\{(a_i,b_{\pi_i}),\, i=1,\ldots,n\}\in\mathcal{R}(A,B)$.*

Example 10 *Let $\phi:X\to Y$ and $\psi:Y\to X$ be given maps. Then, one can construct a correspondence out of these maps, call it $R(\phi,\psi)$ given by*

$$\big\{(x,\phi(x)),\, x\in X\big\}\bigcup\big\{(\psi(y),y),\; y\in Y\big\}.$$

For metric spaces (X,d_X) and (Y,d_Y). Let $\Gamma_{X,Y}:X\times Y\times X\times Y\to\mathbb{R}^+$ be given by

$$(x,y,x',y')\mapsto\big|d_X(x,x')-d_Y(y,y')\big|.$$

Then, by (Burago et al., 2001, Theorem 7.3.25) the **Gromov-Hausdorff distance** between X and Y is equal to

$$d_{\mathcal{GH}}\big(X,Y\big):=\frac{1}{2}\inf_{R\in\mathcal{R}(X,Y)}\sup_{(x,y),(x',y')\in R}\Gamma_{X,Y}(x,y,x',y'). \tag{12}$$

It can be seen (it is an easy computation) that in (12) one can restrict the infimum to those correspondences that arise from maps ϕ and ψ such as those constructed in Example 10. Then, one recovers expression (6) which we gave in §3.5, namely, that actually

$$d_{\mathcal{GH}}\big(X,Y\big):=\frac{1}{2}\inf_{\phi,\psi}\max\big(\mathrm{dis}(\phi),\mathrm{dis}(\psi),\mathrm{dis}(\phi,\psi)\big). \tag{13}$$

Remark 24 *Expression (13) defines a **distance** on the set of (isometry classes of) finite metric spaces (Burago et al., 2001, Theorem 7.3.30). From now on let $\mathcal{G}$ denote the collection of all (isometry classes of) compact metric spaces. We say that $\{(X_n,d_{X_n})\}_{n\in\mathbb{N}}\subset\mathcal{G}$ Gromov-Hausdorff converges to $X\in\mathcal{G}$ if and only if $d_{\mathcal{GH}}\big(X_n,X\big)\to 0$ as $n\uparrow\infty$.*

Example 11 *Fix $(X,d_X)\in\mathcal{G}$. Consider the sequence $\{(X,\frac{1}{n}\cdot d_X)\}_{n\in\mathbb{N}}\subset\mathcal{G}$. Then, X_n Gromov-Hausdorff converges to the metric space consisting of a single point.*

Remark 25 (Gromov-Hausdorff distance and Hausdorff distance) *Let (X,d_X) be a compact metric space. Then, if $X'\subset X$ is compact and we endow X' with the metric $d_{X'}$ equal to the restriction of d_X, then*

$$d_{\mathcal{GH}}\big((X,d_X),(X',d_{X'})\big)\leqslant d_{\mathcal{H}}^X\big(X',X\big).$$

This is easy to see by defining the correspondence R between X and X' given by

$$R=\{(x',x'),x'\in X'\}\cup\{(x,x'),x\in V(x'),x'\in X'\},$$

where $V(x'):=\{x\in X,\, d_X(x,x')\leqslant d_X(x,z),\, z\in X'\backslash\{x'\}\}$. Indeed, since then, for all $(x_1,x_1'),(x_2,x_2')\in R$,

$$\frac{1}{2}\big|d_X(x_1,x_2)-d_X(x_1',x_2')\big|\leqslant\frac{1}{2}\left(d_X(x_1,x_1')+d_X(x_2,x_2')\right)\leqslant\max_{x\in X}\min_{x'\in X'}d_X(x,x')=d_{\mathcal{H}}^X\big(X,X'\big).$$

Example 12 *Consider a finite set M and $d, d' : M \times M \to \mathbb{R}^+$ two metrics on M. Then, the GH distance between (M,d) and (M,d') is bounded above by the L^∞ norm of the difference between d and d':*

$$d_{\mathcal{GH}}\big((M,d),(M,d')\big) \leqslant \frac{1}{2}\|d - d'\|_{L^\infty(M\times M)}.$$

To prove this it is enough to consider the correspondence $R \in \mathcal{R}(M,M)$ given by $R = \{(m,m), m \in M\}$.

Notice that as an application, for the metric spaces $(P, d_{L_n^\varepsilon})$ and $(P, d_{\Delta_n^\varepsilon})$ discussed in §3.6, one has that

$$d_{\mathcal{GH}}\big((P,d_{L_n}),(P,d_{L_n^\varepsilon})\big) \leqslant \frac{\varepsilon}{2} \text{ and}$$

$$d_{\mathcal{GH}}\big((P,d_{\Delta_n}),(P,d_{\Delta_n^\varepsilon})\big) \leqslant \frac{\varepsilon}{2}.$$

5.2 Stability and Convergence Results

Our first result states that SL HC is stable in the Gromov-Hausdorff sense and it is a generalization of Lemma 15.

Proposition 26 *For any two finite metric spaces (X,d_X) and (Y,d_Y)*

$$d_{\mathcal{GH}}\big((X,d_X),(Y,d_Y)\big) \geqslant d_{\mathcal{GH}}\big(\mathfrak{T}^*(X,d_X),\mathfrak{T}^*(Y,d_Y)\big).$$

Remark 27 *This Proposition generalizes Lemma 15. Notice for example that in case X and Y are finite, they need not have the same number of points. This feature is important in order to be able to make sense of situations such as the one depicted in Figure 2 in pp. 1428, where one is trying to capture the connectivity (i.e., clustering) properties of an underlying 'continuous' space by taking finitely (but increasingly) many samples from this space and applying some form of HC to this finite set. Theorem 28 below deals with exactly this situation. See Figure 20.*

Let (Z,d_Z) be a compact metric space. Given a finite index set A and a (finite) collection of disjoint compact subsets of Z, $\{U^{(\alpha)}\}_{\alpha\in A}$, let $W_A : A \times A \to \mathbb{R}^+$ be given by

$$(\alpha,\alpha') \mapsto \min_{\substack{z\in U^{(\alpha)}\\ z'\in U^{(\alpha')}}} d_Z(z,z').$$

A metric space (A,d_A) arises from this construction, where $d_A = \mathcal{L}(W_A)$. We say that (A,d_A) is the metric space with underlying set A **arising** from $\{U^{(\alpha)}\}_{\alpha\in A}$. Notice that $\mathrm{sep}(A,d_A)$ equals the minimal separation between any two sets $U^{(\alpha)}$ and $U^{(\alpha')}$ ($\alpha \neq \alpha'$). More precisely,

$$\mathrm{sep}(A,d_A) = \min_{\substack{\alpha,\alpha'\in A,\\ \alpha\neq\alpha'}} \min_{\substack{z\in U^{(\alpha)}\\ z'\in U^{(\alpha')}}} d_Z(z,z').$$

We now state a metric stability and convergence result, see Figure 20. The proof of this result is deferred to §B.

Theorem 28 *Assume (Z,d_Z) is a compact metric space. Let X and X' be any two finite subsets of Z and let $d_X = d_{Z|_{X\times X}}$ and $d_{X'} = d_{Z|_{X'\times X'}}$. Write $\mathfrak{T}^*(X,d_X) = (X,u_X)$ and $\mathfrak{T}^*(X',d_{X'}) = (X',u_{X'})$. Then,*

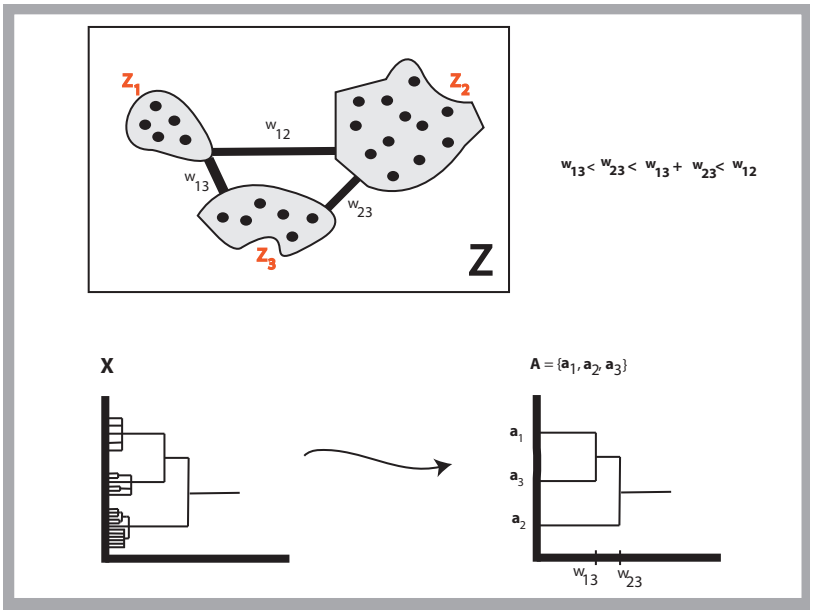


Figure 20: Illustration of Theorem 28. *Top*: A space Z composed of 3 disjoint path connected parts, $Z^{(1)}, Z^{(2)}$ and $Z^{(3)}$. The black dots are the points in the finite sample X. In the figure, $w_{ij} = W(i,j)$, $1 \leqslant i \neq j \leqslant 3$. *Bottom Left*: The dendrogram representation of (X, u_X). *Bottom Right* The dendrogram representation of (Z, u_Z). Note that $d_Z(z_1, z_2) = w_{13} + w_{23}$, $d_Z(z_1, z_3) = w_{13}$ and $d_Z(z_2, z_3) = w_{23}$. As $r \to 0$, $(X, u_X) \to (Z, u_Z)$ in the Gromov-Hausdorff sense, see text for details.

1. *(Finite Stability)* $d_{\mathcal{GH}}\big((X,u_X),(X',u_{X'})\big) \leqslant d_{\mathcal{H}}^Z\big(X,Z\big) + d_{\mathcal{H}}^Z\big(X',Z\big)$.

2. *(Approximation bound) Assume in addition that* $Z = \bigsqcup_{\alpha\in A} Z^{(\alpha)}$ *where* A *is a finite index set and* $Z^{(\alpha)}$ *are compact, disjoint and path-connected sets. Let* (A, d_A) *be the finite metric space with underlying set* A *arising from* $\{Z^{(\alpha)}\}_{\alpha\in A}$. *Let* $\mathfrak{T}^*(A,d_A) = (A,u_A)$. *Then, if* $d_{\mathcal{H}}^Z\big(X,Z\big) < sep(A,d_A)/2$,
$$d_{\mathcal{GH}}\big((X,u_X),(A,u_A)\big) \leqslant d_{\mathcal{H}}^Z\big(X,Z\big).$$

3. *(Convergence) Under the hypotheses of (2), let* $\{X_n\}_{n\in\mathbb{N}}$ *be a sequence of finite subsets of* Z *s.t.* $d_{\mathcal{H}}^Z\big(X_n,Z\big) \to 0$ *as* $n \to \infty$, *and* d_{X_n} *be the metric on* X_n *given by the restriction of* d_Z *to* $X_n \times X_n$. *Then, one has that*
$$d_{\mathcal{GH}}\big(\mathfrak{T}^*(X_n,d_{X_n}),(A,u_A)\big) \to 0 \text{ as } n \to \infty.$$

Remark 29 (Interpretation of the statement) *Assertion (1) guarantees that if* X, X' *are both dense samples of* Z, *then the result of applying* $\mathfrak{T}^*$ *to both sets are very close in the Gromov-Hausdorff sense.*

Assertions (2) and (3) identify the limiting behavior of the construction $\mathfrak{T}^*(X_n, d_{X_n})$ *as* X_n *becomes denser and denser in* X, *see Figure 20.*

5.3 A Probabilistic Convergence Result

In this section, we prove a precise result which describes how the dendrograms attached to compact metric spaces by single linkage clustering can be obtained as the limits of the dendrograms attached to finite subsets of the metric space. The result is by necessity probabilistic in nature. This kind of result is of great importance, since we are often interested in infinite metric spaces but typically do not have access to more than finitely many random samples from the metric space.

Theorem 30 and Corollary 32 below proves that for random i.i.d. observations $\mathbb{X}_n = \{x_1, \ldots, x_n\}$ with probability distribution μ compactly supported in a metric space (X, d), the result $(\mathbb{X}_n, u_{\mathbb{X}_n})$ of applying single linkage clustering to $(\mathbb{X}_n, d)$ converges almost surely in the Gromov-Hausdorff sense to an ultrametric space that recovers the multiscale structure of the *support* of μ, see Figure 20. This is a refinement of a previous observation of Hartigan (1985) that SLHC is insensitive to the distribution of mass of μ in its support.

The proof of this theorem relies on Theorem 34, a probabilistic covering theorem of independent interest. In order to state and prove our theorems we make use of the formalism of **metric measure spaces**.

A triple (X, d_X, μ_X), where (X, d_X) is a metric space and μ_X is a Borel probability measure on X with compact support will be called an **mm-space** (short for measure metric space). The support $\mathrm{supp}[\mu_X]$ of a measure μ_X on X is the minimal closed set A (w.r.t. inclusion) s.t. $\mu_X(X \backslash A) = 0$. Measure metric spaces are considered in the work of Gromov and are useful in different contexts, see (Gromov, 2007, Chapter $3\frac{1}{2}$). For a mm-space X let $f_X : \mathbb{R}^+ \to \mathbb{R}^+$ be defined by

$$r \mapsto \min_{x \in \mathrm{supp}[X]} \mu_X(B_X(x, r)).$$

Note also that by construction $f_X(\cdot)$ in non-decreasing and $f_X(r) > 0$ for all $r > 0$. Let also $F_X : \mathbb{N} \times \mathbb{R}^+ \to \mathbb{R}^+$ be defined by $(n, \delta) \mapsto \frac{e^{-n f_X(\delta/4)}}{f_X(\delta/4)}$. Note that for fixed $\delta_0 > 0$, (1) $F_X(\cdot, \delta_0)$ is decreasing in its argument, and (2) $\sum_{n \in \mathbb{N}} F_X(n, \delta_0) < \infty$.

Theorem 30 *Let (Z, d_Z, μ_Z) be a mm-space and write $\mathrm{supp}[\mu_Z] = \bigcup_{\alpha \in A} U^{(\alpha)}$ for a finite index set A and $\mathbf{U} = \{U^{(\alpha)}\}_{\alpha \in A}$ a collection of disjoint, compact, path-connected subsets of Z. Let (A, d_A) be the metric space arising from $\mathbf{U}$ and let $\delta_A := sep(A, d_A)/2$.*

For each $n \in \mathbb{N}$, let $\mathbb{Z}_n = \{z_1, z_2, \ldots, z_n\}$ be a collection of n independent random variables (defined on some probability space Ω with values in Z) with distribution μ_Z, and let $d_{\mathbb{Z}_n}$ be the restriction of d_Z to $\mathbb{Z}_n \times \mathbb{Z}_n$. Then, for $\zeta \geqslant 0$ and $n \in \mathbb{N}$,

$$\mathbf{P}_{\mu_Z}\left(d_{\mathcal{GH}}\big(\mathfrak{T}^*(\mathbb{Z}_n, d_{\mathbb{Z}_n}), \mathfrak{T}^*(A, d_A)\big) > \zeta \right) \leqslant F_Z\big(n, \min(\zeta, \delta_A/2)\big).$$

Corollary 31 *Under the hypotheses of Theorem 30, for any pre-specified probability level $p \in (0,1)$ and tolerance $\zeta \geqslant 0$, if*

$$n \geqslant \frac{\ln \frac{1}{1-p} - \ln f_X(\delta/4)}{f_X(\delta/4)},$$

then $\mathbf{P}_{\mu_Z}\left(d_{\mathcal{GH}}\big(\mathfrak{T}^*(\mathbb{Z}_n, d_{\mathbb{Z}_n}), \mathfrak{T}^*(A, d_A)\big) \leqslant \zeta \right) \geqslant p$, where $\delta := \min(\zeta, \delta_A/2)$.

Corollary 32 *Under the hypotheses of Theorem 30,* $\mathfrak{T}^*(\mathbb{Z}_n, d_{\mathbb{Z}_n}) \xrightarrow{n} \mathfrak{T}^*(A, d_A)$ *in the Gromov-Hausdorff sense* μ_Z*-almost surely.*

Proof [Proof of Corollary 32] The proof follows immediately from the expression for F_X and the Borel-Cantelli Lemma. ■

Remark 33 *Note that the convergence theorem above implies that in the limit,* $\mathfrak{T}^*(X_n, d_{X_n})$ *only retains information about the support of the probability measure but not about the way the mass is distributed inside the support, compare to Hartigan (1985).*

Example 13 ($Z \subset \mathbb{R}^d$**)** *Let* $\rho : \mathbb{R}^d \to \mathbb{R}^+$ *be a density function with compact, support* Z *and* μ *be its associated probability measure. Then* $(\mathbb{R}^d, \|\cdot\|, \mu)$ *satisfies the assumptions in the theorem. If one makes additional smoothness assumptions on* ρ*, in this particular case one can relate* $F_Z(n, \zeta)$ *to geometrical properties of the boundary of supp* $[\rho]$*.*

Example 14 (Z **is a Riemannian manifold)** *In more generality,* Z *could be a Riemannian manifold and* μ *a probability measure absolutely continuous w.r.t. to the Riemannian area measure on* Z*.*

6. Discussion

We have obtained novel characterization, stability and convergence theorems for SL HC. Our theorems contemplate both the deterministic and the stochastic case. Our characterization theorem can be interpreted as a relaxation of Kleinberg's impossibility result for standard clustering methods in that by allowing the output of clustering methods to be hierarchical, one obtains existence and uniqueness.

Our stability results seem to be novel and complement classical observations that CL and AL are discontinuous as maps from finite metric spaces into dendrograms.

Our convergence results also seem to be novel and they refine a previous observation by Hartigan about the information retained about an underlying density by SL clustering of an i.i.d. collection of samples from that density. Our setting for the stochastic convergence results is quite general in that we do not assume the underlying space to be a smooth manifold and we do not assume the underlying probability measure to have a density with respect to any reference measure.

We understand that SL HC is not sensitive to variations in the density (see also Hartigan, 1981). In our future work we will be looking at ways of further relaxing the notions of clustering that can cope with the problem of detecting "dense" clusters, in the same spirit as Wishart (1969); Stuetzle (2003). A follow up paper (Carlsson and Mémoli, 2009) presents a systematic treatment of this with a more general framework.

Some recent works have also addressed the characterization of clustering schemes in the hierarchical case. The authors of the present paper reported a characterization for proximity dendrograms (Carlsson and Mémoli, 2008) using the language of category theory. Zadeh and Ben-David (2009) gave a characterization for threshold dendrograms.[7] More classical is the work of Jardine and Sibson (1971) who also ultimately view HC methods as maps form finite metric spaces to finite ultrametric spaces.

7. Recall that the difference between these two types of dendrograms is that proximity dendrograms retain the linkage value at which mergings take place whereas threshold dendrograms only record the order, see Remark 3.

It is interesting to consider the situation when one requires the map ϕ in our characterization theorem (Theorem 18) to be 1 to 1 on points. In this case, a much wider class of hierarchical schmemes becomes possible including for example a certain version of *clique clustering*. The restriction on the nature of ϕ would be called restriction of *functoriality* by a mathematician. The classification question of clustering methods that arises becomes mathematically interesting and we are currently exploring it (Carlsson and Mémoli, Stanford, 2009; Carlsson and Mémoli, 2008).

Acknowledgments

We would like to acknowledge support for this project from DARPA grant HR0011-05-1-0007 and ONR grant N00014-09-1-0783.

Appendix A. Notation

Symbol	Meaning
$\mathbb{R}$	Real numbers.
$\mathbb{R}^d$	d-dimensional Euclidean space.
$\mathbb{N}$	Natural numbers.
$((a))$	A square symmetric matrix with elements a_{ij} which are usually distances.
(X,d)	Metric space X with metric d, page 1429.
(X,u)	Ultrametric space X with ultrametric u, page 1429.
$\mathcal{X}$, $\mathcal{X}_n$	Collection of all finite (resp. n point) metric spaces, page 1429.
$\mathcal{U}$, $\mathcal{U}_n$	Collection of all finite (resp. n point) ultrametric spaces, page 1429.
$\mathcal{C}(X)$	Collection of all non-empty subsets of the set X, page 1429.
$\mathcal{U}(X)$	Collection of all ultrametrics over the finite set X, page 1429.
$\mathcal{P}(X)$	Collection of all partitions of the finite set X, page 1429.
$\Pi, \mathcal{B}, \mathcal{A}$	A partition of a finite set and blocks of that partition, respectively, page 1429.
$\sim, [a], A\backslash\sim$	An equivalence relation, the equivalence class of a point and the quotient space, page 1429.
$\sim_r$	An equivalence relation with a parameter $r \geqslant 0$, page 1429.
$S^{k-1}(r)$	Sphere of radius r and dimension $k-1$ embedded in $\mathbb{R}^k$, page 1429.
$\mathcal{L}(W)$	Maximal metric $\leqslant W$, page 1429.
$\theta : [0,\infty) \to \mathcal{P}(X)$	A dendrogram over the finite set X, 1431.
$\mathcal{D}(X)$	Collection of all dendrograms over the finite set X, page 1431.
θ^*	Dendrogram over the finite set X arising from $\sim_r$, 1433.
$\ell^{\mathrm{SL}}, \ell^{\mathrm{CL}}, \ell^{\mathrm{AL}}$	Linkage functions, page 1434.
$\theta^{\mathrm{SL}}, \theta^{\mathrm{AL}}, \theta^{\mathrm{CL}}$	Dendrograms arising from linkage functions, 1434.
$\mathfrak{T}$	A hierarchical clustering method seen as a map $\mathfrak{T} : \mathcal{X} \to \mathcal{U}$, page 1442.
$\mathfrak{T}^*$	A HC method arising from the maximal sub-dominant ultrametric, page 1442.
u_θ	An ultrametric obtained from the dendrogram θ, page 1440.
θ^u	A dendrogram obtained from the ultrametric u, page 1441.
Ψ	A bijective map between $\mathcal{D}(X)$ and $\mathcal{U}(X)$, page 1439.
Δ_n	Metric space isometric to an n point unit simplex, page 1447.

L_n	Metric space isometric to n points on a line, page 1447.
$d_{\mathcal{H}}^Z$	Hausdorff distance between subsets of the metric space Z, page 1455.
$\mathfrak{T}^{\mathrm{SL}}, \mathfrak{T}^{\mathrm{CL}}, \mathfrak{T}^{\mathrm{CL}}$	Standard linkage based HC methods seen as maps from $\mathcal{X}$ to $\mathcal{U}$, page 1442.
$\mathrm{dis}(f), \mathrm{dis}(f,g)$	Distortion of a map f and joint distortion of a pair of maps f and g, page 1445.
$d_{\mathcal{GH}}$	Gromov-Hausdorff distance between metric spaces, pages 1445, 1456.
$\mathrm{sep}(X)$	Separation of the metric space X, page 1429.
$\mathrm{diam}\,(X)$	Diameter of the metric space X, page 1429.
P_n	All the $n!$ permutations of elements of the set $\{1,\ldots,n\}$.
$\Gamma_{X,Y}$	A function used to measure metric distortion, page 1456.
(X,d,μ)	An mm-space, (X,d) a compact metric space, μ a Borel probability measure, page 1459.
$\mathrm{supp}\,[\mu]$	Support of the probability measure μ, page 1459.
$\mathbf{P}_\mu$	Probability with respect to the law μ.

Appendix B. Proofs

Proof [Proof of Proposition 8] The claim follows from the following claim, which we prove by induction on i:

Claim: For all $i \geqslant 2, x, x' \in X$ are s.t. there exists $\mathcal{B} \in \Theta_i$ with $x, x' \in \mathcal{B}$ if and only if $x \sim_{R_{i-1}} x'$.

Proof [Proof of the Claim] For $i = 2$ the claim is clearly true. Fix $i > 2$.

Assume that $x, x' \in X$ and $\mathcal{B} \in \Theta_{i+1}$ are such that $x, x' \in \mathcal{B}$. If x, x' belong to the same block of Θ_i there is nothing to prove. So, assume that $x \in \mathcal{A}$ and $x' \in \mathcal{A}'$ with $\mathcal{A} \neq \mathcal{A}'$ and $\mathcal{A}, \mathcal{A}' \in \Theta_i$. Then, it must be that there exist blocks $\mathcal{A} = \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_s = \mathcal{A}'$ of Θ_i s.t. $\ell^{\mathrm{SL}}(\mathcal{A}_t, \mathcal{A}_{t+1}) \leqslant R_i$ for $t = 1, \ldots, s-1$. Pick $x_1, y_1 \in \mathcal{A}_1, x_2, y_2 \in \mathcal{A}_2, \ldots, x_s, y_s \in \mathcal{A}_s$ s.t. $x_1 = x$ and $y_s = x'$ and $d(y_t, x_{t+1}) = \ell^{\mathrm{SL}}(\mathcal{A}_t, \mathcal{A}_{t+1}) \leqslant R_i$ for $t = 1, \ldots, s-1$, see the Figure 21.

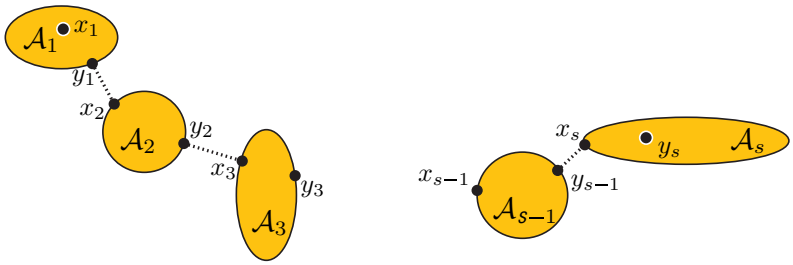


Figure 21: Construction used in the proof of Proposition 8.

Notice that by the inductive hypothesis we have $x_t \sim_{R_{i-1}} y_t$ for $t = 1, \ldots, s$. It follows that $x \sim_r x'$ for $r = \max(R_i, R_{i-1})$. By Proposition 5, $r = R_i$ and hence $x \sim_{R_i} x'$.

Assume now that $x \sim_{R_i} x'$. If x, x' belong to the same block of Θ_i there's nothing to prove since Θ_{i+1} is coarser than Θ_i and hence x, x' will also belong to the same block of Θ_{i+1}. Assume then that $x \in \mathcal{B}$ and $x' \in \mathcal{B}'$ for $\mathcal{B}, \mathcal{B}' \in \Theta_i$ with $\mathcal{B} \neq \mathcal{B}'$. Let $x = x_1, x_2, \ldots, x_s = x'$ be points in X with $d(x_t, x_{t+1}) \leqslant R_i$ for $t = 1, \ldots, s-1$. Also, for $t = 1, \ldots, s-1$ let $\mathcal{B}_t$ be the block of Θ_i to which x_t

belongs. But then, by construction

$$R_i \geqslant d(x_t, x_{t+1}) \geqslant \min_{z\in\mathcal{B}_t, z'\in\mathcal{B}_{t+1}} d(z,z') = \ell^{\mathrm{SL}}(\mathcal{B}_t, \mathcal{B}_{t+1}) \quad \text{for } t = 1,\ldots,s-1,$$

and hence $\mathcal{B}_1 \sim_{\ell^{\mathrm{SL}},R_i} \mathcal{B}_s$. In particular, $\mathcal{B}_1 \cup \mathcal{B}_s \subset \mathcal{A}$ for some $\mathcal{A} \in \Theta_{i+1}$ and thus x, x' belong to the same block in Θ_{i+1}. ■

■

Proof [Proof of Lemma 10] Obviously u_θ is non-negative. Pick $x, x', x'' \in X$ and let $r_1, r_2 \geqslant 0$ be s.t. x, x' belong to the same block of $\theta(r_1)$ and x', x'' belong to the same block of $\theta(r_2)$. These numbers clearly exist by condition (2) in the definition of dendrograms. Then, there exist a block $\mathcal{B}$ of $\theta\big(\max(r_1, r_2)\big)$ s.t. $x, x'' \in \mathcal{B}$ and hence $u_\theta(x, x'') \leqslant max(r_1, r_2)$. The conclusion follows since $r_1 \geqslant u_\theta(x, x')$ and $r_2 \geqslant u_\theta(x', x'')$ are arbitrary.

Now, let $x, x' \in X$ be such that $u_\theta(x, x') = 0$. Then x, x' are in the same block of $\theta(0)$. Condition (1) in the definition of dendrograms implies that $x = x'$. ■

Proof [Proof of Lemma 12] Pick $x, x' \in X$ and let $r := u_{\theta*}(x, x')$. Then, according to (3), there exist $x_0, x_1, \ldots, x_t \in X$ with $x_0 = x$, $x_t = x'$ and $\max_i d(x_i, x_{i+1}) \leqslant r$. From (4) we conclude that then $u^*(x, x') \leqslant r$ as well. Assume now that $u^*(x, x') \leqslant r$ and let $x_0, x_1, \ldots, x_t \in X$ be s.t. $x_0 = x$, $x_t = x'$ and $\max_i d(x_i, x_{i+1}) \leqslant r$. Then, $x \sim_r x'$ and hence again by recalling (3), $u_{\theta*}(x, x') \leqslant r$. This finishes the proof. ■

Proof [Proof of Theorem 18] Pick $(X, d) \in \mathcal{X}$. Write $\mathfrak{T}(X, d) = (X, u)$ and $\mathfrak{T}^*(X, d) = (X, u^*)$.
(A) We prove that $u^*(x, x') \geqslant u(x, x')$ for all $x, x' \in X$. Pick $x, x' \in X$ and let $\delta := u^*(x, x')$. Let $x = x_0, \ldots, x_n = x'$ be s.t.

$$\max_i d(x_i, x_{i+1}) = u^*(x, x') = \delta.$$

Consider the two point metric space $(Z, e) := (\{p, q\}, \left(\begin{smallmatrix} 0 & \delta \\ \delta & 0 \end{smallmatrix}\right))$. Fix $i \in \{0, \ldots, n-1\}$. Consider $\phi : \{p, q\} \to X$ given by $p \mapsto x_i$ and $q \mapsto x_{i+1}$. By condition (I) we have $\mathfrak{T}(Z_\delta) = Z_\delta$. Note that $\delta = e(p, q) \geqslant d(\phi(p), \phi(q)) = d(x_i, x_{i+1})$ and hence by condition (II),

$$\delta \geqslant u(x_i, x_{i+1}).$$

Then, since i was arbitrary, we obtain $\delta \geqslant \max_i u(x_i, x_{i+1})$. Now, since u is an ultrametric on X, we know that $\max_i u(x_i, u_{i+1}) \geqslant u(x, x')$ and hence $\delta \geqslant u(x, x')$.
(B) We prove that $u^*(x, x') \leqslant u(x, x')$ for all $x, x' \in X$. Fix $r > 0$. Let (X_r, d_r) be the metric space with underlying set X_r given by the equivalence classes of X under the relation $x \sim_r x'$. Let $\phi_r : X \to X_r$ be given by $x \mapsto [x]_r$ where $[x]_r$ denotes the equivalence class of x under $\sim_r$. Let $\tilde{d}_r : X_r \times X_r \to \mathbb{R}^+$ be given by

$$\tilde{d}_r(z, z') = \min_{\substack{x \in \phi_r^{-1}(z) \\ x' \in \phi_r^{-1}(z')}} d(x, x')$$

and let $d_r = \mathcal{L}(\tilde{d}_r)$. Note that, by our construction, ϕ_r is such that for all $x,x' \in X$,

$$d(x,x') \geqslant d_r(\phi_r(x),\phi_r(x')).$$

Indeed, assume the contrary. Then for some $x,x' \in X$ one has that $d(x,x') < d_r(\phi_r(x),\phi_r(x'))$. But, from the definition of d_r it follows that $d(x,x') < d_r(\phi_r(x),\phi_r(x')) \leqslant \tilde{d}_r(\phi_r(x),\phi_r(x')) = \min\{d(\bar{x},\bar{x}'), \text{s.t. } \bar{x} \sim_r x; \bar{x}' \sim_r x'\}$. This is a contradiction since $x \sim_r x$ and $x' \sim_r x'$.

Write $\mathfrak{T}(X_r,d_r) = (X_r,u_r)$. Then, by condition (III),

$$u(x,x') \geqslant u_r(\phi_r(x),\phi_r(x')) \tag{14}$$

for all $x,x' \in X$. Note that

$$\operatorname{sep}(X_r,d_r) > r. \tag{15}$$

Indeed, for otherwise, there would be two points $x,x' \in X$ with $[x]_r \neq [x']_r$ and $r \geqslant d(x,x') \geqslant u^*(x,x')$. But this gives a contradiction by Remark 13.
Claim: $u^*(x,x') > r$ implies that $u_r(\phi_r(x),\phi_r(x')) > r$.
Assuming the claim, let $x,x' \in X$ be s.t. $u^*(x,x') > r$, then by Equation (14),

$$u(x,x') \geqslant u_r(\phi_r(x),\phi_r(x')) > r.$$

That is, we have obtained that for any $r > 0$,

$$\{(x,x') \,\text{s.t.}\, u^*(x,x') > r\} \subseteq \{(x,x') \,\text{s.t.}\, u(x,x') > r\},$$

which implies that $u^*(x,x') \leqslant u(x,x')$ for all $x,x' \in X$.
Proof of the claim. Let $x,x' \in X$ be s.t. $u^*(x,x') > r$. Then, $[x]_r \neq [x']_r$. By definition of ϕ_r, also, $\phi_r(x) \neq \phi_r(x')$ and hence, by condition (III) and Equation (15):

$$u_r(\phi_r(x),\phi_r(x')) \geqslant \operatorname{sep}(X_r,d_r) > r.$$

■

Proof [Proof of Proposition 26] Write $\mathfrak{T}^*(X,d_X) = (X,u_X)$ and $\mathfrak{T}^*(Y,d_Y) = (Y,u_Y)$. Let $\eta = d_{\mathcal{GH}}\big((X,d_X),(Y,d_Y)\big)$ and $R \in \mathcal{R}(X,Y)$ s.t. $|d_X(x,x') - d_Y(y,y')| \leqslant 2\eta$ for all $(x,y),(x',y') \in R$. Fix (x,y) and $(x',y') \in R$. Let $x_0,\ldots,x_m \in X$ be s.t. $x_0 = x$, $x_m = x'$ and $d_X(x_i,x_{i+1}) \leqslant u_X(x,x')$ for all $i = 0,\ldots,m-1$. Let $y = y_0,y_1,\ldots,y_{m-1},y_m = y' \in Y$ be s.t. $(x_i,y_i) \in R$ for all $i = 0,\ldots,m$ (this is possible by definition of R). Then, $d_Y(y_i,y_{i+1}) \leqslant d_X(x_i,x_{i+1}) + \eta \leqslant u_X(x,x') + \eta$ for all $i = 0,\ldots,m-1$ and hence $u_Y(y,y') \leqslant u_X(x,x') + 2\eta$. By exchanging the roles of X and Y one obtains the inequality $u_X(x,x') \leqslant u_Y(y,y') + 2\eta$. This means $|u_X(x,x') - u_Y(y,y')| \leqslant 2\eta$. Since $(x,y),(x',y') \in R$ are arbitrary, and upon recalling the expression of the Gromov-Hausdorff distance given by (12) we obtain the desired conclusion. ■

Proof [Proof of Theorem 28] By Proposition 26 and the triangle inequality for the Gromov-Hausdorff distance,

$$d_{\mathcal{GH}}\big(X,Z\big) + d_{\mathcal{GH}}\big(X',Z\big) \geqslant d_{\mathcal{GH}}\big((X,u_X)),(X',u_{X'}))\big).$$

Now, (1) follows from Remark 25.

We now prove the second claim. Let $\delta > 0$ be s.t. $\min_{\alpha\neq\beta} W_A(\alpha,\beta) \geqslant \delta$. For each $z \in Z$ let $\alpha(z)$ denote the index of the path connected component of Z s.t. $z \in Z^{(\alpha(z))}$. Since $r := d_{\mathcal{H}}^{Z}\big(X,Z\big) < \frac{\delta}{2}$, it is clear that $\#\left(Z^{(\alpha)} \cap X\right) \geqslant 1$ for all $\alpha \in A$. It follows that $R = \{(x,\alpha(x))|x \in X\}$ belongs to $\mathcal{R}(X,A)$. We prove below that for all $x,x' \in X$,

$$u_A(\alpha(x),\alpha(x')) \overset{\text{(I)}}{\leqslant} u_X(x,x') \overset{\text{(II)}}{\leqslant} u_A(\alpha(x),\alpha(x')) + 2r.$$

By putting (I) and (II) together we will have $d_{\mathcal{GH}}\big((X,u_X),(A,u_A)\big) \leqslant r$.

Let's prove (I). It follows immediately from the definition of d_A and W_A that for all $y,y' \in X$,

$$W_A(\alpha(y),\alpha(y')) \leqslant d_X(y,y').$$

From the definition of d_A it also follows that $W_A(\alpha,\alpha') \geqslant d_A(\alpha,\alpha')$ for all $\alpha,\alpha' \in A$. Then, in order to prove (I) pick $x_0,\ldots,x_m$ in X with $x_0 = x, x_m = x'$ and $\max_i d_X(x_i,x_{i+1}) \leqslant u_X(x,x')$. Consider the points in A given by

$$\alpha(x) = \alpha(x_0),\alpha(x_1),\ldots,\alpha(x_m) = \alpha(x').$$

Then,

$$d_A(\alpha(x_i),\alpha(x_{i+1})) \leqslant W_A(\alpha(x_i),\alpha(x_{i+1})) \leqslant d_X(x_i,x_{i+1}) \leqslant u_X(x,x')$$

for $i = 0,\ldots,m-1$ by the observations above. Then, $\max_i d_A(\alpha(x_i),\alpha(x_{i+1})) \leqslant d_X(x,x')$ and by recalling the definition of $u_A(\alpha(x),\alpha(x'))$ we obtain (I).

We now prove (II). Assume first that $\alpha(x) = \alpha(x') = \alpha$. Fix $\varepsilon_0 > 0$ small. Let $\gamma : [0,1] \to Z^{(\alpha)}$ be a continuous path s.t. $\gamma(0) = x$ and $\gamma(1) = x'$. Let $z_1,\ldots,z_m$ be points on image(γ) s.t. $z_0 = x, z_m = x'$ and $d_X(z_i,z_{i+1}) \leqslant \varepsilon_0$, $i = 0,\ldots,m-1$. By hypothesis, one can find $x = x_0,x_1,\ldots,x_{m-1},x_m = x'$ s.t. $d_Z(x_i,z_i) \leqslant r$. Thus,

$$\max_i d_X(x_i,x_{i+1}) \leqslant \varepsilon_0 + 2r$$

and hence $u_X(x,x') \leqslant \varepsilon_0 + 2r$. Let $\varepsilon_0 \to 0$ to obtain the desired result.

Now if $\alpha = \alpha(x) \neq \alpha(x') = \beta$, let $\alpha_0,\alpha_1,\ldots,\alpha_l \in A$ be s.t. $\alpha_0 = \alpha(x), \alpha_l = \alpha(x')$ and $d_A(\alpha_j,\alpha_{j+1}) \leqslant u_A(\alpha,\beta)$ for $j = 0,\ldots,l-1$.

By definition of d_A, for each $j = 0,\ldots,l-1$ one can find a *chain*

$$C_j = \left\{\alpha_j^{(0)},\ldots,\alpha_j^{(r_j)}\right\} \quad \text{s.t. } \alpha_j^{(0)} = \alpha_j, \alpha_j^{(r_j)} = \alpha_{j+1}$$

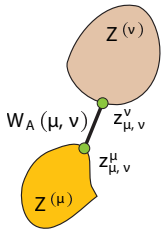
and

$$\sum_{i=0}^{r_j-1} W_A\left(\alpha_j^{(i)},\alpha_j^{(i+1)}\right) = d_A(\alpha_j,\alpha_{j+1}) \leqslant u_A(\alpha,\beta).$$

Since W_A takes non-negative values, then, for fixed $j \in \{0,\ldots,l-1\}$, it follows that

$$W_A(\alpha_j^{(i)},\alpha_j^{(i+1)}) \leqslant u_A(\alpha,\beta) \quad \text{for all } i = 0,\ldots,r_j-1.$$

Consider the chain $C = \{\widehat{\alpha}_0,\ldots,\widehat{\alpha}_s\}$ in A joining α to β given by the concatenation of all the C_j. By eliminating repeated consecutive elements in C, if necessary, one can assume that $\widehat{\alpha}_i \neq \widehat{\alpha}_{i+1}$. By construction $W_A(\widehat{\alpha}_i,\widehat{\alpha}_{i+1}) \leqslant u_A(\alpha,\beta)$ for $i \in \{0,\ldots,s-1\}$, and $\widehat{\alpha}_0 = \alpha$, $\widehat{\alpha}_s = \beta$. We will now lift C into a chain in Z joining x to x'. Note that by compactness, for all $\nu,\mu \in A, \nu \neq \mu$ there exist $z_{\nu,\mu}^{\nu} \in Z^{(\nu)}$ and $z_{\nu,\mu}^{\mu} \in Z^{(\mu)}$ s.t. $W_A(\nu,\mu) = d_Z(z_{\nu,\mu}^{\nu}, z_{\nu,\mu}^{\mu})$.



Consider the chain G in Z given by

$$G = \left\{x, z^{\hat{\alpha}_0}_{\hat{\alpha}_0,\hat{\alpha}_1}, z^{\hat{\alpha}_1}_{\hat{\alpha}_0,\hat{\alpha}_1}, \ldots, z^{\hat{\alpha}_s}_{\hat{\alpha}_{s-1},\hat{\alpha}_s}, x'\right\}.$$

For each point $g \in G \subset Z$ pick a point $x(g) \in X$ s.t. $d_Z(g, x(g)) \leqslant r$. Note that this is possible by definition of r and also, that $x(g) \in Z^{(\alpha(g))}$ since $r < \delta/2$.

Let $G' = \{x_0, x_1, \ldots, x_t\}$ be the resulting path in X. Notice that if $\alpha(x_k) \neq \alpha(x_{k+1})$ then

$$d_X(x_k, x_{k+1}) \leqslant 2r + W_A(\alpha(x_k), \alpha(x_{k+1})) \tag{16}$$

by the triangle inequality. Also, by construction, for $k \in \{0, \ldots, t-1\}$,

$$W_A(\alpha(x_k), \alpha(x_{k+1})) \leqslant u_A(\alpha, \beta). \tag{17}$$

Now, we claim that

$$u_X(x, x') \leqslant \max_k W_A(\alpha(x_k), \alpha(x_{k+1})) + 2r. \tag{18}$$

This claim will follow from (16) and the simple observation that

$$u_X(x, x') \leqslant \max_k u_X(x_k, x_{k+1}) \leqslant \max_k d_X(x_k, x_{k+1})$$

which in turn follows from the fact that u_X is the ultrametric on X defined by (4), see remarks in Example 7. If $\alpha(x_k) = \alpha(x_{k+1})$ we already proved that $u_X(x_k, x_{k+1}) \leqslant 2r$. If on the other hand $\alpha(x_k) \neq \alpha(x_{k+1})$ then (18) holds. Hence, we have that without restriction, for all $x, x' \in X$,

$$u_X(x, x') \leqslant \max_k W_A(\alpha(x_k), \alpha(x_{k+1})) + 2r.$$

and hence the claim. Combine this fact with (17) to conclude the proof of (II). Claim (3) follows immediately from (2).

■

B.1 The Proof of Theorem 30

We will make use of the following general covering theorem in the proof of Theorem 30.

Theorem 34 *Let (X, d, μ) be an mm-space and $\mathbb{X}_n = \{x_1, x_2, \ldots, x_n\}$ a collection of n independent random variables (defined on some probability space Ω, and with values in X) and identically distributed with distribution μ. Then, for any $\delta > 0$,*

$$\mathbf{P}_\mu\left(d^X_{\mathcal{H}}\left(\mathbb{X}_n, supp\left[\mu_X\right]\right) > \delta\right) \leqslant F_X(n, \delta).$$

Proof Consider first a fixed point $x \in \operatorname{supp}[\mu_X]$ and $h > 0$. Then, since $x_1, \ldots, x_n$ are i.i.d., for all i, $\mathbf{P}_\mu(x_i \in B_X(x,h)) = \mu(B_X(x,h))$. We then have:

$$\begin{aligned}
\mathbf{P}_\mu\left(\left\{x \notin \bigcup_{i=1}^n B_X(x_i,h)\right\}\right) &= \mathbf{P}_\mu\left(\bigcap_{i=1}^n \{x \notin B_X(x_i,h)\}\right) \\
&= \mathbf{P}_\mu\left(\bigcap_{i=1}^n \{x_i \notin B_X(x,h)\}\right) \\
&= \prod_{i=1}^n \mathbf{P}_\mu(\{x_i \notin B_X(x,h)\}) \quad \text{(by independence)} \\
&= (1-\mu_X(B_X(x,h)))^n \\
&\leqslant (1-f_X(h)))^n. \qquad (19)
\end{aligned}$$

We now obtain a similar bound for the probability that a ball of radius $\delta/2$ around x is within δ of a point in $\mathbb{X}_n$. Notice that the following inclusion of events holds:

$$\left\{B_X(x,\delta/2) \subset \bigcup_{i=1}^n B_X(x_i,\delta)\right\} \supseteq \left\{x \in \bigcup_{i=1}^n B_X(x_i,\delta/2)\right\}. \qquad (20)$$

Indeed, assume that the event $\{x \in \bigcup_{i=1}^n B_X(x_i,\delta/2)\}$ holds. Then, $x \in B_X(x_i,\delta/2)$ for some $i \in \{1,\ldots,n\}$. Pick any $x' \in B_X(x,\delta/2)$, then by the triangle inequality, $d_X(x',x_i) \leqslant d_X(x',x) + d_X(x,x_i) < \delta/2 + \delta/2 = \delta$, thus $x' \in B_X(x_i,\delta)$. Since x' is an arbitrary point in $B_X(x,\delta/2)$ we are done. Now, from (20) and (19) (for $h = \delta/2$) above, we find

$$\mathbf{P}_\mu\left(\left\{B_X(x,\delta/2) \nsubseteq \bigcup_{i=1}^n B_X(x_i,\delta)\right\}\right) \leqslant (1-f_X(\delta/2))^n. \qquad (21)$$

Now, consider a maximal $\delta/4$-packing of $\operatorname{supp}[\mu_X]$ by balls with centers $\{p_1,\ldots,p_N\}$. Then, clearly, $\operatorname{supp}[\mu_X] = \bigcup_{j=1}^N B_X(p_j,\delta/2)$. Such a packing always exists since $\operatorname{supp}[\mu_X]$ is assumed to be compact (Burago et al., 2001). Notice that N, the cardinality of the packing, can be bounded by $1/f_X(\delta/4)$. Indeed, since $B_X(p_\alpha,\delta/4) \cap B_X(p_\beta,\delta/4) = \varnothing$ for $\alpha \neq \beta$, we have

$$\begin{aligned}
1 = \mu_X(\operatorname{supp}[\mu_X]) &= \mu_X\left(\bigcup_{j=1}^N B_X(p_j,\delta/2)\right) \\
&\geqslant \mu_X\left(\bigcup_{j=1}^N B_X(p_j,\delta/4)\right) \\
&= \sum_{j=1}^N \mu_X(B_X(p_j,\delta/4)) \\
&\geqslant N \cdot f_X(\delta/4)
\end{aligned}$$

and the claim follows. Now, we finish the proof by first noting that since $\mathbb{X}_n \subset \mathrm{supp}\,[\mu_X]$, the following inclusion of events holds:

$$\{d_{\mathcal{H}}^X(\mathbb{X}_n, \mathrm{supp}\,[\mu_X]) > \delta\} \subseteq \left\{ X \nsubseteq \bigcup_{i=1}^n B_X(x_i, \delta) \right\}$$

and hence, using the union bound, then (21) and the bound on N, we find:

$$\begin{aligned}
\mathbf{P}_\mu\left(d_{\mathcal{H}}^X(\mathbb{X}_n, \mathrm{supp}\,[\mu_X]) > \delta\right) &\leqslant \mathbf{P}_\mu\left(X \nsubseteq \bigcup_{i=1}^n B_X(x_i,\delta)\right) \\
&= \mathbf{P}_\mu\left(\bigcup_{j=1}^N \left\{ B_X(p_j,\delta/2) \nsubseteq \bigcup_{i=1}^n B_X(x_i,\delta)\right\}\right) \\
&\leqslant N \cdot \max_{j=1,\ldots,N} \mathbf{P}_\mu\left(B_X(p_j,\delta/2) \nsubseteq \bigcup_{i=1}^n B_X(x_i,\delta)\right) \\
&\leqslant \frac{1}{f_X(\delta/4)} \cdot (1 - f_X(\delta/2))^n \\
&\leqslant \frac{1}{f_X(\delta/4)} \cdot (1 - f_X(\delta/4))^n \quad \text{(since } f_X(\cdot) \text{ is non-decreasing)} \\
&\leqslant \frac{1}{f_X(\delta/4)} e^{-n f_X(\delta/4)} \quad \text{(by the inequality } (1-t) \leqslant e^{-t}, \forall t \in \mathbb{R}) \\
&= F_X(n,\delta)
\end{aligned}$$

thus concluding the proof. ■

Proof [Proof of Theorem 30] For each $n \in \mathbb{N}$, introduce the random variables $r_n := d_{\mathcal{H}}^Z(\mathbb{Z}_n, \mathrm{supp}\,[\mu_Z])$ and $g_n := d_{\mathcal{GH}}(\mathfrak{T}^*(\mathbb{Z}_n, d_{\mathbb{Z}_n}), \mathfrak{T}^*(A, d_A))$. Fix $\zeta' = \delta_A/2$. Note that by Theorem 28 (2) once $r_n \leqslant \zeta$ for some $\zeta \leqslant \zeta'$ we know that $g_n \leqslant r_n$ a.s. Hence, we have

$$\mathbf{P}(g_n > \zeta) \leqslant \mathbf{P}(r_n > \zeta) \leqslant F_X(n,\zeta), \tag{22}$$

where the last inequality follows from Lemma 34.

Meanwhile, if $\zeta \geqslant \zeta'$ is arbitrary, then $\mathbf{P}(g_n > \zeta) \leqslant \mathbf{P}(g_n > \zeta')$. By (22) (for $\zeta = \zeta'$) we find $\mathbf{P}(g_n > \zeta) \leqslant \mathbf{P}(r_n > \zeta') \leqslant F_X(n,\zeta')$ for all $\zeta \geqslant \zeta'$. Thus, we have found that

$$\mathbf{P}(g_n > \zeta) \leqslant \begin{cases} F_X(n,\zeta') & \text{for } \zeta \geqslant \zeta'. \\ F_X(n,\zeta) & \text{for } \zeta \leqslant \zeta'. \end{cases}$$

The conclusion now follows. ■

References

S. Ben-David, U. von Luxburg, and D. Pál. A sober look at clustering stability. In Gábor Lugosi and Hans-Ulrich Simon, editors, *COLT*, volume 4005 of *Lecture Notes in Computer Science*, pages 5–19. Springer, 2006. ISBN 3-540-35294-5.

F. L. Bookstein, B. Chernoff, R. L. Elder, J. M. Humphries Jr., G. R. Smith, and R. E. Strauss. *Morphometrics in Evolutionary Biology: the Geometry of Size and Shape Change, with Examples from Fishes*. Academy of Natural Sciences of Philadelphia., 1985.

M. R. Bridson and A. Haefliger. *Metric Spaces of Non-positive Curvature*, volume 319 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1999. ISBN 3-540-64324-9.

D. Burago, Y. Burago, and S. Ivanov. *A Course in Metric Geometry*, volume 33 of *AMS Graduate Studies in Math*. American Mathematical Society, 2001.

G. Carlsson and F. Mémoli. Persistent clustering and a theorem of J. Kleinberg. *ArXiv e-prints*, August 2008.

G. Carlsson and F. Mémoli. Classifying clustering schemes. Technical report, Stanford, 2009.

G. Carlsson and F. Mémoli. Multiparameter clustering methods. In Claus Weihs Hermann Locarek-Junge, editor, *'Classification as a Tool for Research'. Proceedings of the 11th IFCS Biennial Conference and 33rd Annual Conference of the Gesellschaft fr Klassifikation e.V.*, Berlin-Heidelberg-New York, to appear 2009. Springer.

M. Gromov. *Metric Structures for Riemannian and non-Riemannian spaces*. Modern Birkhäuser Classics. Birkhäuser Boston Inc., Boston, MA, english edition, 2007. ISBN 978-0-8176-4582-3; 0-8176-4582-9. Based on the 1981 French original, With appendices by M. Katz, P. Pansu and S. Semmes, Translated from the French by Sean Michael Bates.

M. Gromov. Hyperbolic groups. In *Essays in Group Theory*, volume 8 of *Math. Sci. Res. Inst. Publ.*, pages 75–263. Springer, New York, 1987.

J. A. Hartigan. Consistency of single linkage for high-density clusters. *J. Amer. Statist. Assoc.*, 76 (374):388–394, 1981. ISSN 0162-1459.

J. A. Hartigan. Statistical theory in clustering. *J. Classification*, 2(1):63–76, 1985. ISSN 0176-4268.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall Advanced Reference Series. Prentice Hall Inc., Englewood Cliffs, NJ, 1988. ISBN 0-13-022278-X.

N. Jardine and R. Sibson. *Mathematical Taxonomy*. John Wiley & Sons Ltd., London, 1971. Wiley Series in Probability and Mathematical Statistics.

D. G. Kendall, D. Barden, T. K. Carne, and H. Le. *Shape and Shape Theory*. Wiley Series in Probability and Statistics. John Wiley & Sons Ltd., Chichester, 1999. ISBN 0-471-96823-4.

J. M. Kleinberg. An impossibility theorem for clustering. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, pages 446–453. MIT Press, 2002. ISBN 0-262-02550-7.

G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies 1. Hierarchical systems. *Computer Journal*, 9(4):373–380, February 1967. ISSN 0010-4620.

W. Stuetzle. Estimating the cluster type of a density by analyzing the minimal spanning tree of a sample. *J. Classification*, 20(1):25–47, 2003. ISSN 0176-4268.

U. von Luxburg and S. Ben-David. Towards a statistical theory of clustering. Technical report, PASCAL workshop on clustering, London, 2005.

D. Wishart. Mode analysis: a generalization of nearest neighbor which reduces chaining effects. In *Numerical Taxonomy*, pages 282–311. Academic Press, 1969.

R. Zadeh and S. Ben-David. A uniqueness theorem for clustering. In *Proceedings of the Proceedings of the Twenty-Fifth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*, page 8, Corvallis, Oregon, 2009. AUAI Press.

Journal of Machine Learning Research 11 (2010) 1471-1490 Submitted 8/09; Revised 1/10; Published 4/10

Training and Testing Low-degree Polynomial Data Mappings via Linear SVM

Yin-Wen Chang B92059@CSIE.NTU.EDU.TW
Cho-Jui Hsieh B92085@CSIE.NTU.EDU.TW
Kai-Wei Chang B92084@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Michael Ringgaard RINGGAARD@GOOGLE.COM
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: Sathiya Keerthi

Abstract

Kernel techniques have long been used in SVM to handle linearly inseparable problems by transforming data to a high dimensional space, but training and testing large data sets is often time consuming. In contrast, we can efficiently train and test much larger data sets using linear SVM without kernels. In this work, we apply fast linear-SVM methods to the explicit form of polynomially mapped data and investigate implementation issues. The approach enjoys fast training and testing, but may sometimes achieve accuracy close to that of using highly nonlinear kernels. Empirical experiments show that the proposed method is useful for certain large-scale data sets. We successfully apply the proposed method to a natural language processing (NLP) application by improving the testing accuracy under some training/testing speed requirements.

Keywords: decomposition methods, low-degree polynomial mapping, kernel functions, support vector machines, dependency parsing, natural language processing

1. Introduction

Support vector machines (SVMs) (Boser et al., 1992; Cortes and Vapnik, 1995) have been popular for data classification. An SVM often maps data to a high dimensional space and then employs kernel techniques. We refer to such an approach as nonlinear SVM. Training nonlinear SVM is usually performed through the use of popular decomposition methods. However, these decomposition approaches require considerable time for large data sets. In addition, the testing procedure is slow due to the kernel calculation involving support vectors and testing instances.

For some applications with data in a rich dimensional space (e.g., document classification), people have shown that testing accuracy is similar with/without a nonlinear mapping. If data are not

mapped, recently some methods have been proposed to efficiently train much larger data sets. We refer to such cases as linear SVM.

Among the recent advances in training large linear SVM, Hsieh et al. (2008) discuss decomposition methods for training linear and nonlinear SVM. If l is the number of training data, $\bar{n}$ is the average number of non-zero features per instance, and each kernel evaluation takes $O(\bar{n})$ time, then the cost per decomposition iteration for nonlinear SVM is $O(l\bar{n})$. Taking the property of linear SVM, Hsieh et al.'s approach runs one iteration in only $O(\bar{n})$. If the number of iterations is not significantly more than that for the nonlinear case, their method is very efficient for training linear SVM.

Motivated by the above $O(l\bar{n})$ and $O(\bar{n})$ difference, in this work, we investigate the performance of applying linear-SVM methods to low-degree data mappings. By considering the explicit form of the mapping, we directly train a linear SVM. The cost per decomposition iteration is $O(\hat{n})$, where $\hat{n}$ is the new average number of non-zero elements in the mapped vector. If $\hat{n} < l\bar{n}$, the new strategy may be faster than the training using kernels.

Currently, polynomial kernels are less widely used than the RBF (Gaussian) kernel, which maps data to an infinite dimensional space. This might be because under similar training and testing cost, a polynomial kernel may not give higher accuracy. We show for some data, the testing accuracy of using low-degree polynomial mappings is only slightly worse than RBF, but training/testing via linear-SVM strategies is much faster. Therefore, our approach takes advantages of linear methods, while still preserves a certain degree of nonlinearity. Some early works (e.g., Gertz and Griffin, 2005; Jung et al., 2008; Moh and Buhmann, 2008) have employed this idea in their experiments. Here we aim at a more detailed study on large-scale scenarios.

An exception where polynomial kernels have been popular is NLP (natural language processing). Some have explored the fast calculation of low-degree polynomial kernels to save the testing time (e.g., Isozaki and Kazawa, 2002; Kudo and Matsumoto, 2003; Goldberg and Elhadad, 2008). However, these works still suffer from the slow training because of not applying some recently developed training techniques.

This paper is organized as follows. We introduce SVM in Section 2. In Section 3, we discuss the proposed method for efficiently training and testing SVM for low-degree polynomial data mappings. A particular emphasis is on the degree-2 polynomial mapping. Section 4 presents the empirical studies. We give an NLP application on dependency parsing in Section 5. Conclusions are in Section 6.

Notation: we list some notation related to the number of features.

n: number of features (dimensionality of data); $\mathbf{x}_i \in R^n$ is the ith training instance

n_i: number of non-zero feature values of $\mathbf{x}_i$

$\bar{n}$: average number of non-zero elements in $\mathbf{x}_i$; see (9)

$\hat{n}$: average number of non-zero elements in the mapped vector $\phi(\mathbf{x}_i)$

$\tilde{n}$: number of non-zero elements in the weight vector $\mathbf{w}$

2. Linear and Nonlinear SVM

Assume training instance-label pairs are $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, l$, where $\mathbf{x}_i \in R^n$ and $y_i \in \{1, -1\}$. We consider the following SVM problem with a penalty parameter $C > 0$:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \max(1 - y_i\mathbf{w}^T\phi(\mathbf{x}_i), 0). \tag{1}$$

The function $\phi(\mathbf{x})$ maps an instance to a higher dimensional space for handling linearly inseparable data. We refer to such a setting as nonlinear SVM. For some applications, $\phi(\mathbf{x}) = \mathbf{x}$ can already properly separate data; we call such cases linear SVM. Many SVM studies consider $\mathbf{w}^T\mathbf{x}_i + b$ instead of $\mathbf{w}^T\mathbf{x}_i$ in (1). In general this bias term b does not affect the performance much, so here we omit it for the simplicity.

Due to the high dimensionality of $\phi(\mathbf{x})$ and the possible difficulty of obtaining the explicit form of $\phi(\mathbf{x})$, SVM is often solved through the dual problem with the kernel trick:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^T Q\alpha - \mathbf{e}^T\alpha \\ \text{subject to} \quad & 0 \le \alpha_i \le C, i = 1,\ldots,l, \end{aligned} \tag{2}$$

where $Q_{ij} = y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) = y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ and $\mathbf{e} = [1,\ldots,1]^T$. $K(\mathbf{x}_i,\mathbf{x}_j)$ is called the kernel function and α is the dual variable.

The matrix Q in the dual problem (2) is dense and may be too large to be stored in the computer memory. Currently, decomposition methods (e.g., Joachims, 1998; Keerthi et al., 2001; Chang and Lin, 2001) are the major approach to solve (2). However, if linear SVM is considered, we can more easily solve both the primal and the dual problems. Early studies (e.g., Mangasarian and Musicant, 1999; Ferris and Munson, 2003) have demonstrated that many traditional optimization methods can be applied. They focus on data with many instances but a small number of features. Recently, an active research topic is to train linear SVM with both large numbers of instances and features (e.g., Joachims, 2006; Shalev-Shwartz et al., 2007; Bottou, 2007; Hsieh et al., 2008; Langford et al., 2009).

3. Using Linear SVM for Low-degree Polynomial Data Mappings

In this section, we discuss the methods and issues in training/testing low-degree data mappings using linear SVM. We are interested in when the training via linear-SVM techniques is faster than nonlinear SVM. We put an emphasis on the degree-2 polynomial mapping.

3.1 Low-degree Polynomial Mappings

A polynomial kernel takes the following form

$$K(\mathbf{x}_i,\mathbf{x}_j) = (\gamma\mathbf{x}_i^T\mathbf{x}_j + r)^d, \tag{3}$$

where γ and r are parameters and d is the degree. The polynomial kernel is the product between two vectors $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$. For example, if $d = 2$ and $r = 1$, then

$$\phi(\mathbf{x}) = [1, \sqrt{2\gamma}x_1,\ldots,\sqrt{2\gamma}x_n, \gamma x_1^2,\ldots,\gamma x_n^2, \sqrt{2}\gamma x_1x_2,\ldots,\sqrt{2}\gamma x_{n-1}x_n]^T. \tag{4}$$

The coefficient $\sqrt{2}$ in (4) is only used to make $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ have a simple form. Without using kernels, we can consider more flexible mapping vectors. For example, if $\gamma = 1$, removing $\sqrt{2}$ in (4) results in a simple mapping vector:

$$\phi(\mathbf{x}) = [1, x_1,\ldots,x_n, x_1^2,\ldots,x_n^2, x_1x_2,\ldots,x_{n-1}x_n]^T. \tag{5}$$

For the polynomial kernel (3), the dimensionality of $\phi(\mathbf{x})$ is

$$C(n+d,d) = \frac{(n+d)(n+d-1)\cdots(n+1)}{d!},$$

which is obtained by counting the number of terms in (3).

3.2 Training by Linear SVM Methods

The training time for SVM depends on the number of data instances and the number of features. Due to the high dimensionality of $\phi(\mathbf{x})$, we must judge whether it is better to choose an explicit mapping or an implicit way by kernels. We explore this issue by investigating the difference between applying decomposition methods to solve the dual problem of linear and nonlinear SVM. Though many optimization methods have been applied to train SVM, we discuss decomposition methods because of the following reasons. First, they are the major approach for nonlinear SVM. Second, efficient decomposition methods for linear SVM have been developed (e.g., Hsieh et al., 2008).

A decomposition method iteratively updates a small subset of variables. We consider the situation of updating one variable at a time.[1] If α is the current solution and the ith component is selected for update, then we minimize the following one-variable problem:

$$\begin{aligned} \min_d \quad & \frac{1}{2}(\alpha + d\mathbf{e}_i)^T Q(\alpha + d\mathbf{e}_i) - \mathbf{e}^T(\alpha + d\mathbf{e}_i) \\ & = \frac{1}{2}Q_{ii}d^2 + (Q\alpha - \mathbf{e})_i d + \text{constant} \\ \text{subject to} \quad & 0 \le \alpha_i + d \le C. \end{aligned} \tag{6}$$

This minimization is easy, but to construct (6), we must calculate

$$(Q\alpha - \mathbf{e})_i = \sum_{j=1}^{l} Q_{ij}\alpha_j - 1 = \sum_{j=1}^{l} y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)\alpha_j - 1. \tag{7}$$

If each kernel element costs $O(\bar{n})$, where $\bar{n}$ is the average number of non-zero features, then (7) needs $O(l\bar{n})$ operations.

If using the explicit mapping vectors, we can calculate $(Q\alpha - \mathbf{e})_i$ by

$$\sum_{j=1}^{l} Q_{ij}\alpha_j - 1 = y_i \mathbf{w}^T \phi(\mathbf{x}_i) - 1, \text{ where } \mathbf{w} = \sum_{j=1}^{l} y_j \alpha_j \phi(\mathbf{x}_j). \tag{8}$$

If $\mathbf{w}$ is available, (8) requires $O(\hat{n})$ operations, where $\hat{n}$ is the average number of non-zero elements in $\phi(\mathbf{x}_i), \forall i$. To maintain $\mathbf{w}$, Hsieh et al. (2008) use

$$\mathbf{w} \leftarrow \mathbf{w} + y_i(\alpha_i^{\text{new}} - \alpha_i^{\text{old}})\phi(\mathbf{x}_i),$$

so the cost is also $O(\hat{n})$. Therefore, the above discussion indicates the tradeoff between $O(l\bar{n})$ and $O(\hat{n})$ cost by implicit and explicit mappings, respectively.

Practical implementations of decomposition methods involve other issues. For example, if using the kernel trick, we may develop better techniques for selecting the working variable at each iteration. Then the number of iterations is smaller. More details can be found in Hsieh et al. (2008, Section 4). Nevertheless, checking $O(l\bar{n})$ and $O(\hat{n})$ can roughly indicate if using an explicit mapping leads to faster training.

1. If using standard SVM with the bias b, the dual form contains an equality and at least two variables must be considered.

3.3 Number of Non-zero Features per Instance

From the discussion in Section 3.2, it is important to know the value $\hat{n}$. If the input data are dense, then the number of non-zero elements in $\phi(\mathbf{x})$ is $O(n^d)$, where d is the degree of the polynomial mapping.

If the input data are sparse, the number of non-zero elements is smaller than the dimensionality. Assume n_i is the number of non-zero elements of the ith training instance. Then the average number in $\mathbf{x}_i$ $\forall i$ is

$$\bar{n} = \frac{n_1 + \cdots + n_l}{l}. \tag{9}$$

If $d = 2$, the average number of non-zero elements in $\phi(\mathbf{x}_i)$, $\forall i$ is

$$\hat{n} = \frac{1}{l}\sum_{i=1}^{l} \frac{(n_i+2)(n_i+1)}{2} \approx \frac{1}{l}\sum_{i=1}^{l} \frac{n_i^2}{2} = \frac{1}{2}\bar{n}^2 + \frac{1}{2l}\sum_{i=1}^{l}(n_i - \bar{n})^2. \tag{10}$$

The second term in (10) is the variance of $n_1, \ldots, n_l$. If the variance is small, comparing $l\bar{n}$ and $\bar{n}^2/2$ can possibly indicate if one should train a linear or a nonlinear SVM. In Section 4, we give more analysis on real data.

3.4 Implementation Issues

Due to the high dimensionality of $\phi(\mathbf{x}_i)$, some implementation issues must be addressed. To begin, we discuss various ways to handle the new data $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_l)$. A vector $\phi(\mathbf{x})$ now contains terms like $x_r x_s$, which can be calculated using $\mathbf{x}$. We consider three methods:

1. Calculate and store $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_l)$ as the new input data.
2. Use $\mathbf{x}_1, \ldots, \mathbf{x}_l$ as the input data and calculate all $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_l)$ before training.
3. Use $\mathbf{x}_1, \ldots, \mathbf{x}_l$ as the input data and calculate $\phi(\mathbf{x}_i)$ in a training algorithm (e.g., decomposition method).

These methods have advantages/disadvantages. The first method does not require any modification of linear-SVM solvers, but needs a large $O(l\hat{n})$ disk/memory space to store $\phi(\mathbf{x}_i)$, $\forall i$. The second method also needs extra memory spaces, but avoids the long time for loading data from disk. The third method does not need extra memory, but requires some modifications of the decomposition implementation. That is, we need to calculate $\phi(\mathbf{x}_i)$ in (8). These three methods are useful under different circumstances. In Section 4.3, we experimentally show that for data with not too large n, the third way is the fastest. Although $\phi(\mathbf{x}_i)$ $\forall i$ can be stored in memory, accessing data from memory to cache and then CPU may be slower than performing the calculation. However, for an application with very large n and very small $\bar{n}$, we demonstrate that the first or the second method may be more suitable. See the discussion later in this section.

While we may avoid storing $\phi(\mathbf{x}_i)$, one memory bottleneck remains. The vector $\mathbf{w}$ has a huge number of $O(n^d)$ components. If some features of $\phi(\mathbf{x}_i)$, $\forall i$ are zero, their corresponding elements in $\mathbf{w}$ are useless. Hence we can implement a sparse $\mathbf{w}$ to save the storage. In the following we analyze the possibility of having a sparse $\mathbf{w}$ by considering $d = 2$ and assuming that features of an instance have an equal opportunity to be zeros. The probability that $(\mathbf{x}_i)_r(\mathbf{x}_i)_s$ is zero for all $\phi(\mathbf{x}_i)$, $i = 1, \ldots, l$ is

$$\prod_{i=1}^{l}\left(1 - \frac{n_i(n_i-1)}{n(n-1)}\right).$$

Note that $n_i(n_i-1)/n(n-1)$ is the probability that $(\mathbf{x}_i)_r(\mathbf{x}_i)_s$ is non-zero. Then the expected number of non-zero elements in $\mathbf{w}$ can be approximated by

$$C(n+2,2)-\frac{n(n-1)}{2}\prod_{i=1}^{l}\left(1-\frac{n_i(n_i-1)}{n(n-1)}\right), \tag{11}$$

where $n(n-1)/2$ is the number of $x_r x_s$ terms in (4). This number is usually close to $C(n+2,2)$ due to the product of l values in (11). Moreover, this estimate is only accurate if features are independent. The assumption may hold for data sets with features from bitmaps or word frequencies, but is wrong for data with exclusive features (e.g., binary representation of a nominal feature). For data with known structures of features, in Section 4.2 we use real examples to illustrate how to more accurately estimate the number of $\mathbf{w}$'s non-zero elements.

In Section 5, we present an example of using a sparse $\mathbf{w}$. The dimensionality of the input data is $n = 46,155$. If $d = 2$, then storing $\mathbf{w}$ as a dense vector takes almost 20 GBytes of memory (assuming double precision). This problem has a very small $n_i \approx 13.3, \forall i$. Many $x_r x_s$ terms are zero in all $\phi(\mathbf{x}_i), i = 1,\ldots,l$, so $\mathbf{w}$ is very sparse. However, a naive implementation can be very inefficient. Assume $\tilde{n}$ is the number of non-zero elements in $\mathbf{w}$, where for this example $\tilde{n} = 1,438,456$. Accessing an element in a sparse vector requires an expensive $O(\tilde{n})$ linear search. We can use a hash table to store $\mathbf{w}$, but experiments show that the access time of $\mathbf{w}$ is still high. If $\phi(\mathbf{x}_i)$, $\forall i$ can be stored in memory, we construct a hash mapping from (r,s) to $j \in \{1,\ldots,\tilde{n}\}$ and re-generate training data using feature index j. That is, we construct a condensed representation for $\phi(\mathbf{x}_i), \forall i$ and train a linear SVM with a dense $\mathbf{w} \in R^{\tilde{n}}$. The training is much more efficient because we can easily access any element of $\mathbf{w}$. In prediction, for any testing instance $\mathbf{x}$, we use the same hash table to conduct the inner product $\mathbf{w}^T\phi(\mathbf{x})$. This strategy corresponds to the first/second methods in the above discussion of handling $\phi(\mathbf{x}_i)\ \forall i$.

The above technique to condense $\phi(\mathbf{x}_i)$ has been used in recent works on hash kernels (e.g., Shi et al., 2009). They differ from us in several aspects. First, their condensed representation is an approximation to $\phi(\mathbf{x}_i)$. Second, with an online setting, they may not accurately solve the problem (1).

3.5 Relations with the RBF kernel

RBF kernel (Gaussian kernel) may be the most used kernel in training nonlinear SVM. It takes the following form:

$$K(\mathbf{x}_i,\mathbf{x}_j) = e^{\frac{-\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}}.$$

Keerthi and Lin (2003) show that, as $\sigma^2 \to \infty$, SVM with the RBF kernel and the penalty parameter C approaches linear SVM with the penalty parameter $C/(2\sigma^2)$. This result implies that with a suitable parameter selection, the testing accuracy of using the RBF kernel is at least as good as using the linear kernel.

For polynomial kernels, Lippert and Rifkin (2006) discuss the relation with RBF. They consider the penalty parameter $(1/(2\sigma^2))^{-2d}C$ and check the situation as $\sigma^2 \to \infty$. For a positive integer d, the limit of SVM with the RBF kernel approaches SVM with a degree-d polynomial mapping of data. The polynomial mapping is related only to the degree d. This result seems to indicate that RBF is again at least as good as polynomial kernels. However, the polynomial mapping for (3) is more general due to two additional parameters γ and r. Thus the situation is unclear if parameter

Data set	n	$\bar{n}$	l	# testing
a9a	123	13.9	32,561	16,281
real-sim	20,958	51.5	57,848	14,461
news20	1,355,181	455.5	15,997	3,999
ijcnn1	22	13.0	49,990	91,701
MNIST38	752	168.2	11,982	1,984
covtype	54	11.9	464,810	116,202
webspam	254	85.1	280,000	70,000

Table 1: Summary of the data sets. n is the number of features, and $\bar{n}$ is the average number of non-zero features for each data instance. l is the number of data instances. The last column shows the number of testing data.

selections have been applied to both kernels. In Section 4, we give a detailed comparison between degree-2 polynomial mappings and RBF.

3.6 Parameter Selection

The polynomial kernel defined in (3) has three parameters (d, γ, and r). Now we intend to use low-degree mappings so d should be 2 or 3. Selecting the two remaining parameters is still complicated. Fortunately, we show in Appendix A that r can be fixed to one, so the number of parameters is the same as that of the RBF kernel. This result is obtained by proving that a polynomial kernel

$$\bar{K}(\mathbf{x}_i, \mathbf{x}_j) = (\bar{\gamma}\mathbf{x}_i^T\mathbf{x}_j + r)^d \text{ with parameters } (\bar{C}, \bar{\gamma}, r) \tag{12}$$

results in the same model as the polynomial kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\mathbf{x}_i^T\mathbf{x}_j + 1)^d \text{ with parameters } \gamma = \frac{\bar{\gamma}}{r} \text{ and } C = r^d\bar{C}. \tag{13}$$

3.7 Prediction

Assume a degree-d polynomial mapping is considered and #SV is the number of support vectors. For any testing data $\mathbf{x}$, the prediction time with/without kernels is

$$\sum_{i:\alpha_i>0} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) \quad \Rightarrow \quad O(\#\text{SV} \cdot \bar{n}), \tag{14}$$

$$\mathbf{w}^T\phi(\mathbf{x}) \quad \Rightarrow \quad O(\hat{n}), \tag{15}$$

where $\bar{n}$ and $\hat{n}$ are respectively the average number of non-zero elements in $\mathbf{x}$ and $\phi(\mathbf{x})$. If $\hat{n} \leq \#\text{SV} \cdot \bar{n}$, then (15) is more efficient than (14). Several NLP studies (e.g., Isozaki and Kazawa, 2002) have used (15) for efficient testing.

4. Experiments

In this section, we experimentally analyze the proposed approach for degree-2 polynomial mappings. We use two-class data sets, but in Section 5, we consider multi-class problems from an NLP application. We briefly discuss extensions to L1-regularized SVM in Section 4.5.

Data set	Analysis of $\phi(\mathbf{x}_i)$			# non-zeros in $\mathbf{w}$			
	$\bar{n}^2/2$	$\hat{n}$	$l\bar{n}$	$C(n+2,2)$	Estimated by (11)	Estimated by (16)	Real
a9a	96.2	118.1	4.52e+05	7.75e+03	7.75e+03	6.60e+03	5.56e+03
real-sim	1,325.1	2,923.6	2.98e+06	2.20e+08	1.15e+08		3.01e+07
news20	103,750.6	327,051.6	7.29e+06	9.18e+11	5.20e+09		3.13e+09
ijcnn1	84.5	105.0	6.50e+05	2.76e+02	2.76e+02	2.31e+02	2.31e+02
MNIST38	14,147.2	14,965.1	2.02e+06	2.84e+05	2.84e+05		1.54e+05
covtype	71.3	90.3	5.55e+06	1.54e+03	1.54e+03	7.54e+02	6.69e+02
webspam	3,623.5	3,836.4	2.38e+07	3.26e+04	3.26e+04		9.44e+03

Table 2: Analysis of $\phi(\mathbf{x}_i), i = 1,\ldots,l$ and number of non-zero elements in $\mathbf{w}$.

Except programs used in Section 5, all sources for experiments are available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/exp.html`.

4.1 Data Sets and Implementations

We select the following problems from LIBSVM tools[2] for experiments: a9a, real-sim, news20, ijcnn1, MNIST, covtype and webspam. The summary of data sets is in Table 1. Problems real-sim, news20, covtype and webspam have no original test sets, so we use a 80/20 split for training and testing. MNIST is a 10-class problem; we consider classes 3 and 8 to form a two-class data set MNIST38. While covtype is originally multi-class, we use a two-class version at LIBSVM tools.

We do not further scale these data sets as some of them have been pre-processed. Problems real-sim, news20 and webspam are document sets and each instance is normalized to unit length. We use a scaled version of covtype at LIBSVM tools, where each feature is linearly scaled to $[0,1]$. The original MNIST data have all feature values in the range $[0,255]$, but the version we download is scaled to $[0,1]$ by dividing every value by 255.

We compare implicit mappings (kernel) and explicit mappings of data by LIBSVM (Chang and Lin, 2001) and an extension of LIBLINEAR (Fan et al., 2008), respectively. The two packages use similar stopping conditions, and we set the same stopping tolerance 0.1. Experiments are conducted on a 2.5G Xeon L5420 machine with 16G RAM using gcc compiler. Our experiments are run on a single CPU.

4.2 Analysis of $\phi(\mathbf{x})$ and $\mathbf{w}$

Following the discussion in Section 3.2, we check $l\bar{n}$ and $\hat{n}$ to see if using the explicit mapping of data may be helpful. Table 2 presents these two values for the degree-2 polynomial mapping. We also present $\bar{n}^2/2$ as from (10) it can be a rough estimate of $\hat{n}$.

From Table 2, except document data real-sim and news20, $\bar{n}^2/2$ is close to $\hat{n}$. The huge difference between $\hat{n}$ and $l\bar{n}$ indicates that using explicit mappings is potentially faster.

Next we investigate the number of non-zero elements in $\mathbf{w}$. Table 2 presents the dimensionality of $\mathbf{w}$, two estimated numbers of non-zero elements and the actual number. For most data, the first estimation by (11) pessimistically predicts that $\mathbf{w}$ is fully dense. Two exceptions are real-sim and

2. LIBSVM tools can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

Data set	$\hat{n}/\bar{n}$	Storing $\phi(\mathbf{x}_i)$		Calculating $\phi(\mathbf{x}_i)$	
		L2 cache misses	Training time (s)	L2 cache misses	Training time (s)
a9a	8.51	5.62e+07	2.2	2.51e+06	1.6
real-sim	56.79	2.60e+09	63.3	1.84e+09	59.8
ijcnn1	8.08	3.62e+08	14.0	2.32e+07	10.7
MNIST38	88.97	9.08e+08	20.4	7.90e+06	8.6
covtype	7.56	1.55e+11	6,422.4	2.98e+10	5,211.9
webspam[4]	45.07	1.30e+11	4,219.3	3.20e+09	3,228.1

Table 3: A comparison between storing and calculating $\phi(\mathbf{x}_i)$. The column $\hat{n}/\bar{n}$ indicates the ratio between the memory consumption for storing $\phi(\mathbf{x}_i)$ and $\mathbf{x}_i$. Time is in seconds.

news20, where (11) is quite accurate. These two sparse document sets seem to have independent features (word occurrence) so that the assumption for (11) holds. For data with known structures, we demonstrate that a better estimate than (11) can be achieved. The problem a9a contains 14 groups of features[3] and each group contains several exclusive binary features (e.g., age in various ranges). Within each group, $x_r x_s = 0$ if $r \neq s$, so an upper bound of $\mathbf{w}$'s number of non-zero elements is

$$C(n+2,2) - \sum_{\text{feature groups}} C(\#\text{features in each group}, 2). \tag{16}$$

We show in Table 2 that (16) is closer to the actual number. The situation for two other problems (ijcnn1 and covtype) is similar.

As storing news20's non-zero $\mathbf{w}$ elements requires more memory than our machine's capacity, we do not include this set in subsequent experiments.

4.3 Calculating or Storing $\phi(\mathbf{x}_i)$

In Section 3.4, we discuss three methods to handle $\phi(\mathbf{x}_i)$. Table 3 compares the first/second and the third methods. We select C and kernel parameters by a validation procedure and present the training time of using optimal parameters. For the first/second methods, we count CPU time after all $\phi(\mathbf{x}_i)$ have been loaded/generated. For the third, we show CPU time after loading all $\mathbf{x}_i$, as this method calculates $\phi(\mathbf{x}_i), \forall i$ in the middle of the training procedure.

Table 3 lists $\hat{n}/\bar{n}$ to show the ratio between two methods' memory consumption on storing $\phi(\mathbf{x}_i)$ and $\mathbf{x}_i$, $\forall i$. In the same table we present each method's number of L2 cache misses and training time. The number of L2 cache misses is obtained by the simulation tool cachegrind in valgrind.[5] The method by calculating $\phi(\mathbf{x}_i)$ is faster in all cases. Moreover, it has a smaller number of L2 cache misses. Data can be more easily located in the cache when $\phi(\mathbf{x}_i)$ is not stored. We consider the method of calculating $\phi(\mathbf{x}_i)$ for subsequent experiments in this section.

3. See descriptions in the beginning of each file from `http://research.microsoft.com/en-us/um/people/jplatt/adult.zip`.
4. For webspam, as $\phi(\mathbf{x}_i)$ $\forall i$ require more memory than what our machine has, we use single precision floating-point numbers to store the data. All other experiments in this work use double precision.
5. cachegrind can be found at `http://valgrind.org/`.

Data set	Linear (LIBLINEAR)			RBF (LIBSVM)			
	C	Time (s)	Accuracy	C	γ	Time (s)	Accuracy
a9a	32	5.4	84.98	8	0.03125	98.9	85.03
real-sim	1	0.3	97.51	8	0.5	973.7	97.90
ijcnn1	32	1.6	92.21	32	2	26.9	98.69
MNIST38	0.03125	0.1	96.82	2	0.03125	37.6	99.70
covtype	0.0625	1.4	76.35	32	32	54,968.1	96.08
webspam	32	25.5	93.15	8	32	15,571.1	99.20

Table 4: Comparison of linear SVM and nonlinear SVM with RBF kernel. Time is in seconds.

Data set	Degree-2 Polynomial					Accuracy diff.	
	C	γ	Training time (s) LIBLINEAR	Training time (s) LIBSVM	Accuracy	Linear	RBF
a9a	8	0.03125	1.6	89.8	85.06	0.07	0.02
real-sim	0.03125	8	59.8	1,220.5	98.00	0.49	0.10
ijcnn1	0.125	32	10.7	64.2	97.84	5.63	−0.85
MNIST38	2	0.3125	8.6	18.4	99.29	2.47	−0.40
covtype	2	8	5,211.9	NA	80.09	3.74	−15.98
webspam	8	8	3,228.1	NA	98.44	5.29	−0.76

Table 5: Training time (in seconds) and testing accuracy of using the degree-2 polynomial mapping. The last two columns show the accuracy difference to results using linear and RBF. NA indicates that programs do not terminate after 300,000 seconds.

4.4 Accuracy and Time of Using Linear, Degree-2 Polynomial, and RBF

We compare training time, testing time, and testing accuracy of using three mappings: linear, degree-2 polynomial, and RBF. We use LIBLINEAR for linear, LIBSVM for RBF, and both for degree-2 polynomial. For each data set, we choose parameters C and γ by a five-fold cross validation on a grid of points. The best (C,γ) are then used to train the whole training set and obtain the testing accuracy. To reduce the training time, LIBSVM allocates some memory space, called kernel cache, to store recently used kernel elements. In contrast, LIBLINEAR does not require this space. All it needs is to store $\mathbf{w}$. In this work, we run LIBSVM using 1 GBytes of kernel cache.

Using linear and RBF mappings, Table 4 presents the training time, testing accuracy, and the correspondent parameters. Linear and RBF have similar testing accuracy on data sets a9a and real-sim. The set real-sim contains document data with many features. Linear classifiers have been observed to perform well on such data with much less training time. For other data sets, the testing accuracy of using linear is clearly inferior to that of using RBF. Degree-2 polynomial mappings may be useful for these data. We can possibly improve the accuracy over linear while achieving faster training time than RBF.

We then explore the performance of the degree-2 polynomial mapping. The first part of Table 5 shows the training time, testing accuracy, and optimal parameters using LIBLINEAR. As a com-

Data set	LIBLINEAR		LIBSVM	
	linear	degree-2	degree-2	RBF
a9a	0.00	0.01	19.28	32.42
real-sim	0.02	1.13	107.67	84.52
ijcnn1	0.02	0.07	14.07	20.38
MNIST38	0.00	0.12	2.41	5.76
covtype	0.03	0.09	NA	998.68
webspam	0.05	1.14	NA	846.77

Table 6: Testing time (in seconds) using decomposition methods for linear and nonlinear SVM. Parameters in Tables 4 and 5 are used to build SVM models for prediction. NA: SVM models are not available due to lengthy training time (see Table 5).

parison, we run LIBSVM with the same parameters and report training time.[6] Table 5 also presents the testing accuracy difference between degree-2 polynomial and linear/RBF. It is observed that for nearly all problems, the performance of the degree-2 polynomial mapping can compete with RBF, while for covtype, the performance is only similar to the linear mapping. Apparently, a degree-2 mapping does not give rich enough information to separate data in covtype.

Regarding the training time, LIBLINEAR with degree-2 polynomial mappings is faster than LIBSVM with RBF. Therefore, the proposed method may achieve fast training, while preserving some benefits of nonlinear mappings. Next, we compare the training time between LIBLINEAR and LIBSVM when the same degree-2 polynomial mapping is used. From Table 5, LIBLINEAR is much faster than LIBSVM. Thus for applications needing to use low-degree polynomial kernels, the training time can be significantly reduced.

We present testing time in Table 6. The explicit mapping approach is much faster as it calculates only $\mathbf{w}^T\mathbf{x}$ or $\mathbf{w}^T\phi(\mathbf{x})$.

4.5 L1-regularized SVM with Linear and Degree-2 Polynomial Mappings

Recently, L1-regularized SVM has gained attention because it can produce a sparse model (see, for example, the survey by Yuan et al., 2009, and references therein). An L1-regularized SVM[7] solves

$$\min_{\mathbf{w}} \quad \|\mathbf{w}\|_1 + C\sum_{i=1}^{l} \max(1 - y_i\mathbf{w}^T\phi(\mathbf{x}_i), 0)^2, \tag{17}$$

where $\|\cdot\|_1$ denotes the 1-norm. As discussed in Section 3.4, after a degree-2 polynomial mapping the number of features may be very large. A sparse $\mathbf{w}$ reduces the memory consumption. In this section, we conduct a preliminary investigation on training degree-2 polynomial mappings via (17).

Due to the non-differentiable term $\|\mathbf{w}\|_1$, optimization techniques for (17) are different from those for L2-regularized SVM. If we pre-compute $\phi(\mathbf{x}_i), \forall i$ (i.e., methods 1 and 2 in Section 3.4 for handling $\phi(\mathbf{x}_i)$), then any optimization technique for (17) can be directly applied. Recall that Section 4.3 shows that method 3 (calculating $\phi(\mathbf{x}_i)$ in the training algorithm) is faster if n is not large. We show an interesting example where this method significantly increases the number of operations. In

6. LIBSVM solves SVM with bias b, but LIBLINEAR solves (1). As the difference is minor, we run LIBSVM with the same parameters for LIBLINEAR. Moreover, the testing accuracy of LIBSVM is almost the same as LIBLINEAR.

7. We consider L2-loss in (17) by following Yuan et al. (2009).

Data set	Linear			Degree-2 polynomial				
	Time (s)	Sparsity (%)	Accuracy (%)	Time (s): $\phi(\mathbf{x}_i)$ is stored	Time (s): $\phi(\mathbf{x}_i)$ is calculated	Sparsity (%)	Accuracy (%)	L2 SVM Sparsity
a9a	0.73	83.74	85.00	3.94	19.33	10.43	85.10	71.74
real-sim	1.06	25.16	97.04	524.54	2,288.27	0.01	97.43	26.17
ijcnn1	0.86	100.00	91.79	9.17	18.09	83.70	97.59	83.70
MNIST38	0.86	55.85	96.93	38.11	81.10	0.23	99.50	54.23
covtype	60.07	98.15	75.66	70.13	2,196.08	39.22	79.73	43.44
webspam	47.08	38.98	92.55	772.92	1,296.40	12.60	98.32	28.96

Table 7: L1-regularized SVM: a comparison between linear and degree-2 polynomial mappings. Time is in seconds. Sparsity is the percentage of non-zero elements in $\mathbf{w}$. For the degree-2 polynomial mapping, we show the training time of both calculating and storing $\phi(\mathbf{x})$.

Yuan et al. (2009), a primal decomposition (coordinate descent) method is considered the fastest for solving (17). It updates one element of $\mathbf{w}$ at a time. If $\mathbf{w}$ is the current solution and the jth element is selected, the following one-variable problem is minimized:

$$\min_d \quad |w_j + d| + C\sum_{i=1}^{l} \max\left(1 - y_i \mathbf{w}^T \phi(\mathbf{x}_i) - y_i d\phi(\mathbf{x}_i)_j, 0\right)^2.$$

Assume $\phi(\mathbf{x})_j$ involves $x_r x_s$. To obtain all $\phi(\mathbf{x}_i)_j, \forall i$, we must go through non-zero elements of the rth and the sth features of the original data. This operation costs $O(\bar{l})$, where $\bar{l}$ is the average number of non-zero elements per feature of $\mathbf{x}_i$, $\forall i$. However, the expected number of non-zero elements of $\phi(\mathbf{x}_i)_j, \forall i$ is only

$$\left(\bar{l}/l\right)^2 \cdot l = \bar{l}^2/l.$$

If data are sparse, $\bar{l}^2/l$ is much smaller than $\bar{l}$. Therefore, the cost by using methods 1 and 2 to pre-compute $\phi(\mathbf{x}_i), \forall i$ is less than method 3. This is mainly because the primal coordinate descent approach accesses data in a feature-based way. The sparse patterns of two features are needed. In contrast, decomposition methods used earlier for solving the dual problem of L2-regularized SVM is instance-based. To obtain $x_r x_s$, one needs only the sparse pattern of an instance $\mathbf{x}$. Some optimization approaches discussed in Yuan et al. (2009) for (17) are instance-based. An interesting future study would be to investigate their performances.

We extend a primal decomposition implementation for (17) in LIBLINEAR to handle degree-2 polynomial mappings. We use default settings (e.g., stopping tolerance) in LIBLINEAR. Table 7 compares linear and degree-2 polynomial mappings by showing training time, $\mathbf{w}$'s sparsity, and testing accuracy. For the training time of using degree-2 polynomials, we present results by storing and calculating $\phi(\mathbf{x}_i)$. Clearly, calculating $\phi(\mathbf{x}_i)$ is much slower, a result consistent with our analysis. The training time for real-sim is much longer than that in Table 5 (L2-regularized SVM). This result is due to the huge number of variables in solving the primal problem (17). There are some tricks to improve the training speed for this problem though we do not get into details. For sparsity, we also show the result using L2-regularized SVM as a comparison.[8] L1-regularized SVM gives excellent

8. The sparsity of L2-regularized SVM is in fact the column of "real" numbers of non-zero elements in Table 2 divided by the dimensionality $C(n+2,2)$.

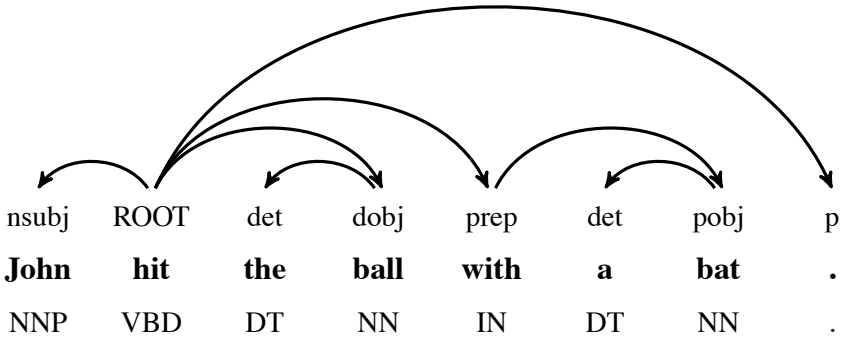


Figure 1: A dependency graph with arc labels and part-of-speech tags for a sentence.

sparsity for some problems. For example, MNIST38 has $n = 752$ features. By solving (17) with linear mapping, 420 features remain. If using a degree-2 polynomial mapping, the dimensionality is $C(n+2,2) = 283,881$. Solving an L2-regularized SVM gives a $\mathbf{w}$ with 153,564 non-zero elements, but using L1 regularization $\mathbf{w}$ has only a very small number of 650 non-zero elements. Finally, for testing accuracy, results are similar to (or slightly lower than) those in Tables 4-5.

Due to the nice sparsity results, L1 regularization for low-degree polynomial mappings may be a promising future direction.

5. An NLP Application: Data-driven Dependency Parsing

In this section we study a real-world natural language processing (NLP) task on dependency parsing. Given a sentence, a dependency graph represents each word and its syntactic modifiers through labeled directed edges. Figure 1 shows an example. Data-driven dependency parsing is a common method to construct dependency graphs. Different from grammar-based parsing, it learns to produce the dependency graph solely by using the training data. Data-driven dependency parsing has become popular because it is chosen as the shared task at CONLL-X[9] and CONLL2007.[10] More information about dependency parsing can be found in, for example, McDonald and Nivre (2007).

Dependency parsing appears in many online NLP applications. In such cases testing (parsing) speed is very important. We will see that this requirement for testing speed makes our approach very useful for this application.

5.1 A Multi-class Problem

We study a transition-based parsing method proposed by Nivre (2003). The parsing algorithm builds a labeled dependency graph in one left-to-right pass over the input with a stack to store partially processed tokens. At each step, we need to decide which transition to perform. As in Nivre et al. (2007), we use the following transitions:

- SHIFT: Pushes the next input token to the top of the stack and advances to the next input token.
- REDUCE: Pops the top element from the stack.

9. CONLL-X can be found at `http://nextens.uvt.nl/~conll/`.
10. CONLL2007 can be found at `http://nextens.uvt.nl/depparse-wiki/SharedTaskWebsite`.

input(1).tag	stack.tag	stack.leftmost-child.label
input(2).tag	stack(1).tag	stack.rightmost-child.label
input(3).tag	stack.label	input.leftmost-child.label
input.word	stack.word	
input(1).word	stack.head.word	

Table 8: Feature types used by the dependency parser.

- LEFT-ARC(r): Adds an arc with label r from the next input token to the token on top of the stack and pops the top element off the stack.

- RIGHT-ARC(r): Adds an arc with label r from the top token on the stack to the next input token. Then pushes the current input token to the stack and advances to the next input token.

The parser decides the next transition by extracting features from the current parse state. The parse state consists of the stack, the remaining input tokens, and the partially built dependency graph. We use the standard method for converting symbolic features into numerical features by binarization. For each feature type F (see Table 8), that has value v in the current state, we generate a feature predicate, $F = v$. These feature predicates are used as binary features in the classifier. It is this expansion of feature types to binary feature predicates that leads to the large number of features in the classifiers. Especially the lexicalized (i.e., word-based) features generate large numbers of sparse features.

Thus the core of the parser is a multi-class classification problem, which maps features to transitions. Nivre et al. (2006) use LIBSVM with a degree-2 polynomial kernel to train the multi-class classification problems, and get good results at CONLL-X.

In this experiment, we use data from the English Penn Treebank (Marcus et al., 1993). The treebank is converted to dependency format using Penn2Malt,[11] and the data is split into sections 02–21 for training and section 23 for testing. During training we construct a canonical transition sequence from the dependency graph of each sentence in the training corpus, adopting an arc-eager approach for disambiguation. For each transition, we extract the features in Table 8 from the current parse state, and use this for training the classifiers.

When parsing a sentence, the classifiers are used for predicting the next transition, based on the features extracted from the current parse state. When all the input tokens have been processed the dependency graph is extracted from the transition sequence.

In order to reduce training time the data is split into multiple sets. For example, if a feature j takes two values a and b, we can divide the training data into $\{\mathbf{x} \mid x_j = a\}$ and $\{\mathbf{x} \mid x_j = b\}$, and get two models M^a and M^b. Then in the prediction phase, we decide to use M^a or M^b according to x_j of the testing instance. Yamada and Matsumoto (2003) mention that applying this method reduces the training time without a significant loss in accuracy. We divide the training data into 125 smaller training sets according to the part-of-speech tag of the current input token. Also, the label for RIGHT-ARC and LEFT-ARC transitions is predicted separately from the transition. The number of classes ranges from 2 to 12. Table 9 lists the statistics of the largest multi-class problem among 125.

11. See `http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html`

n	$\bar{n}$	l	#nz
46,155	13.3	294,582	3,913,845

Table 9: Summary of the dependency parsing data set. We show statistics of the largest problem among the 125 sets divided from the training data. The column #nz ($= l\bar{n}$) indicates the total number of non-zero feature values in the training set.

5.2 Implementations

We consider the degree-2 polynomial mapping in (5). Since the original $\mathbf{x}_i, \forall i$ have 0/1 feature values, we use (5) instead of (4) to preserve this property.[12] In our implementation, by extending LIBLINEAR, 0/1 values are still stored as double-precision numbers. However, for large data sets, we can save memory by storing only non-zero indices. Due to using (5), the only parameter is C.

The dimensionality of using the degree-2 polynomial mapping is huge. We discussed in Section 3.4 that 20 GBytes memory is needed to store a dense $\mathbf{w}$. Assume the "one-against-the rest" approach is applied for multi-class classification. We need 125 $\times$ (# classes) vectors of $\mathbf{w}$ in the prediction (parsing) stage as training data are separated into 125 sets. Obviously we do not have enough memory for them. As the data are very sparse, the actual number of non-zero elements in $\mathbf{w}$ is merely 1,438,456 (considering the largest of the 125 training sets). Only these non-zero features in $\phi(\mathbf{x})$ and $\mathbf{w}$ are needed in training/testing, so the memory issue is solved. In the practical implementation, we construct a hash table to collect non-zero features of $\phi(\mathbf{x}_i), i = 1, \ldots, l$ as a new set for training.[13] In prediction, we use the same hash map to calculate $\mathbf{w}^T\phi(\mathbf{x})$. This implementation corresponds to the first/second methods discussed in Section 3.4. See more details in the end of Section 3.4.

Other settings (e.g., stopping tolerance and LIBSVM's kernel cache) are the same as those in Section 4. LIBSVM uses the "one-against-one" approach for training multi-class problems, while LIBLINEAR uses "one-against-the rest." We use a dependency parsing system at Google, which calls LIBSVM/LIBLINEAR for training/testing. Parts of this experiment were performed when some authors worked at Google.

As the whole parsing system is quite complex, we have not conducted a complete parameter optimization. Instead, we have roughly tuned each parameter to produce good results.

5.3 Experiments

We compare two approaches. The first uses kernels, while the second does not.

- LIBSVM: RBF and degree-2 polynomial kernels.
- LIBLINEAR: linear mapping (i.e., the original input data) and the degree-2 polynomial mapping via (5).

12. In fact, if using (4), we can still use some ways so that $\sqrt{2}$ is not stored. However, the implementation is more complicated.
13. For a fair comparison, the reported training time includes time for this pre-processing stage.

	LIBSVM		LIBLINEAR	
	RBF	Poly ($d=2$)	Linear	Poly: Eq. (5)
Parameters	$C=0.5$ $1/(2\sigma^2)=0.18$	$C=0.5$ $\gamma=0.18$ $r=0.3$	$C=0.5$	$C=0.05$
Training time	3h34m53s	3h21m51s	3m36s	3m43s
Parsing speed	0.7x	1x	1652x	103x
UAS	89.92	91.67	89.11	91.71
LAS	88.55	90.60	88.07	90.71

Table 10: Accuracy, training time, and parsing speed (relative to LIBSVM with polynomial kernel) for the dependency parsing.

Table 10 lists parameters for various kernels, training/testing time, and testing accuracy. Training and testing are done using gold standard part-of-speech tags, and only non-punctuation tokens are used for scoring. The accuracy of dependency parsing is measured by two evaluation metrics:

1. Labeled attachment score (LAS): the percentage of tokens with correct dependency head and dependency label.

2. Unlabeled attachment score (UAS): the percentage of tokens with correct dependency head.

For LIBSVM the polynomial kernel gives better accuracy than the RBF kernel, consistent with previous observations, that polynomial mappings are important for parsing (Kudo and Matsumoto, 2000; McDonald and Pereira, 2006; Yamada and Matsumoto, 2003; Goldberg and Elhadad, 2008). Moreover, LIBSVM using degree-2 polynomial kernel produces better results in terms of UAS/LAS than LIBLINEAR using just a linear mapping of features. However, parsing using LIBSVM is slow compared to LIBLINEAR. We can speed up parsing by a factor of 1,652 with only a 2.5% drop in accuracy. With a degree-2 polynomial mapping (5), we achieve UAS/LAS results similar to LIBSVM, while still maintaining high parsing speed, 103 times faster than LIBSVM.

From Table 10, training LIBLINEAR is a lot faster than LIBSVM. This large reduction in training time allows us to easily conduct experiments and improve the settings. Some may criticize that the comparison on training time is not fair as LIBSVM uses "one-against-one" for multi-class classification, while LIBLINEAR uses "one-against-the rest." It is known (e.g., Hsu and Lin, 2002) that for nonlinear SVM, LIBSVM with "one-against-one" is faster than "one-against-the rest." Thus even if we modify LIBSVM to perform "one-against-the rest," its training is still much slower than LIBLINEAR.

5.4 Related Work

Earlier works have improved the testing speed of SVM with low-degree polynomial kernels. Most of them target natural language processing (NLP) applications. Isozaki and Kazawa (2002) propose an approach similar to obtaining $\mathbf{w}$ by the expression in (8).[14] A direct implementation of their

14. They do not really form $\mathbf{w}$, but their result by expanding the degree-2 polynomial kernel leads to something very similar.

method requires a memory space as large as the dimensionality of $\mathbf{w}$, but we cannot afford such a space for our application. Kudo and Matsumoto (2003) consider the expression of $\mathbf{w}$ in (8) and propose an approximate prediction scheme using only a sub-vector of $\mathbf{w}$. Their method is useful for data with 0/1 features. Goldberg and Elhadad (2008) propose speeding up the calculation of low-degree polynomial kernels by separating features to rare and common ones. Goldberg and Elhadad's approach is motivated by some observations of NLP data. It avoids the memory problem, but the effectiveness on general data is not clear yet.

The above existing works focus on improving testing speed. They suffer from the slow training of using traditional SVM solvers. For example, Kudo and Matsumoto (2000) mention that "the experiments . . . have actually taken long training time," so they must select a subset using properties of dependency parsing. Our approach considers linear SVM on explicitly mapped data, applies state of the art training techniques, and can simultaneously achieve fast training and testing.

6. Discussions and Conclusions

Past research has shown that SVM using linear and highly nonlinear mappings of data has the following properties:

Linear	Highly nonlinear
Fast training/testing	Slow training/testing via kernels
Low accuracy	High accuracy

Many have attempted to develop techniques in the between. Most start from the nonlinear side. They propose methods to manipulate the kernels (e.g., Lee and Mangasarian, 2001; Keerthi et al., 2006). In contrast, ours is from the linear side. The strategy is simple and requires only minor modifications of existing packages for linear SVM.

This work focuses on the degree-2 polynomial mapping. An interesting future study is the efficient implementation for degree-3 mappings. Considering other mapping functions to expand data vectors could be investigated as well. As kernels are not used, we might have a greater flexibility to design the mapping function.

Table 2 shows that storing $\mathbf{w}$ may require a huge amount of memory. For online training, some (e.g., Langford et al., 2009) have designed feature hashing techniques to control the memory use of $\mathbf{w}$. Recently, feature hashing has been popular for projecting a high dimensional feature vector to a lower dimensional one (e.g., Weinberger et al., 2009; Shi et al., 2009). For certain sequence data, one can consider n-gram (i.e., n consecutive features) instead of general polynomial mappings. Then not only the number of features becomes smaller, but also controlling $\mathbf{w}$'s sparsity is easier. Existing experiments on document data can be found in, for example, Ifrim et al. (2008) and Shi et al. (2009).

We successfully apply the proposed procedure to an NLP application. It has certain requirements on the training and testing speed, but we also hope to achieve better testing accuracy. The proposed procedure is very useful for applications of this type.

Acknowledgments

The authors thank Aditya Menon, Ming-Wei Chang, anonymous reviewers, and associate editor for helpful comments. They also thank Naomi Bilodeau for proofreading the paper. This work was supported in part by the National Science Council of Taiwan via the grant 98-2221-E-002-136-MY3.

Appendix A. Connection Between (12) and (13)

We prove the result by showing that the dual optimization problems of using (12) and (13) are the same. Since

$$(\bar{\gamma}\mathbf{x}_i^T\mathbf{x}_j + r)^d = r^d\left(\frac{\bar{\gamma}}{r}\mathbf{x}_i^T\mathbf{x}_j + 1\right)^d,$$

we have

$$\bar{Q}_{ij} = y_i y_j \bar{K}(\mathbf{x}_i, \mathbf{x}_j) = r^d Q_{ij}.$$

The dual optimization problem of using $\bar{Q}$ can be written as

$$\begin{aligned} \min_{\bar{\alpha}} \quad & \frac{1}{r^d}\left(\frac{1}{2}(r^d\bar{\alpha})^T Q(r^d\bar{\alpha}) - \mathbf{e}^T r^d\bar{\alpha}\right) \\ \text{subject to} \quad & 0 \le r^d\bar{\alpha}_i \le r^d\bar{C}, \quad i = 1\ldots, l. \end{aligned}$$

Using $\alpha = r^d\bar{\alpha}$ and $C = r^d\bar{C}$, this problem becomes the dual problem when using Q.

References

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.

Leon Bottou. Stochastic gradient descent examples, 2007. `http://leon.bottou.org/projects/sgd`.

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`.

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`.

Michael Ferris and Todd Munson. Interior point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783–804, 2003.

E. Michael Gertz and Joshua D. Griffin. Support vector machine classifiers for large data sets. Technical Report ANL/MCS-TM-289, Argonne National Laboratory, 2005.

Yoav Goldberg and Michael Elhadad. splitSVM: Fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications. In *Proceedings of the 46st Annual Meeting of the Association of Computational Linguistics (ACL)*, 2008.

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`.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

Georgiana Ifrim, Gökhan Bakır, and Gerhard Weikum. Fast logistic regression for text categorization with variable-length n-grams. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 354–362, 2008.

Hideki Isozaki and Hideto Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of COLING*, pages 390–396, 2002.

Thorsten Joachims. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

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*, Cambridge, MA, 1998. MIT Press.

Jin Hyuk Jung, Dianne P. O'Leary, and André L. Tits. Adaptive constraint reduction for training support vector machines. *Electronic Transactions on Numerical Analysis*, 31:156–177, 2008.

S. Sathiya Keerthi and Chih-Jen Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 15(7):1667–1689, 2003.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13:637–649, 2001.

S. Sathiya Keerthi, Olivier Chapelle, and Dennis DeCoste. Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, 7:1493–1515, 2006.

Taku Kudo and Yuji Matsumoto. Japanese dependency structure analysis based on support vector machines. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, 2000.

Taku Kudo and Yuji Matsumoto. Fast methods for kernel-based text analysis. In *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics (ACL)*, 2003.

John Langford, Lihong Li, and Tong Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:771–801, 2009.

Yuh-Jye Lee and Olvi L. Mangasarian. RSVM: Reduced support vector machines. In *Proceedings of the First SIAM International Conference on Data Mining*, 2001.

Ross A. Lippert and Ryan M. Rifkin. Infinite-σ limits for Tikhonov regularization. *Journal of Machine Learning Research*, 7:855–876, 2006.

Olvi L. Mangasarian and David R. Musicant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5):1032–1037, 1999.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.

Ryan McDonald and Joakim Nivre. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the Joint Conferences on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

Ryan McDonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 81–88, 2006.

Yvonne Moh and Joachim M. Buhmann. Kernel expansion for online preference tracking. In *Proceedings of The International Society for Music Information Retrieval (ISMIR)*, pages 167–172, 2008.

Joakim Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, 2003.

Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*, pages 221–225, 2006.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gulsen Eryigit, Sandra Kubler, Svetoslav Marinov, and Erwin Marsi. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135, 2007.

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.

Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alex Smola, and S.V.N. Vishwanathan. Hash kernels for structured data. *Journal of Machine Learning Research*, 10:2615–2637, 2009.

Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the Twenty Sixth International Conference on Machine Learning (ICML)*, pages 1113–1120, 2009.

Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, 2003.

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. 2009. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/l1.pdf`. Under revision for Journal of Machine Learning Research.

Journal of Machine Learning Research 11 (2010) 1491-1516 Submitted 6/09; Revised 11/09; Published 4/10

Quadratic Programming Feature Selection

Irene Rodriguez-Lujan IRENE.RODRIGUEZ@IIC.UAM.ES
Departamento de Ingeniería Informática and IIC
Universidad Autónoma de Madrid
28049 Madrid, Spain

Ramon Huerta RHUERTA@UCSD.EDU
BioCircuits Institute
University of California, San Diego
La Jolla, CA 92093-0402, USA

Charles Elkan ELKAN@CS.UCSD.EDU
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404, USA

Carlos Santa Cruz CARLOS.SANTACRUZ@IIC.UAM.ES
Departamento de Ingeniería Informática and IIC
Universidad Autónoma de Madrid
28049 Madrid, Spain

Editor: Lyle Ungar

Abstract

Identifying a subset of features that preserves classification accuracy is a problem of growing importance, because of the increasing size and dimensionality of real-world data sets. We propose a new feature selection method, named Quadratic Programming Feature Selection (QPFS), that reduces the task to a quadratic optimization problem. In order to limit the computational complexity of solving the optimization problem, QPFS uses the Nyström method for approximate matrix diagonalization. QPFS is thus capable of dealing with very large data sets, for which the use of other methods is computationally expensive. In experiments with small and medium data sets, the QPFS method leads to classification accuracy similar to that of other successful techniques. For large data sets, QPFS is superior in terms of computational efficiency.

Keywords: feature selection, quadratic programming, Nyström method, large data set, high-dimensional data

1. Introduction

The task of feature selection is to reduce the number of variables used in training a classifier. Three main benefits can be drawn from successful feature selection: first, a substantial gain in computational efficiency (especially important for any application that requires classifier execution in real-time); second, scientific discovery by determining which features are most correlated with the class labels (which may in turn reveal unknown relationships among features); and, third, reduction of the risk of overfitting if too few training instances are available (a serious problem particularly in situations with high dimensionalities relative to training set sizes). Document categorization (Forman, 2008), prosthesis control (Momen et al., 2007; Shenoy et al., 2008), cardiac arrhythmia classifica-

tion (Rodriguez et al., 2005), fMRI analysis, gene selection from microarray data (Ding and Peng, 2005; Li et al., 2004; Zhang et al., 2008), real-time identification of polymers (Leitner et al., 2003), and credit card fraud detection are some real-life domains where these gains are especially meaningful.

Many methods have been suggested to solve the variable selection problem. They can be categorized into three groups. *Filter* methods perform feature selection that is independent of the classifier (Bekkerman et al., 2003; Forman, 2003, 2008). *Wrapper* methods use search techniques to select candidate subsets of variables and evaluate their fitness based on classification accuracy (John et al., 1994; Kohavi and John, 1997; Langley, 1994). Finally, *embedded* methods incorporate feature selection in the classifier objective function or algorithm (Breiman et al., 1984; Weston et al., 2001).

Filter methods are often preferable to other selection methods because of their usability with alternative classifiers, their computational speed, and their simplicity (Guyon, 2003; Yu and Liu, 2003). But filter algorithms often score variables separately from each other, so they do not achieve the goal of finding combinations of variables that give the best classification performance. It has been shown that simply combining good variables does not necessary lead to good classification accuracy (Cover, 1974; Cover and Thomas, 1991; Jain et al., 2000). Therefore, one common improvement direction for filter algorithms is to consider dependencies among variables. In this direction approaches based on mutual information, in particular Maximal Dependency (MaxDep) and minimal-Redundancy-Maximum-Relevance (mRMR), have been significant advances (Peng et al., 2005).

The central idea of the MaxDep approach is to find a subset of features which jointly have the largest dependency on the target class. However, it is often infeasible to compute the joint density functions of all features and of all features with the class. One approach to making the MaxDep approach practical is Maximal Relevance (MaxRel) feature selection (Peng et al., 2005). This approach selects those features that have highest relevance (mutual information) to the target class. The main limitation of MaxRel is not accounting for redundancy among features. The mRMR criterion is another version of MaxDep that chooses a subset of features with both minimum redundancy (approximated as the mean value of the mutual information between each pair of variables in the subset) and maximum relevance (estimated as the mean value of the mutual information between each feature and the target class). Given the prohibitive cost of considering all possible subsets of features, the mRMR algorithm selects features greedily, minimizing their redundancy with features chosen in previous steps and maximizing their relevance to the class.

The new method proposed in this paper aims at dealing with very large data sets with high dimensionality providing a time complexity improvement respect to current methods.We show how to build on well-established mathematical methods to reduce time and space complexity. The new method is named Quadratic Programming Feature Selection (QPFS) because it is based on efficient quadratic programming (Bertsekas, 1999). We introduce an objective function with quadratic and linear terms. The quadratic term captures the dependence (that is, similarity, correlation, or mutual information) between each pair of variables, whereas the linear term captures the relationship between each feature and the class label. For large data sets, solving a quadratic programming problem can have high time and space complexity. Therefore, we show how to reformulate the optimization problem in a lower dimensional subspace using the Nyström method for matrix diagonalization (Fowlkes et al., 2001). The Nyström approximation allows the variables to be sampled, without losing much information but with a great improvement in the speed of the algorithm. Ex-

perimental results show that the QPFS method achieves accuracy similar to that of other methods on medium-size data sets, while on the well-known large MNIST data set, QPFS is more efficient than its predecessors.

The present manuscript is organized as follows. Section 2 presents the QPFS algorithm, including the Nyström approximation, error estimation, theoretical complexity, and implementation issues. Section 3 provides a description of data sets, and experimental results in terms of classification accuracy and running time.

2. The QPFS Algorithm

Our goal is to develop a feature selection method capable of succeeding with very large data sets. To achieve this goal, our first contribution is a novel formulation of the task. The new formulation uses quadratic programming, a methodology that has previously been successful for a broad range of other quite different applications (Bertsekas, 1999). Assume the classifier learning problem involves N training samples and M variables (also called attributes or features). A quadratic programming problem is to minimize a multivariate quadratic function subject to linear constraints as follows:

$$\min_x \left\{ \frac{1}{2} x^T Q x - F^T x \right\} . \tag{1}$$

Above, x is an M-dimensional vector, $Q \in \mathbb{R}^{M \times M}$ is a symmetric positive semidefinite matrix, and F is a vector in $\mathbb{R}^M$ with non-negative entries. Applied to the feature selection task, Q represents the similarity among variables (redundancy), and F measures how correlated each feature is with the target class (relevance).

After the quadratic programming optimization problem has been solved, the components of x represent the weight of each feature. Features with higher weights are better variables to use for subsequent classifier training. Since x_i represents the weight of each variable, it is reasonable to enforce the following constraints:

$$\begin{aligned} x_i &\geqslant 0 \text{ for all } i = 1, \ldots, M \\ \sum_{i=1}^{M} x_i &= 1 . \end{aligned}$$

Depending on the learning problem, the quadratic and linear terms can have different relative purposes in the objective function. Therefore, we introduce a scalar parameter α as follows:

$$\min_x \left\{ \frac{1}{2}(1-\alpha) x^T Q x - \alpha F^T x \right\} \tag{2}$$

where x, Q and F are defined as before and $\alpha \in [0,1]$. If $\alpha = 1$, only relevance is considered; the quadratic programming problem becomes linear and equivalent to the MaxRel criterion. On the contrary, if $\alpha = 0$, then only independence between features is considered that is, features with higher weights are those which have lower similarity coefficients with the rest of features. Every data set has its best choice of α to extract the minimum number of features for classification purposes. Nevertheless, a reasonable choice of α must balance the linear and quadratic terms of Equation 2. Thus, we estimate the mean value $\bar{q}$ of the elements of the matrix Q and on the mean value $\bar{f}$ of the

elements of the vector F as

$$\bar{q} = \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M} q_{ij} ,$$
$$\bar{f} = \frac{1}{M}\sum_{i=1}^{M} f_i .$$

Since the relevance and redundancy terms in Equation 2 are balanced when $(1-\hat{\alpha})\bar{q} = \hat{\alpha}\bar{f}$, a reasonable initial estimate of α is

$$\hat{\alpha} = \frac{\bar{q}}{\bar{q}+\bar{f}} .$$

The goal of balancing both terms in the QPFS objective function, Equation 2, is to ensure that both redundancy and relevance are taken into account. If features are only slightly redundant, that is, they have low correlation with each other, then the linear term in Equation 1 is dominant: $\bar{f} \gg \bar{q}$. Making α small reduces this dominance. On the other hand, if the features have a high level of redundancy relative to relevance ($\bar{q} \gg \bar{f}$), then the quadratic term in Equation 1 can dominate the linear one. In this case, overweighting the linear term (α close to 1) makes the objective function be balanced.

Experimental results in Section 3 show that using $\hat{\alpha}$ leads to good results. Alternatively, it is possible to use a validation subset to determine an appropriate value for α. However, that approach requires evaluating the accuracy of the underlying classifier for each point in a grid of α values. In this case, QPFS would become a wrapper feature selection method instead of a filter method because it would need the classifier accuracy to determine the proper value of α.

2.1 Similarity Measures

One advantage of the problem formulation above is that it is sufficiently general to permit any symmetric similarity measure to be used. In the remainder of this paper, the Pearson correlation coefficient and mutual information are chosen, because they are conventional and because they are representative ways to measure similarity.

The Pearson correlation coefficient is simple and has been shown to be effective in a wide variety of feature selection methods, including correlation based feature selection (CFS) (Hall, 2000) and principal component analysis (PCA) (Duda et al., 2000). Formally, the Pearson correlation coefficient is defined as

$$\rho_{ij} = \frac{cov(v_i, v_j)}{\sqrt{var(v_i)var(v_j)}}$$

where *cov* is the covariance of variables and *var* is the variance of each variable. The sample correlation is calculated as

$$\hat{\rho}_{ij} = \frac{\sum_{k=1}^{N}(v_{ki}-\bar{v}_i)(v_{kj}-\bar{v}_j)}{\sqrt{\sum_{k=1}^{N}(v_{ki}-\bar{v}_i)^2 \sum_{k=1}^{N}(v_{kj}-\bar{v}_j)^2}} \tag{3}$$

where N is the number of samples, v_{ki} is the k-th sample of random variable v_i, and $\bar{v}_i$ is the sample mean of the random variable v_i.

Each matrix element q_{ij} is defined to be the absolute value of the Pearson correlation coefficient of the pair of variables v_i and v_j, that is, $q_{ij} = |\hat{\rho}_{ij}|$. Suppose a classifier learning problem with C classes, the relevance weight of variable v_i, F_i, is computed using a modified correlation coefficient (Hall, 2000) which is an extension to the C-class classification scenario. The modified definition is

$$F_i = \sum_{k=1}^{C} \hat{p}(K=k)|\hat{\rho}_{iC_k}|$$

where K is the target class variable, C_k is a binary variable taking the value 1 when $K = k$ and 0 otherwise, $\hat{p}(K=k)$ is the empirical prior probability of class k, and $\hat{\rho}_{iC_k}$ is the correlation between feature v_i and binary variable C_k, computed according to Equation 3.

Because the correlation coefficient only measures the *linear* relationship between two random variables, it may not be suitable for some classification problems. Mutual information can capture nonlinear dependencies between variables. Formally, the mutual information between two random variables v_i and v_j is defined as

$$I(v_i;v_j) = \int\int p(v_i,v_j)\log\frac{p(v_i,v_j)}{p(v_i)p(v_j)}dv_i dv_j \ .$$

Computing mutual information is based on estimating the probability distributions $p(v_i)$, $p(v_j)$ and $p(v_i,v_j)$. These distributions can be either discretized or estimated by density function methods (Duda et al., 2000). When mutual information is used, the quadratic term is $q_{ij} = I(v_i,v_j)$ and the linear one is $F_i = I(v_i,c)$.

QPFS using mutual information as its similarity measure resembles mRMR, but there is an important difference. The mRMR method selects features greedily, as a function of features chosen in previous steps. In contrast, QPFS is not greedy and provides a ranking of features that takes into account simultaneously the mutual information between all pairs of features and the relevance of each feature to the class label.

2.2 Approximate Solution of the Quadratic Programming Problem

In high-dimensional domains, it is likely that the feature space is redundant. If so, the symmetric matrix Q is singular. We show now how Equation 2 can then be simplified and solved in a space of dimension less than M, thus reducing the computational cost.

Given the diagonalization $Q = U\Lambda U^T$ in decreasing order of eigenvalues, Equation 2 is equivalent to

$$\min_x \left\{ \frac{1}{2}(1-\alpha)x^T U\Lambda U^T x - \alpha F^T x \right\} . \tag{4}$$

If the rank of Q is $k \ll M$, then the diagonalization $Q = U\Lambda U^T$ can be written as $Q = \bar{U}\bar{\Lambda}\bar{U}^T$, where $\bar{\Lambda}$ is a diagonal square matrix consisting of the highest k eigenvalues of Q in decreasing order and $\bar{U}$ is a $M \times k$ matrix consisting of the first k eigenvectors of Q. Then, Equation 4 can be rewritten as

$$\min_x \left\{ \frac{1}{2}(1-\alpha)x^T \bar{U}\bar{\Lambda}\bar{U}^T x - \alpha F^T x \right\} .$$

Let $y = \bar{U}^T x$ be a vector in $\mathbb{R}^k$. The optimization problem is reduced to minimizing a derived quadratic function in a k-dimensional space:

$$\min_y \left\{ \frac{1}{2}(1-\alpha) y^T \bar{\Lambda} y - \alpha F^T \bar{U} y \right\}$$

under $M+1$ constraints:

$$\begin{aligned} \bar{U} y &\geq \overrightarrow{0} \\ \sum_{i=1}^{M} \sum_{j=1}^{k} \bar{u}_{ij} y_j &= 1. \end{aligned}$$

We can approximate the original vector x as $x \approx \bar{U} y$.

The matrix Q is seldom precisely singular for real world data sets. However, Q can normally be reasonably approximated by a low-rank matrix formed from its $\tilde{k}$ eigenvectors whose eigenvalues are greater than a fixed threshold $\delta > 0$ (Fine et al., 2001). More precisely, let $\tilde{Q} = U \Gamma U^T$ be the $\tilde{k}$-rank approximation of Q, where $\Gamma \in \mathbb{R}^{M \times M}$ is a diagonal matrix consisting of the $\tilde{k}$ highest eigenvalues of Q and the rest of diagonal entries are zero. Then, the approximate quadratic programming problem is formulated as

$$\min_x \left\{ \frac{1}{2}(1-\alpha) x^T U \Gamma U^T x - \alpha F^T x \right\} .$$

Equivalently,

$$\min_y \left\{ \frac{1}{2}(1-\alpha) y^T \widetilde{\Gamma} y - \alpha F^T \widetilde{U} y \right\} \tag{5}$$

where $y = \widetilde{U}^T x \in \mathbb{R}^{\tilde{k}}$, $\widetilde{\Gamma} \in \mathbb{R}^{\tilde{k} \times \tilde{k}}$ is a diagonal matrix with the nonzero eigenvalues of Γ and $\widetilde{U} \in \mathbb{R}^{M \times \tilde{k}}$ the first $\tilde{k}$ eigenvectors of U. The $M+1$ constraints of the optimization problem are defined as

$$\begin{aligned} \widetilde{U} y &\geq \overrightarrow{0} \\ \sum_{i=1}^{M} \sum_{j=1}^{\tilde{k}} \tilde{u}_{ij} y_j &= 1 . \end{aligned}$$

Given the solutions x^* of Equation 2 and $\tilde{x}^*$ of Equation 5, the error of the approximation can be estimated using the following theorem.

Theorem 1 *(Fine et al., 2001) Given $\tilde{Q}$ a $\tilde{k}$-rank approximation of Q, if $(Q - \tilde{Q})$ is positive semidefinite and $tr(Q - \tilde{Q}) \leq \varepsilon$ then the optimal value of the original problem is larger than the optimal objective value of the perturbed problem and their difference is bounded by*

$$\tilde{g}(\tilde{x}^*) \leq g(x^*) \leq \tilde{g}(\tilde{x}^*) + \frac{d^2 l \varepsilon}{2} \tag{6}$$

where l is the number of active constraints in the perturbed problem and d is an upper bound for the coefficients of the original solution.

In our case, $0 \leq x_i \leq 1$ and $d = 1$. The matrix $(Q - \tilde{Q})$ is positive semidefinite since $(Q - \tilde{Q}) = U(\Lambda - \Gamma)U^T$ and $(\Lambda - \Gamma)$ is a diagonal matrix with positive eigenvalues upper bounded by δ. Moreover $\varepsilon \leq (M - \tilde{k})\delta$ and $l \leq M + 1$, so

$$g(x^*) - \tilde{g}(\tilde{x}^*) \leq \frac{l(M-\tilde{k})\delta}{2} \leq \frac{(M+1)(M-\tilde{k})\delta}{2} = \gamma$$

where $g(x)$ and $\tilde{g}(x)$ are defined as

$$g(x) = \frac{1}{2}(1-\alpha)x^T Q x - \alpha F^T x \text{ for } x \in \mathbb{R}^M \tag{7}$$

$$\tilde{g}(x) = \frac{1}{2}(1-\alpha)x^T \tilde{\Gamma} x - \alpha F^T \tilde{U} x \text{ for } x \in \mathbb{R}^{\tilde{k}} . \tag{8}$$

Although the quadratic programming formulation of the feature selection problem is elegant and provides insight, the formulation by itself does not significantly reduce the computational complexity of feature selection. Thus we introduce the idea of applying a Nyström approximation to take advantage of the redundancy that typically makes the matrix Q almost singular. When this is true, the rank of Q is much smaller than M and the Nyström method can approximate eigenvalues and eigenvectors of Q by solving a smaller eigenproblem using only a subset of rows and columns of Q (Fowlkes et al., 2001). Suppose that $k < M$ is the rank of Q which is represented as

$$Q = \begin{pmatrix} A & B \\ B^T & E \end{pmatrix}$$

where $A \in \mathbb{R}^{k \times k}$, $B \in \mathbb{R}^{k \times (M-k)}$, $E \in \mathbb{R}^{(M-k)\times(M-k)}$, and the rows of $[A\ B]$ are independent. Then, the eigenvalues and eigenvectors of Q can be calculated exactly from the submatrix $[A\ B]$ and the diagonalization of A. Let $S = A + A^{-\frac{1}{2}} B B^T A^{-\frac{1}{2}}$ and its diagonalization $S = R\hat{\Sigma}R^T$ then, the highest k eigenvalues of Q are given by $\tilde{\Lambda} = \hat{\Sigma}$ and its associated eigenvectors $\tilde{U}$ are calculated as,

$$\tilde{U} = \begin{pmatrix} A \\ B^T \end{pmatrix} A^{-\frac{1}{2}} R \hat{\Sigma}^{-\frac{1}{2}} .$$

The application of the Nyström method entails some practical issues. First, a prior knowledge of the rank k of Q is, in general, unfeasible and it is necessary to estimate the number of subsamples r to be used in the Nyström approximation. Second, the r rows of $[A\ B]$ should be, ideally, linearly independent. If the rank of Q is greater than r or the rows of $[A\ B]$ are not linearly independent, an approximation of the diagonalization of Q is obtained whose error can be quantified, in general, as $\|E - B^T A^{-1} B\|$. Although the Nyström approximation is not error-free, if the redundancy of the feature space is large enough, then good approximations can be achieved, as shown in the following sections.

When QPFS+Nyström is used, the rule for setting the value of the α parameter is slightly different. In this case, only the the $[A\ B]$ submatrix of Q is known, and it is necessary to use the Nyström approximation $\hat{Q}$ of the original matrix Q,

$$\hat{Q} = (\hat{q_{ij}}) = \begin{pmatrix} A & B \\ B^T & B^T A^{-1} B \end{pmatrix} .$$

Therefore, the mean value of $\hat{Q}$ is computed as

$$\bar{\hat{q}} = \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j=1}^{M} \hat{q_{ij}}.$$

The mean value $\bar{f}$ of the vector F is still calculated using Equation 3, since QPFS+Nyström needs to know all coordinates of F. To sum up, the value of α for the QPFS+Nyström method is

$$\hat{\alpha} = \frac{\bar{\hat{q}}}{\bar{\hat{q}} + \bar{f}}. \tag{9}$$

The algorithm QPFS+Nyström has two levels of approximation.

1. The first level is to approximate the eigenvalues and eigenvectors of the original matrix Q based on only a subset of rows, applying the Nyström method: $\hat{Q} = \hat{U}\hat{\Lambda}\hat{U}^T$. One of the critical issues with the Nyström method is how to choose the subset of rows to use (Fowlkes et al., 2001). Ideally, the number of linearly independent rows of $[A\,B]$ should be the rank of Q. We use uniform sampling without replacement. This technique has been used successfully in other applications (Fowlkes et al., 2001; Williams and Seeger, 2001). Moreover, theoretical performance bounds for the Nyström method with uniform sampling without replacement are known (Kumar et al., 2009). In particular, we use the following theorem.

   Theorem 2 *(Kumar et al., 2009) Let $Q \in \mathbb{R}^{M\times M}$ be a symmetric positive semidefinite Gram (or kernel) matrix. Assume that r columns of Q are sampled uniformly at random without replacement ($r > k$), let $\hat{Q}$ be the rank-k Nyström approximation to Q, and let $\tilde{Q}$ the best rank-k approximation to Q. For $\varepsilon > 0$, if $r \geq \frac{64k}{\varepsilon^4}$, then*

$$E\left[\|Q - \hat{Q}\|_F\right] \leq \|Q - \tilde{Q}\|_F + \varepsilon \left[\left(\frac{M}{r} \sum_{i \in D(r)} Q_{ii}\right) \sqrt{M \sum_{i=1}^{M} Q_{ii}^2}\right]^{\frac{1}{2}} \tag{10}$$

   where $\sum_{i\in D(r)} Q_{ii}$ is the sum of the largest r diagonal entries of Q and $\|\cdot\|_F$ represents the Frobenius norm.

   As $\left(Q - \tilde{Q}\right)$ is a real symmetric positive semidefinite matrix, it is easy to prove that $\|Q - \tilde{Q}\|_F \leq \text{trace}\left(Q - \tilde{Q}\right)$.
   Equation 10 shows that the error in the Nyström approximation decreases with the number of sampled rows, r.

2. The second level of approximation is to solve the quadratic programming problem using the Nyström approximation. As stated in Section 2.2, only eigenvalues higher than a fixed threshold $\delta > 0$ are considered in the rank of matrix $\hat{Q}$. Then, these top $\tilde{k}$ eigenvalues of matrix $\hat{Q}$ are taken to make up a diagonal matrix $\hat{\tilde{\Lambda}} \in \mathbb{R}^{\tilde{k}\times\tilde{k}}$ and let $\hat{\tilde{U}} \in \mathbb{R}^{M\times\tilde{k}}$ the matrix consisting of the eigenvectors associated to $\hat{\tilde{\Lambda}}$. Therefore, the QPFS+Nyström method approximates Q by $\hat{\tilde{Q}} = \hat{\tilde{U}}\hat{\tilde{\Lambda}}\hat{\tilde{U}}^T$ and the quadratic programming problem is defined as,

$$\hat{\tilde{g}}(x) \quad = \quad \frac{1}{2}(1-\alpha)x^T\hat{\tilde{\Lambda}}x - \alpha F^T\hat{\tilde{U}}x \text{ for } x \in \mathbb{R}^{\tilde{k}}$$

and let $\hat{\hat{x}}^*$ be its optimal solution, and $g(x)$ and $\tilde{g}(x)$ be as described in Equations 7 and 8, respectively. The best rank-$\tilde{k}$ approximation to Q is $\tilde{Q} = U\Gamma U^T$ as given in Section 2.2. A bound on the total error in the QPFS+Nyström approximation is obtained following the reasoning in Fine et al. (2001):

$$\begin{aligned} E\left[g(x^*) - \hat{\hat{g}}(\hat{\hat{x}}^*)\right] &\leq E\left[g(\hat{\hat{x}}^*) - \hat{\hat{g}}(\hat{\hat{x}}^*)\right] \\ &\leq \frac{1}{2}(1-\alpha)E\left[(\hat{\hat{x}}^*)^T\left(Q - \hat{\hat{Q}}\right)(\hat{\hat{x}}^*)\right] \\ &\leq \frac{1}{2}E\left[\|Q - \hat{\hat{Q}}\|_2 \|\hat{\hat{x}}^*\|_2^2\right] \\ &\leq \frac{1}{2}(M+1)\,E\left[\|Q - \hat{\hat{Q}}\|_F\right]. \end{aligned}$$

Applying the bound for the Nyström method with uniform sampling without replacement (Equation 10) and the inequality $\|Q - \tilde{Q}\|_F \leq \text{trace}\left(Q - \tilde{Q}\right) \leq (M - \tilde{k})\delta$ yields

$$\begin{aligned} E\left[g(x^*) - \hat{\hat{g}}(\hat{\hat{x}}^*)\right] &\leq \frac{1}{2}(M+1)\left((M-\tilde{k})\delta + \varepsilon\left[\left(\frac{M}{r}\sum_{i\in D(r)} Q_{ii}\right)\sqrt{M\sum_{i=1}^{M} Q_{ii}^2}\right]^{\frac{1}{2}}\right) \\ &\leq \gamma + \frac{\varepsilon}{2}(M+1)\left[\left(\frac{M}{r}\sum_{i\in D(r)} Q_{ii}\right)\sqrt{M\sum_{i=1}^{M} Q_{ii}^2}\right]^{\frac{1}{2}}. \end{aligned}$$

The total error is the sum of the error γ obtained from the approximation of the quadratic programming problem in a subspace (Equation 6) and the error due to the Nyström method.

2.3 Summary of the QPFS+Nyström Method

Figure 1 shows a diagram of the proposed feature selection method, which can be summarized as follows:

1. Compute the F vector representing the dependence of each variable with the class.
2. Choose r rows of Q according to some criterion (typically, uniform sampling without replacement). Arrange the Q matrix so that these r rows are the first ones. Define the $[A\,B]$ matrix to be the first r rows of Q.
3. Set the value of the α parameter according to Equation 9.
4. Apply the Nyström method knowing $[A\,B]$. Obtain an approximation of the eigenvalues and eigenvectors of Q, $\hat{Q} = \hat{U}\hat{\Lambda}\hat{U}^T$.
5. Formulate the quadratic programming (QP) problem in the lower dimensional space $\hat{\hat{Q}} = \hat{\hat{U}}\hat{\hat{\Lambda}}\hat{\hat{U}}^T$.
6. Solve the QP in the subspace to obtain the solution vector y.
7. Return to the original space via $x = \hat{\hat{U}}y$.
8. Rank the variables according to the coefficients of vector x. In case of equal coefficients, rank them by decreasing relevance F_k to the class.

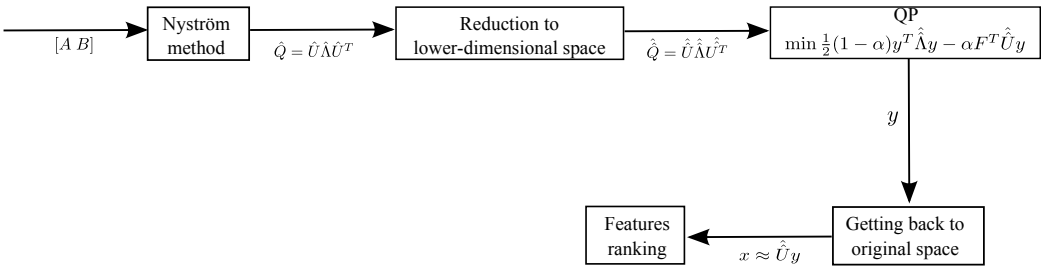


Figure 1: Diagram of the QPFS algorithm using the Nyström method. $[A\ B]$ is the upper $r \times M$ submatrix of Q.

2.4 Complexity Analysis

As already mentioned, the mRMR method is one of the most successful previous methods for feature selection. The main advantage of the QPFS+Nyström method versus mRMR is the time complexity reduction. The time complexities of mRMR and QPFS both have two components, the time needed to compute the matrices Q and F *(Similarity)*, and the time needed to perform variable ranking *(Rank)*. The computational cost of evaluating correlations or mutual information for all variable pairs is $O(NM^2)$ for both mRMR and QPFS. Table 1 shows the time complexities of the three algorithms mRMR, QPFS and QPFS+Nyström.

	mRMR		**QPFS**		**QPFS+Nyström**	
	Similarity	Rank	Similarity	Rank	Similarity	Rank
M large $N \ll pM$	$\mathbf{O(NM^2)}$	$O(M^2)$	$O(NM^2)$	$\mathbf{O(M^3)}$	$O(NpM^2)$	$\mathbf{O(p^2M^3)}$
M medium $N \gg pM$	$\mathbf{O(NM^2)}$	$O(M^2)$	$O(NM^2)$	$\mathbf{O(M^3)}$	$\mathbf{O(NpM^2)}$	$O(p^2M^3)$
M small $N \gg M$	$\mathbf{O(NM^2)}$	$O(M^2)$	$\mathbf{O(NM^2)}$	$O(M^3)$	$\mathbf{O(NpM^2)}$	$O(p^2M^3)$

Table 1: Time complexity of algorithms as a function of training set size N, number of variables M, and Nyström sampling rate p. The predominant cost term is indicated in boldface.

The order-of-magnitude time complexity of QPFS is greater than or similar to that of mRMR. However, the QPFS+Nyström time complexity is lower for $N \gg pM$ and $N \gg M$. When $N \ll pM$, QPFS+Nyström is faster than mRMR if $p^2M^3 \ll NM^2$, that is, when $N \gg p^2M$. For example, if $p = 10^{-2}$ then QPFS+Nyström is more efficient than mRMR if $N \gg 10^{-4}M$, that is, if the size of the training set is greater than 10^{-4} times the number of variables.

2.5 Implementation

We implemented QPFS and mRMR in C using *LAPACK* for matrix operations (Anderson et al., 1990). Quadratic optimization is performed by the Goldfarb and Idnani algorithm implemented in Fortran and used in the R *quadprog* package (Goldfarb and Idnani, 1983; Turlach and Weingessel, 2000).

As mentioned in Section 2.1, in general mutual information computation requires estimating density functions for continuous variables. For simplicity, each variable is discretized in three segments $(-\infty, \mu-\sigma]$, $(\mu-\sigma, \mu+\sigma]$, and $(\mu+\sigma, +\infty)$, where μ is the sample mean of training data and σ its standard deviation. The linear SVM provided by the *LIBSVM* package (Chang and Lin, 2001) was the underlying classifier in all experiments. A linear kernel is used to reduce the number of SVM parameters, thus making meaningful results easier to obtain. Note that mRMR and QPFS can be used with any classifier learning algorithms. We expect results obtained with linear SVMs to be representative.

3. Experiments

The aim of the experiments described here is twofold: first, to compare classification accuracy achieved using mRMR versus QPFS; and second, to compare their computational cost.

3.1 Experimental Design

The data sets used for experiments are shown in Table 2. These data sets were chosen because they are representative of multiple types of classification problems. with respect to the number of samples, the number of features, and the achievable classification accuracy. Moreover, these data sets have been used in other research on the feature selection task (Hua et al., 2009; Lauer et al., 2007; Lecun et al., 1998; Li et al., 2004; Peng et al., 2005; Zhang et al., 2008; Zhu et al., 2008).

In order to estimate classification accuracy, for the ARR, NCI60, SRBCT and GCM data sets 10-fold cross-validation (10CV) and 100 runs were used (Duda et al., 2000). Mean error rates are comparable to the results reported in Li et al. (2004), Peng et al. (2005), Zhang et al. (2008) and Zhu et al. (2008). In the case of the RAT data set, 120 training samples (61 for test) and 300 runs were used, following Hua et al. (2009). The MNIST data set is divided into training and testing subsets as proposed by Chang and Lin (2001), with 60000 and 10000 patterns respectively. Therefore, cross-validation is not done with MNIST.

Time complexity is measured as a function of training set size (N), dimensionality (M), and the Nyström sampling rate ($p = \frac{r}{M}$). In all cases, times are averages over 50 runs. In order to measure time complexity as a function of training set size, the number of SRBCT examples was artificially increased 4 times ($N = 332$) and dimensionality reduced to $M = 140$. Time complexity as a function of dimensionality was measured using the original SRBCT data set, that is, with $N = 83$ and $M = 2308$.

As mentioned above, MaxRel, mRMR and QPFS are all filter methods that can be used with any classifier. Figure 2 shows that the choice of the SVM regularization parameter c does not influence the comparison between mRMR and QPFS. This figure displays the performance of mRMR and QPFS for the ARR data set and different c values. Our goal is not to determine the optimal c value for each data set, but to compare mRMR and QPFS. Therefore, c is set to 1.0 in all experiments.

Data Set	N	M	C	Baseline Error Rate	References
ARR	422	278	2	21.81%	(Peng et al., 2005; Zhang et al., 2008)
NCI60	60	1123	9	38.67%	(Li et al., 2004; Zhang et al., 2008) (Zhu et al., 2008)
SRBCT	83	2308	4	0.22%	(Li et al., 2004; Zhu et al., 2008)
GCM	198	16063	14	33.85%	(Li et al., 2004; Zhang et al., 2008) (Zhu et al., 2008)
RAT	181	8460	2	8.61%	(Hua et al., 2009)
MNIST	60000	780	10	6.02%	(Lauer et al., 2007; Lecun et al., 1998)

Table 2: Description of the data sets used in experiments. N is the number of examples, M is the number of variables, and C is the number of classes. Baseline error rate is the rate obtained taking into account all variables. The last column cites papers where the data sets have been mentioned.

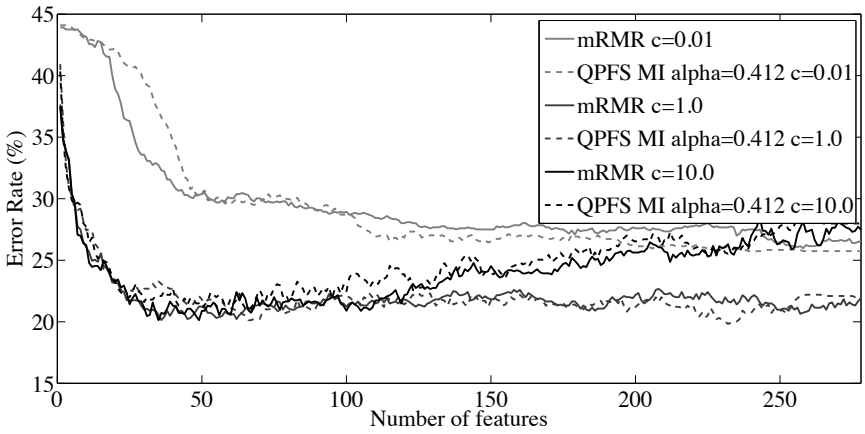


Figure 2: Classification error as a function of the number of features for the ARR data set and different regularization parameter values c in linear SVM. The figure shows that for $c = 0.01$, the SVM is too regularized. The effect when $c = 10.0$ is the opposite and the SVM overfits the training data. A value of $c = 1.0$ is a good tradeoff.

The value of the α parameter chosen for each data set is shown in Table 3. This value is obtained according to Equations 3 and 9. Our hypothesis is that high values of α are better for data sets with high redundancy among variables. On the other hand, if there is low redundancy then small α should yield better results. The Nyström sampling rate p is chosen as large as possible while still yielding a reasonable running time, since larger values reduce error in the approximation of the Q matrix. Other values of the α parameter, $\alpha \in \{0.0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, were considered in all experiments in order to verify that the proposed method of setting α provides good results.

Data Set	p	$\bar{q}$	$\bar{f}$	$\hat{\alpha}$
ARR (cor)	-	0.0889	0.0958	0.481
NCI60 (cor)	-	0.267	0.165	0.618
ARR (MI)	-	0.0106	0.0152	0.411
NCI60 (MI)	-	0.0703	0.253	0.217
SRBCT (MI)	-	0.0188	0.0861	0.179
GCM (MI)	0.05	0.0284	0.158	0.152
RAT (MI)	0.1	0.0346	0.0187	0.649
MNIST (MI)	-	0.0454	0.0515	0.469

Table 3: Values of the α parameter for each data set. Correlation (cor) and mutual information (MI) were used as similarity measures for ARR and NCI60 data sets. Only mutual information was used for SRBCT, GCM, RAT and MNIST data sets. p is the subsampling rate in the Nyström method, $\bar{q}$ is the mean value of the elements of the matrix Q (similarity among each pair of features), and $\bar{f}$ is the mean value of the elements of the F vector (similarity of each feature with the target class). For the MNIST data set only nonzero values have been considered for the statistics due to the high level of sparsity of its features (80.78% sparsity in average).

3.2 Classification Accuracy Results

The aim of the experiments described in this section is to compare classification accuracy achieved with mRMR and with QPFS, with and without Nyström approximation. The MaxRel algorithm (Peng et al., 2005) is also included in the comparison. Two similarity measures, mutual information (MI) and correlation are considered. Classification error is measured as a function of the number of features. We also give results from a baseline method that does random selection of features, in order to determine the absolute advantage of using any feature selection method.

Figure 3 shows the average classification error rate for the ARR data set as a function of the number of features. In Figure 3a, correlation is the similarity measure while mutual information (MI) is applied for Figure 3b. In both cases, the best accuracy is obtained with $\alpha = 0.5$, which means that an equal tradeoff between relevance and redundancy is best. However, accuracies using the values of α specified by our heuristic are similar.

Better accuracy is obtained when MI is used, in which case (Figure 3b) the error rate curve for $\alpha = 0.5$ is similar to that obtained with mRMR. The random selection method yields results significantly worse than those obtained with other algorithms. Comparison with this method shows that the other methods provide a significant benefit up to about 150 features.

For the NCI60 data set (Figure 4), the best accuracy is obtained when mutual information is used (Figure 4b) and α is set to 0.217 according to Table 3. In this case, the accuracy of QPFS is slightly better than the accuracy of mRMR. The value of α close to zero indicates that it is appropriate to give more weight to the quadratic term in QPFS. When correlation is used (Figure 4a), the best accuracy is obtained when α is set according to Equation 3.

Generally, MI as similarity measure leads to better accuracy than correlation. This finding is reasonable given that MI can capture nonlinear relationships between variables. MI is used in the experiments described in the remainder of this section.

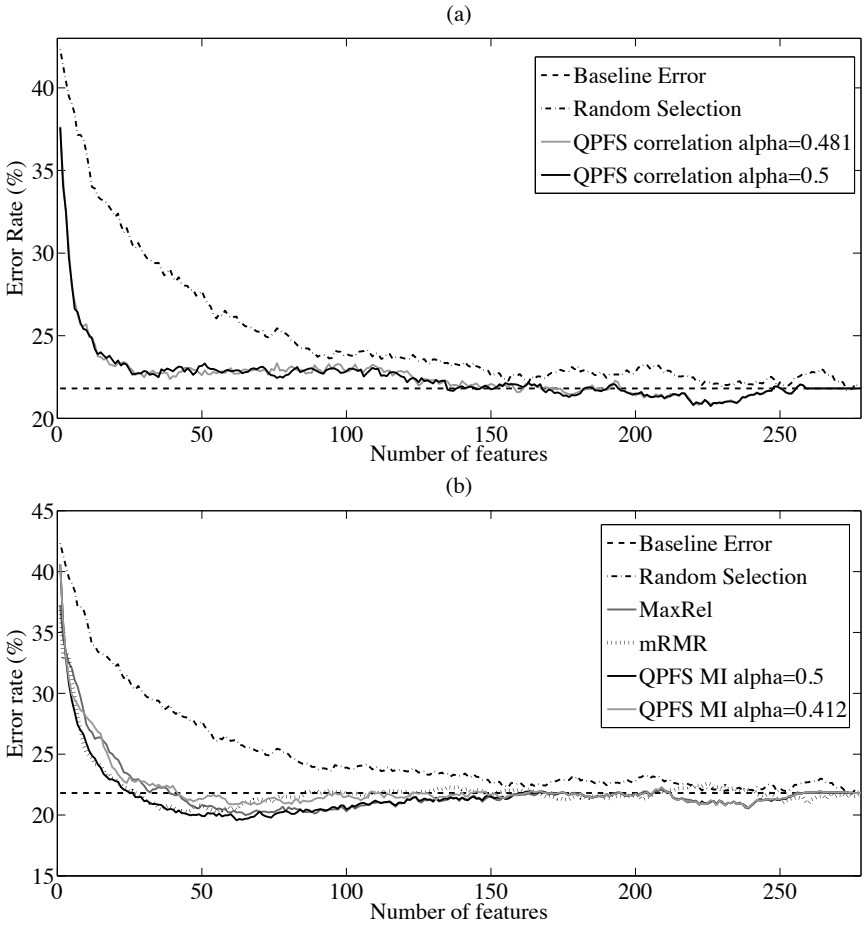


Figure 3: Classification error as a function of the number of features for the ARR data set. **(a)** QPFS results using correlation as similarity measure with different α values. **(b)** MaxRel, mRMR and QPFS results using mutual information as similarity measure and different values of α for QPFS.

Average error rate for the SRBCT data set and different sampling rates as a function of the number of features is shown in Figure 5. Results for the best α value in the grid $\{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, $\alpha = 0.1$, and the estimated $\hat{\alpha} = 0.179$ are shown in Figure 5a. Accuracies for both α values are similar. The fact that a low value of α is best indicates low redundancy among variables compared to their relevance with the target class. QPFS classification accuracy is similar to that of mRMR. As shown in Figure 5b, when the QPFS+Nyström method is used, the higher the parameter p, the closer the Nyström approximation is to complete diagonalization. QPFS+Nyström gives classification accuracy similar to that of QPFS when $p > 0.1$.

Figure 6 shows error rates for the GCM data set using the algorithms MaxRel, mRMR, and QPFS+Nyström with $\alpha = 0.1$ and $\hat{\alpha} = 0.152$. When the number of features is over 60, accuracy achieved with QPFS+Nyström is better than with mRMR. A sampling rate of 3% is adequate for

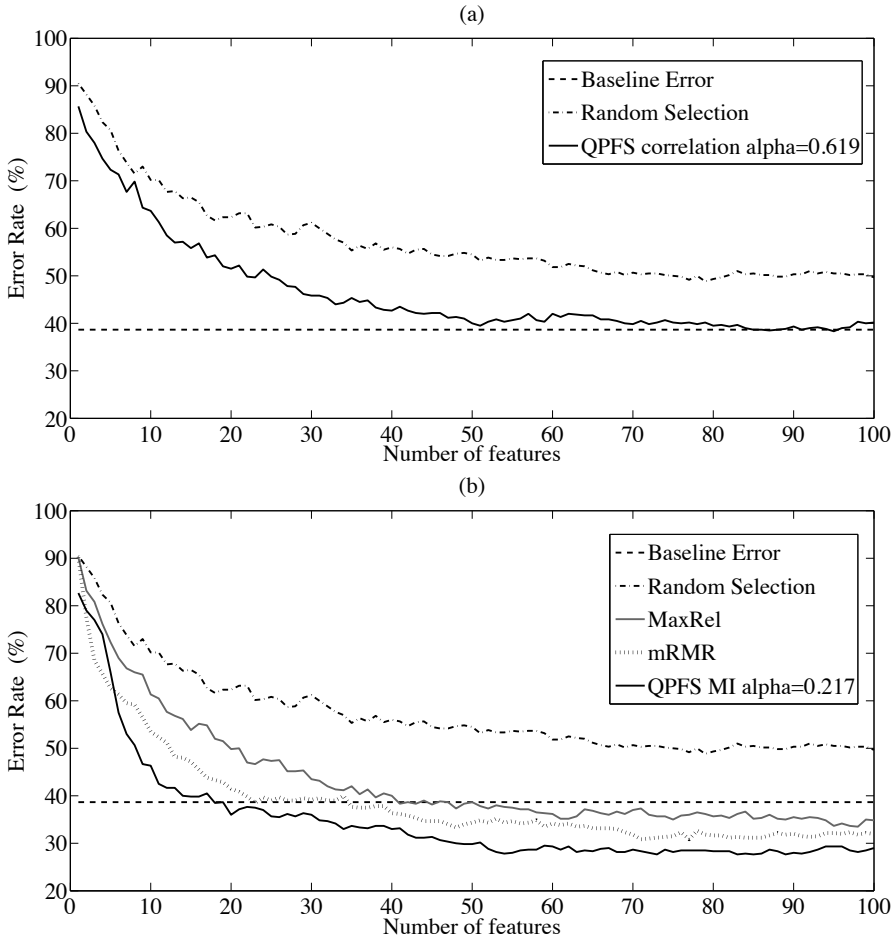


Figure 4: Classification error as a function of number of features for the NCI60 data set. **(a)** QPFS results using correlation as similarity measure with different α values. **(b)** MaxRel, mRMR and QPFS results using mutual information as similarity measure and different values of α for QPFS.

this data set, which represents a major time complexity reduction given a feature space of 16063 variables.

Another data set with many features is the RAT data set, for which Figure 7 shows results. In this case, QPFS+Nyström gives classification accuracy similar to that of mRMR when the subset size is over 80 and the sampling rate is 10%. Given the good performance of the MaxRel algorithm for this data set, it is not surprising that a large α value $\alpha = 0.9$ or $\hat{\alpha} = 0.649$ is best, considering also that QPFS with $\alpha = 1.0$ is equivalent to MaxRel.

The MNIST data set has a high number of training examples. Results for it are shown in Figure 8 for the QPFS with $\alpha = 0.3$, the estimation $\hat{\alpha} = 0.469$ and the QPFS+Nyström with $\hat{\alpha}$ and $p \in \{0.1, 0.2, 0.5\}$. Our C code of mRMR is used instead of the code on the mRMR web site (Peng et al., 2005) which takes a long time to read the training file. The error rate for all algorithms

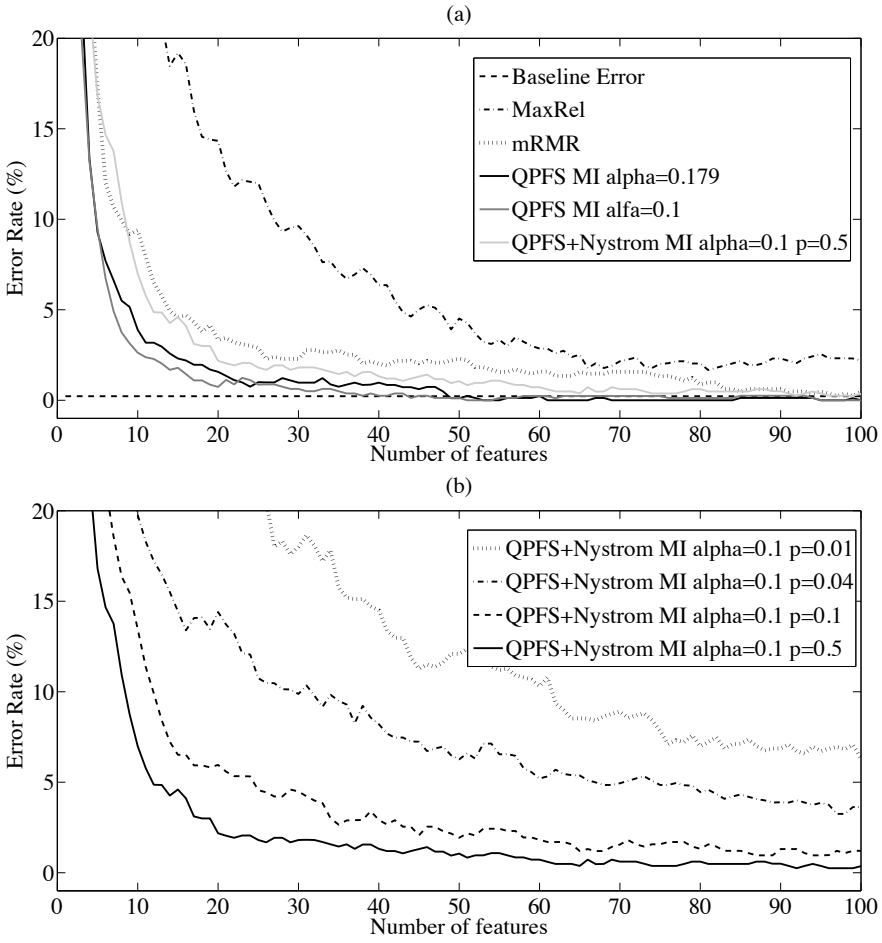


Figure 5: Error rates using MaxRel, mRMR and QPFS+Nyström methods, with mutual information as similarity measure for the SRBCT data set.

reaches a minimum when about 350 features are selected. This is not a surprising fact: analyzing the sparsity of the MNIST features, approximately 400 of them have a level of sparsity higher than 70%. But if the feature space needs a greater reduction, significant differences appears between the studied methods as shown in Figure 8. mRMR and QPFS with $\hat{\alpha} = 0.469$ have similiar performance and close to the best results obtained by QPFS with $\alpha = 0.3$. For this data set, the number of samples is much greater than the number of features, $N \gg M$, and therefore , the time complexity of mRMR and QPFS is the same ($O(NM^2)$). When QPFS+Nyström is applied with $p = 0.2$, the error rate is competitive and the MNIST provides an example of the ability of QPFS+Nyström to handle large data sets reducing the computational cost of mRMR and QPFS by a factor of 5. Note that the error rates shown for the MNIST data set are obtained using a linear kernel. The radial basis function kernel for SVM classifiers is known to lead to lower error rates for the full MNIST data set, but the choice of kernel is an issue separate from feature selection, which is the focus of this paper.

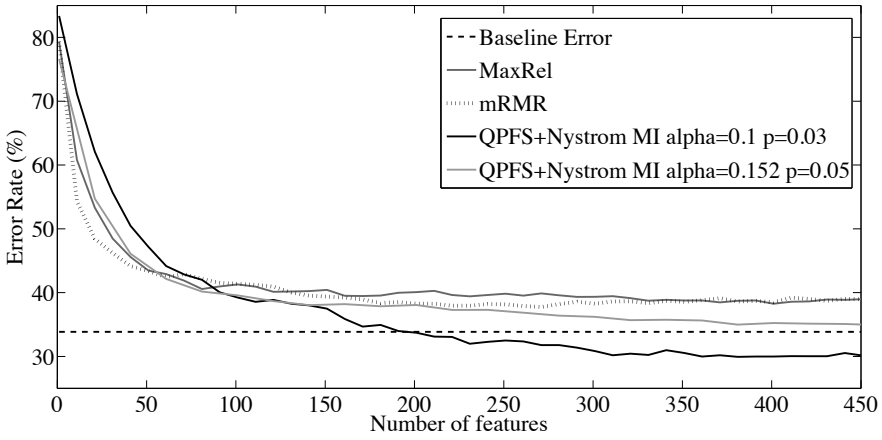


Figure 6: Error rates using MaxRel, mRMR and QPFS+Nyström methods, with mutual information as similarity measure for the GCM data set.

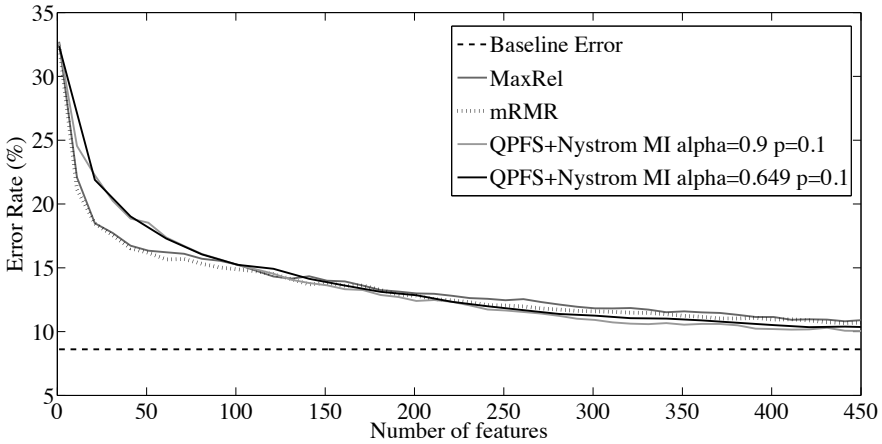


Figure 7: Error rates using MaxRel, mRMR and QPFS+Nyström methods, with mutual information as similarity measure for the RAT data set.

Figure 9 shows a grid of 780 pixels arrayed in the same way as the images in the MNIST data sets. A pixel is black if it corresponds to one of the top 100 (Figure 9a) and 350 (Figure 9b) selected features, and white otherwise. Black pixels are more dense towards the middle of the grid, because that is where the most informative features are. Pixels sometimes appear in a black/white/black checkerboard pattern, because neighboring pixels tend to make each other redundant.

Table 4 evaluates the statistical significance of error rate differences. For each data set, 100 classifiers were trained using the stated number M of selected features. The 100 classifiers arise

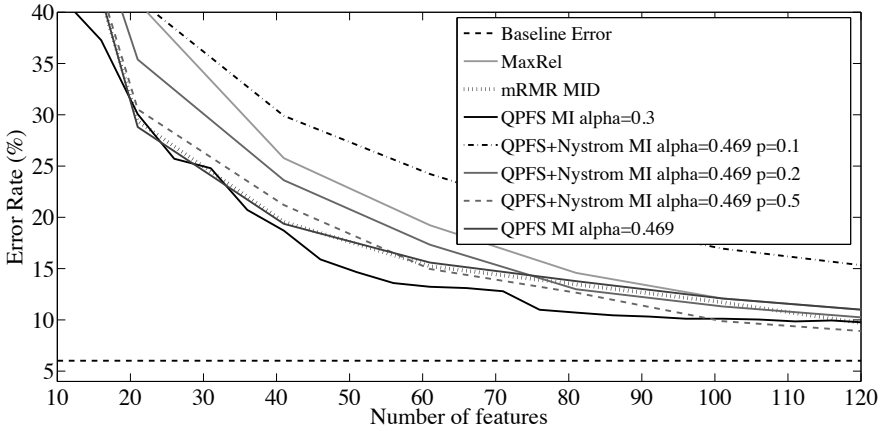


Figure 8: Error rates using MaxRel, mRMR and QPFS+Nyström methods, with mutual information as similarity measure for the MNIST data set.

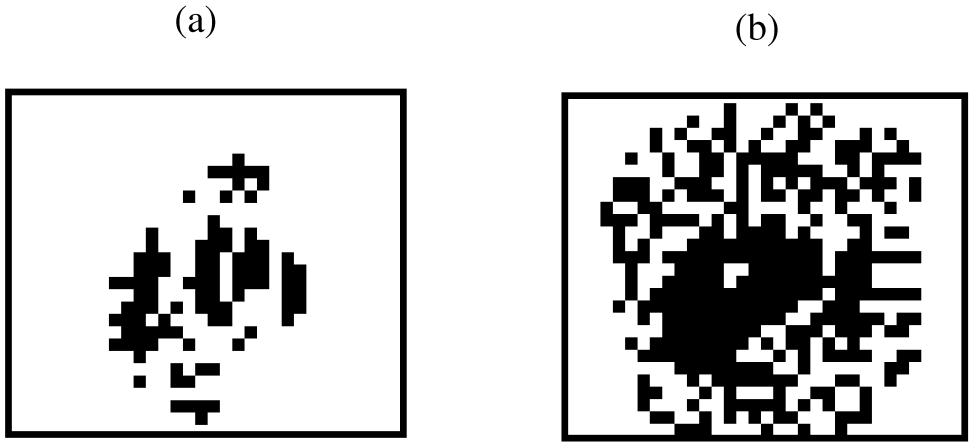


Figure 9: First **(a)** 100 and **(b)** 350 features selected by QPFS+Nyström ($\hat{\alpha} = 0.469$ and $p = 0.5$) for the MNIST data set (black pixels).

from 10 repetitions of 10-fold cross-validation, so which M features are used may be different for each classifier. The one-tailed paired t-test for equal means is applied to the two sets of error rates, one set for mRMR and one set for QPFS. The test is one-tailed because the null hypothesis is that the mRMR method is as good or better than the QPFS method. The test is paired because both methods were applied to the same 100 data set versions. Results of the test are given in the row labeled *significant?*.

For the NCI60 and SRBCT data sets, the best result is obtained when QPFS is used and it is statistically significantly better than mRMR. When 200 to 400 variables are considered, mRMR and QPFS are not statistically significantly different but the accuracy is not as good as in the case of 100 features, probably due to overfitting. In the case of the GCM data set, the mRMR method is statistically significantly better when fewer than 50 variables are considered. If the number of features is over 100, the accuracy with QPFS is significantly better than with mRMR, and the best performance is obtained in this case. For the ARR data set, mRMR is statistically significantly better than QPFS if fewer than 10 features are considered but the error rate obtained can be improved if more features are taken into account. When more than 50 featues are selected, the two methods are not statistically significantly different. The RAT data set behavior is quite similar. When fewer than 100 features are used, the mRMR algorithm is satatistically better than QPFS, but the error rate can be reduced adding more features. The two algorithms are not statistically significantly different in the other cases, except if more than 400 features are involved in which case QPFS is statistically significantly better than mRMR. Note that the error rates shown for QPFS are obtained with the proposed estimation of $\hat{\alpha}$. In some cases, as shown in Figures 3 to 7, this α value is not the best choice.

Beyond simple binary statistical significance, Table 4 indicates that the QPFS method is statistically significantly better when the value of $\hat{\alpha}$ is small. A possible explanation for this finding is the following. When $\hat{\alpha}$ is small, features are highly correlated with the label ($\bar{f} \gg \bar{q}$). The mRMR method is greedy, and only takes into account redundancy among features selected in previous iterations. When features are highly correlated with the label, then mRMR selects features with high relevance and mostly ignores redundancy. In contrast, QPFS evaluates all variables simultaneously, and always balances relevance and redundancy.

3.2.1 COMPARISON WITH OTHER FEATURE SELECTION METHODS

The experiments of this work are focused in comparing QPFS with the greedy filter-type method mRMR (difference form, named MID) which also takes into account the difference between redundancy and relevance. Nevertheless, other feature selection methods independent of the classifier have been considered in the described experiments:

- **mRMR (quotient form, named MIQ)** (Ding and Peng, 2005). While in mRMR (MID form) the difference between the estimation of redundancy and relevance is considered, in the case of mRMR (MIQ form) the quotient of both approximations is calculated.

- **reliefF** (Robnik-Šikonja and Kononenko, 2003). The main idea of ReliefF is to evaluate the quality of a feature according to how well it distinguishes between instances that are near to each other. This algorithm is efficient in problems with strong dependencies between attributes.

- **Streamwise Feature Selection (SFS)** (Zhou et al., 2006). SFS selects a feature if the *benefit* of adding it to the model is greater than the increase in the model complexity. The algorithm scales well to large feature sets and considers features sequentially for addition to a model making unnecessary to know all the features in advance.

Average error rates for MaxRel, mRMR (MID), mRMR (MIQ), reliefF and QPFS using linear SVM ($c = 1.0$) and different number of features are shown in Table 5. Table 6 shows the error rate

	M				
	10	50	100	200	400
RAT					
mRMR	21.15 ± 0.31	16.18 ± 0.27	14.88 ± 0.24	12.81 ± 0.25	10.95 ± 0.23
QPFS $\hat{\alpha} = 0.65$	27.13 ± 0.33	18.16 ± 0.29	15.24 ± 0.27	12.85 ± 0.26	10.51 ± 0.21
significant?	no	no	no	no	yes
p value	1.00	1.00	0.89	0.56	1.7×10^{-2}
ARR					
mRMR	25.19 ± 0.65	20.76 ± 0.63	21.71 ± 0.61	21.64 ± 0.61	-
QPFS $\hat{\alpha} = 0.41$	28.05 ± 0.65	21.30 ± 0.65	21.52 ± 0.65	21.76 ± 0.58	-
significant?	no	no	no	no	-
p value	1.00	0.96	0.69	0.39	-
NCI60					
mRMR	53.50 ± 2.17	34.33 ± 1.74	32.00 ± 1.93	32.83 ± 1.84	33.64 ± 1.80
QPFS $\hat{\alpha} = 0.22$	46.33 ± 2.19	29.83 ± 1.68	29.00 ± 1.83	34.67 ± 1.81	35.17 ± 1.95
significant?	yes	yes	yes	no	no
p value	1.6×10^{-3}	7.5×10^{-3}	3.3×10^{-2}	0.95	0.96
SRBCT					
mRMR	9.38 ± 1.06	2.31 ± 0.51	0.47 ± 0.23	0.24 ± 0.17	0.49 ± 0.30
QPFS $\hat{\alpha} = 0.18$	3.89 ± 0.75	0.11 ± 0.11	0.05 ± 0.11	0.11 ± 0.11	0.35 ± 0.25
significant?	yes	yes	yes	no	no
p value	5.4×10^{-9}	5.6×10^{-5}	2.3×10^{-2}	0.27	0.36
GCM					
mRMR	54.26 ± 1.19	43.38 ± 1.18	41.38 ± 1.08	38.26 ± 1.06	38.50 ± 1.10
QPFS $\hat{\alpha} = 0.15$	65.66 ± 1.03	44.11 ± 1.11	39.57 ± 1.24	38.06 ± 1.16	35.23 ± 1.17
significant?	no	no	yes	no	yes
p value	1.00	0.81	0.037	0.40	1.42×10^{-4}

Table 4: Average error rates using the mRMR and QPFS methods, for classifiers based on M features. The parameter $\hat{\alpha}$ of the QPFS method is indicated; rows are ordered according to this value. The Nyström approximation was used for the GCM and RAT data sets.

and the average number of features selected by Streamwise Feature Selection. SFS was applied to the binary data sets ARR and RAT and was used only as a feature selection method (a feature generation step was not included).

Table 5 shows that for ARR, NCI60, SRBCT and GCM data sets, the best selector is mRMR or QPFS. A statistical study of the performance of both methods is given in Table 4. In the case of the RAT data set, the best methods are MaxRel and reliefF. The fact that the best results are obtained with methods which only consider relevance with the target class fits in with the analysis of Figure 7. Finally, for the MNIST data set the best choice is the mRMR (MIQ) algorithm. Nevertheless, the

performance of MIQ in some data sets is not competitive (see, for instance, the ARR and NCI60 results). The accuracy of QPFS+Nyström ($p = 0.2$) is good if a high enough number of features is used, and it has lower computational cost than mRMR and QPFS.

Regarding SFS, Table 6 shows that SFS provides a competitive error rate for the ARR data set with few features (around 11) but its effectiveness in the RAT data set is improved by other feature selection algorithms when more than 6 attributes are considered. It is noticeable the efficiency of SFS getting acceptable accuracies using a small number of features.

ReliefF and SFS are feature selection methods which need to establish the value of some parameters like in QPFS. In ReliefF all instances were used (not random subsampling) and the number of neighbors was set to 3 for all data sets, except for MNIST where 10 neighbors were considered. In the case of the SFS algorithm, the default values (wealth $= 0.5$ and $\triangle\ \alpha = 0.5$) were used.

3.3 Time Complexity Results

Since the previous subsection has established the effectiveness of the QPFS method, it is useful now to compare mRMR and QPFS experimentally with respect to time complexity. As stated in Table 1 in Section 2.4, the running times of mRMR and QPFS with and without Nyström all depend linearly on N when M and p are fixed. In order to confirm experimentally this theoretical dependence, time consumption as a function of the number of training examples is measured on the SRBCT data set.

Figure 10a shows the time consumed for the modified SRBCT data set, averaged over 50 runs, as a function of the number of samples, N, for the mRMR, QPFS and QPFS+Nyström methods.

As expected, both mRMR and QPFS show a linear dependence on the number of patterns. For QPFS+Nyström, Table 1 shows that the slope of this linear dependence is proportional to the sampling rate p. Over the range $p = 0.01$ to $p = 0.5$, a decrease in p leads to a decrease in the slope of the linear dependence on N. Therefore, although all algorithms are linearly dependent on N, the QPFS+Nyström is computationally the most efficient. The time cost advantage increases with increasing number of training examples because the slope is greater for mRMR than for QPFS.

The next question is the impact on performance of the number of features, M. Table 1 shows that mRMR and QPFS have quadratic and cubic dependence on M, respectively. However, the QPFS+Nyström cubic coefficient is proportional to the square of the sampling rate. When small value of p are sufficient, which is the typical case, the cubic terms are not dominant.

These results are illustrated in the experiments shown in Figure 10b. This figure shows the average time cost for the SRBCT data set as a function of the problem dimension, M, for the mRMR, QPFS, and QPFS+Nyström methods. As expected from Table 1, mRMR and QPFS empirically show quadratic and cubic dependence on problem dimension. QPFS+Nyström shows only quadratic dependence on problem dimension, with a decreasing coefficient for decreasing p values. In all cases, a t-test has been used to verify the order of the polynomial that best fits each curve by least-squares fitting (Neter and Wasserman, 1974). Overall, for small Nyström sampling rates, QPFS+Nyström is computationally the most efficient.

Last but not least important, Table 1 shows there should be a quadratic dependence on sampling rate for the QPFS+Nyström algorithm. Figure 10c shows the empirical average time cost for the SRBCT data set as a function of the sampling rate p. As expected, there is quadratic dependence on p and cubic dependence on the problem dimension M.

Data Set	Method	M						
		10	20	40	50	100	200	400
ARR	MaxRel	27.48	24.68	21.70	20.82	**20.31**	21.73	-
	MID	**25.19**	**22.99**	**20.64**	**20.76**	21.71	**21.64**	-
	MIQ	29.79	27.78	23.89	23.32	21.53	21.74	-
	reliefF	30.64	24.48	21.54	21.34	20.90	21.66	-
	QPFS	28.05	23.72	22.39	21.30	21.52	21.76	-
NCI60	MaxRel	61.33	49.83	40.00	38.67	34.83	35.50	34.17
	MID	53.50	41.50	36.33	34.33	32.00	32.83	**33.67**
	MIQ	56.50	47.50	38.83	38.17	32.83	35.50	35.17
	reliefF	56.93	54.17	48.49	48.49	38.07	**32.13**	34.36
	QPFS	**46.33**	**36.00**	**33.00**	**29.83**	**29.00**	34.67	35.17
SRBCT	MaxRel	21.58	14.33	6.36	4.51	2.19	0.24	**0.13**
	MID	9.39	3.33	2.01	2.31	0.47	0.24	0.49
	MIQ	10.11	2.18	**0.47**	0.72	0.24	0.25	0.72
	reliefF	6.38	4.18	1.65	1.79	0.96	0.40	0.40
	QPFS	**3.89**	**1.57**	0.97	**0.11**	**0.05**	**0.11**	0.35
GCM	MaxRel	79.32	60.78	48.46	45.58	40.98	39.98	38.77
	MID	**54.26**	**48.45**	**44.16**	**43.38**	41.38	38.26	35.50
	MIQ	79.32	56.48	46.64	43.96	41.80	38.46	38.05
	reliefF	61.25	51.61	46.36	43.83	**39.35**	39.75	37.08
	QPFS+N $p = 0.05$	65.66	54.72	46.09	44.11	39.57	**38.06**	**35.26**
RAT	MaxRel	**19.95**	**17.32**	**15.40**	**15.16**	14.34	13.54	11.97
	MID	21.15	18.46	16.53	16.18	14.88	12.81	10.95
	MIQ	23.69	19.62	17.23	16.61	15.07	12.46	10.96
	reliefF	22.16	20.40	17.44	16.45	**13.68**	**11.43**	**9.85**
	QPFS+N $p = 0.1$	27.13	21.89	19.02	18.16	15.24	12.85	10.51
MNIST	MaxRel	59.19	40.98	25.77	22.5	12.09	7.64	6.72
	MID	53.39	29.37	19.56	17.40	11.72	7.55	6.66
	MIQ	51.69	**25.98**	**11.79**	**10.87**	**7.78**	**6.90**	**6.33**
	reliefF	**50.91**	40.20	23.81	19.56	12.31	8.47	6.86
	QPFS+N $p = 0.2$	57.00	35.39	23.62	20.48	11.31	7.71	6.54

Table 5: Error rates for different feature selection methods and Linear SVM. The best result in each case is marked in bold. QPFS+N indicates that the Nyström approximation is used in the QPFS method and p represents the subsampling rate in Nyström method. In all cases, the α parameter of QPFS is set to $\hat{\alpha}$.

4. Conclusions

This paper has presented and studied a new feature selection method for multiclass classifier learning problems. The new method, named Quadratic Programming Feature Selection (QPFS), is based

Data Set	Number of Selected Features (average)	Error rate (%)
ARR	10.75 ± 0.155	23.34 ± 0.63
RAT	6.12 ± 0.13	22.87 ± 0.33

Table 6: Streamwise Feature Selection error rates.

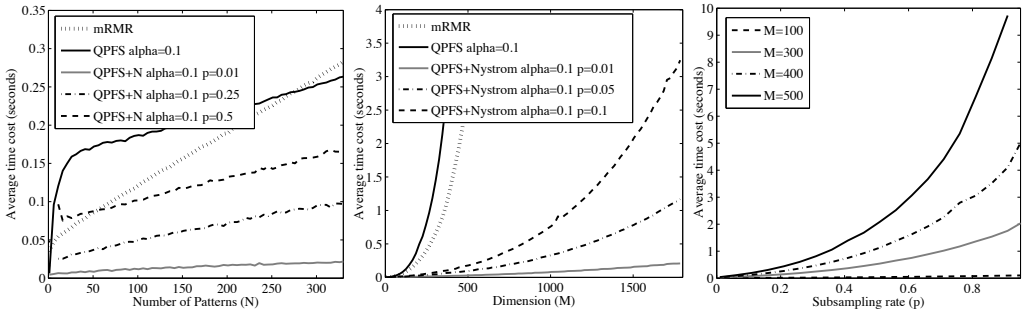


Figure 10: Time cost in seconds for mRMR and QPFS as a function of: **(a)** the number of patterns, N; **(b)** the dimension, M; and **(c)** the sampling rate, p. QPFS+N indicates that the Nyström approximation is used in the QPFS method.

on the optimization of a quadratic function that is reformulated in a lower-dimensional space using the Nyström approximation (QPFS+Nyström). The QPFS+Nyström method, using either Pearson correlation coefficient or mutual information as the underlying similarity measure, is computationally more efficient than the leading previous methods, mRMR and MaxRel.

With respect to classification accuracy, the QPFS method is similar to MaxRel and mRMR when mutual information is used, and yields slightly better results if there is high redundancy. In all experiments, mutual information yields better classification accuracy than correlation, presumably because mutual information better captures nonlinear dependencies. Small sampling rates in the Nyström method still lead to reasonable approximations of exact matrix diagonalization, sharply reducing the time complexity of QPFS. In summary, the new QPFS+Nyström method for selecting a subset of features is a competitive and efficient filter-type feature selection algorithm for high-dimensional classifier learning problems.

Acknowledgments

I.R.-L. is supported by an FPU grant from Universidad Autónoma de Madrid, and partially supported by the Universidad Autónoma de Madrid-IIC Chair. R.H. acknowledges partial support by ONR N00014-07-1-0741.

References

E. Anderson, Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. Du Croz, S. Hammarling, J. Demmel, C. H. Bischof, and Danny C. Sorensen. LAPACK: a portable linear algebra library for high-performance computers. In *SC*, pages 2–11, 1990.

R. Bekkerman, N. Tishby, Y. Winter, I. Guyon, and A. Elisseeff. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, 2003.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, September 1999.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J.Stone. *Classification and Regression Trees*. Chapman & Hall/CRC, January 1984.

C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

T.M. Cover. The best two independent measurements are not the two best. *IEEE Trans. Systems, Man, and Cybernetics*, 4:116–117, 1974.

C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *J Bioinform Comput Biol*, 3(2):185–205, April 2005.

R. O. Duda, P.E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000.

S. Fine, K. Scheinberg, N. Cristianini, J. Shawe-Taylor, and B. Williamson. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

G. Forman. BNS feature scaling: an improved representation over TF-IDF for SVM text classification. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge mining*, pages 263–270, New York, NY, USA, 2008. ACM.

G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289–1305, 2003.

C. Fowlkes, S. Belongie, and J. Malik. Efficient spatiotemporal grouping using the Nyström method. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, pages 231–238, 2001.

D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27(1):1–33, 1983.

I. Guyon. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the International Conference on Machine Learning*, pages 359–366. Morgan Kaufmann, 2000.

J. Hua, W. D. Tembe, and E. R. Dougherty. Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recogn.*, 42(3):409–424, 2009.

A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):4–37, January 2000.

G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Machine Learning: Proceedings of the Eleventh International Conference*, San Francisco, 1994.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2): 273–324, 1997.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling techniques for the Nyström method. In *Proceeding of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.

P. Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance*, pages 140–144. AAAI Press, 1994.

F. Lauer, C. Y. Suen, and G. Bloch. A trainable feature extractor for handwritten digit recognition. *Pattern Recogn.*, 40(6):1816–1824, 2007.

Y. Lecun, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

R. Leitner, H. Mairer, and A. Kercek. Real-time classification of polymers with NIR spectral imaging and blob analysis. *Real-Time Imaging*, 9:245 – 251, 2003.

T. Li, C. Zhang, and M. Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20:2429–2437, 2004.

K. Momen, S. Krishnan, and T. Chau. Real-time classification of forearm electromyographic signals corresponding to user-selected intentional movements for multifunction prosthesis control. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 15(4):535–542, Dec. 2007.

J. Neter and W. Wasserman. *Applied Linear Statistical Models*. Richard D. Irwin, INC., 1974.

H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell*, 27: 1226–1238, 2005. Software available at `http://research.janelia.org/peng/proj/mRMR/`.

M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Mach. Learn.*, 53(1-2):23–69, 2003.

J. Rodriguez, A. Goni, and A. Illarramendi. Real-time classification of ECGs on a PDA. *IEEE Transactions on Information Technology in Biomedicine*, 9(1):23–34, 2005.

P. Shenoy, K. J. Miller, B. Crawford, and R. N. Rao. Online electromyographic control of a robotic prosthesis. *IEEE Trans Biomed Eng*, 55(3):1128–1135, Mar 2008.

B. A. Turlach and A. Weingessel. The quadprog package, version 1.4-11, 2000. Available electronically at `cran.r-project.org/web/packages/quadprog/index.html`.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems 13*, pages 668–674. MIT Press, 2001.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.

L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-03)*, 2003.

Y. Zhang, C. Ding, and T. Li. Gene selection algorithm by combining reliefF and mRMR. *BMC Genomics*, 9(Suppl 2), 2008.

J. Zhou, D. P. Foster, R. A. Stine, and L. H. Ungar. Streamwise feature selection. *J. Mach. Learn. Res.*, 7:1861–1885, 2006.

S. Zhu, D. Wang, and T. Li. Feature selection for gene expression using model-based entropy. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 99(1), 2008.

Journal of Machine Learning Research 11 (2010) 1517-1561 Submitted 8/09; Revised 3/10; Published 4/10

Hilbert Space Embeddings and Metrics on Probability Measures

Bharath K. Sriperumbudur BHARATHSV@UCSD.EDU
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

Arthur Gretton* ARTHUR@TUEBINGEN.MPG.DE
MPI for Biological Cybernetics
Spemannstraße 38
72076, Tübingen, Germany

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
The Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa
Tokyo 190-8562, Japan

Bernhard Schölkopf BERNHARD.SCHOELKOPF@TUEBINGEN.MPG.DE
MPI for Biological Cybernetics
Spemannstraße 38
72076, Tübingen, Germany

Gert R. G. Lanckriet GERT@ECE.UCSD.EDU
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

Editor: Ingo Steinwart

Abstract

A Hilbert space embedding for probability measures has recently been proposed, with applications including dimensionality reduction, homogeneity testing, and independence testing. This embedding represents any probability measure as a mean element in a reproducing kernel Hilbert space (RKHS). A pseudometric on the space of probability measures can be defined as the distance between distribution embeddings: we denote this as γ_k, indexed by the kernel function k that defines the inner product in the RKHS.

We present three theoretical properties of γ_k. First, we consider the question of determining the conditions on the kernel k for which γ_k is a metric: such k are denoted *characteristic kernels*. Unlike pseudometrics, a metric is zero only when two distributions coincide, thus ensuring the RKHS embedding maps all distributions uniquely (i.e., the embedding is injective). While previously published conditions may apply only in restricted circumstances (e.g., on compact domains), and are difficult to check, our conditions are straightforward and intuitive: *integrally strictly positive definite kernels* are characteristic. Alternatively, if a bounded continuous kernel is translation-invariant on $\mathbb{R}^d$, then it is characteristic if and only if the support of its Fourier transform is the entire $\mathbb{R}^d$. Second, we show that the distance between distributions under γ_k results from an interplay between the properties of the kernel and the distributions, by demonstrating that distributions are close in the embedding space when their differences occur at higher frequencies. Third, to understand the

*. Also at Carnegie Mellon University, Pittsburgh, PA 15213, USA.

nature of the topology induced by γ_k, we relate γ_k to other popular metrics on probability measures, and present conditions on the kernel k under which γ_k metrizes the weak topology.

Keywords: probability metrics, homogeneity tests, independence tests, kernel methods, universal kernels, characteristic kernels, Hilbertian metric, weak topology

1. Introduction

The concept of distance between probability measures is a fundamental one and has found many applications in probability theory, information theory and statistics (Rachev, 1991; Rachev and Rüschendorf, 1998; Liese and Vajda, 2006). In statistics, distances between probability measures are used in a variety of applications, including hypothesis tests (homogeneity tests, independence tests, and goodness-of-fit tests), density estimation, Markov chain monte carlo, etc. As an example, homogeneity testing, also called the two-sample problem, involves choosing whether to accept or reject a null hypothesis $H_0 : \mathbb{P} = \mathbb{Q}$ versus the alternative $H_1 : \mathbb{P} \neq \mathbb{Q}$, using random samples $\{X_j\}_{j=1}^m$ and $\{Y_j\}_{j=1}^n$ drawn i.i.d. from probability distributions $\mathbb{P}$ and $\mathbb{Q}$ on a topological space $(M, \mathcal{A})$. It is easy to see that solving this problem is equivalent to testing $H_0 : \gamma(\mathbb{P}, \mathbb{Q}) = 0$ versus $H_1 : \gamma(\mathbb{P}, \mathbb{Q}) > 0$, where γ is a metric (or, more generally, a semi-metric[1]) on the space of all probability measures defined on M. The problems of testing independence and goodness-of-fit can be posed in an analogous form. In non-parametric density estimation, $\gamma(p_n, p_0)$ can be used to study the quality of the density estimate, p_n, that is based on the samples $\{X_j\}_{j=1}^n$ drawn i.i.d. from p_0. Popular examples for γ in these statistical applications include the *Kullback-Leibler divergence*, the *total variation distance*, the *Hellinger distance* (Vajda, 1989)—these three are specific instances of the generalized ϕ-divergence (Ali and Silvey, 1966; Csiszár, 1967)—the *Kolmogorov distance* (Lehmann and Romano, 2005, Section 14.2), the *Wasserstein distance* (del Barrio et al., 1999), etc.

In probability theory, the distance between probability measures is used in studying limit theorems, the popular example being the central limit theorem. Another application is in metrizing the weak convergence of probability measures on a separable metric space, where the *Lévy-Prohorov distance* (Dudley, 2002, Chapter 11) and *dual-bounded Lipschitz distance* (also called the *Dudley metric*) (Dudley, 2002, Chapter 11) are commonly used.

In the present work, we will consider a particular pseudometric[1] on probability distributions which is an instance of an *integral probability metric* (IPM) (Müller, 1997). Denoting $\mathscr{P}$ the set of all Borel probability measures on $(M, \mathcal{A})$, the IPM between $\mathbb{P} \in \mathscr{P}$ and $\mathbb{Q} \in \mathscr{P}$ is defined as

$$\gamma_{\mathcal{F}}(\mathbb{P}, \mathbb{Q}) = \sup_{f \in \mathcal{F}} \left| \int_M f \, d\mathbb{P} - \int_M f \, d\mathbb{Q} \right|, \tag{1}$$

where $\mathcal{F}$ is a class of real-valued bounded measurable functions on M. In addition to the general application domains discussed earlier for metrics on probabilities, IPMs have been used in proving central limit theorems using Stein's method (Stein, 1972; Barbour and Chen, 2005), and are popular in empirical process theory (van der Vaart and Wellner, 1996). Since most of the applications listed

1. Given a set M, a *metric* for M is a function $\rho : M \times M \to \mathbb{R}_+$ such that *(i)* $\forall x, \rho(x,x) = 0$, *(ii)* $\forall x, y, \rho(x,y) = \rho(y,x)$, *(iii)* $\forall x, y, z, \rho(x,z) \leq \rho(x,y) + \rho(y,z)$, and *(iv)* $\rho(x,y) = 0 \Rightarrow x = y$. A semi-metric only satisfies *(i)*, *(ii)* and *(iv)*. A pseudometric only satisfies *(i)-(iii)* of the properties of a metric. Unlike a metric space (M, ρ), points in a pseudometric space need not be distinguishable: one may have $\rho(x,y) = 0$ for $x \neq y$.

Now, in the two-sample test, though we mentioned that γ is a metric/semi-metric, it is sufficient that γ satisfies *(i)* and *(iv)*.

above require $\gamma_{\mathcal{F}}$ to be a metric on $\mathscr{P}$, the choice of $\mathcal{F}$ is critical (note that irrespective of $\mathcal{F}$, $\gamma_{\mathcal{F}}$ is a pseudometric on $\mathscr{P}$). The following are some examples of $\mathcal{F}$ for which $\gamma_{\mathcal{F}}$ is a metric.

(a) $\mathcal{F} = C_b(M)$, the space of bounded continuous functions on (M,ρ), where ρ is a metric (Shorack, 2000, Chapter 19, Definition 1.1).

(b) $\mathcal{F} = C_{bu}(M)$, the space of bounded ρ-uniformly continuous functions on (M,ρ)—Portmonteau theorem (Shorack, 2000, Chapter 19, Theorem 1.1).

(c) $\mathcal{F} = \{f : \|f\|_\infty \leq 1\} =: \mathcal{F}_{TV}$, where $\|f\|_\infty = \sup_{x\in M} |f(x)|$. $\gamma_{\mathcal{F}}$ is called the *total variation distance* (Shorack, 2000, Chapter 19, Proposition 2.2), which we denote as TV, that is, $\gamma_{\mathcal{F}_{TV}} =: TV$.

(d) $\mathcal{F} = \{f : \|f\|_L \leq 1\} =: \mathcal{F}_W$, where $\|f\|_L := \sup\{|f(x) - f(y)|/\rho(x,y) : x \neq y \text{ in } M\}$. $\|f\|_L$ is the Lipschitz semi-norm of a real-valued function f on M and $\gamma_{\mathcal{F}}$ is called the *Kantorovich metric*. If (M,ρ) is separable, then $\gamma_{\mathcal{F}}$ equals the *Wasserstein distance* (Dudley, 2002, Theorem 11.8.2), denoted as $W := \gamma_{\mathcal{F}_W}$.

(e) $\mathcal{F} = \{f : \|f\|_{BL} \leq 1\} =: \mathcal{F}_\beta$, where $\|f\|_{BL} := \|f\|_L + \|f\|_\infty$. $\gamma_{\mathcal{F}}$ is called the *Dudley metric* (Shorack, 2000, Chapter 19, Definition 2.2), denoted as $\beta := \gamma_{\mathcal{F}_\beta}$.

(f) $\mathcal{F} = \{\mathbb{1}_{(-\infty,t]} : t \in \mathbb{R}^d\} =: \mathcal{F}_{KS}$. $\gamma_{\mathcal{F}}$ is called the *Kolmogorov distance* (Shorack, 2000, Theorem 2.4).

(g) $\mathcal{F} = \{e^{\sqrt{-1}\langle \omega, \cdot \rangle} : \omega \in \mathbb{R}^d\} =: \mathcal{F}_c$. This choice of $\mathcal{F}$ results in the maximal difference between the characteristic functions of $\mathbb{P}$ and $\mathbb{Q}$. That $\gamma_{\mathcal{F}_c}$ is a metric on $\mathscr{P}$ follows from the *uniqueness theorem* for characteristic functions (Dudley, 2002, Theorem 9.5.1).

Recently, Gretton et al. (2007b) and Smola et al. (2007) considered $\mathcal{F}$ to be the unit ball in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ (Aronszajn, 1950), with k as its reproducing kernel (r.k.), that is, $\mathcal{F} = \{f : \|f\|_{\mathcal{H}} \leq 1\} =: \mathcal{F}_k$ (also see Chapter 4 of Berlinet and Thomas-Agnan, 2004, and references therein for related work): we denote $\gamma_{\mathcal{F}_k} =: \gamma_k$. While we have seen many possible $\mathcal{F}$ for which $\gamma_{\mathcal{F}}$ is a metric, $\mathcal{F}_k$ has a number of important advantages:

- **Estimation of $\gamma_{\mathcal{F}}$:** In applications such as hypothesis testing, $\mathbb{P}$ and $\mathbb{Q}$ are known only through the respective random samples $\{X_j\}_{j=1}^m$ and $\{Y_j\}_{j=1}^n$ drawn i.i.d. from each, and $\gamma_{\mathcal{F}}(\mathbb{P},\mathbb{Q})$ is estimated based on these samples. One approach is to compute $\gamma_{\mathcal{F}}(\mathbb{P},\mathbb{Q})$ using the empirical measures $\mathbb{P}_m = \frac{1}{m}\sum_{j=1}^m \delta_{X_j}$ and $\mathbb{Q}_n = \frac{1}{n}\sum_{j=1}^n \delta_{Y_j}$, where δ_x represents a Dirac measure at x. It can be shown that choosing $\mathcal{F}$ as $C_b(M)$, $C_{bu}(M)$, $\mathcal{F}_{TV}$ or $\mathcal{F}_c$ results in this approach not yielding consistent estimates of $\gamma_{\mathcal{F}}(\mathbb{P},\mathbb{Q})$ for all $\mathbb{P}$ and $\mathbb{Q}$ (Devroye and Györfi, 1990). Although choosing $\mathcal{F} = \mathcal{F}_W$ or $\mathcal{F}_\beta$ yields consistent estimates of $\gamma_{\mathcal{F}}(\mathbb{P},\mathbb{Q})$ for all $\mathbb{P}$ and $\mathbb{Q}$ when $M = \mathbb{R}^d$, the rates of convergence are dependent on d and become slow for large d (Sriperumbudur et al., 2009b). On the other hand, $\gamma_k(\mathbb{P}_m,\mathbb{Q}_n)$ is a $\sqrt{mn/(m+n)}$-consistent estimator of $\gamma_k(\mathbb{P},\mathbb{Q})$ if k is measurable and bounded, for all $\mathbb{P}$ and $\mathbb{Q}$. If k is translation invariant on $M = \mathbb{R}^d$, the rate is independent of d (Gretton et al., 2007b; Sriperumbudur et al., 2009b), an important property when dealing with high dimensions. Moreover, $\gamma_{\mathcal{F}}$ is not straightforward to compute when $\mathcal{F}$ is $C_b(M)$, $C_{bu}(M)$, $\mathcal{F}_W$ or $\mathcal{F}_\beta$ (Weaver, 1999, Section 2.3): by contrast, $\gamma_k^2(\mathbb{P},\mathbb{Q})$ is simply a sum of expectations of the kernel k (see (9) and Theorem 1).

- **Comparison to ϕ-divergences:** Instead of using $\gamma_{\mathcal{F}}$ in statistical applications, one can also use ϕ-divergences. However, the estimators of ϕ-divergences (especially the Kullback-Leibler divergence) exhibit arbitrarily slow rates of convergence depending on the distributions (see Wang et al., 2005; Nguyen et al., 2008, and references therein for details), while, as noted above, $\gamma_k(\mathbb{P}_m, \mathbb{Q}_n)$ exhibits good convergence behavior.

- **Structured domains:** Since γ_k is dependent only on the kernel (see Theorem 1) and kernels can be defined on arbitrary domains M (Aronszajn, 1950), choosing $\mathcal{F} = \mathcal{F}_k$ provides the flexibility of measuring the distance between probability measures defined on structured domains (Borgwardt et al., 2006) like graphs, strings, etc., unlike $\mathcal{F} = \mathcal{F}_{KS}$ or $\mathcal{F}_c$, which can handle only $M = \mathbb{R}^d$.

The distance measure γ_k has appeared in a wide variety of applications. These include statistical hypothesis testing, of homogeneity (Gretton et al., 2007b), independence (Gretton et al., 2008), and conditional independence (Fukumizu et al., 2008); as well as in machine learning applications including kernel independent component analysis (Bach and Jordan, 2002; Gretton et al., 2005a; Shen et al., 2009) and kernel based dimensionality reduction for supervised learning (Fukumizu et al., 2004). In these applications, kernels offer a linear approach to deal with higher order statistics: given the problem of homogeneity testing, for example, differences in higher order moments are encoded as differences in the means of nonlinear features of the variables. To capture all nonlinearities that are relevant to the problem at hand, the embedding RKHS therefore has to be "sufficiently large" that differences in the embeddings correspond to differences of interest in the distributions. Thus, a natural question is how to guarantee k provides a sufficiently rich RKHS so as to detect *any* difference in distributions. A second problem is to determine what properties of distributions result in their being proximate or distant in the embedding space. Finally, we would like to compare γ_k to the classical integral probability metrics listed earlier, when used to measure convergence of distributions. In the following section, we describe the contributions of the present paper, addressing each of these three questions in turn.

1.1 Contributions

The contributions in this paper are three-fold and explained in detail below.

1.1.1 WHEN IS $\mathcal{H}$ CHARACTERISTIC?

Recently, Fukumizu et al. (2008) introduced the concept of a *characteristic kernel*, that is, a reproducing kernel for which $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0 \Leftrightarrow \mathbb{P} = \mathbb{Q}$, $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$, that is, γ_k is a metric on $\mathscr{P}$. The corresponding RKHS, $\mathcal{H}$ is referred to as a *characteristic RKHS*. The following are two characterizations for characteristic RKHSs that have already been studied in literature:

1. When M is compact, Gretton et al. (2007b) showed that $\mathcal{H}$ is characteristic if k is *universal* in the sense of Steinwart (2001, Definition 4), that is, $\mathcal{H}$ is dense in the Banach space of bounded continuous functions with respect to the supremum norm. Examples of such $\mathcal{H}$ include those induced by the Gaussian and Laplacian kernels on every compact subset of $\mathbb{R}^d$.

2. Fukumizu et al. (2008, 2009a) extended this characterization to non-compact M and showed that $\mathcal{H}$ is characteristic if and only if the direct sum of $\mathcal{H}$ and $\mathbb{R}$ is dense in the Banach space of r-integrable (for some $r \geq 1$) functions. Using this characterization, they showed

that the RKHSs induced by the Gaussian and Laplacian kernels (supported on the entire $\mathbb{R}^d$) are characteristic.

In the present study, we provide alternative conditions for characteristic RKHSs which address several limitations of the foregoing. First, it can be difficult to verify the conditions of denseness in both of the above characterizations. Second, universality is in any case an overly restrictive condition because universal kernels assume M to be compact, that is, they induce a metric only on the space of probability measures that are supported on compact M.

In Section 3.1, we present the simple characterization that *integrally strictly positive definite* (pd) kernels (see Section 1.2 for the definition) are characteristic, that is, the induced RKHS is characteristic (also see Sriperumbudur et al., 2009a, Theorem 4). This condition is more natural—strict pd is a natural property of interest for kernels, unlike the denseness condition—and much easier to understand than the characterizations mentioned above. Examples of integrally strictly pd kernels on $\mathbb{R}^d$ include the Gaussian, Laplacian, inverse multiquadratics, Matérn kernel family, B_{2n+1}-splines, etc.

Although the above characterization of integrally strictly pd kernels being characteristic is simple to understand, it is only a sufficient condition and does not provide an answer for kernels that are not integrally strictly pd,[2] for example, a Dirichlet kernel. Therefore, in Section 3.2, we provide an easily checkable condition, after making some assumptions on the kernel. We present a complete characterization of characteristic kernels when the kernel is translation invariant on $\mathbb{R}^d$. We show that a bounded continuous translation invariant kernel on $\mathbb{R}^d$ is characteristic if and only if the support of the Fourier transform of the kernel is the entire $\mathbb{R}^d$. This condition is easy to check compared to the characterizations described above. An earlier version of this result was provided by Sriperumbudur et al. (2008): by comparison, we now present a simpler and more elegant proof. We also show that all compactly supported translation invariant kernels on $\mathbb{R}^d$ are characteristic. Note, however, that the characterization of integral strict positive definiteness in Section 3.1 does not assume M to be $\mathbb{R}^d$ nor k to be translation invariant.

We extend the result of Section 3.2 to M being a d-Torus, that is, $\mathbb{T}^d = S^1 \times \overset{d}{\ldots} \times S^1 \equiv [0, 2\pi)^d$, where S^1 is a circle. In Section 3.3, we show that a translation invariant kernel on $\mathbb{T}^d$ is characteristic if and only if the Fourier series coefficients of the kernel are positive, that is, the support of the Fourier spectrum is the entire $\mathbb{Z}^d$. The proof of this result is similar in flavor to the one in Section 3.2. As examples, the Poisson kernel can be shown to be characteristic, while the Dirichlet kernel is not.

Based on the discussion so far, it is clear that the characteristic property of k can be determined in many ways. In Section 3.4, we summarize the relations between various kernel families (e.g., the universal kernels and the strictly pd kernels), and show how they relate in turn to characteristic kernels. A summary is depicted in Figure 1.

1.1.2 DISSIMILAR DISTRIBUTIONS WITH SMALL γ_k

As we have seen, the characteristic property of a kernel is critical in distinguishing between distinct probability measures. Suppose, however, that for a given characteristic kernel k and for any $\varepsilon > 0$, there exist $\mathbb{P}$ and $\mathbb{Q}$, $\mathbb{P} \neq \mathbb{Q}$, such that $\gamma_k(\mathbb{P}, \mathbb{Q}) < \varepsilon$. Though k distinguishes between such $\mathbb{P}$ and $\mathbb{Q}$, it can be difficult to tell the distributions apart in applications (even with characteristic kernels), since $\mathbb{P}$ and $\mathbb{Q}$ are then replaced with finite samples, and the distance between them may not be

2. It can be shown that integrally strictly pd kernels are strictly pd (see Footnote 4). Therefore, examples of kernels that are not integrally strictly pd include those kernels that are not strictly pd.

statistically significant (Gretton et al., 2007b). Therefore, given a characteristic kernel, it is of interest to determine the properties of distributions $\mathbb{P}$ and $\mathbb{Q}$ that will cause their embeddings to be close. To this end, in Section 4, we show that given a kernel k (see Theorem 19 for conditions on the kernel), for any $\varepsilon > 0$, there exists $\mathbb{P} \neq \mathbb{Q}$ (with non-trivial differences between them) such that $\gamma_k(\mathbb{P},\mathbb{Q}) < \varepsilon$. These distributions are constructed so as to differ at a sufficiently high frequency, which is then penalized by the RKHS norm when computing γ_k.

1.1.3 WHEN DOES γ_k METRIZE THE WEAK TOPOLOGY ON $\mathscr{P}$?

Given γ_k, which is a metric on $\mathscr{P}$, a natural question of theoretical and practical importance is "how is γ_k related to other probability metrics, such as the Dudley metric (β), Wasserstein distance (W), total variation metric (TV), etc?" For example, in applications like density estimation, where the unknown density is estimated based on finite samples drawn i.i.d. from it, the quality of the estimate is measured by computing the distance between the true density and the estimated density. In such a setting, given two probability metrics, ρ_1 and ρ_2, one might want to use the *stronger*[3] of the two to determine this distance, as the convergence of the estimated density to the true density in the stronger metric implies the convergence in the weaker metric, while the converse is not true. On the other hand, one might need to use a metric of weaker topology (i.e., coarser topology) to show convergence of some estimators, as the convergence might not occur w.r.t. a metric of strong topology. Clarifying and comparing the topology of a metric on the probabilities is, thus, important in the analysis of density estimation. Based on this motivation, in Section 5, we analyze the relation between γ_k and other probability metrics, and show that γ_k is weaker than all these other metrics.

It is well known in probability theory that β is weaker than W and TV, and it metrizes the weak topology (we will provide formal definitions in Section 5) on $\mathscr{P}$ (Shorack, 2000; Gibbs and Su, 2002). Since γ_k is weaker than all these other probability metrics, that is, the topology induced by γ_k is coarser than the one induced by these metrics, the next interesting question to answer would be, "When does γ_k metrize the weak topology on $\mathscr{P}$?" In other words, for what k, does the topology induced by γ_k coincide with the weak topology? Answering this question would show that γ_k is equivalent to β, while it is weaker than W and TV. In probability theory, the metrization of weak topology is of prime importance in proving results related to the weak convergence of probability measures. Therefore, knowing the answer to the above question will help in using γ_k as a theoretical tool in probability theory. To this end, in Section 5, we show that universal kernels on compact (M,ρ) metrize the weak topology on $\mathscr{P}$. For the non-compact setting, we assume $M = \mathbb{R}^d$ and provide sufficient conditions on the kernel such that γ_k metrizes the weak topology on $\mathscr{P}$.

In the following section, we introduce the notation and some definitions that are used throughout the paper. Supplementary results used in proofs are collected in Appendix A.

1.2 Definitions and Notation

For a measurable space, M and μ a Borel measure on M, $L^r(M,\mu)$ denotes the Banach space of r-power ($r \geq 1$) μ-integrable functions. We will also use $L^r(M)$ for $L^r(M,\mu)$ and dx for $d\mu(x)$ if μ is

3. Two metrics $\rho_1 : Y \times Y \to \mathbb{R}_+$ and $\rho_2 : Y \times Y \to \mathbb{R}_+$ are said to be equivalent if $\rho_1(x,y) = 0 \Leftrightarrow \rho_2(x,y) = 0, \forall x,y \in Y$. On the other hand, ρ_1 is said to be stronger than ρ_2 if $\rho_1(x,y) = 0 \Rightarrow \rho_2(x,y) = 0, \forall x,y \in Y$ but not vice-versa. If ρ_1 is stronger than ρ_2, then we say ρ_2 is weaker than ρ_1. Note that if ρ_1 is stronger (*resp.* weaker) than ρ_2, then the topology induced by ρ_1 is finer (*resp.* coarser) than the one induced by ρ_2.

the Lebesgue measure on $M \subset \mathbb{R}^d$. $C_b(M)$ denotes the space of all bounded, continuous functions on M. The space of all r-continuously differentiable functions on M is denoted by $C^r(M), 0 \leq r \leq \infty$. For $x \in \mathbb{C}$, $\bar{x}$ represents the complex conjugate of x. We denote as i the imaginary unit $\sqrt{-1}$.

For a measurable function f and a signed measure $\mathbb{P}$, $\mathbb{P}f := \int f \, d\mathbb{P} = \int_M f(x) \, d\mathbb{P}(x)$. δ_x represents the Dirac measure at x. The symbol δ is overloaded to represent the Dirac measure, the Dirac-delta distribution, and the Kronecker-delta, which should be distinguishable from the context. For $M = \mathbb{R}^d$, the characteristic function, $\phi_{\mathbb{P}}$ of $\mathbb{P} \in \mathscr{P}$ is defined as $\phi_{\mathbb{P}}(\omega) := \int_{\mathbb{R}^d} e^{i\omega^T x} \, d\mathbb{P}(x)$, $\omega \in \mathbb{R}^d$.

Support of a Borel measure: The support of a finite regular Borel measure, μ on a Hausdorff space, M is defined to be the closed set,

$$\text{supp}(\mu) := M \backslash \bigcup \{U \subset M : U \text{ is open}, \, \mu(U) = 0\}. \tag{2}$$

Vanishing at infinity and $C_0(M)$: A complex function f on a locally compact Hausdorff space M is said to *vanish at infinity* if for every $\varepsilon > 0$ there exists a compact set $K \subset M$ such that $|f(x)| < \varepsilon$ for all $x \notin K$. The class of all continuous f on M which vanish at infinity is denoted as $C_0(M)$.

Holomorphic and entire functions: Let $D \subset \mathbb{C}^d$ be an open subset and $f : D \to \mathbb{C}$ be a function. f is said to be *holomorphic* (or *analytic*) at the point $z_0 \in D$ if

$$f'(z_0) := \lim_{z \to z_0} \frac{f(z_0) - f(z)}{z_0 - z}$$

exists. Moreover, f is called holomorphic if it is holomorphic at every $z_0 \in D$. f is called an *entire function* if f is holomorphic and $D = \mathbb{C}^d$.

Positive definite and strictly positive definite: A function $k : M \times M \to \mathbb{R}$ is called *positive definite* (pd) if, for all $n \in \mathbb{N}$, $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and all $x_1, \ldots, x_n \in M$, we have

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0. \tag{3}$$

Furthermore, k is said to be *strictly pd* if, for mutually distinct $x_1, \ldots, x_n \in X$, equality in (3) only holds for $\alpha_1 = \cdots = \alpha_n = 0$. ψ is said to be a positive definite function on $\mathbb{R}^d$ if $k(x, y) = \psi(x - y)$ is positive definite.

Integrally strictly positive definite: Let M be a topological space. A measurable and bounded kernel, k is said to be integrally strictly positive definite if

$$\int\int_M k(x, y) \, d\mu(x) \, d\mu(y) > 0,$$

for all finite non-zero signed Borel measures μ defined on M.

The above definition is a generalization of *integrally strictly positive definite functions* on $\mathbb{R}^d$ (Stewart, 1976, Section 6): $\int\int_{\mathbb{R}^d} k(x, y) f(x) f(y) \, dx \, dy > 0$ for all $f \in L^2(\mathbb{R}^d)$, which is the strictly positive definiteness of the integral operator given by the kernel. Note that the above definition is *not* equivalent to the definition of strictly pd kernels: if k is integrally strictly pd, then it is strictly pd, while the converse is not true.[4]

4. Suppose k is not strictly pd. This means for some $n \in \mathbb{N}$ and for mutually distinct $x_1, \ldots, x_n \in M$, there exists $\mathbb{R} \ni \alpha_j \neq 0$ for some $j \in \{1, \ldots, n\}$ such that $\sum_{j,l=1}^{n} \alpha_j \alpha_l k(x_j, x_l) = 0$. By defining $\mu = \sum_{j=1}^{n} \alpha_j \delta_{x_j}$, it is easy to see that there exists $\mu \neq 0$ such that $\int\int_M k(x, y) \, d\mu(x) \, d\mu(y) = 0$, which means k is not integrally strictly pd. Therefore, if k is integrally strictly pd, then it is strictly pd. However, the converse is not true. See Steinwart and Christmann (2008, Proposition 4.60, Theorem 4.62) for an example.

Fourier transform in $\mathbb{R}^d$: For $f \in L^1(\mathbb{R}^d)$, $\widehat{f}$ and $f^\vee$ represent the Fourier transform and inverse Fourier transform of f respectively, defined as

$$\widehat{f}(y) := \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{-iy^T x} f(x)\, dx,\ y \in \mathbb{R}^d, \tag{4}$$

$$f^\vee(x) := \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{ix^T y} f(y)\, dy,\ x \in \mathbb{R}^d. \tag{5}$$

Convolution: If f and g are complex functions in $\mathbb{R}^d$, their convolution $f * g$ is defined by

$$(f * g)(x) := \int_{\mathbb{R}^d} f(y) g(x-y)\, dy,$$

provided that the integral exists for almost all $x \in \mathbb{R}^d$, in the Lebesgue sense. Let μ be a finite Borel measure on $\mathbb{R}^d$ and f be a bounded measurable function on $\mathbb{R}^d$. The convolution of f and μ, $f * \mu$, which is a bounded measurable function, is defined by

$$(f * \mu)(x) := \int_{\mathbb{R}^d} f(x-y)\, d\mu(y).$$

Rapidly decaying functions, $\mathscr{D}_d$ and $\mathscr{S}_d$: Let $\mathscr{D}_d$ be the space of compactly supported infinitely differentiable functions on $\mathbb{R}^d$, that is, $\mathscr{D}_d = \{f \in C^\infty(\mathbb{R}^d) \,|\, \mathrm{supp}(f) \text{ is bounded}\}$, where $\mathrm{supp}(f) = \mathrm{cl}\left(\{x \in \mathbb{R}^d \,|\, f(x) \neq 0\}\right)$. A function $f : \mathbb{R}^d \to \mathbb{C}$ is said to decay rapidly, or be rapidly decreasing, if for all $N \in \mathbb{N}$,

$$\sup_{\|\alpha\|_1 \leq N} \sup_{x \in \mathbb{R}^d} (1 + \|x\|_2^2)^N |(T_\alpha f)(x)| < \infty,$$

where $\alpha = (\alpha_1, \ldots, \alpha_d)$ is an ordered d-tuple of non-negative α_j, $\|\alpha\|_1 = \sum_{j=1}^d \alpha_j$ and $T_\alpha = \left(\frac{1}{i}\frac{\partial}{\partial x_1}\right)^{\alpha_1} \cdots \left(\frac{1}{i}\frac{\partial}{\partial x_d}\right)^{\alpha_d}$. $\mathscr{S}_d$, called the Schwartz class, denotes the vector space of rapidly decreasing functions. Note that $\mathscr{D}_d \subset \mathscr{S}_d$. Also, for any $p \in [1, \infty]$, $\mathscr{S}_d \subset L^p(\mathbb{R}^d)$. It can be shown that for any $f \in \mathscr{S}_d$, $\widehat{f} \in \mathscr{S}_d$ and $f^\vee \in \mathscr{S}_d$ (see Folland, 1999, Chapter 9 and Rudin, 1991, Chapter 6 for details).

Distributions, $\mathscr{D}'_d$: A linear functional on $\mathscr{D}_d$ which is continuous with respect to the Fréchet topology (see Rudin, 1991, Definition 6.3) is called a *distribution* in $\mathbb{R}^d$. The space of all distributions in $\mathbb{R}^d$ is denoted by $\mathscr{D}'_d$.

As examples, if f is *locally integrable* on $\mathbb{R}^d$ (this means that f is Lebesgue measurable and $\int_K |f(x)|\, dx < \infty$ for every compact $K \subset \mathbb{R}^d$), then the functional D_f defined by

$$D_f(\varphi) = \int_{\mathbb{R}^d} f(x)\varphi(x)\, dx,\ \varphi \in \mathscr{D}_d, \tag{6}$$

is a distribution. Similarly, if μ is a Borel measure on $\mathbb{R}^d$, then

$$D_\mu(\varphi) = \int_{\mathbb{R}^d} \varphi(x)\, d\mu(x),\ \varphi \in \mathscr{D}_d,$$

defines a distribution D_μ in $\mathbb{R}^d$, which is identified with μ.

Support of a distribution: For an open set $U \subset \mathbb{R}^d$, $\mathscr{D}_d(U)$ denotes the subspace of $\mathscr{D}_d$ consisting of the functions with support contained in U. Suppose $D \in \mathscr{D}'_d$. If U is an open set of $\mathbb{R}^d$ and if

$D(\varphi) = 0$ for every $\varphi \in \mathscr{D}_d(U)$, then D is said to *vanish* or be *null* in U. Let W be the union of all open $U \subset \mathbb{R}^d$ in which D vanishes. The complement of W is the *support* of D.

Tempered distributions, $\mathscr{S}'_d$ and Fourier transform on $\mathscr{S}'_d$: A linear continuous functional (with respect to the Fréchet topology) over the space $\mathscr{S}_d$ is called a *tempered distribution* and the space of all tempered distributions in $\mathbb{R}^d$ is denoted by $\mathscr{S}'_d$. For example, every compactly supported distribution is tempered.

For any $f \in \mathscr{S}'_d$, the Fourier and inverse Fourier transforms are defined as

$$\begin{aligned} \widehat{f}(\varphi) &:= f(\hat{\varphi}), \varphi \in \mathscr{S}_d, \\ f^{\vee}(\varphi) &:= f(\varphi^{\vee}), \varphi \in \mathscr{S}_d, \end{aligned}$$

respectively. The Fourier transform is a linear, one-to-one, bicontinuous mapping from $\mathscr{S}'_d$ to $\mathscr{S}'_d$.

For complete details on distribution theory and Fourier transforms of distributions, we refer the reader to Folland (1999, Chapter 9) and Rudin (1991, Chapter 6).

2. Hilbert Space Embedding of Probability Measures

Embeddings of probability distributions into reproducing kernel Hilbert spaces were introduced in the late 70's and early 80's, generalizing the notion of mappings of individual points: see Berlinet and Thomas-Agnan (2004, Chapter 4) for a survey. Following Gretton et al. (2007b) and Smola et al. (2007), γ_k can be alternatively expressed as a pseudometric between such distribution embeddings. The following theorem describes this relation.

Theorem 1 *Let $\mathscr{P}_k := \{\mathbb{P} \in \mathscr{P} : \int_M \sqrt{k(x,x)}\, d\mathbb{P}(x) < \infty\}$, where k is measurable on M. Then for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}_k$,*

$$\gamma_k(\mathbb{P}, \mathbb{Q}) = \left\| \int_M k(\cdot, x)\, d\mathbb{P}(x) - \int_M k(\cdot, x)\, d\mathbb{Q}(x) \right\|_{\mathcal{H}} =: \|\mathbb{P}k - \mathbb{Q}k\|_{\mathcal{H}}, \tag{7}$$

where $\mathcal{H}$ is the RKHS generated by k.

Proof Let $T_{\mathbb{P}} : \mathcal{H} \to \mathbb{R}$ be the linear functional defined as $T_{\mathbb{P}}[f] := \int_M f(x)\, d\mathbb{P}(x)$ with $\|T_{\mathbb{P}}\| := \sup_{f \in \mathcal{H}, f \neq 0} \frac{|T_{\mathbb{P}}[f]|}{\|f\|_{\mathcal{H}}}$. Consider

$$|T_{\mathbb{P}}[f]| = \left| \int_M f\, d\mathbb{P} \right| \leq \int_M |f(x)|\, d\mathbb{P}(x) = \int_M |\langle f, k(\cdot, x)\rangle_{\mathcal{H}}|\, d\mathbb{P}(x) \leq \int_M \sqrt{k(x,x)} \|f\|_{\mathcal{H}}\, d\mathbb{P}(x),$$

which implies $\|T_{\mathbb{P}}\| < \infty, \forall \mathbb{P} \in \mathscr{P}_k$, that is, $T_{\mathbb{P}}$ is a bounded linear functional on $\mathcal{H}$. Therefore, by the Riesz representation theorem (Reed and Simon, 1980, Theorem II.4), for each $\mathbb{P} \in \mathscr{P}_k$, there exists a unique $\lambda_{\mathbb{P}} \in \mathcal{H}$ such that $T_{\mathbb{P}}[f] = \langle f, \lambda_{\mathbb{P}}\rangle_{\mathcal{H}}, \forall f \in \mathcal{H}$. Let $f = k(\cdot, u)$ for some $u \in M$. Then, $T_{\mathbb{P}}[k(\cdot, u)] = \langle k(\cdot, u), \lambda_{\mathbb{P}}\rangle_{\mathcal{H}} = \lambda_{\mathbb{P}}(u)$, which implies $\lambda_{\mathbb{P}} = \int_M k(\cdot, x)\, d\mathbb{P}(x) =: \mathbb{P}k$. Therefore, with

$$|\mathbb{P}f - \mathbb{Q}f| = |T_{\mathbb{P}}[f] - T_{\mathbb{Q}}[f]| = |\langle f, \lambda_{\mathbb{P}}\rangle_{\mathcal{H}} - \langle f, \lambda_{\mathbb{Q}}\rangle_{\mathcal{H}}| = |\langle f, \lambda_{\mathbb{P}} - \lambda_{\mathbb{Q}}\rangle_{\mathcal{H}}|,$$

we have

$$\gamma_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} |\mathbb{P}f - \mathbb{Q}f| = \|\lambda_{\mathbb{P}} - \lambda_{\mathbb{Q}}\|_{\mathcal{H}} = \|\mathbb{P}k - \mathbb{Q}k\|_{\mathcal{H}}.$$

Note that this holds for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}_k$. ■

Given a kernel, k, (7) holds for all $\mathbb{P} \in \mathscr{P}_k$. However, in practice, especially in statistical inference applications, it is not possible to check whether $\mathbb{P} \in \mathscr{P}_k$ as $\mathbb{P}$ is not known. Therefore, one would prefer to have a kernel such that

$$\int_M \sqrt{k(x,x)}\, d\mathbb{P}(x) < \infty, \forall \mathbb{P} \in \mathscr{P}. \tag{8}$$

The following proposition shows that (8) is equivalent to the kernel being bounded. Therefore, combining Theorem 1 and Proposition 2 shows that if k is measurable and bounded, then $\gamma_k(\mathbb{P},\mathbb{Q}) = \|\mathbb{P}k - \mathbb{Q}k\|_{\mathcal{H}}$ for any $\mathbb{P},\mathbb{Q} \in \mathscr{P}$.

Proposition 2 *Let f be a measurable function on M. Then $\int_M f(x)\, d\mathbb{P}(x) < \infty$ for all $\mathbb{P} \in \mathscr{P}$ if and only if f is bounded.*

Proof One direction is straightforward because if f is bounded, then $\int_M f(x)\, d\mathbb{P}(x) < \infty$ for all $\mathbb{P} \in \mathscr{P}$. Let us consider the other direction. Suppose f is not bounded. Then there exists a sequence $\{x_n\} \subset M$ such that $f(x_n) \overset{n\to\infty}{\longrightarrow} \infty$. By taking a subsequence, if necessary, we can assume $f(x_n) > n^2$ for all n. Then, $A := \sum_{n=1}^{\infty} \frac{1}{f(x_n)} < \infty$. Define a probability measure $\mathbb{P}$ on M by $\mathbb{P} = \sum_{n=1}^{\infty} \frac{1}{Af(x_n)} \delta_{x_n}$, where δ_{x_n} is a Dirac measure at x_n. Then, $\int_M f(x)\, d\mathbb{P}(x) = \frac{1}{A}\sum_{n=1}^{\infty} \frac{f(x_n)}{f(x_n)} = \infty$, which means if f is not bounded, then there exists a $\mathbb{P} \in \mathscr{P}$ such that $\int_M f(x)\, d\mathbb{P}(x) = \infty$. ■

The representation of γ_k in (7) yields the embedding,

$$\Pi : \mathscr{P} \to \mathcal{H} \qquad \mathbb{P} \mapsto \int_M k(\cdot,x)\, d\mathbb{P}(x),$$

as proposed by Berlinet and Thomas-Agnan (2004, Chapter 4, Section 1.1) and Gretton et al. (2007b); Smola et al. (2007). Berlinet and Thomas-Agnan derived this embedding as a generalization of $\delta_x \mapsto k(\cdot,x)$, while Gretton et al. arrived at the embedding by choosing $\mathcal{F} = \mathcal{F}_k$ in (1). Since $\gamma_k(\mathbb{P},\mathbb{Q}) = \|\Pi[\mathbb{P}] - \Pi[\mathbb{Q}]\|_{\mathcal{H}}$, the question "When is γ_k a metric on $\mathscr{P}$?" is equivalent to the question "When is Π injective?" Addressing these questions is the central focus of the paper and is discussed in Section 3.

Before proceeding further, we present a number of equivalent representations of γ_k, which will improve our understanding of γ_k and be helpful in its computation. First, Gretton et al. have shown the reproducing property of k leads to

$$\begin{aligned}
\gamma_k^2(\mathbb{P},\mathbb{Q}) &= \left\| \int_M k(\cdot,x)\, d\mathbb{P}(x) - \int_M k(\cdot,x)\, d\mathbb{Q}(x) \right\|_{\mathcal{H}}^2 \\
&= \left\langle \int_M k(\cdot,x)\, d\mathbb{P}(x) - \int_M k(\cdot,x)\, d\mathbb{Q}(x), \int_M k(\cdot,y)\, d\mathbb{P}(y) - \int_M k(\cdot,y)\, d\mathbb{Q}(y) \right\rangle_{\mathcal{H}} \\
&= \left\langle \int_M k(\cdot,x)\, d\mathbb{P}(x), \int_M k(\cdot,y)\, d\mathbb{P}(y) \right\rangle_{\mathcal{H}} + \left\langle \int_M k(\cdot,x)\, d\mathbb{Q}(x), \int_M k(\cdot,y)\, d\mathbb{Q}(y) \right\rangle_{\mathcal{H}} \\
&\qquad -2\left\langle \int_M k(\cdot,x)\, d\mathbb{P}(x), \int_M k(\cdot,y)\, d\mathbb{Q}(y) \right\rangle_{\mathcal{H}} \\
&\overset{(a)}{=} \iint_M k(x,y)\, d\mathbb{P}(x)\, d\mathbb{P}(y) + \iint_M k(x,y)\, d\mathbb{Q}(x)\, d\mathbb{Q}(y) \\
&\qquad -2\iint_M k(x,y)\, d\mathbb{P}(x)\, d\mathbb{Q}(y) \qquad (9) \\
&= \iint_M k(x,y)\, d(\mathbb{P}-\mathbb{Q})(x)\, d(\mathbb{P}-\mathbb{Q})(y), \qquad (10)
\end{aligned}$$

where (a) follows from the fact that $\int_M f(x)\,d\mathbb{P}(x) = \langle f, \int_M k(\cdot,x)\,d\mathbb{P}(x)\rangle_{\mathcal{H}}$ for all $f \in \mathcal{H}$, $\mathbb{P} \in \mathscr{P}$ (see proof of Theorem 1), applied with $f = \int_M k(\cdot,y)\,d\mathbb{P}(y)$. As motivated in Section 1, γ_k^2 is a straightforward sum of expectations of k, and can be computed easily, for example, using (9) either in closed form or using numerical integration techniques, depending on the choice of k, $\mathbb{P}$ and $\mathbb{Q}$. It is easy to show that, if k is a Gaussian kernel with $\mathbb{P}$ and $\mathbb{Q}$ being normal distributions on $\mathbb{R}^d$, then γ_k can be computed in a closed form (see Song et al., 2008 and Sriperumbudur et. al., 2009b, Section III-C for examples). In the following corollary to Theorem 1, we prove three results which provide a nice interpretation for γ_k when $M = \mathbb{R}^d$ and k is translation invariant, that is, $k(x,y) = \psi(x-y)$, where ψ is a positive definite function. We provide a detailed explanation for Corollary 4 in Remark 5. Before stating the results, we need a famous result due to Bochner, that characterizes ψ. We quote this result from Wendland (2005, Theorem 6.6).

Theorem 3 (Bochner) *A continuous function $\psi : \mathbb{R}^d \to \mathbb{R}$ is positive definite if and only if it is the Fourier transform of a finite nonnegative Borel measure Λ on $\mathbb{R}^d$, that is,*

$$\psi(x) = \int_{\mathbb{R}^d} e^{-ix^T\omega}\,d\Lambda(\omega),\ x \in \mathbb{R}^d. \tag{11}$$

Corollary 4 (Different interpretations of γ_k) *(i) Let $M = \mathbb{R}^d$ and $k(x,y) = \psi(x-y)$, where $\psi : M \to \mathbb{R}$ is a bounded, continuous positive definite function. Then for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$,*

$$\gamma_k(\mathbb{P},\mathbb{Q}) = \sqrt{\int_{\mathbb{R}^d} |\phi_{\mathbb{P}}(\omega) - \phi_{\mathbb{Q}}(\omega)|^2\,d\Lambda(\omega)} =: \|\phi_{\mathbb{P}} - \phi_{\mathbb{Q}}\|_{L^2(\mathbb{R}^d,\Lambda)}, \tag{12}$$

where $\phi_{\mathbb{P}}$ and $\phi_{\mathbb{Q}}$ represent the characteristic functions of $\mathbb{P}$ and $\mathbb{Q}$ respectively.

(ii) Suppose $\theta \in L^1(\mathbb{R}^d)$ is a continuous bounded positive definite function and $\int_{\mathbb{R}^d}\theta(x)\,dx = 1$. Let $\psi(x) := \psi_t(x) = t^{-d}\theta(t^{-1}x)$, $t > 0$. Assume that p and q are bounded uniformly continuous Radon-Nikodym derivatives of $\mathbb{P}$ and $\mathbb{Q}$ w.r.t. the Lebesgue measure, that is, $d\mathbb{P} = p\,dx$ and $d\mathbb{Q} = q\,dx$. Then,

$$\lim_{t\to 0}\gamma_k(\mathbb{P},\mathbb{Q}) = \|p-q\|_{L^2(\mathbb{R}^d)}. \tag{13}$$

In particular, if $|\theta(x)| \le C(1+\|x\|_2)^{-d-\varepsilon}$ for some $C, \varepsilon > 0$, then (13) holds for all bounded p and q (not necessarily uniformly continuous).

(iii) Suppose $\psi \in L^1(\mathbb{R}^d)$ and $\sqrt{\widehat{\psi}} \in L^1(\mathbb{R}^d)$. Then,

$$\gamma_k(\mathbb{P},\mathbb{Q}) = (2\pi)^{-d/4}\|\Phi * \mathbb{P} - \Phi * \mathbb{Q}\|_{L^2(\mathbb{R}^d)}, \tag{14}$$

where $\Phi := \left(\sqrt{\widehat{\psi}}\right)^{\vee}$ and $d\Lambda = (2\pi)^{-d/2}\widehat{\psi}\,d\omega$. Here, $\Phi * \mathbb{P}$ represents the convolution of Φ and $\mathbb{P}$.

Proof *(i)* Let us consider (10) with $k(x,y) = \psi(x-y)$. Then, we have

$$\begin{aligned}
\gamma_k^2(\mathbb{P},\mathbb{Q}) &= \iint_{\mathbb{R}^d} \psi(x-y)\,d(\mathbb{P}-\mathbb{Q})(x)\,d(\mathbb{P}-\mathbb{Q})(y)\\
&\overset{(a)}{=} \iiint_{\mathbb{R}^d} e^{-i(x-y)^T\omega}\,d\Lambda(\omega)\,d(\mathbb{P}-\mathbb{Q})(x)\,d(\mathbb{P}-\mathbb{Q})(y)\\
&\overset{(b)}{=} \iint_{\mathbb{R}^d} e^{-ix^T\omega}\,d(\mathbb{P}-\mathbb{Q})(x)\int_{\mathbb{R}^d} e^{iy^T\omega}\,d(\mathbb{P}-\mathbb{Q})(y)\,d\Lambda(\omega)\\
&= \int_{\mathbb{R}^d}(\phi_{\mathbb{P}}(\omega)-\phi_{\mathbb{Q}}(\omega))\left(\overline{\phi_{\mathbb{P}}(\omega)} - \overline{\phi_{\mathbb{Q}}(\omega)}\right)d\Lambda(\omega) = \int_{\mathbb{R}^d}|\phi_{\mathbb{P}}(\omega)-\phi_{\mathbb{Q}}(\omega)|^2\,d\Lambda(\omega),
\end{aligned}$$

where Bochner's theorem (Theorem 3) is invoked in (a), while Fubini's theorem (Folland, 1999, Theorem 2.37) is invoked in (b).

(ii) Consider (9) with $k(x,y) = \psi_t(x-y)$,

$$\begin{aligned}\gamma_k^2(\mathbb{P},\mathbb{Q}) &= \iint_{\mathbb{R}^d} \psi_t(x-y)p(x)p(y)\,dx\,dy + \iint_{\mathbb{R}^d} \psi_t(x-y)q(x)q(y)\,dx\,dy \\ &\qquad -2\iint_{\mathbb{R}^d} \psi_t(x-y)p(x)q(y)\,dx\,dy \\ &= \int_{\mathbb{R}^d} (\psi_t * p)(x)p(x)\,dx + \int_{\mathbb{R}^d} (\psi_t * q)(x)q(x)\,dx - 2\int_{\mathbb{R}^d} (\psi_t * q)(x)p(x)\,dx. \qquad (15)\end{aligned}$$

Note that $\lim_{t\to 0}\int_{\mathbb{R}^d}(\psi_t * p)(x)p(x)\,dx = \int_{\mathbb{R}^d}\lim_{t\to 0}(\psi_t * p)(x)p(x)\,dx$, by invoking the dominated convergence theorem. Since p is bounded and uniformly continuous, by Theorem 25 (see Appendix A), we have $p * \psi_t \to p$ uniformly as $t \to 0$, which means $\lim_{t\to 0}\int_{\mathbb{R}^d}(\psi_t * p)(x)p(x)\,dx = \int_{\mathbb{R}^d} p^2(x)\,dx$. Using this in (15), we have

$$\lim_{t\to 0}\gamma_k^2(\mathbb{P},\mathbb{Q}) = \int_{\mathbb{R}^d}(p^2(x)+q^2(x)-2p(x)q(x))\,dx = \|p-q\|^2_{L^2(\mathbb{R}^d)}.$$

Suppose $|\theta(x)| \le (1+\|x\|_2)^{-d-\varepsilon}$ for some $C, \varepsilon > 0$. Since $p \in L^1(\mathbb{R}^d)$, by Theorem 26 (see Appendix A), we have $(p * \psi_t)(x) \to p(x)$ as $t \to 0$ for almost every x. Therefore $\lim_{t\to 0}\int_{\mathbb{R}^d}(\psi_t * p)(x)p(x)\,dx = \int_{\mathbb{R}^d} p^2(x)\,dx$ and the result follows.

(iii) Since ψ is positive definite, $\widehat{\psi}$ is nonnegative and therefore $\sqrt{\widehat{\psi}}$ is valid. Since $\sqrt{\widehat{\psi}} \in L^1(\mathbb{R}^d)$, Φ exists. Define $\phi_{\mathbb{P},\mathbb{Q}} := \phi_{\mathbb{P}} - \phi_{\mathbb{Q}}$. Now, consider

$$\begin{aligned}\|\Phi * \mathbb{P} - \Phi * \mathbb{Q}\|^2_{L^2(\mathbb{R}^d)} &= \int_{\mathbb{R}^d} |(\Phi * (\mathbb{P}-\mathbb{Q}))(x)|^2\,dx \\ &= \int_{\mathbb{R}^d}\left|\int_{\mathbb{R}^d}\Phi(x-y)\,d(\mathbb{P}-\mathbb{Q})(y)\right|^2 dx \\ &= \frac{1}{(2\pi)^d}\int_{\mathbb{R}^d}\left|\iint_{\mathbb{R}^d}\sqrt{\widehat{\psi}(\omega)}\,e^{i(x-y)^T\omega}\,d\omega\,d(\mathbb{P}-\mathbb{Q})(y)\right|^2 dx \\ &\overset{(c)}{=} \frac{1}{(2\pi)^d}\int_{\mathbb{R}^d}\left|\int_{\mathbb{R}^d}\sqrt{\widehat{\psi}(\omega)}(\overline{\phi_{\mathbb{P}}(\omega)} - \overline{\phi_{\mathbb{Q}}(\omega)})\,e^{ix^T\omega}\,d\omega\right|^2 dx \\ &= \frac{1}{(2\pi)^d}\iiint_{\mathbb{R}^d}\sqrt{\widehat{\psi}(\omega)}\sqrt{\widehat{\psi}(\xi)}\,\overline{\phi_{\mathbb{P},\mathbb{Q}}(\omega)}\,\phi_{\mathbb{P},\mathbb{Q}}(\xi)\,e^{i(\omega-\xi)^Tx}\,d\omega\,d\xi\,dx \\ &\overset{(d)}{=} \iint_{\mathbb{R}^d}\sqrt{\widehat{\psi}(\omega)}\sqrt{\widehat{\psi}(\xi)}\,\overline{\phi_{\mathbb{P},\mathbb{Q}}(\omega)}\,\phi_{\mathbb{P},\mathbb{Q}}(\xi)\left[\frac{1}{(2\pi)^d}\int_{\mathbb{R}^d}e^{i(\omega-\xi)^Tx}\,dx\right]d\omega\,d\xi \\ &= \iint_{\mathbb{R}^d}\sqrt{\widehat{\psi}(\omega)}\sqrt{\widehat{\psi}(\xi)}\,\overline{\phi_{\mathbb{P},\mathbb{Q}}(\omega)}\,\phi_{\mathbb{P},\mathbb{Q}}(\xi)\,\delta(\omega-\xi)\,d\omega\,d\xi \\ &= \int_{\mathbb{R}^d}\widehat{\psi}(\omega)\,|\phi_{\mathbb{P}}(\omega)-\phi_{\mathbb{Q}}(\omega)|^2\,d\omega \\ &= (2\pi)^{d/2}\gamma_k^2(\mathbb{P},\mathbb{Q}),\end{aligned}$$

where (c) and (d) are obtained by invoking Fubini's theorem. ■

Remark 5 *(a) (12) shows that γ_k is the L^2-distance between the characteristic functions of $\mathbb{P}$ and $\mathbb{Q}$ computed w.r.t. the non-negative finite Borel measure, Λ, which is the Fourier transform of ψ. If $\psi \in L^1(\mathbb{R}^d)$, then (12) rephrases the well known fact (Wendland, 2005, Theorem 10.12) that for any $f \in \mathcal{H}$,*

$$\|f\|_{\mathcal{H}}^2 = \int_{\mathbb{R}^d} \frac{|\widehat{f}(\omega)|^2}{\widehat{\psi}(\omega)}\, d\omega. \tag{16}$$

Choosing $f = (\mathbb{P} - \mathbb{Q}) * \psi$ in (16) yields $\widehat{f} = (\phi_{\mathbb{P}} - \phi_{\mathbb{Q}})\widehat{\psi}$ and therefore the result in (12).

(b) Suppose $d\Lambda(\omega) = (2\pi)^{-d}\, d\omega$. Assume $\mathbb{P}$ and $\mathbb{Q}$ have p and q as Radon-Nikodym derivatives w.r.t. the Lebesgue measure, that is, $d\mathbb{P} = p\, dx$ and $d\mathbb{Q} = q\, dx$. Using these in (12), it can be shown that $\gamma_k(\mathbb{P}, \mathbb{Q}) = \|p - q\|_{L^2(\mathbb{R}^d)}$. However, this result should be interpreted in a limiting sense as mentioned in Corollary 4(ii) because the choice of $d\Lambda(\omega) = (2\pi)^{-d}\, d\omega$ implies $\psi(x) = \delta(x)$, which does not satisfy the conditions of Corollary 4(i). It can be shown that $\psi(x) = \delta(x)$ is obtained in a limiting sense (Folland, 1999, Proposition 9.1): $\psi_t \to \delta$ in $\mathscr{D}_d'$ as $t \to 0$.

(c) Choosing $\theta(x) = (2\pi)^{-d/2} e^{-\|x\|_2^2/2}$ in Corollary 4(ii) corresponds to ψ_t being a Gaussian kernel (with appropriate normalization such that $\int_{\mathbb{R}^d} \psi_t(x)\, dx = 1$). Therefore, (13) shows that as the bandwidth, t of the Gaussian kernel approaches zero, γ_k approaches the L^2-distance between the densities p and q. The same result also holds for choosing ψ_t as the Laplacian kernel, B_{2n+1}-spline, inverse multiquadratic, etc. Therefore, $\gamma_k(\mathbb{P}, \mathbb{Q})$ can be seen as a generalization of the L^2-distance between probability measures, $\mathbb{P}$ and $\mathbb{Q}$.

(d) The result in (13) holds if p and q are bounded and uniformly continuous. Since any condition on $\mathbb{P}$ and $\mathbb{Q}$ is usually difficult to check in statistical applications, it is better to impose conditions on ψ rather than on $\mathbb{P}$ and $\mathbb{Q}$. In Corollary 4(ii), by imposing additional conditions on ψ_t, the result in (13) is shown to hold for all $\mathbb{P}$ and $\mathbb{Q}$ with bounded densities p and q. The condition, $|\theta(x)| \leq C(1 + \|x\|_2)^{-d-\varepsilon}$ for some $C, \varepsilon > 0$, is, for example, satisfied by the inverse multiquadratic kernel, $\theta(x) = \widetilde{C}(1 + \|x\|_2^2)^{-\tau}$, $x \in \mathbb{R}^d$, $\tau > d/2$, where $\widetilde{C} = \left(\int_{\mathbb{R}^d} (1 + \|x\|_2^2)^{-\tau}\, dx\right)^{-1}$.

(e) The result in Corollary 4(ii) has connections to the kernel density estimation in L^2-sense using Parzen windows (Rosenblatt, 1975), where ψ can be chosen as the Parzen window: see Gretton et al. (2007a, Section 7.1) for further discussion. Note in particular that when γ_k is used in a homogeneity test, a constant kernel bandwidth results in a faster decrease of the Type II error with increasing sample size (Anderson et al., 1994, p. 43). A decreasing bandwidth is required for a consistent estimate of $\|p - q\|_{L^2(\mathbb{R}^d)}$, however.

(f) (14) shows that γ_k is proportional to the L^2-distance between $\Phi * \mathbb{P}$ and $\Phi * \mathbb{Q}$. Let Φ be such that Φ is nonnegative and $\Phi \in L^1(\mathbb{R}^d)$. Then, defining $\widetilde{\Phi} := \left(\int_{\mathbb{R}^d} \Phi(x)\, dx\right)^{-1} \Phi = \Phi / \sqrt{\widehat{\psi}(0)} = \left(\int_{\mathbb{R}^d} \psi(x)\, dx\right)^{-1/2} \Phi$ and using this in (14), we have

$$\gamma_k(\mathbb{P}, \mathbb{Q}) = (2\pi)^{-d/4} \sqrt{\widehat{\psi}(0)} \left\| \widetilde{\Phi} * \mathbb{P} - \widetilde{\Phi} * \mathbb{Q} \right\|_{L^2(\mathbb{R}^d)}. \tag{17}$$

The r.h.s. of (17) can be interpreted as follows. Let X, Y and N be independent random variables such that $X \sim \mathbb{P}$, $Y \sim \mathbb{Q}$ and $N \sim \widetilde{\Phi}$. This means γ_k is proportional to the L^2-distance computed between the densities associated with the perturbed random variables, $X + N$ and $Y + N$. Note that $\|p - q\|_{L^2(\mathbb{R}^d)}$ is the L^2-distance between the densities of X and Y. Examples of ψ that satisfy the conditions in Corollary 4(iii) in addition to the conditions on Φ as mentioned here include the

Gaussian and Laplacian kernels on $\mathbb{R}^d$. The result in (14) holds even if $\sqrt{\widehat{\psi}} \notin L^1(\mathbb{R}^d)$ as the proof of (iii) can be handled using distribution theory. However, we assumed $\sqrt{\widehat{\psi}} \in L^1(\mathbb{R}^d)$ to keep the proof simple, without delving into distribution theory.

Although we will not be using all the results of Corollary 4 in deriving our main results in the following sections, Corollary 4 was presented to provide a better intuitive understanding of γ_k. To summarize, the core results of this section are Theorem 1 (combined with Proposition 2), which provides a closed form expression for γ_k in terms of the measurable and bounded k, and Corollary 4*(i)*, which provides an alternative representation for γ_k when k is bounded, continuous and translation invariant on $\mathbb{R}^d$.

3. Conditions for Characteristic Kernels

In this section, we address the question, "When is γ_k a metric on $\mathscr{P}$?". In other words, "When is Π injective?" or "Under what conditions is k characteristic?". To this end, we start with the definition of characteristic kernels and provide some examples where k is such that γ_k is not a metric on $\mathscr{P}$. As discussed in Section 1.1.1, although some characterizations are available for k so that γ_k is a metric on $\mathscr{P}$, they are difficult to check in practice. In Section 3.1, we provide the characterization that if k is integrally strictly pd, then γ_k is a metric on $\mathscr{P}$. In Section 3.2, we present more easily checkable conditions wherein we show that if $\text{supp}(\Lambda) = \mathbb{R}^d$ (see (2) for the definition of the support of a Borel measure), then γ_k is a metric on $\mathscr{P}$. This result is extended in a straightforward way to $\mathbb{T}^d$ (d-Torus) in Section 3.3. The main results of this section are summarized in Table 1.

We start by defining characteristic kernels.

Definition 6 (Characteristic kernel) *A bounded measurable positive definite kernel k is characteristic to a set $\mathscr{Q} \subset \mathscr{P}$ of probability measures defined on $(M, \mathcal{A})$ if for $\mathbb{P}, \mathbb{Q} \in \mathscr{Q}$, $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0 \Leftrightarrow \mathbb{P} = \mathbb{Q}$. k is simply said to be characteristic if it is characteristic to $\mathscr{P}$. The RKHS $\mathcal{H}$ induced by such a k is called a characteristic RKHS.*

As mentioned before, the injectivity of Π is related to the characteristic property of k. If k is characteristic, then $\gamma_k(\mathbb{P}, \mathbb{Q}) = \|\Pi[\mathbb{P}] - \Pi[\mathbb{Q}]\|_{\mathcal{H}} = 0 \Rightarrow \mathbb{P} = \mathbb{Q}$, which means $\mathbb{P} \mapsto \int_M k(\cdot, x)\, d\mathbb{P}(x)$, that is, Π is injective. Therefore, when $M = \mathbb{R}^d$, the embedding of a distribution to a characteristic RKHS can be seen as a generalization of the characteristic function, $\phi_{\mathbb{P}} = \int_{\mathbb{R}^d} e^{i\langle \cdot, x\rangle}\, d\mathbb{P}(x)$. This is because, by the uniqueness theorem for characteristic functions (Dudley, 2002, Theorem 9.5.1), $\phi_{\mathbb{P}} = \phi_{\mathbb{Q}} \Rightarrow \mathbb{P} = \mathbb{Q}$, which means $\mathbb{P} \mapsto \int_{\mathbb{R}^d} e^{i\langle \cdot, x\rangle}\, d\mathbb{P}(x)$ is injective. So, in this context, intuitively $e^{i\langle y, x\rangle}$ can be treated as the characteristic kernel, k, although, formally, this is not true as $e^{i\langle y, x\rangle}$ is not a pd kernel.

Before we get to the characterization of characteristic kernels, the following examples show that there exist bounded measurable kernels that are not characteristic.

Example 1 (Trivial kernel) *Let $k(x, y) = \psi(x - y) = C$, $\forall x, y \in \mathbb{R}^d$ with $C > 0$. Using this in (9), we have $\gamma_k^2(\mathbb{P}, \mathbb{Q}) = C + C - 2C = 0$ for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$, which means k is not characteristic.*

Example 2 (Dot product kernel) *Let $k(x, y) = x^T y$, $x, y \in \mathbb{R}^d$. Using this in (9), we have*

$$\gamma_k^2(\mathbb{P}, \mathbb{Q}) = \mu_{\mathbb{P}}^T \mu_{\mathbb{P}} + \mu_{\mathbb{Q}}^T \mu_{\mathbb{Q}} - 2\mu_{\mathbb{P}}^T \mu_{\mathbb{Q}} = \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_2^2,$$

where $\mu_{\mathbb{P}}$ and $\mu_{\mathbb{Q}}$ represent the means associated with $\mathbb{P}$ and $\mathbb{Q}$ respectively, that is, $\mu_{\mathbb{P}} := \int_{\mathbb{R}^d} x\, d\mathbb{P}(x)$. It is clear that k is not characteristic as $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0 \Rightarrow \mu_{\mathbb{P}} = \mu_{\mathbb{Q}} \not\Rightarrow \mathbb{P} = \mathbb{Q}$ for all $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$.

Summary of Main Results				
Domain	Property	$\mathscr{Q}$	Characteristic	Reference
M	k is integrally strictly pd	$\mathscr{P}$	Yes	Theorem 7
$\mathbb{R}^d$	$\Omega = \mathbb{R}^d$	$\mathscr{P}$	Yes	Theorem 9
$\mathbb{R}^d$	$\text{supp}(\psi)$ is compact	$\mathscr{P}$	Yes	Corollary 10
$\mathbb{R}^d$	$\Omega \subsetneq \mathbb{R}^d, \text{int}(\Omega) \neq \emptyset$	$\mathscr{P}_1$	Yes	Theorem 12
$\mathbb{R}^d$	$\Omega \subsetneq \mathbb{R}^d$	$\mathscr{P}$	No	Theorem 9
$\mathbb{T}^d$	$A_\psi(0) \geq 0, A_\psi(n) > 0, \forall n \neq 0$	$\mathscr{P}$	Yes	Theorem 14
$\mathbb{T}^d$	$\exists n \neq 0 \,\vert\, A_\psi(n) = 0$	$\mathscr{P}$	No	Theorem 14

Table 1: The table should be read as: If "Property" is satisfied on "Domain", then k is characteristic (or not) to $\mathscr{Q}$. $\mathscr{P}$ is the set of all Borel probability measures on a topological space, M. See Section 1.2 for the definition of integrally strictly pd kernels. When $M = \mathbb{R}^d$, $k(x,y) = \psi(x-y)$, where ψ is a bounded, continuous positive definite function on $\mathbb{R}^d$. ψ is the Fourier transform of a finite nonnegative Borel measure, Λ, and $\Omega := \text{supp}(\Lambda)$ (see Theorem 3 and (2) for details). $\mathscr{P}_1 := \{\mathbb{P} \in \mathscr{P} : \phi_\mathbb{P} \in L^1(\mathbb{R}^d) \cup L^2(\mathbb{R}^d), \mathbb{P} \ll \lambda \text{ and } \text{supp}(\mathbb{P}) \text{ is compact}\}$, where $\phi_\mathbb{P}$ is the characteristic function of $\mathbb{P}$ and λ is the Lebesgue measure. $\mathbb{P} \ll \lambda$ denotes that $\mathbb{P}$ is absolutely continuous w.r.t. λ. When $M = \mathbb{T}^d$, $k(x,y) = \psi(x-y)$, where ψ is a bounded, continuous positive definite function on $\mathbb{T}^d$. $\{A_\psi(n)\}_{n\in\mathbb{Z}^d}$ are the Fourier series coefficients of ψ which are nonnegative and summable (see Theorem 13 for details).

Example 3 (Polynomial kernel of order 2) *Let $k(x,y) = (1+x^T y)^2$, $x, y \in \mathbb{R}^d$. Using this in (10), we have*

$$\begin{aligned}\gamma_k^2(\mathbb{P},\mathbb{Q}) &= \int\!\!\int_{\mathbb{R}^d} (1 + 2x^T y + x^T y y^T x)\, d(\mathbb{P}-\mathbb{Q})(x)\, d(\mathbb{P}-\mathbb{Q})(y) \\ &= 2\|\mu_\mathbb{P} - \mu_\mathbb{Q}\|_2^2 + \|\Sigma_\mathbb{P} - \Sigma_\mathbb{Q} + \mu_\mathbb{P}\mu_\mathbb{P}^T - \mu_\mathbb{Q}\mu_\mathbb{Q}^T\|_F^2,\end{aligned}$$

where $\Sigma_\mathbb{P}$ and $\Sigma_\mathbb{Q}$ represent the covariance matrices associated with $\mathbb{P}$ and $\mathbb{Q}$ respectively, that is, $\Sigma_\mathbb{P} := \int_{\mathbb{R}^d} xx^T\, d\mathbb{P}(x) - \mu_\mathbb{P}\mu_\mathbb{P}^T$. $\|\cdot\|_F$ represents the Frobenius norm. Since $\gamma_k(\mathbb{P},\mathbb{Q}) = 0 \Rightarrow (\mu_\mathbb{P} = \mu_\mathbb{Q}$ and $\Sigma_\mathbb{P} = \Sigma_\mathbb{Q}) \not\Rightarrow \mathbb{P} = \mathbb{Q}$ for all $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$, k is not characteristic.

In the following sections, we address the question of when k is characteristic, that is, for what k is γ_k a metric on $\mathscr{P}$?

3.1 Integrally Strictly Positive Definite Kernels are Characteristic

Compared to the existing characterizations in literature (Gretton et al., 2007b; Fukumizu et al., 2008, 2009a), the following result provides a more natural and easily understandable characterization for characteristic kernels, namely that integrally strictly pd kernels are characteristic to $\mathscr{P}$.

Theorem 7 (Integrally strictly pd kernels are characteristic) *Let k be an integrally strictly positive definite kernel on a topological space M. Then k is characteristic to $\mathscr{P}$.*

Before proving Theorem 7, we provide a supplementary result in Lemma 8 that provides necessary and sufficient conditions for a kernel *not* to be characteristic. We show that choosing k to be integrally strictly pd violates the conditions in Lemma 8, and k is therefore characteristic to $\mathscr{P}$.

Lemma 8 *Let k be measurable and bounded on a topological space, M. Then $\exists \mathbb{P} \neq \mathbb{Q}$ where $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$ if and only if there exists a finite non-zero signed Borel measure μ that satisfies:*

(i) $\int\int_M k(x,y)\, d\mu(x)\, d\mu(y) = 0$,

(ii) $\mu(M) = 0$.

Proof ($\Leftarrow$) Suppose there exists a finite non-zero signed Borel measure, μ that satisfies *(i)* and *(ii)* in Lemma 8. By the Jordan decomposition theorem (Dudley, 2002, Theorem 5.6.1), there exist unique positive measures μ^+ and μ^- such that $\mu = \mu^+ - \mu^-$ and $\mu^+ \perp \mu^-$ (μ^+ and μ^- are singular). By *(ii)*, we have $\mu^+(M) = \mu^-(M) =: \alpha$. Define $\mathbb{P} = \alpha^{-1}\mu^+$ and $\mathbb{Q} = \alpha^{-1}\mu^-$. Clearly, $\mathbb{P} \neq \mathbb{Q}$, $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$. Then, by (10), we have

$$\gamma_k^2(\mathbb{P}, \mathbb{Q}) = \int\int_M k(x,y)\, d(\mathbb{P}-\mathbb{Q})(x)\, d(\mathbb{P}-\mathbb{Q})(y) = \alpha^{-2} \int\int_M k(x,y)\, d\mu(x)\, d\mu(y) \overset{(a)}{=} 0,$$

where *(a)* is obtained by invoking *(i)*. So, we have constructed $\mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$.
($\Rightarrow$) Suppose $\exists \mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$. Let $\mu = \mathbb{P} - \mathbb{Q}$. Clearly μ is a finite non-zero signed Borel measure that satisfies $\mu(M) = 0$. Note that by (10),

$$\gamma_k^2(\mathbb{P}, \mathbb{Q}) = \int\int_M k(x,y)\, d(\mathbb{P}-\mathbb{Q})(x)\, d(\mathbb{P}-\mathbb{Q})(y) = \int\int_M k(x,y)\, d\mu(x)\, d\mu(y),$$

and therefore *(i)* follows. ■

Proof (of Theorem 7) Since k is integrally strictly pd on M, we have

$$\int\int_M k(x,y)\, d\eta(x) d\eta(y) > 0,$$

for any finite non-zero signed Borel measure η. This means there does not exist a finite non-zero signed Borel measure that satisfies *(i)* in Lemma 8. Therefore, by Lemma 8, there does not exist $\mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$, which implies k is characteristic. ■

Examples of integrally strictly pd kernels on $\mathbb{R}^d$ include the Gaussian, $\exp(-\sigma\|x-y\|_2^2)$, $\sigma > 0$; the Laplacian, $\exp(-\sigma\|x-y\|_1)$, $\sigma > 0$; inverse multiquadratics, $(\sigma^2 + \|x-y\|_2^2)^{-c}$, $c > 0$, $\sigma > 0$,

etc, which are translation invariant kernels on $\mathbb{R}^d$. A *translation variant* integrally strictly pd kernel, $\widetilde{k}$, can be obtained from a translation invariant integrally strictly pd kernel, k, as $\widetilde{k}(x,y) = f(x)k(x,y)f(y)$, where $f : M \to \mathbb{R}$ is a bounded continuous function. A simple example of a translation variant integrally strictly pd kernel on $\mathbb{R}^d$ is $\widetilde{k}(x,y) = \exp(\sigma x^T y)$, $\sigma > 0$, where we have chosen $f(\cdot) = \exp(\sigma\|\cdot\|_2^2/2)$ and $k(x,y) = \exp(-\sigma\|x-y\|_2^2/2)$, $\sigma > 0$. Clearly, this kernel is characteristic on compact subsets of $\mathbb{R}^d$. The same result can also be obtained from the fact that $\widetilde{k}$ is universal on compact subsets of $\mathbb{R}^d$ (Steinwart, 2001, Section 3, Example 1), recalling that universal kernels are characteristic (Gretton et al., 2007b, Theorem 3).

Although the condition for characteristic k in Theorem 7 is easy to understand compared to other characterizations in literature, it is not always easy to check for integral strict positive definiteness of k. In the following section, we assume $M = \mathbb{R}^d$ and k to be translation invariant and present a complete characterization for characteristic k which is simple to check.

3.2 Characterization for Translation Invariant k on $\mathbb{R}^d$

The complete, detailed proofs of the main results in this section are provided in Section 3.5. Compared to Sriperumbudur et al. (2008), we now present simple proofs for these results without resorting to distribution theory. Let us start with the following assumption.

Assumption 1 *$k(x,y) = \psi(x-y)$ where ψ is a bounded continuous real-valued positive definite function on $M = \mathbb{R}^d$.*

The following theorem characterizes all translation invariant kernels in $\mathbb{R}^d$ that are characteristic.

Theorem 9 *Suppose k satisfies Assumption 1. Then k is characteristic if and only if* $\text{supp}(\Lambda) = \mathbb{R}^d$, *where Λ is defined as in (11).*

First, note that the condition $\text{supp}(\Lambda) = \mathbb{R}^d$ is easy to check compared to all other, aforementioned characterizations for characteristic k. Table 2 shows some popular translation invariant kernels on $\mathbb{R}$ along with their Fourier spectra, $\widehat{\psi}$ and its support: Gaussian, Laplacian, B_{2n+1}-spline[5] (Schölkopf and Smola, 2002) and Sinc kernels are aperiodic while Poisson (Brémaud, 2001; Steinwart, 2001; Vapnik, 1998), Dirichlet (Brémaud, 2001; Schölkopf and Smola, 2002), Féjer (Brémaud, 2001) and cosine kernels are periodic. Although the Gaussian and Laplacian kernels are shown to be characteristic by all the characterizations we have mentioned so far, the case of B_{2n+1}-splines is addressed only by Theorem 9, which shows them to be characteristic (note that B_{2n+1}-splines being integrally strictly pd also follows from Theorem 9). In fact, one can provide a more general result on compactly supported translation invariant kernels, which we do later in Corollary 10. The Matérn class of kernels (Rasmussen and Williams, 2006, Section 4.2.1), given by

$$k(x,y) = \psi(x-y) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\|x-y\|_2}{\sigma}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}\|x-y\|_2}{\sigma}\right), \nu > 0, \sigma > 0, \qquad (18)$$

5. A B_{2n+1}-spline is a B_n-spline of odd order. Only B_{2n+1}-splines are admissible, that is, B_n-splines of odd order are positive definite kernels whereas those of even order have negative components in their Fourier spectrum $\widehat{\psi}$, and therefore are not admissible kernels. In Table 2, the symbol $*_1^{(2n+2)}$ represents the $(2n+2)$-fold convolution. An important point to be noted with the B_{2n+1}-spline kernel is that $\widehat{\psi}$ has vanishing points at $\omega = 2\pi\alpha, \alpha \in \mathbb{Z}\backslash\{0\}$, unlike Gaussian and Laplacian kernels which do not have vanishing points in their Fourier spectrum. Nevertheless, the spectrum of all these kernels has support $\mathbb{R}$.

Kernel	$\psi(x)$	$\widehat{\psi}(\omega)$	$\text{supp}(\widehat{\psi})$
Gaussian	$\exp\left(-\frac{x^2}{2\sigma^2}\right)$	$\sigma\exp\left(-\frac{\sigma^2\omega^2}{2}\right)$	$\mathbb{R}$
Laplacian	$\exp(-\sigma\lvert x\rvert)$	$\sqrt{\frac{2}{\pi}}\frac{\sigma}{\sigma^2+\omega^2}$	$\mathbb{R}$
B_{2n+1}-spline	$*_1^{(2n+2)}\mathbb{1}_{\left[-\frac{1}{2},\frac{1}{2}\right]}(x)$	$\frac{4^{n+1}}{\sqrt{2\pi}}\frac{\sin^{2n+2}\left(\frac{\omega}{2}\right)}{\omega^{2n+2}}$	$\mathbb{R}$
Sinc	$\frac{\sin(\sigma x)}{x}$	$\sqrt{\frac{\pi}{2}}\mathbb{1}_{[-\sigma,\sigma]}(\omega)$	$[-\sigma,\sigma]$
Poisson	$\frac{1-\sigma^2}{\sigma^2-2\sigma\cos(x)+1}, 0<\sigma<1$	$\sqrt{2\pi}\sum_{j=-\infty}^{\infty}\sigma^{\lvert j\rvert}\delta(\omega-j)$	$\mathbb{Z}$
Dirichlet	$\frac{\sin\frac{(2n+1)x}{2}}{\sin\frac{x}{2}}$	$\sqrt{2\pi}\sum_{j=-n}^{n}\delta(\omega-j)$	$\{0,\pm1,\ldots,\pm n\}$
Féjer	$\frac{1}{n+1}\frac{\sin^2\frac{(n+1)x}{2}}{\sin^2\frac{x}{2}}$	$\sqrt{2\pi}\sum_{j=-n}^{n}\left(1-\frac{\lvert j\rvert}{n+1}\right)\delta(\omega-j)$	$\{0,\pm1,\ldots,\pm n\}$
Cosine	$\cos(\sigma x)$	$\sqrt{\frac{\pi}{2}}\left[\delta(\omega-\sigma)+\delta(\omega+\sigma)\right]$	$\{-\sigma,\sigma\}$

Table 2: Translation invariant kernels on $\mathbb{R}$ defined by ψ, their spectra, $\widehat{\psi}$ and its support, $\text{supp}(\widehat{\psi})$. The first four are aperiodic kernels while the last four are periodic. The domain is considered to be $\mathbb{R}$ for simplicity. For $x\in\mathbb{R}^d$, the above formulae can be extended by computing $\psi(x)=\prod_{j=1}^d\psi(x_j)$ where $x=(x_1,\ldots,x_d)$ and $\widehat{\psi}(\omega)=\prod_{j=1}^d\widehat{\psi}(\omega_j)$ where $\omega=(\omega_1,\ldots,\omega_d)$. δ represents the Dirac-delta distribution.

is characteristic as the Fourier spectrum of ψ, given by

$$\widehat{\psi}(\omega)=\frac{2^{d+\nu}\pi^{d/2}\Gamma(\nu+d/2)\nu^\nu}{\Gamma(\nu)\sigma^{2\nu}}\left(\frac{2\nu}{\sigma^2}+4\pi^2\|\omega\|_2^2\right)^{-(\nu+d/2)},\ \omega\in\mathbb{R}^d, \tag{19}$$

is positive for any $\omega\in\mathbb{R}^d$. Here, Γ is the Gamma function, K_ν is the modified Bessel function of the second kind of order ν, where ν controls the smoothness of k. The case of $\nu=\frac{1}{2}$ in the Matérn class gives the exponential kernel, $k(x,y)=\exp(-\|x-y\|_2/\sigma)$, while $\nu\to\infty$ gives the Gaussian kernel. Note that $\widehat{\psi}(x-y)$ in (19) is actually the inverse multiquadratic kernel, which is characteristic both by Theorem 7 and Theorem 9.

By Theorem 9, the Sinc kernel in Table 2 is not characteristic, which is not easy to show using other characterizations. By combining Theorem 7 with Theorem 9, it can be shown that the Sinc, Poisson, Dirichlet, Féjer and cosine kernels are not integrally strictly pd. Therefore, for translation invariant kernels on $\mathbb{R}^d$, the integral strict positive definiteness of the kernel (or the lack of it) can be tested using Theorems 7 and 9.

Of all the kernels shown in Table 2, only the Gaussian, Laplacian and B_{2n+1}-spline kernels are integrable and their corresponding $\widehat{\psi}$ are computed using (4). The other kernels shown in Table 2

are not integrable and their corresponding $\widehat{\psi}$ have to be treated as distributions (see Folland, 1999, Chapter 9 and Rudin, 1991, Chapter 6 for details), except for the Sinc kernel whose Fourier transform can be computed in the L^2 sense.[6]

Proof (Theorem 9) We provide an outline of the complete proof, which is presented in Section 3.5. The sufficient condition in Theorem 9 is simple to prove and follows from Corollary 4*(i)*, whereas we need a supplementary result to prove its necessity, which is presented in Lemma 16 (see Section 3.5). Proving the necessity of Theorem 9 is equivalent to showing that if $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$, then $\exists \mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$. In Lemma 16, we present equivalent conditions for the existence of $\mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$ if $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$, using which we prove the necessity of Theorem 9. ■

The whole family of compactly supported translation invariant continuous bounded kernels on $\mathbb{R}^d$ is characteristic, as shown by the following corollary to Theorem 9.

Corollary 10 *Suppose $k \neq 0$ satisfies Assumption 1 and* $\text{supp}(\psi)$ *is compact. Then k is characteristic.*

Proof Since $\psi \in C_b(\mathbb{R}^d)$ is compactly supported on $\mathbb{R}^d$, by (6), $\psi \in \mathscr{D}'_d$. Therefore, by the Paley-Wiener theorem (Theorem 29 in Appendix A), $\widehat{\psi}$ is the restriction to $\mathbb{R}^d$ of an entire function on $\mathbb{C}^d$, which means $\widehat{\psi}$ is an analytic function on $\mathbb{R}^d$. Suppose $\text{supp}(\widehat{\psi})$ is compact, which means there exists an open set, $U \subset \mathbb{R}^d$ such that $\widehat{\psi}(x) = 0, \forall x \in U$. But being analytic, this implies that $\widehat{\psi}(x) = 0, \forall x \in \mathbb{R}^d$, that is, $\psi = 0$, which leads to a contradiction. Therefore, $\widehat{\psi}$ cannot be compactly supported, that is, $\text{supp}(\widehat{\psi}) = \mathbb{R}^d$, and the result follows from Theorem 9. ■

The above result is interesting in practice because of the computational advantage in dealing with compactly supported kernels. Note that proving such a general result for compactly supported kernels on $\mathbb{R}^d$ is not straightforward (maybe not even possible) with the other characterizations.

As a corollary to Theorem 9, the following result provides a method to construct new characteristic kernels from a given one.

Corollary 11 *Let k, k_1 and k_2 satisfy Assumption 1. Suppose k is characteristic and $k_2 \neq 0$. Then $k + k_1$ and $k \cdot k_2$ are characteristic.*

Proof Since k, k_1 and k_2 satisfy Assumption 1, $k + k_1$ and $k_2 \cdot k$ also satisfy Assumption 1. In addition,

$$\begin{aligned}
(k+k_1)(x,y) &:= k(x,y) + k_1(x,y) = \psi(x-y) + \psi_1(x-y) = \int_{\mathbb{R}^d} e^{-i(x-y)^T\omega}\, d(\Lambda + \Lambda_1)(\omega), \\
(k \cdot k_2)(x,y) &:= k(x,y)k_2(x,y) = \psi(x-y)\psi_2(x-y) = \iint_{\mathbb{R}^d} e^{-i(x-y)^T(\omega+\xi)}\, d\Lambda(\omega)\, d\Lambda_2(\xi) \\
&\overset{(a)}{=}: \int_{\mathbb{R}^d} e^{-i(x-y)^T\omega}\, d(\Lambda * \Lambda_2)(\omega),
\end{aligned}$$

6. If $f \in L^2(\mathbb{R}^d)$, the Fourier transform $\mathcal{F}[f] := \widehat{f}$ of f is defined to be the limit, in the L^2-norm, of the sequence $\{\widehat{f}_n\}$ of Fourier transforms of any sequence $\{f_n\}$ of functions belonging to $\mathscr{L}_d$, such that f_n converges in the L^2-norm to the given function $f \in L^2(\mathbb{R}^d)$, as $n \to \infty$. The function $\widehat{f}$ is defined almost everywhere on $\mathbb{R}^d$ and belongs to $L^2(\mathbb{R}^d)$. Thus, $\mathcal{F}$ is a linear operator, mapping $L^2(\mathbb{R}^d)$ into $L^2(\mathbb{R}^d)$. See Gasquet and Witomski (1999, Chapter IV, Lesson 22) for details.

where (a) follows from the definition of convolution of measures (see Rudin 1991, Section 9.14 for details). Since k is characteristic, that is, $\text{supp}(\Lambda) = \mathbb{R}^d$, and $\text{supp}(\Lambda) \subset \text{supp}(\Lambda + \Lambda_1)$, we have $\text{supp}(\Lambda + \Lambda_1) = \mathbb{R}^d$ and therefore $k + k_1$ is characteristic. Similarly, since $\text{supp}(\Lambda) \subset \text{supp}(\Lambda * \Lambda_2)$, we have $\text{supp}(\Lambda * \Lambda_2) = \mathbb{R}^d$ and therefore, $k \cdot k_2$ is characteristic. ■

Note that in the above result, we do not need k_1 or k_2 to be characteristic. Therefore, one can generate all sorts of kernels that are characteristic by starting with a characteristic kernel, k.

So far, we have considered characterizations for k such that it is characteristic to $\mathscr{P}$. We showed in Theorem 9 that kernels with $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ are not characteristic to $\mathscr{P}$. Now, we can question whether such kernels can be characteristic to some proper subset $\mathscr{Q}$ of $\mathscr{P}$. The following result addresses this. Note that these kernels, that is, the kernels with $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ are usually not useful in practice, especially in statistical inference applications, because the conditions on $\mathscr{Q}$ are usually not easy to check. On the other hand, the following result is of theoretical interest: along with Theorem 9, it completes the characterization of characteristic kernels that are translation invariant on $\mathbb{R}^d$. Before we state the result, we denote $\mathbb{P} \ll \mathbb{Q}$ to mean that $\mathbb{P}$ is absolutely continuous w.r.t. $\mathbb{Q}$.

Theorem 12 *Let* $\mathscr{P}_1 := \{\mathbb{P} \in \mathscr{P} : \phi_{\mathbb{P}} \in L^1(\mathbb{R}^d) \cup L^2(\mathbb{R}^d),\ \mathbb{P} \ll \lambda$ *and* $\text{supp}(\mathbb{P})$ *is compact*$\}$*, where* λ *is the Lebesgue measure. Suppose* k *satisfies Assumption 1 and* $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ *has a non-empty interior, where* Λ *is defined as in (11). Then* k *is characteristic to* $\mathscr{P}_1$.

Proof See Section 3.5. ■

Although, by Theorem 9, the kernels with $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ are not characteristic to $\mathscr{P}$, Theorem 12 shows that there exists a subset of $\mathscr{P}$ to which a subset of these kernels are characteristic. This type of result is not available for the previously mentioned characterizations. An example of a kernel that satisfies the conditions in Theorem 12 is the Sinc kernel, $\psi(x) = \frac{\sin(\sigma x)}{x}$ which has $\text{supp}(\Lambda) = [-\sigma, \sigma]$. The condition that $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ has a non-empty interior is important for Theorem 12 to hold. If $\text{supp}(\Lambda)$ has an empty interior (examples include periodic kernels), then one can construct $\mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}_1$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$. This is illustrated in Example 5 of Section 3.5.

So far, we have characterized the characteristic property of kernels that satisfy (a) $\text{supp}(\Lambda) = \mathbb{R}^d$ or (b) $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ with $\text{int}(\text{supp}(\Lambda)) \neq \emptyset$. In the following section, we investigate kernels that have $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$ with $\text{int}(\text{supp}(\Lambda)) = \emptyset$, examples of which include periodic kernels on $\mathbb{R}^d$. This discussion uses the fact that a periodic function on $\mathbb{R}^d$ can be treated as a function on $\mathbb{T}^d$, the d-Torus.

3.3 Characterization for Translation Invariant k on $\mathbb{T}^d$

Let $M = \times_{j=1}^d [0, \tau_j)$ and $\tau := (\tau_1, \ldots, \tau_d)$. A function defined on M with periodic boundary conditions is equivalent to considering a periodic function on $\mathbb{R}^d$ with period τ. With no loss of generality, we can choose $\tau_j = 2\pi, \forall\, j$ which yields $M = [0, 2\pi)^d =: \mathbb{T}^d$, called the d-Torus. The results presented here hold for any $0 < \tau_j < \infty, \forall\, j$ but we choose $\tau_j = 2\pi$ for simplicity. Similar to Assumption 1, we now make the following assumption.

Assumption 2 $k(x, y) = \psi((x - y)_{mod\ 2\pi})$*, where* ψ *is a continuous real-valued positive definite function on* $M = \mathbb{T}^d$.

Similar to Theorem 3, we now state Bochner's theorem on $M = \mathbb{T}^d$.

Theorem 13 (Bochner) *A continuous function $\psi : \mathbb{T}^d \to \mathbb{R}$ is positive definite if and only if*

$$\psi(x) = \sum_{n \in \mathbb{Z}^d} A_\psi(n)\, e^{ix^T n},\ x \in \mathbb{T}^d, \tag{20}$$

where $A_\psi : \mathbb{Z}^d \to \mathbb{R}_+$, $A_\psi(-n) = A_\psi(n)$ and $\sum_{n\in\mathbb{Z}^d} A_\psi(n) < \infty$. A_ψ are called the Fourier series coefficients of ψ.

Examples for ψ include the Poisson, Dirichlet, Féjer and cosine kernels, which are shown in Table 2. We now state the result that defines characteristic kernels on $\mathbb{T}^d$.

Theorem 14 *Suppose k satisfies Assumption 2. Then k is characteristic (to the set of all Borel probability measures on $\mathbb{T}^d$) if and only if $A_\psi(0) \geq 0$, $A_\psi(n) > 0, \forall\, n \neq 0$.*

The proof is provided in Section 3.5 and the idea is similar to that of Theorem 9. Based on the above result, one can generate characteristic kernels by constructing an infinite sequence of positive numbers that are summable and then using them in (20). It can be seen from Table 2 that the Poisson kernel on $\mathbb{T}$ is characteristic while the Dirichlet, Féjer and cosine kernels are not. Some examples of characteristic kernels on $\mathbb{T}$ are:

(1) $k(x,y) = e^{\alpha\cos(x-y)}\cos(\alpha\sin(x-y)), 0 < \alpha \leq 1 \leftrightarrow A_\psi(0) = 1, A_\psi(n) = \frac{\alpha^{|n|}}{2|n|!}, \forall\, n \neq 0.$

(2) $k(x,y) = -\log(1 - 2\alpha\cos(x-y) + \alpha^2), |\alpha| < 1 \leftrightarrow A_\psi(0) = 0, A_\psi(n) = \frac{\alpha^n}{n}, \forall\, n \neq 0.$

(3) $k(x,y) = (\pi - (x-y)_{mod\,2\pi})^2 \leftrightarrow A_\psi(0) = \frac{\pi^2}{3}, A_\psi(n) = \frac{2}{n^2}, \forall\, n \neq 0.$

(4) $k(x,y) = \frac{\sinh\alpha}{\cosh\alpha - \cos(x-y)}, \alpha > 0 \leftrightarrow A_\psi(0) = 1, A_\psi(n) = e^{-\alpha|n|}, \forall\, n \neq 0.$

(5) $k(x,y) = \frac{\pi\cosh(\alpha(\pi - (x-y)_{mod\,2\pi}))}{\alpha\sinh(\pi\alpha)} \leftrightarrow A_\psi(0) = \frac{1}{\alpha^2}, A_\psi(n) = \frac{1}{n^2+\alpha^2}, \forall\, n \neq 0.$

The following result relates characteristic kernels and universal kernels defined on $\mathbb{T}^d$.

Corollary 15 *Let k be a characteristic kernel satisfying Assumption 2 with $A_\psi(0) > 0$. Then k is also universal.*

Proof Since k is characteristic with $A_\psi(0) > 0$, we have $A_\psi(n) > 0, \forall\, n$. Therefore, by Corollary 11 of Steinwart (2001), k is universal. ■

Since k being universal implies that it is characteristic, the above result shows that the converse is not true (though almost true except that $A_\psi(0)$ can be zero for characteristic kernels). The condition on A_ψ in Theorem 14, that is, $A_\psi(0) \geq 0, A_\psi(n) > 0, \forall\, n \neq 0$ can be equivalently written as $\text{supp}(A_\psi) = \mathbb{Z}^d$ or $\text{supp}(A_\psi) = \mathbb{Z}^d \backslash \{0\}$. Therefore, Theorems 9 and 14 are of similar flavor. In fact, these results can be generalized to locally compact Abelian groups. Fukumizu et al. (2009b) shows that a bounded continuous translation invariant kernel on a locally compact Abelian group G is characteristic to the set of all probability measures on G if and only if the support of the Fourier transform of the translation invariant kernel is the dual group of G. In our case, $(\mathbb{R}^d, +)$ and $(\mathbb{T}^d, +)$ are locally compact Abelian groups with $(\mathbb{R}^d, +)$ and $(\mathbb{Z}^d, +)$ as their respective dual groups. In Fukumizu et al. (2009b), these results are also extended to translation invariant kernels on non-Abelian compact groups and the semigroup $\mathbb{R}^d_+$.

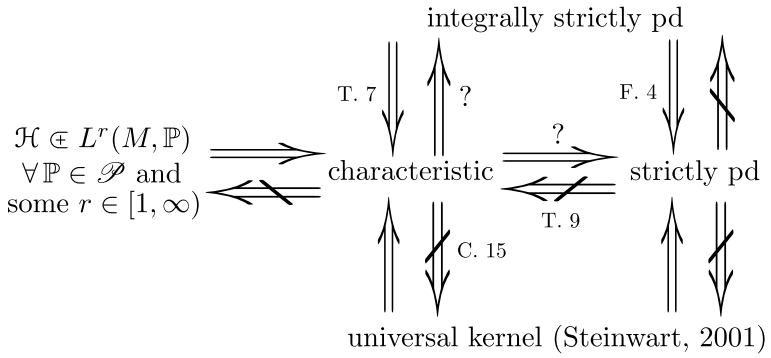


Figure 1: Summary of the relations between various families of kernels is shown along with the reference. The letters "C", "F", and "T" refer to Corollary, Footnote and Theorem respectively. For example, T. 7 refers to Theorem 7. The implications which are open problems are shown with "?". $A \Subset B$ indicates that A is a dense subset of B. Refer to Section 3.4 for details.

3.4 Overview of Relations Between Families of Kernels

So far, we have presented various characterizations of characteristic kernels, which are easily checkable compared with characterizations proposed in the earlier literature (Gretton et al., 2007b; Fukumizu et al., 2008, 2009b). We now provide an overview of various useful conditions one can impose on kernels (to be universal, strictly pd, integrally strictly pd, or characteristic), and the implications that relate some of these conditions. A summary is provided in Figure 1.

Characteristic kernels vs. Integrally strictly pd kernels: It is clear from Theorem 7 that integrally strictly pd kernels on a topological space M are characteristic, whereas the converse remains undetermined. When k is translation invariant on $\mathbb{R}^d$, however, then the converse holds. This is because if k is characteristic, then by Theorem 9, $\text{supp}(\Lambda) = \mathbb{R}^d$, where Λ is defined as in (11). It is easy to check that if $\text{supp}(\Lambda) = \mathbb{R}^d$, then k is integrally strictly pd.

Integrally strictly pd kernels vs. Strictly pd kernels: The relation between integrally strictly pd and strictly pd kernels shown in Figure 1 is straightforward, as one direction follows from Footnote 4, while the other direction is not true, which follows from Steinwart and Christmann (2008, Proposition 4.60, Theorem 4.62). However, if M is a finite set, then k being strictly pd also implies it is integrally strictly pd.

Characteristic kernels vs. Strictly pd kernels: Since integrally strictly pd kernels are characteristic and are also strictly pd, a natural question to ask is, "What is the relation between characteristic and strictly pd kernels?" It can be seen that strictly pd kernels need not be characteristic because the sinc-squared kernel, $k(x,y) = \frac{\sin^2(\sigma(x-y))}{(x-y)^2}$ on $\mathbb{R}$, which has $\text{supp}(\Lambda) = [-\sigma,\sigma] \subsetneq \mathbb{R}$ is strictly pd (Wendland, 2005, Theorem 6.11), while it is not characteristic by Theorem 9. However, for any general M, it is not clear whether k being characteristic implies that it is strictly pd. As a special case, if $M = \mathbb{R}^d$ or $M = \mathbb{T}^d$, then by Theorems 9 and 12, it follows that a translation invariant k being characteristic also implies that it is strictly pd.

Universal kernels vs. Characteristic kernels: Gretton et al. (2007b) have shown that if k is universal in the sense of Steinwart (2001), then it is characteristic. As mentioned in Section 3.3, the converse is not true, that is, if a kernel is characteristic, then it need not be universal, which

follows from Corollary 15. Note that in this case, M is assumed to be a compact metric space. The notion of universality of kernels was extended to non-compact domains by Micchelli et al. (2006): k is said to be universal on a non-compact Hausdorff space, M, if for any compact $Z \subset M$, the set $K(Z) := \overline{\text{span}}\{k(\cdot, y) : y \in Z\}$ is dense in $C_b(Z)$ w.r.t. the supremum norm. It is to be noted that when M is compact, this notion of universality is same as that of Steinwart (2001). Micchelli et al. (2006, Proposition 15) have provided a characterization of universality for translation invariant kernels on $\mathbb{R}^d$: k is universal if $\lambda(\text{supp}(\Lambda)) > 0$, where λ is the Lebesgue measure and Λ is defined as in (11). This means if a translation invariant kernel on $\mathbb{R}^d$ is characteristic, that is, $\text{supp}(\Lambda) = \mathbb{R}^d$, then it is also universal in the sense of Micchelli et al. (2006), while the converse is not true (e.g., sinc-squared kernel is not characteristic as $\text{supp}(\Lambda) = [-\sigma, \sigma] \subsetneq \mathbb{R}$ but universal in the sense of Micchelli as $\lambda(\text{supp}(\Lambda)) = 2\sigma > 0$). The relation between these notions for a general non-compact Hausdorff space M (other than $\mathbb{R}^d$) remains to be determined (whether or not the kernel is translation invariant).

Fukumizu et al. (2008, 2009b) have shown that k is characteristic if and only if $\mathcal{H} + \mathbb{R}$ is dense in $L^r(M, \mathbb{P})$ for all $\mathbb{P} \in \mathscr{P}$ and for some $r \in [1, \infty)$. Using this, it is easy to see that if $\mathcal{H}$ is dense in $L^r(M, \mathbb{P})$ for all $\mathbb{P} \in \mathscr{P}$ and for some $r \in [1, \infty)$, then k is characteristic. Clearly, the converse is not true. However, if constant functions are included in $\mathcal{H}$, then it is easy to see that the converse is also true.

Universal kernels vs. Strictly pd kernels: If a kernel is universal, then it is strictly pd, which follows from Steinwart and Christmann (2008, Definition 4.53, Proposition 4.54, Exercise 4.11). On the other hand, if a kernel is strictly pd, then it need not be universal, which follows from the results due to Dahmen and Micchelli (1987) and Pinkus (2004) for Taylor kernels (Steinwart and Christmann, 2008, Lemma 4.8, Corollary 4.57). Refer to Steinwart and Christmann (2008, Section 4.7, p. 161) for more details.

Recently, Sriperumbudur et al. (2010a,b) carried out a thorough study relating characteristic kernels to various notions of universality, addressing some open questions mentioned in the above discussion and Figure 1. This is done by relating universality to the injective embedding of regular Borel measures into an RKHS, which can therefore be seen as a generalization of the notion of characteristic kernels, as the latter deal with the injective RKHS embedding of probability measures.

3.5 Proofs

First, we present a supplementary result in Lemma 16 that will be used to prove Theorem 9. The idea of Lemma 16 is to characterize the equivalent conditions for the existence of $\mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$ when $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$. Its proof relies on the properties of characteristic functions, which we have collected in Theorem 27 in Appendix A.

Lemma 16 *Let $\mathscr{P}_0 := \{\mathbb{P} \in \mathscr{P} : \phi_{\mathbb{P}} \in L^1(\mathbb{R}^d) \cup L^2(\mathbb{R}^d)$ and $\mathbb{P} \ll \lambda\}$, where λ is the Lebesgue measure. Suppose k satisfies Assumption 1 and $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$, where Λ is defined as in (11). Then, for any $\mathbb{Q} \in \mathscr{P}_0$, $\exists\, \mathbb{P} \neq \mathbb{Q}$, $\mathbb{P} \in \mathscr{P}_0$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$ if and only if there exists a non-zero function $\theta : \mathbb{R}^d \to \mathbb{C}$ that satisfies the following conditions:*

(i) $\theta \in (L^1(\mathbb{R}^d) \cup L^2(\mathbb{R}^d)) \cap C_b(\mathbb{R}^d)$ *is conjugate symmetric, that is,* $\overline{\theta(x)} = \theta(-x), \forall x \in \mathbb{R}^d$,

(ii) $\theta^\vee \in L^1(\mathbb{R}^d) \cap (L^2(\mathbb{R}^d) \cup C_b(\mathbb{R}^d))$,

(iii) $\int_{\mathbb{R}^d} |\theta(x)|^2 \, d\Lambda(x) = 0$,

(iv) $\theta(0) = 0$,

(v) $\inf_{x\in\mathbb{R}^d}\{\theta^\vee(x) + q(x)\} \geq 0$.

Proof Define $L^1 := L^1(\mathbb{R}^d)$, $L^2 := L^2(\mathbb{R}^d)$ and $C_b := C_b(\mathbb{R}^d)$.

($\Leftarrow$) Suppose there exists a non-zero function θ satisfying *(i) – (v)*. For any $\mathbb{Q} \in \mathscr{P}_0$, we have $\phi_\mathbb{Q} \in L^1 \cup L^2$ and $\phi_\mathbb{Q} \in C_b$ (by Theorem 27), that is, $\phi_\mathbb{Q} \in (L^1 \cup L^2) \cap C_b$. Now, consider the case of $\phi_\mathbb{Q} \in L^1 \cap C_b$. Since $\phi_\mathbb{Q} \in L^1$, by the inversion theorem for characteristic functions (Dudley, 2002, Theorem 9.5.4), $\mathbb{Q}$ is absolutely continuous w.r.t. λ. If q is the Radon-Nikodym derivative of $\mathbb{Q}$ w.r.t. λ, then $q = [\overline{\phi_\mathbb{Q}}]^\vee \in L^1$. In addition, by the Riemann-Lebesgue lemma (Lemma 28 in Appendix A), we have $q \in C_0(\mathbb{R}^d) \subset C_b$, which therefore implies $q \in L^1 \cap C_b$. When $\phi_\mathbb{Q} \in L^2 \cap C_b$, the Fourier transform in the L^2 sense (see Footnote 6) implies that $q = [\overline{\phi_\mathbb{Q}}]^\vee \in L^1 \cap L^2$. Therefore, $q \in L^1 \cap (L^2 \cup C_b)$. Define $p := q + \theta^\vee$. Clearly $p \in L^1 \cap (L^2 \cup C_b)$. In addition, $\overline{\phi_\mathbb{P}} = \widehat{p} = \widehat{q} + \widehat{\theta^\vee} = \overline{\phi_\mathbb{Q}} + \theta \in (L^1 \cup L^2) \cap C_b$. Since θ is conjugate symmetric, $\theta^\vee$ is real valued and so is p. Consider

$$\int_{\mathbb{R}^d} p(x)\,dx = \int_{\mathbb{R}^d} q(x)\,dx + \int_{\mathbb{R}^d} \theta^\vee(x)\,dx = 1 + \theta(0) = 1.$$

(v) implies that p is non-negative. Therefore, p is the Radon-Nikodym derivative of a probability measure $\mathbb{P}$ w.r.t. λ, where $\mathbb{P}$ is such that $\mathbb{P} \neq \mathbb{Q}$ and $\mathbb{P} \in \mathscr{P}_0$. By (12), we have

$$\gamma_k^2(\mathbb{P},\mathbb{Q}) = \int_{\mathbb{R}^d} |\phi_\mathbb{P}(x) - \phi_\mathbb{Q}(x)|^2\, d\Lambda(x) = \int_{\mathbb{R}^d} |\theta(x)|^2\, d\Lambda(x) = 0.$$

($\Rightarrow$) Suppose that there exists $\mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}_0$ such that $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$. Define $\theta := \phi_\mathbb{P} - \phi_\mathbb{Q}$. We need to show that θ satisfies *(i) – (v)*. Recalling Theorem 27 in the appendix, $\mathbb{P}, \mathbb{Q} \in \mathscr{P}_0$ implies $\phi_\mathbb{P}, \phi_\mathbb{Q} \in (L^1 \cup L^2) \cap C_b$ and $p, q \in L^1 \cap (L^2 \cup C_b)$. Therefore, $\theta = \overline{\phi}_\mathbb{P} - \overline{\phi}_\mathbb{Q} \in (L^1 \cup L^2) \cap C_b$ and $\theta^\vee = p - q \in L^1 \cap (L^2 \cup C_b)$. By Theorem 27 (see Appendix A), $\phi_\mathbb{P}$ and $\phi_\mathbb{Q}$ are conjugate symmetric and so is θ. Therefore θ satisfies *(i)* and $\theta^\vee$ satisfies *(ii)*. θ satisfies *(iv)* as

$$\theta(0) = \int_{\mathbb{R}^d} \theta^\vee(x)\,dx = \int_{\mathbb{R}^d} (p(x) - q(x))\,dx = 0.$$

Non-negativity of p yields *(v)*. By (12), $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$ implies *(iii)*. ■

Remark 17 *Note that the dependence of* θ *on the kernel appears in the form of (iii) in Lemma 16. This condition shows that* $\lambda(\mathrm{supp}(\theta) \cap \mathrm{supp}(\Lambda)) = 0$*, that is, the supports of* θ *and* Λ *are disjoint w.r.t. the Lebesgue measure,* λ*. In other words,* $\mathrm{supp}(\theta) \subset \mathrm{cl}(\mathbb{R}^d \backslash \mathrm{supp}(\Lambda))$*. So, the idea is to introduce the perturbation,* θ *over an open set,* U *where* $\Lambda(U) = 0$*. The remaining conditions characterize the nature of this perturbation so that the constructed measure,* $p = q + \theta^\vee$*, is a valid probability measure. Conditions (i), (ii) and (iv) simply follow from* $\theta = \phi_\mathbb{P} - \phi_\mathbb{Q}$*, while (v) ensures that* $p(x) \geq 0, \forall x$*.*

Using Lemma 16, we now present the proof of Theorem 9.

Proof(Theorem 9) The sufficiency follows from (12): if $\mathrm{supp}(\Lambda) = \mathbb{R}^d$, then $\gamma_k^2(\mathbb{P},\mathbb{Q}) = \int_{\mathbb{R}^d} |\phi_\mathbb{P}(x) - \phi_\mathbb{Q}(x)|^2\, d\Lambda(x) = 0 \Rightarrow \phi_\mathbb{P} = \phi_\mathbb{Q}$, a.e. Recalling from Theorem 27 that $\phi_\mathbb{P}$ and $\phi_\mathbb{Q}$ are uniformly continuous on $\mathbb{R}^d$, we have that $\mathbb{P} = \mathbb{Q}$, and therefore k is characteristic. To prove necessity, we need to show that if $\mathrm{supp}(\Lambda) \subsetneq \mathbb{R}^d$, then there exists $\mathbb{P} \neq \mathbb{Q}$, $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$ such that $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$. By Lemma 16, this is equivalent to showing that there exists a non-zero θ satisfying the conditions in

Lemma 16. Below, we provide a constructive procedure for such a θ when $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$, thereby proving the result.

Consider the following function, $f_{\beta,\omega_0} \in C^\infty(\mathbb{R}^d)$ supported in $[\omega_0 - \beta, \omega_0 + \beta]$,

$$f_{\beta,\omega_0}(\omega) = \prod_{j=1}^{d} h_{\beta_j,\omega_{0,j}}(\omega_j) \text{ with } h_{a,b}(y) := \mathbb{1}_{[-a,a]}(y-b)\, e^{-\frac{a^2}{a^2-(y-b)^2}},$$

where $\omega = (\omega_1, \ldots, \omega_d)$, $\omega_0 = (\omega_{0,1}, \ldots, \omega_{0,d})$, $\beta = (\beta_1, \ldots, \beta_d)$, $a \in \mathbb{R}_{++}$, $b \in \mathbb{R}$ and $y \in \mathbb{R}$. Since $\text{supp}(\Lambda) \subsetneq \mathbb{R}^d$, there exists an open set $U \subset \mathbb{R}^d$ such that $\Lambda(U) = 0$. So, there exists $\beta \in \mathbb{R}^d_{++}$ and $\omega_0 > \beta$ (element-wise inequality) such that $[\omega_0 - \beta, \omega_0 + \beta] \subset U$. Let

$$\theta = \alpha(f_{\beta,\omega_0} + f_{\beta,-\omega_0}), \alpha \in \mathbb{R}\backslash\{0\},$$

which implies $\text{supp}(\theta) = [-\omega_0 - \beta, -\omega_0 + \beta] \cup [\omega_0 - \beta, \omega_0 + \beta]$ is compact. Clearly $\theta \in \mathscr{D}_d \subset \mathscr{S}_d$ which implies $\theta^\vee \in \mathscr{S}_d \subset L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d)$. Therefore, by construction, θ satisfies *(i) – (iv)* in Lemma 16. Since $\int_{\mathbb{R}^d} \theta^\vee(x)\, dx = \theta(0) = 0$ (by construction), $\theta^\vee$ will take negative values, so we need to show that there exists $\mathbb{Q} \in \mathscr{P}_0$ such that *(v)* in Lemma 16 holds. Let $\mathbb{Q}$ be such that it has a density given by

$$q(x) = C_l \prod_{j=1}^{d} \frac{1}{(1+|x_j|^2)^l}, l \in \mathbb{N} \text{ where } C_l = \prod_{j=1}^{d} \left(\int_{\mathbb{R}} (1+|x_j|^2)^{-l}\, dx_j \right)^{-1},$$

and $x = (x_1, \ldots, x_d)$. It can be verified that choosing α such that

$$0 < |\alpha| \leq \frac{C_l}{2 \sup_x \left| \prod_{j=1}^{d} h^\vee_{\beta_j,0}(x_j)(1+|x_j|^2)^l \cos(\omega_0^T x) \right|} < \infty,$$

ensures that θ satisfies *(v)* in Lemma 16. The existence of finite α is guaranteed as $h_{a,0} \in \mathscr{D}_1 \subset \mathscr{S}_1$ which implies $h^\vee_{a,0} \in \mathscr{S}_1, \forall a$. We conclude there exists a non-zero θ as claimed earlier, which completes the proof. ■

To elucidate the necessity part in the above proof, in the following, we present a simple example that provides an intuitive understanding about the construction of θ such that for a given $\mathbb{Q}, \mathbb{P} \neq \mathbb{Q}$ can be constructed with $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$.

Example 4 *Let $\mathbb{Q}$ be a Cauchy distribution in $\mathbb{R}$, that is, $q(x) = \frac{1}{\pi(1+x^2)}$ with characteristic function, $\phi_{\mathbb{Q}}(\omega) = \frac{1}{\sqrt{2\pi}} e^{-|\omega|}$ in $L^1(\mathbb{R})$. Let ψ be a Sinc kernel, that is, $\psi(x) = \sqrt{\frac{2}{\pi}} \frac{\sin(\beta x)}{x}$ with Fourier transform given by $\widehat{\psi}(\omega) = \mathbb{1}_{[-\beta,\beta]}(\omega)$ and $\text{supp}(\widehat{\psi}) = [-\beta, \beta] \subsetneq \mathbb{R}$. Let θ be*

$$\theta(\omega) = \frac{\alpha}{2i} \left[*_1^N \mathbb{1}_{\left[-\frac{\beta}{2}, \frac{\beta}{2}\right]}(\omega) \right] * [\delta(\omega - \omega_0) - \delta(\omega + \omega_0)],$$

where $|\omega_0| \geq \left(\frac{N+2}{2}\right)\beta, N \geq 2$ and $\alpha \neq 0$. $*_1^N$ represents the N-fold convolution. Note that θ is such that $\text{supp}(\theta) \cap \text{supp}(\widehat{\psi})$ is a null set w.r.t. the Lebesgue measure, which satisfies (iii) in Lemma 16. It is easy to verify that $\theta \in L^1(\mathbb{R}) \cap L^2(\mathbb{R}) \cap C_b(\mathbb{R})$ also satisfies conditions (i) and (iv) in Lemma 16. $\theta^\vee$ can be computed as

$$\theta^\vee(x) = \frac{2^N \alpha}{\sqrt{2\pi}} \sin(\omega_0 x) \frac{\sin^N\left(\frac{\beta x}{2}\right)}{x^N},$$

and $\theta^\vee \in L^1(\mathbb{R}) \cap L^2(\mathbb{R}) \cap C_b(\mathbb{R})$ *satisfies* (ii) *in Lemma 16. Choose*

$$0 < |\alpha| \leq \frac{\sqrt{2}}{\sqrt{\pi}\beta^N \sup_x \left| (1+x^2) \sin(\omega_0 x) \mathrm{sinc}^N\left(\frac{\beta x}{2\pi}\right)\right|},$$

where $\mathrm{sinc}(x) := \frac{\sin(\pi x)}{\pi x}$. *Define* $g(x) := \sin(\omega_0 x)\mathrm{sinc}^N\left(\frac{\beta x}{2\pi}\right)$. *Since* $g \in \mathscr{S}_1$, $0 < \sup_x |(1+x^2)g(x)| < \infty$ *and, therefore,* α *is a finite non-zero number. It is easy to see that* θ *satisfies* (v) *of Lemma 16. Then, by Lemma 16, there exists* $\mathbb{P} \neq \mathbb{Q}$, $\mathbb{P} \in \mathscr{P}_0$, *given by*

$$p(x) = \frac{1}{\pi(1+x^2)} + \frac{2^N \alpha}{\sqrt{2\pi}} \sin(\omega_0 x) \frac{\sin^N\left(\frac{\beta x}{2}\right)}{x^N},$$

with $\phi_{\mathbb{P}} = \phi_{\mathbb{Q}} + \theta = \phi_{\mathbb{Q}} + i\theta_I$ *where* $\theta_I = \mathrm{Im}[\theta]$ *and* $\phi_{\mathbb{P}} \in L^1(\mathbb{R})$. *So, we have constructed* $\mathbb{P} \neq \mathbb{Q}$, *such that* $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$. *Figure 2 shows the plots of* ψ, $\widehat{\psi}$, θ, $\theta^\vee$, q, $\phi_{\mathbb{Q}}$, p *and* $|\phi_{\mathbb{P}}|$ *for* $\beta = 2\pi$, $N = 2$, $\omega_0 = 4\pi$ *and* $\alpha = \frac{1}{50}$.

We now prove Theorem 12.

Proof(Theorem 12) Suppose $\exists\, \mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}_1$ such that $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$. Since any positive Borel measure on $\mathbb{R}^d$ is a distribution (Rudin, 1991, p. 157), $\mathbb{P}$ and $\mathbb{Q}$ can be treated as distributions with compact support. By the Paley-Wiener theorem (Theorem 29 in Appendix A), $\phi_{\mathbb{P}}$ and $\phi_{\mathbb{Q}}$ are restrictions to $\mathbb{R}^d$ of entire functions on $\mathbb{C}^d$. Let $\theta := \phi_{\mathbb{P}} - \phi_{\mathbb{Q}}$. Since $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$, we have from (12) that $\int_{\mathbb{R}^d} |\theta(\omega)|^2 d\Lambda(\omega) = 0$. From Remark 17, it follows that $\mathrm{supp}(\theta) \subset \mathrm{cl}(\mathbb{R}^d \backslash \mathrm{supp}(\Lambda))$. Since $\mathrm{supp}(\Lambda)$ has a non-empty interior, we have $\mathrm{supp}(\theta) \subsetneq \mathbb{R}^d$. Thus, there exists an open set, $U \subset \mathbb{R}^d$ such that $\theta(x) = 0, \forall x \in U$. Since θ is analytic on $\mathbb{R}^d$, we have $\theta = 0$, which means $\phi_{\mathbb{P}} = \phi_{\mathbb{Q}} \Rightarrow \mathbb{P} = \mathbb{Q}$, leading to a contradiction. So, there does not exist $\mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}_1$ such that $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$, and k is therefore characteristic to $\mathscr{P}_1$. ■

The condition that $\mathrm{supp}(\Lambda)$ has a non-empty interior is important for Theorem 12 to hold. In the following, we provide a simple example to show that $\mathbb{P} \neq \mathbb{Q}, \mathbb{P}, \mathbb{Q} \in \mathscr{P}_1$ can be constructed such that $\gamma_k(\mathbb{P},\mathbb{Q}) = 0$, if k is a periodic translation invariant kernel for which $\mathrm{int}(\mathrm{supp}(\Lambda)) = \emptyset$.

Example 5 *Let* $\mathbb{Q}$ *be a uniform distribution on* $[-\beta,\beta] \subset \mathbb{R}$, *that is,* $q(x) = \frac{1}{2\beta}\mathbb{1}_{[-\beta,\beta]}(x)$ *with its characteristic function,* $\phi_{\mathbb{Q}}(\omega) = \frac{1}{\beta\sqrt{2\pi}} \frac{\sin(\beta\omega)}{\omega} \in L^2(\mathbb{R})$. *Let* ψ *be the Dirichlet kernel with period* τ, *where* $\tau \leq \beta$, *that is,* $\psi(x) = \frac{\sin\frac{(2l+1)\pi x}{\tau}}{\sin\frac{\pi x}{\tau}}$ *and* $\widehat{\psi}(\omega) = \sqrt{2\pi}\sum_{j=-l}^{l} \delta\left(\omega - \frac{2\pi j}{\tau}\right)$ *with* $\mathrm{supp}(\widehat{\psi}) = \left\{\frac{2\pi j}{\tau} : j \in \{0, \pm 1, \ldots, \pm l\}\right\}$. *Clearly,* $\mathrm{supp}(\widehat{\psi})$ *has an empty interior. Let* θ *be*

$$\theta(\omega) = \frac{8\sqrt{2}\alpha}{i\sqrt{\pi}} \sin\left(\frac{\omega\tau}{2}\right) \frac{\sin^2\left(\frac{\omega\tau}{4}\right)}{\tau\omega^2},$$

with $\alpha \leq \frac{1}{2\beta}$. *It is easy to verify that* $\theta \in L^1(\mathbb{R}) \cap L^2(\mathbb{R}) \cap C_b(\mathbb{R})$, *so* θ *satisfies* (i) *in Lemma 16. Since* $\theta(\omega) = 0$ *at* $\omega = \frac{2\pi l}{\tau}$, $l \in \mathbb{Z}$, $\mathrm{supp}(\theta) \cap \mathrm{supp}(\widehat{\psi}) \subset \mathrm{supp}(\widehat{\psi})$ *is a set of Lebesgue measure zero, so* (iii) *and* (iv) *in Lemma 16 are satisfied.* $\theta^\vee$ *is given by*

$$\theta^\vee(x) = \begin{cases} \frac{2\alpha\left|x+\frac{\tau}{2}\right|}{\tau} - \alpha, & -\tau \leq x \leq 0 \\ \alpha - \frac{2\alpha\left|x-\frac{\tau}{2}\right|}{\tau}, & 0 \leq x \leq \tau \\ 0, & \text{otherwise,} \end{cases}$$

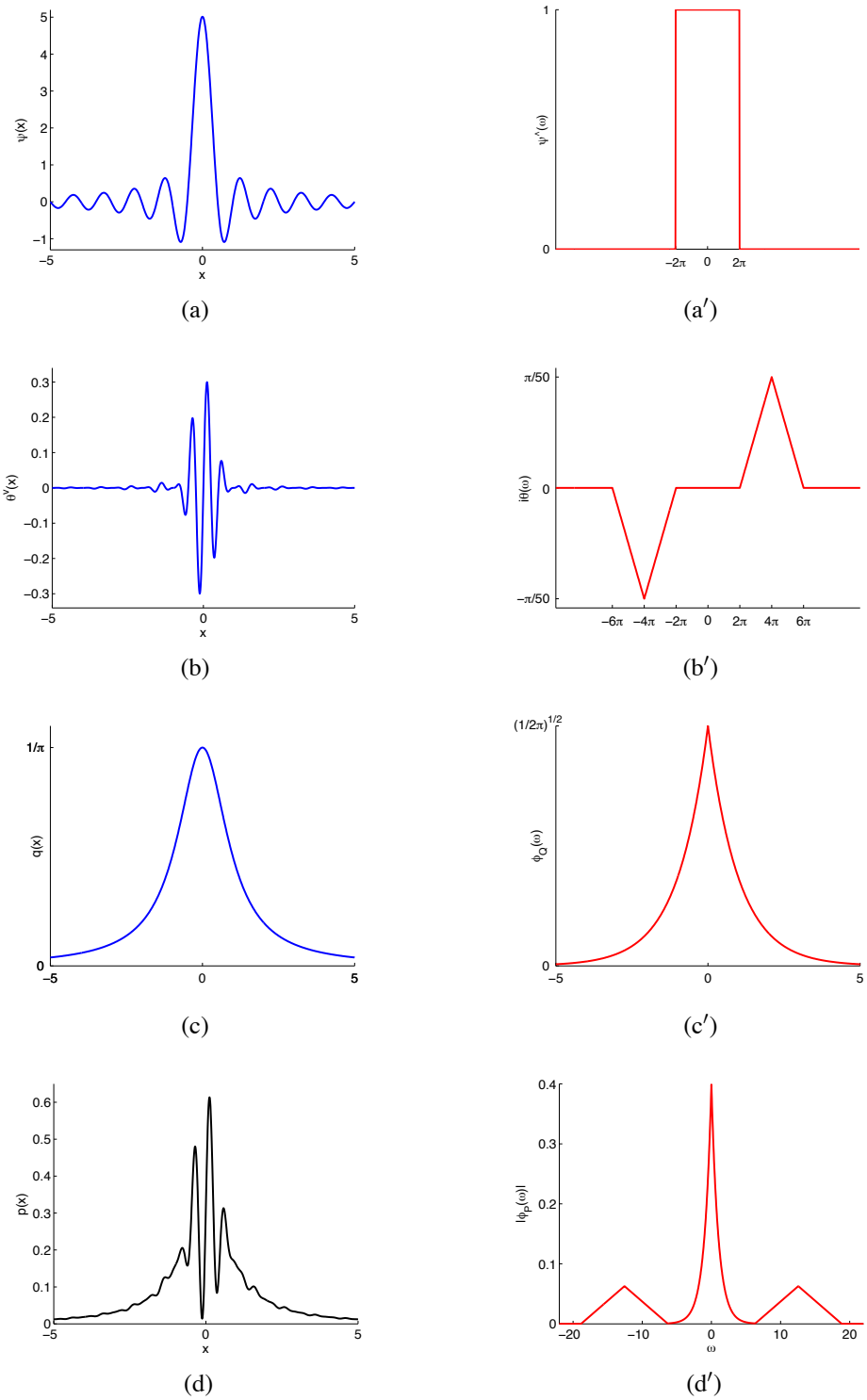


Figure 2: (a-a′) ψ and its Fourier spectrum $\widehat{\psi}$, (b-b′) $\theta^\vee$ and $i\theta$, (c-c′) the Cauchy distribution, q and its characteristic function $\phi_{\mathbb{Q}}$, and (d-d′) $p = q + \theta^\vee$ and $|\phi_{\mathbb{P}}|$. See Example 4 for details.

where $\theta^\vee \in L^1(\mathbb{R}) \cap L^2(\mathbb{R}) \cap C_b(\mathbb{R})$ *satisfies* (ii) *in Lemma 16. Now, consider* $p = q + \theta^\vee$, *which is given as*

$$p(x) = \begin{cases} \frac{1}{2\beta}, & x \in [-\beta, -\tau] \cup [\tau, \beta] \\ \frac{2\alpha\left|x+\frac{\tau}{2}\right|}{\tau} + \frac{1}{2\beta} - \alpha, & x \in [-\tau, 0] \\ \alpha + \frac{1}{2\beta} - \frac{2\alpha\left|x-\frac{\tau}{2}\right|}{\tau}, & x \in [0, \tau] \\ 0, & \text{otherwise}. \end{cases}$$

Clearly, $p(x) \geq 0, \forall x$ *and* $\int_{\mathbb{R}} p(x)\,dx = 1$. $\phi_{\mathbb{P}} = \phi_{\mathbb{Q}} + \theta = \phi_{\mathbb{Q}} + i\theta_I$ *where* $\theta_I = \mathrm{Im}[\theta]$ *and* $\phi_{\mathbb{P}} \in L^2(\mathbb{R})$. *We have therefore constructed* $\mathbb{P} \neq \mathbb{Q}$, *such that* $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$, *where* $\mathbb{P}$ *and* $\mathbb{Q}$ *are compactly supported in* $\mathbb{R}$ *with characteristic functions in* $L^2(\mathbb{R})$, *that is,* $\mathbb{P}, \mathbb{Q} \in \mathscr{P}_1$. *Figure 3 shows the plots of* ψ, $\widehat{\psi}$, θ, $\theta^\vee$, q, $\phi_{\mathbb{Q}}$, p *and* $|\phi_{\mathbb{P}}|$ *for* $\tau = 2$, $l = 2$, $\beta = 3$ *and* $\alpha = \frac{1}{8}$.

We now present the proof of Theorem 14, which is similar to that of Theorem 9.

Proof (Theorem 14) ($\Leftarrow$) From (10), we have

$$\begin{aligned} \gamma_k^2(\mathbb{P}, \mathbb{Q}) &= \iint_{\mathbb{T}^d} \psi(x-y)\,d(\mathbb{P}-\mathbb{Q})(x)\,d(\mathbb{P}-\mathbb{Q})(y) \\ &\overset{(a)}{=} \iint_{\mathbb{T}^d} \sum_{n\in\mathbb{Z}^d} A_\psi(n)\, e^{i(x-y)^T n}\,d(\mathbb{P}-\mathbb{Q})(x)\,d(\mathbb{P}-\mathbb{Q})(y) \\ &\overset{(b)}{=} \sum_{n\in\mathbb{Z}^d} A_\psi(n) \left| \int_{\mathbb{T}^d} e^{-ix^T n}\,d(\mathbb{P}-\mathbb{Q})(x) \right|^2 \\ &\overset{(c)}{=} (2\pi)^{2d} \sum_{n\in\mathbb{Z}^d} A_\psi(n)\,|A_{\mathbb{P}}(n) - A_{\mathbb{Q}}(n)|^2, \end{aligned} \tag{21}$$

where we have invoked Bochner's theorem (Theorem 13) in (a), Fubini's theorem in (b) and

$$A_{\mathbb{P}}(n) := \frac{1}{(2\pi)^d} \int_{\mathbb{T}^d} e^{-in^T x}\,d\mathbb{P}(x),\ n \in \mathbb{Z}^d,$$

in (c). $A_{\mathbb{P}}$ is the Fourier transform of $\mathbb{P}$ in $\mathbb{T}^d$. Since $A_\psi(0) \geq 0$ and $A_\psi(n) > 0, \forall n \neq 0$, we have $A_{\mathbb{P}}(n) = A_{\mathbb{Q}}(n), \forall n$. Therefore, by the uniqueness theorem of Fourier transform, we have $\mathbb{P} = \mathbb{Q}$.

($\Rightarrow$) Proving the necessity is equivalent to proving that if $A_\psi(0) \geq 0$, $A_\psi(n) > 0$, $\forall n \neq 0$ is violated, then k is not characteristic, which is equivalent to showing that $\exists\, \mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$. Let $\mathbb{Q}$ be a uniform probability measure with $q(x) = \frac{1}{(2\pi)^d}, \forall x \in \mathbb{T}^d$. Let k be such that $A_\psi(n) = 0$ for some $n = n_0 \neq 0$. Define

$$A_{\mathbb{P}}(n) := \begin{cases} A_{\mathbb{Q}}(n), & n \neq \pm n_0 \\ A_{\mathbb{Q}}(n) + \theta(n), & n = \pm n_0 \end{cases},$$

where $A_{\mathbb{Q}}(n) = \frac{1}{(2\pi)^d}\delta_{0n}$ and $\theta(-n_0) = \overline{\theta(n_0)}$. So,

$$p(x) = \sum_{n\in\mathbb{Z}^d} A_{\mathbb{P}}(n) e^{ix^T n} = \frac{1}{(2\pi)^d} + \theta(n_0) e^{ix^T n_0} + \theta(-n_0) e^{-ix^T n_0}.$$

Choose $\theta(n_0) = i\alpha$, $\alpha \in \mathbb{R}$. Then, $p(x) = \frac{1}{(2\pi)^d} - 2\alpha \sin(x^T n_0)$. It is easy to check that p integrates to one. Choosing $|\alpha| \leq \frac{1}{2(2\pi)^d}$ ensures that $p(x) \geq 0, \forall x \in \mathbb{T}^d$. By using $A_{\mathbb{P}}(n)$ in (21), it is clear that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$. Therefore, $\exists\, \mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) = 0$, which means k is not characteristic. ■

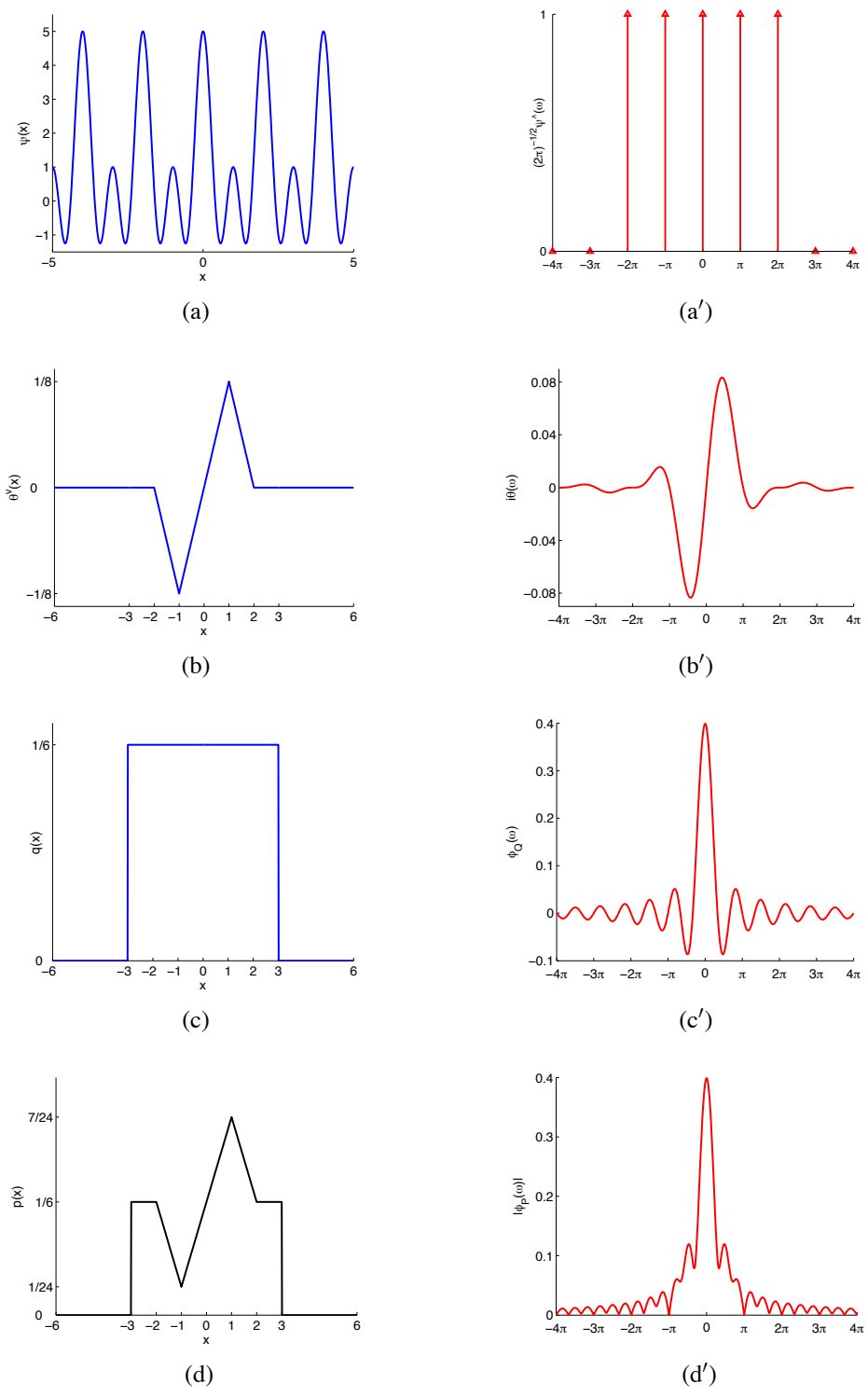


Figure 3: (a-a′) ψ and its Fourier spectrum $\widehat{\psi}$, (b-b′) $\theta^\vee$ and $i\theta$, (c-c′) the uniform distribution, q and its characteristic function $\phi_{\mathbb{Q}}$, and (d-d′) $p = q + \theta^\vee$ and $|\phi_{\mathbb{P}}|$. See Example 5 for details.

4. Dissimilar Distributions with Small γ_k

So far, we have studied different characterizations for the kernel k such that γ_k is a metric on $\mathscr{P}$. As mentioned in Section 1, the metric property of γ_k is crucial in many statistical inference applications like hypothesis testing. Therefore, in practice, it is important to use characteristic kernels. However, in this section, we show that characteristic kernels, while guaranteeing γ_k to be a metric on $\mathscr{P}$, may nonetheless have difficulty in distinguishing certain distributions on the basis of finite samples. More specifically, in Theorem 19 we show that for a given kernel k and for any $\varepsilon > 0$, there exist $\mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P},\mathbb{Q}) < \varepsilon$. Before proving the result, we motivate it through the following example.

Example 6 *Let $\mathbb{P}$ be absolutely continuous w.r.t. the Lebesgue measure on $\mathbb{R}$ with the Radon-Nikodym derivative defined as*

$$p(x) = q(x) + \alpha q(x)\sin(\nu\pi x), \tag{22}$$

where q is the Radon-Nikodym derivative of $\mathbb{Q}$ w.r.t. the Lebesgue measure satisfying $q(x) = q(-x)$, $\forall x$ and $\alpha \in [-1,1]\backslash\{0\}$, $\nu \in \mathbb{R}\backslash\{0\}$. It is obvious that $\mathbb{P} \neq \mathbb{Q}$. The characteristic function of $\mathbb{P}$ is given as

$$\phi_{\mathbb{P}}(\omega) = \phi_{\mathbb{Q}}(\omega) - \frac{i\alpha}{2}\left[\phi_{\mathbb{Q}}(\omega-\nu\pi) - \phi_{\mathbb{Q}}(\omega+\nu\pi)\right], \omega \in \mathbb{R},$$

where $\phi_{\mathbb{Q}}$ is the characteristic function associated with $\mathbb{Q}$. Note that with increasing $|\nu|$, p has higher frequency components in its Fourier spectrum, as shown in Figure 4. In Figure 4, (a-c) show the plots of p when $q = \mathcal{U}[-1,1]$ (uniform distribution) and (a′-c′) show the plots of p when $q = \mathcal{N}(0,2)$ (zero mean normal distribution with variance 2) for $\nu = 0, 2$ and 7.5 with $\alpha = \frac{1}{2}$.

Consider the B_1-spline kernel on $\mathbb{R}$ given by $k(x,y) = \psi(x-y)$ where

$$\psi(x) = \begin{cases} 1-|x|, & |x| \leq 1 \\ 0, & \text{otherwise} \end{cases}, \tag{23}$$

with its Fourier transform given by

$$\widehat{\psi}(\omega) = \frac{2\sqrt{2}}{\sqrt{\pi}}\frac{\sin^2\frac{\omega}{2}}{\omega^2}.$$

Since ψ is characteristic to $\mathscr{P}$, $\gamma_k(\mathbb{P},\mathbb{Q}) > 0$ (see Theorem 9). However, it would be of interest to study the behavior of $\gamma_k(\mathbb{P},\mathbb{Q})$ as a function of ν. We study the behavior of $\gamma_k^2(\mathbb{P},\mathbb{Q})$ through its unbiased, consistent estimator,[7] $\gamma_{k,u}^2(m,m)$ as considered by Gretton et al. (2007b, Lemma 7).

Figure 5(a) shows the behavior of $\gamma_{k,u}^2(m,m)$ as a function of ν for $q = \mathcal{U}[-1,1]$ and $q = \mathcal{N}(0,2)$ using the B_1-spline kernel in (23). Since the Gaussian kernel, $k(x,y) = e^{-(x-y)^2}$ is also a characteristic kernel, its effect on the behavior of $\gamma_{k,u}^2(m,m)$ is shown in Figure 5(b) in comparison to that of the B_1-spline kernel.

In Figure 5, we observe two circumstances under which γ_k^2 may be small. First, $\gamma_{k,u}^2(m,m)$ decays with increasing $|\nu|$, and can be made as small as desired by choosing a sufficiently large $|\nu|$. Second,

7. Let $\{X_j\}_{j=1}^m$ and $\{Y_j\}_{j=1}^m$ be random samples drawn i.i.d. from $\mathbb{P}$ and $\mathbb{Q}$ respectively. An unbiased *empirical estimate* of $\gamma_k^2(\mathbb{P},\mathbb{Q})$, denoted as $\gamma_{k,u}^2(m,m)$ is given by $\gamma_{k,u}^2(m,m) = \frac{1}{m(m-1)}\sum_{l\neq j}^m h(Z_l,Z_j)$, which is a one-sample U-statistic with $h(Z_l,Z_j) := k(X_l,X_j) + k(Y_l,Y_j) - k(X_l,Y_j) - k(X_j,Y_l)$, where $Z_1,\ldots,Z_m$ are m i.i.d. random variables with $Z_j := (X_j,Y_j)$. See Gretton et al. (2007b, Lemma 7) for details.

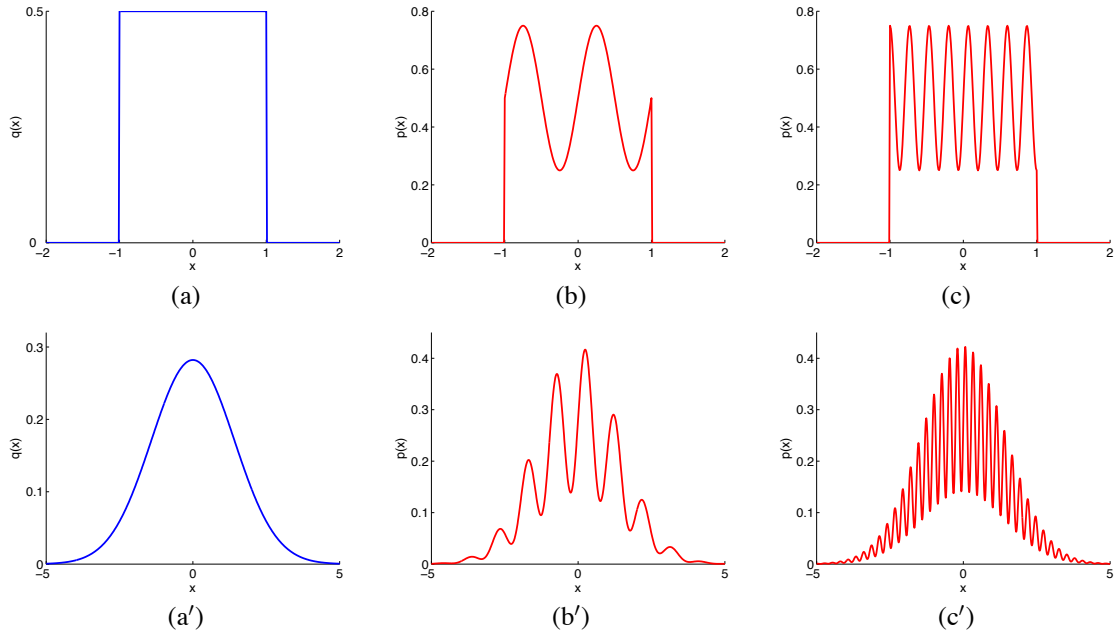


Figure 4: (a) $q = \mathcal{U}[-1,1]$, (a′) $q = \mathcal{N}(0,2)$. (b-c) and (b′-c′) denote $p(x)$ computed as $p(x) = q(x) + \frac{1}{2}q(x)\sin(\nu\pi x)$ with $q = \mathcal{U}[-1,1]$ and $q = \mathcal{N}(0,2)$ respectively. ν is chosen to be 2 in (b,b′) and 7.5 in (c,c′). See Example 6 for details.

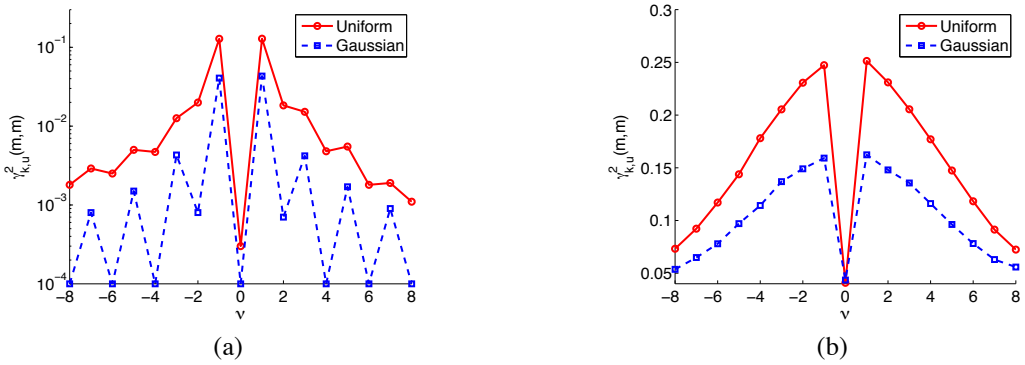


Figure 5: Behavior of the empirical estimate of $\gamma_k^2(\mathbb{P},\mathbb{Q})$ w.r.t. ν for (a) the B_1-spline kernel and (b) the Gaussian kernel. $\mathbb{P}$ is constructed from $\mathbb{Q}$ as defined in (22). "Uniform" corresponds to $\mathbb{Q} = \mathcal{U}[-1,1]$ and "Gaussian" corresponds to $\mathbb{Q} = \mathcal{N}(0,2)$. $m = 1000$ samples are generated from $\mathbb{P}$ and $\mathbb{Q}$ to estimate $\gamma_k^2(\mathbb{P},\mathbb{Q})$ through $\gamma_{k,u}^2(m,m)$. This is repeated 100 times and the average $\gamma_{k,u}^2(m,m)$ is plotted in both figures. Since the quantity of interest is the average behavior of $\gamma_{k,u}^2(m,m)$, we omit the error bars. See Example 6 for details.

in Figure 5(a), $\gamma_{k,u}^2(m,m)$ has troughs at $\nu = \frac{\omega_0}{\pi}$ where $\omega_0 = \{\omega : \widehat{\psi}(\omega) = 0\}$. Since $\gamma_{k,u}^2(m,m)$ is a consistent estimate of $\gamma_k^2(\mathbb{P},\mathbb{Q})$, one would expect similar behavior from $\gamma_k^2(\mathbb{P},\mathbb{Q})$. This means that, although the B_1-spline kernel is characteristic to $\mathscr{P}$, in practice, it becomes harder to distinguish

between $\mathbb{P}$ and $\mathbb{Q}$ with finite samples, when $\mathbb{P}$ is constructed as in (22) with $\nu = \frac{\omega_0}{\pi}$. In fact, one can observe from a straightforward spectral argument that the troughs in $\gamma_k^2(\mathbb{P},\mathbb{Q})$ can be made arbitrarily deep by widening q, when q is Gaussian.

For characteristic kernels, although $\gamma_k(\mathbb{P},\mathbb{Q}) > 0$ when $\mathbb{P} \neq \mathbb{Q}$, Example 6 demonstrates that one can construct distributions such that $\gamma_{k,u}^2(m,m)$ is indistinguishable from zero with high probability, for a given sample size m. Below, in Theorem 19, we explicitly construct $\mathbb{P} \neq \mathbb{Q}$ such that $|\mathbb{P}\varphi_l - \mathbb{Q}\varphi_l|$ is large for some large l, but $\gamma_k(\mathbb{P},\mathbb{Q})$ is arbitrarily small, making it hard to detect a non-zero value of $\gamma_k(\mathbb{P},\mathbb{Q})$ based on finite samples. Here, $\varphi_l \in L^2(M)$ represents the bounded orthonormal eigenfunctions of a positive definite integral operator associated with k. Based on this theorem, for example, in Example 6, the decay mode of γ_k for large $|\nu|$ can be investigated.

Consider the formulation of $\gamma_{\mathcal{F}}$ with $\mathcal{F} = \mathcal{F}_k$ in (1). The construction of $\mathbb{P}$ for a given $\mathbb{Q}$ such that $\gamma_k(\mathbb{P},\mathbb{Q})$ is small, though not zero, can be intuitively understood by re-writing (1) as

$$\gamma_k(\mathbb{P},\mathbb{Q}) = \sup_{f \in \mathcal{H}} \frac{|\mathbb{P}f - \mathbb{Q}f|}{\|f\|_{\mathcal{H}}}.$$

When $\mathbb{P} \neq \mathbb{Q}$, $|\mathbb{P}f - \mathbb{Q}f|$ can be large for some $f \in \mathcal{H}$. However, $\gamma_k(\mathbb{P},\mathbb{Q})$ can be made small by selecting $\mathbb{P}$ such that the maximization of $\frac{|\mathbb{P}f-\mathbb{Q}f|}{\|f\|_{\mathcal{H}}}$ over $\mathcal{H}$ requires an f with large $\|f\|_{\mathcal{H}}$. More specifically, higher order eigenfunctions of the kernel (φ_l for large l) have large RKHS norms, so, if they are prominent in $\mathbb{P}$ and $\mathbb{Q}$ (i.e., highly non-smooth distributions), one can expect $\gamma_k(\mathbb{P},\mathbb{Q})$ to be small even when there exists an l for which $|\mathbb{P}\varphi_l - \mathbb{Q}\varphi_l|$ is large. To this end, we need the following lemma, which we quote from Gretton et al. (2005b, Lemma 4).

Lemma 18 (Gretton et al., 2005b) *Let $\mathcal{F}$ be the unit ball in an RKHS $(\mathcal{H},k)$ defined on a compact topological space, M, with k being measurable. Let $\varphi_l \in L^2(M,\mu)$ be absolutely bounded orthonormal eigenfunctions and λ_l be the corresponding eigenvalues (arranged in decreasing order for increasing l) of a positive definite integral operator associated with k and a σ-finite measure, μ. Assume λ_l^{-1} increases super-linearly with l. Then, for $f \in \mathcal{F}$ where $f(x) = \sum_{j=1}^{\infty} \widetilde{f}_j \varphi_j(x)$, $\widetilde{f}_j := \langle f, \varphi_j \rangle_{L^2(M,\mu)}$, we have $\sum_{j=1}^{\infty} |\widetilde{f}_j| < \infty$ and for every $\varepsilon > 0$, $\exists\, l_0 \in \mathbb{N}$ such that $|\widetilde{f}_l| < \varepsilon$ if $l > l_0$.*

Theorem 19 ($\mathbb{P} \neq \mathbb{Q}$ can have arbitrarily small γ_k) *Suppose the conditions in Lemma 18 hold. Then there exist probability measures $\mathbb{P} \neq \mathbb{Q}$ defined on M such that $\gamma_k(\mathbb{P},\mathbb{Q}) < \varepsilon$ for any arbitrarily small $\varepsilon > 0$.*

Proof Suppose q be the Radon-Nikodym derivative associated with $\mathbb{Q}$ w.r.t. the σ-finite measure, μ (see Lemma 18). Let us construct $p(x) = q(x) + \alpha_l e(x) + \tau\varphi_l(x)$ where $e(x) = \mathbb{1}_M(x)$. For $\mathbb{P}$ to be a probability measure, the following conditions need to be satisfied:

$$\begin{aligned} \int_M [\alpha_l e(x) + \tau\varphi_l(x)]\, d\mu(x) &= 0, \\ \min_{x \in M} [q(x) + \alpha_l e(x) + \tau\varphi_l(x)] &\geq 0. \end{aligned} \tag{24}$$

Expanding $e(x)$ and $f(x)$ in the orthonormal basis $\{\varphi_l\}_{l=1}^{\infty}$, we get $e(x) = \sum_{l=1}^{\infty} \widetilde{e}_l \varphi_l(x)$ and $f(x) = \sum_{l=1}^{\infty} \widetilde{f}_l \varphi_l(x)$, where $\widetilde{e}_l := \langle e, \varphi_l \rangle_{L^2(M,\mu)}$ and $\widetilde{f}_l := \langle f, \varphi_l \rangle_{L^2(M,\mu)}$. Therefore,

$$\begin{aligned} \mathbb{P}f - \mathbb{Q}f &= \int_M f(x)\, [\alpha_l e(x) + \tau\varphi_l(x)]\, d\mu(x) \\ &= \int_M \left[\alpha_l \sum_{j=1}^{\infty} \widetilde{e}_j \varphi_j(x) + \tau\varphi_l(x)\right] \left[\sum_{t=1}^{\infty} \widetilde{f}_t \varphi_t(x)\right] d\mu(x) = \alpha_l \sum_{j=1}^{\infty} \widetilde{e}_j \widetilde{f}_j + \tau \widetilde{f}_l, \end{aligned} \tag{25}$$

where we used the fact that $\langle \varphi_j, \varphi_t \rangle_{L^2(M,\mu)} = \delta_{jt}$ (here, δ is used in the Kronecker sense). Rewriting (24) and substituting for $e(x)$ gives

$$\int_M [\alpha_l e(x) + \tau\varphi_l(x)]\, d\mu(x) = \int_M e(x)[\alpha_l e(x) + \tau\varphi_l(x)]\, d\mu(x) = \alpha_l \sum_{j=1}^{\infty} \widetilde{e}_j^2 + \tau\widetilde{e}_l = 0,$$

which implies

$$\alpha_l = -\frac{\tau\widetilde{e}_l}{\sum_{j=1}^{\infty} \widetilde{e}_j^2}. \tag{26}$$

Now, let us consider $\mathbb{P}\varphi_t - \mathbb{Q}\varphi_t = \alpha_l \widetilde{e}_t + \tau\delta_{tl}$. Substituting for α_l gives

$$\mathbb{P}\varphi_t - \mathbb{Q}\varphi_t = \tau\delta_{tl} - \tau\frac{\widetilde{e}_t\widetilde{e}_l}{\sum_{j=1}^{\infty} \widetilde{e}_j^2} = \tau\delta_{tl} - \tau\rho_{tl},$$

where $\rho_{tl} := \frac{\widetilde{e}_t\widetilde{e}_l}{\sum_{j=1}^{\infty} \widetilde{e}_j^2}$. By Lemma 18, $\sum_{l=1}^{\infty} |\widetilde{e}_l| < \infty \Rightarrow \sum_{j=1}^{\infty} \widetilde{e}_j^2 < \infty$, and choosing large enough l gives $|\rho_{tl}| < \eta, \forall t$, for any arbitrary $\eta > 0$. Therefore, $|\mathbb{P}\varphi_t - \mathbb{Q}\varphi_t| > \tau - \eta$ for $t = l$ and $|\mathbb{P}\varphi_t - \mathbb{Q}\varphi_t| < \eta$ for $t \neq l$, which means $\mathbb{P} \neq \mathbb{Q}$. In the following, we prove that $\gamma_k(\mathbb{P}, \mathbb{Q})$ can be arbitrarily small, though non-zero.

Recall that $\gamma_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} |\mathbb{P}f - \mathbb{Q}f|$. Substituting (26) in (25) and replacing $|\mathbb{P}f - \mathbb{Q}f|$ by (25) in $\gamma_k(\mathbb{P}, \mathbb{Q})$, we have

$$\gamma_k(\mathbb{P}, \mathbb{Q}) = \sup_{\{\widetilde{f}_j\}_{j=1}^{\infty}} \left\{ \tau \sum_{j=1}^{\infty} \nu_{jl} \widetilde{f}_j \ : \ \sum_{j=1}^{\infty} \frac{\widetilde{f}_j^2}{\lambda_j} \leq 1 \right\}, \tag{27}$$

where we used the definition of RKHS norm as $\|f\|_{\mathcal{H}} := \sum_{j=1}^{\infty} \frac{\widetilde{f}_j^2}{\lambda_j}$ and $\nu_{jl} := \delta_{jl} - \rho_{jl}$. (27) is a convex quadratically constrained quadratic program in $\{\widetilde{f}_j\}_{j=1}^{\infty}$. Solving the Lagrangian yields $\widetilde{f}_j = \frac{\nu_{jl}\lambda_j}{\sqrt{\sum_{j=1}^{\infty} \nu_{jl}^2 \lambda_j}}$. Therefore,

$$\gamma_k(\mathbb{P}, \mathbb{Q}) = \tau \sqrt{\sum_{j=1}^{\infty} \nu_{jl}^2 \lambda_j} = \tau \sqrt{\lambda_l - 2\rho_{ll}\lambda_l + \sum_{j=1}^{\infty} \rho_{jl}^2 \lambda_j} \xrightarrow{l \to \infty} 0,$$

because *(i)* by choosing sufficiently large l, $|\rho_{jl}| < \varepsilon, \forall\, j$, for any arbitrary $\varepsilon > 0$, and *(ii)* $\lambda_l \to 0$ as $l \to \infty$ (Schölkopf and Smola, 2002, Theorem 2.10). Therefore, we have constructed $\mathbb{P} \neq \mathbb{Q}$ such that $\gamma_k(\mathbb{P}, \mathbb{Q}) < \varepsilon$ for any arbitrarily small $\varepsilon > 0$. ■

5. Metrization of the Weak Topology

So far, we have shown that a characteristic kernel k induces a metric γ_k on $\mathscr{P}$. As motivated in Section 1.1.3, an important question to consider that is useful both in theory and practice would be: "How strong or weak is γ_k related to other metrics on $\mathscr{P}$?" This question is addressed in Theorem 21, where we compare γ_k to other metrics on $\mathscr{P}$ like the Dudley metric (β), Wasserstein distance (W), total variation distance (TV), and show that γ_k is weaker than all these metrics (see Footnote 3 for the definition of "strong" and "weak" metrics). Since γ_k is weaker than the Dudley metric, which

is known to induce a topology on $\mathscr{P}$ that coincides with the standard topology on $\mathscr{P}$, called the weak-* (weak-star) topology (usually called the weak topology in probability theory), the next question we are interested in is to understand the topology that is being induced by γ_k. In particular, we are interested in determining the conditions on k for which the topology induced by γ_k coincides with the weak topology on $\mathscr{P}$. This is answered in Theorems 23 and 24, where Theorem 23 deals with compact M and Theorem 24 provides a sufficient condition on k when $M = \mathbb{R}^d$. The proofs of all these results are provided in Section 5.1. Before we motivate the need for this study and its implications, we present some preliminaries.

The *weak topology* on $\mathscr{P}$ is the weakest topology such that the map $\mathbb{P} \mapsto \int_M f\,d\mathbb{P}$ is continuous for all $f \in C_b(M)$. For a metric space (M,ρ), a sequence $\mathbb{P}_n$ of probability measures is said to *converge weakly* to $\mathbb{P}$, written as $\mathbb{P}_n \xrightarrow{w} \mathbb{P}$, if and only if $\int_M f\,d\mathbb{P}_n \to \int_M f\,d\mathbb{P}$ for every $f \in C_b(M)$. A metric γ on $\mathscr{P}$ is said to *metrize* the weak topology if the topology induced by γ coincides with the weak topology, which is defined as follows: if, for $\mathbb{P},\mathbb{P}_1,\mathbb{P}_2,\ldots \in \mathscr{P}$, $(\mathbb{P}_n \xrightarrow{w} \mathbb{P} \Leftrightarrow \gamma(\mathbb{P}_n,\mathbb{P}) \xrightarrow{n\to\infty} 0)$ holds, then the topology induced by γ coincides with the weak topology.

In the following, we collect well-known results on the relation between various metrics on $\mathscr{P}$, which will be helpful in understanding the behavior of these metrics, both with respect to each other and to ours. Let (M,ρ) be a separable metric space. The *Prohorov metric* on (M,ρ), defined as

$$\varsigma(\mathbb{P},\mathbb{Q}) := \inf\{\varepsilon > 0 : \mathbb{P}(A) \le \mathbb{Q}(A^\varepsilon) + \varepsilon, \forall \text{Borel sets } A\},$$

metrizes the weak topology on $\mathscr{P}$ (Dudley, 2002, Theorem 11.3.3), where $\mathbb{P},\mathbb{Q} \in \mathscr{P}$ and $A^\varepsilon := \{y \in M : \rho(x,y) < \varepsilon \text{ for some } x \in A\}$. Since the Dudley metric is related to the Prohorov metric as

$$\frac{1}{2}\beta(\mathbb{P},\mathbb{Q}) \le \varsigma(\mathbb{P},\mathbb{Q}) \le 2\sqrt{\beta(\mathbb{P},\mathbb{Q})}, \tag{28}$$

it also metrizes the weak topology on $\mathscr{P}$ (Dudley, 2002, Theorem 11.3.3). The Wasserstein distance and total variation distance are related to the Prohorov metric as

$$\varsigma^2(\mathbb{P},\mathbb{Q}) \le W(\mathbb{P},\mathbb{Q}) \le (\operatorname{diam}(M)+1)\varsigma(\mathbb{P},\mathbb{Q}), \tag{29}$$

and

$$\varsigma(\mathbb{P},\mathbb{Q}) \le TV(\mathbb{P},\mathbb{Q}),$$

where $\operatorname{diam}(M) := \sup\{\rho(x,y) : x,y \in M\}$ (Gibbs and Su, 2002, Theorem 2). This means W and TV are stronger than ς, while W and ς are equivalent (i.e., induce the same topology) when M is bounded. By Theorem 4 in Gibbs and Su (2002), TV and W are related as

$$W(\mathbb{P},\mathbb{Q}) \le \operatorname{diam}(M) TV(\mathbb{P},\mathbb{Q}),$$

which means W and TV are comparable if M is bounded. See Shorack (2000, Chapter 19, Theorem 2.4) and Gibbs and Su (2002) for further detail on the relationship between various metrics on $\mathscr{P}$.

Let us now consider a sequence of of probability measures on $\mathbb{R}$, $\mathbb{P}_n := \left(1-\frac{1}{n}\right)\delta_0 + \frac{1}{n}\delta_n$ and let $\mathbb{P} := \delta_0$. It can be shown that $\beta(\mathbb{P}_n,\mathbb{P}) \to 0$ as $n \to \infty$ which means $\mathbb{P}_n \xrightarrow{w} \mathbb{P}$, while $W(\mathbb{P}_n,\mathbb{P}) = 1$ and $TV(\mathbb{P}_n,\mathbb{P}) = 1$ for all n. $\gamma_k(\mathbb{P}_n,\mathbb{P})$ can be computed as

$$\gamma_k^2(\mathbb{P}_n,\mathbb{P}) = \frac{1}{n^2}\iint_{\mathbb{R}} k(x,y)\,d(\delta_0-\delta_n)(x)\,d(\delta_0-\delta_n)(y) = \frac{k(0,0)+k(n,n)-2k(0,n)}{n^2}.$$

If k is, for example, a Gaussian, Laplacian or inverse multiquadratic kernel, then $\gamma_k(\mathbb{P}_n, \mathbb{P}) \to 0$ as $n \to \infty$. This example shows that γ_k is weaker than W and TV. It also shows that, for certain choices of k, γ_k behaves similarly to β, which leads to several questions: Does γ_k metrize the weak topology on $\mathscr{P}$? What is the general behavior of γ_k compared to other metrics? In other words, depending on k, how weak or strong is γ_k compared to other metrics on $\mathscr{P}$? Understanding the answer to these questions is important both in theory and practice. If k is such that γ_k metrizes the weak topology on $\mathscr{P}$, then it can be used as a theoretical tool in probability theory, similar to the Prohorov and Dudley metrics. On the other hand, the answer to these questions is critical in applications as it will have a bearing on the choice of kernels to be used. In applications like density estimation, one would need a strong metric to ascertain that the density estimate is a good representation of the true underlying density. For this reason, the total variation distance, Hellinger distance or Kullback-Leibler distance are generally used. However, it is not always possible to show the convergence of a density estimate to the true underlying density using a stronger metric and so, in such cases, one would need a weak metric to analyze the quality of estimate. Therefore, studying the relation between γ_k and these other metrics will provide an understanding of the choice of kernels to be used, depending on the application.

With the above motivation, we first compare γ_k to β, W and TV. Since β is equivalent to ς, we do not compare γ_k to ς. Before we provide the main result in Theorem 21 that compares γ_k to other metrics, we present an upper bound on γ_k in terms of the coupling formulation (Dudley, 2002, Section 11.8), which is not only useful in deriving the main result but also interesting in its own right.

Proposition 20 (Coupling bound) *Let k be measurable and bounded on M. Then, for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$,*

$$\gamma_k(\mathbb{P}, \mathbb{Q}) \leq \inf_{\mu \in \mathcal{L}(\mathbb{P}, \mathbb{Q})} \iint_M \|k(\cdot, x) - k(\cdot, y)\|_{\mathcal{H}} d\mu(x, y), \tag{30}$$

where $\mathcal{L}(\mathbb{P}, \mathbb{Q})$ represents the set of all laws on $M \times M$ with marginals $\mathbb{P}$ and $\mathbb{Q}$.

Proof For any $\mu \in \mathcal{L}(\mathbb{P}, \mathbb{Q})$, we have

$$\begin{aligned} \left| \int_M f \, d(\mathbb{P} - \mathbb{Q}) \right| &= \left| \iint_M (f(x) - f(y)) \, d\mu(x, y) \right| \leq \iint_M |f(x) - f(y)| \, d\mu(x, y) \\ &= \iint_M |\langle f, k(\cdot, x) - k(\cdot, y) \rangle_{\mathcal{H}}| \, d\mu(x, y) \leq \|f\|_{\mathcal{H}} \iint_M \|k(\cdot, x) - k(\cdot, y)\|_{\mathcal{H}} d\mu(x, y). \end{aligned} \tag{31}$$

Taking the supremum over $f \in \mathcal{F}_k$ and the infimum over $\mu \in \mathcal{L}(\mathbb{P}, \mathbb{Q})$ in (31), where $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$, gives the result in (30). ■

We now present the main result that compares γ_k to β, W and TV.

Theorem 21 (Comparison of γ_k to β, W and TV) *Assume $\sup_{x \in M} k(x, x) \leq C < \infty$, where k is measurable on M. Let*

$$\widetilde{\rho}(x, y) = \|k(\cdot, x) - k(\cdot, y)\|_{\mathcal{H}}. \tag{32}$$

Then, for any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$,

(i) $\gamma_k(\mathbb{P}, \mathbb{Q}) \leq W(\mathbb{P}, \mathbb{Q}) \leq \sqrt{\gamma_k^2(\mathbb{P}, \mathbb{Q}) + 4C}$ *if $(M, \widetilde{\rho})$ is separable.*

(ii) $\frac{\gamma_k(\mathbb{P},\mathbb{Q})}{(1+\sqrt{C})} \leq \beta(\mathbb{P},\mathbb{Q}) \leq 2(\gamma_k^2(\mathbb{P},\mathbb{Q})+4C)^{\frac{1}{3}}$ *if* $(M,\widetilde{\rho})$ *is separable.*

(iii) $\gamma_k(\mathbb{P},\mathbb{Q}) \leq \sqrt{C}\,TV(\mathbb{P},\mathbb{Q})$.

The proof is provided in Section 5.1. Below are some remarks on Theorem 21.

Remark 22 *(a) First, note that, since k is bounded,* $(M,\widetilde{\rho})$ *is a bounded metric space. In addition, the metric,* $\widetilde{\rho}$*, which depends on the kernel as in (32), is a Hilbertian metric*[8] *(Berg et al., 1984, Chapter 3, Section 3) on M. A popular example of such a metric is* $\widetilde{\rho}(x,y) = \|x-y\|_2$*, which can be obtained by choosing M to be a compact subset of* $\mathbb{R}^d$ *and* $k(x,y) = x^T y$.

(b) Theorem 21 shows that γ_k *is weaker than* β*, W and TV for the assumptions being made on k and* $\widetilde{\rho}$*. Note that the result holds irrespective of whether or not the kernel is characteristic, as we have not assumed anything about the kernel except it being measurable and bounded. Also, it is important to remember that the result holds when* $\widetilde{\rho}$ *is Hilbertian, as mentioned in (32) (see Remark 22(d)).*

(c) Apart from showing that γ_k *is weaker than* β*, W and TV, the result in Theorem 21 can be used to bound these metrics in terms of* γ_k*. For* β*, which is primarily of theoretical interest, we do not know a closed form expression, and likewise a closed form expression for W is known only for* $\mathbb{R}$ *(Vallander, 1973).*[9] *Since* γ_k *is easy to compute (see (9) and (10)), bounds on W can be obtained from Theorem 21 in terms of* γ_k*. A closed form expression for TV is available if* $\mathbb{P}$ *and* $\mathbb{Q}$ *have Radon-Nikodym derivatives w.r.t. a* σ*-finite measure. However, from Theorem 21, a simple lower bound can be obtained on TV in terms of* γ_k *for any* $\mathbb{P},\mathbb{Q} \in \mathscr{P}$.

(d) In Theorem 21, the kernel is fixed and $\widetilde{\rho}$ *is defined as in (32), which is a Hilbertian metric. On the other hand, suppose a Hilbertian metric* $\widetilde{\rho}$ *is given. Then the associated kernel k can be obtained from* $\widetilde{\rho}$ *(Berg et al., 1984, Chapter 3, Lemma 2.1) as*

$$k(x,y) = \frac{1}{2}[\widetilde{\rho}^2(x,x_0) + \widetilde{\rho}^2(y,x_0) - \widetilde{\rho}^2(x,y)],\ x,y,x_0 \in M, \tag{33}$$

which can then be used to compute γ_k.

The discussion so far has been devoted to relating γ_k to β, W and TV to understand the strength or weakness of γ_k w.r.t. these metrics. In a next step, we address the second question of when γ_k metrizes the weak topology on $\mathscr{P}$. This question would have been answered had the result in Theorem 21 shown that under some conditions on k, γ_k is equivalent to β. Since Theorem 21 does not help in this regard, we approach the problem differently. In the following, we provide two results related to the question. The first result states that when (M,ρ) is compact, γ_k induced by universal kernels metrizes the weak topology. In the second result, we relax the assumption of compactness but restrict ourselves to $M = \mathbb{R}^d$ and provide a sufficient condition on k such that γ_k metrizes the weak topology on $\mathscr{P}$. The proofs of both theorems are provided in Section 5.1.

Theorem 23 (Weak convergence-I) *Let* (M,ρ) *be a compact metric space. If k is universal, then* γ_k *metrizes the weak topology on* $\mathscr{P}$.

8. A metric ρ on M is said to be *Hilbertian* if there exists a Hilbert space, H and a mapping Φ such that $\rho(x,y) = \|\Phi(x) - \Phi(y)\|_H$, $\forall x,y \in M$. In our case, $H = \mathcal{H}$ and $\Phi : M \to \mathcal{H}, x \mapsto k(\cdot,x)$.
9. The explicit form for the Wasserstein distance is known for $(M,\rho(x,y)) = (\mathbb{R},|x-y|)$, and is $W(\mathbb{P},\mathbb{Q}) = \int_{\mathbb{R}} |F_{\mathbb{P}}(x) - F_{\mathbb{Q}}(x)|\,dx$, where $F_{\mathbb{P}}(x) = \mathbb{P}((-\infty,x])$. It is easy to show that this explicit form can be extended to $(\mathbb{R}^d, \|\cdot\|_1)$.

From Theorem 23, it is clear that γ_k is equivalent to ς, β and W (see (28) and (29)) when M is compact and k is universal.

Theorem 24 (Weak convergence-II) *Let $M = \mathbb{R}^d$ and $k(x,y) = \psi(x-y)$, where $\psi \in C_0(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$ is a real-valued bounded strictly positive definite function. If there exists an $l \in \mathbb{N}$ such that*

$$\int_{\mathbb{R}^d} \frac{1}{\widehat{\psi}(\omega)(1+\|\omega\|_2)^l}\, d\omega < \infty, \tag{34}$$

then γ_k metrizes the weak topology on $\mathscr{P}$.

The entire Matérn class of kernels in (18) satisfies the conditions of Theorem 24 and, therefore, the corresponding γ_k metrizes the weak topology on $\mathscr{P}$. Note that Gaussian kernels on $\mathbb{R}^d$ do not satisfy the condition in Theorem 24. The characterization of k for general non-compact domains M (not necessarily $\mathbb{R}^d$), such that γ_k metrizes the weak topology on $\mathscr{P}$, still remains an open problem.

5.1 Proofs

We now present the proofs of Theorems 21, 23 and 24.

Proof (Theorem 21) *(i)* When (M,ρ) is separable, $W(\mathbb{P},\mathbb{Q})$ has a coupling formulation (Dudley, 2002, p. 420), given as

$$W(\mathbb{P},\mathbb{Q}) = \inf_{\mu \in \mathcal{L}(\mathbb{P},\mathbb{Q})} \iint_M \rho(x,y)\, d\mu(x,y), \tag{35}$$

where $\mathbb{P},\mathbb{Q} \in \{\mathbb{P} \in \mathscr{P} : \int_M \rho(x,y)\, d\mathbb{P}(y) < \infty, \forall x \in M\}$. In our case $\rho(x,y) = \|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}}$. In addition, (M,ρ) is bounded, which means (35) holds for all $\mathbb{P},\mathbb{Q} \in \mathscr{P}$. The lower bound therefore follows from (30). The upper bound can be obtained as follows. Consider $W(\mathbb{P},\mathbb{Q}) = \inf_{\mu \in \mathcal{L}(\mathbb{P},\mathbb{Q})} \iint_M \|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}} d\mu(x,y)$, which can be bounded as

$$\begin{aligned}
W(\mathbb{P},\mathbb{Q}) &\leq \iint_M \|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}} d\mathbb{P}(x)\, d\mathbb{Q}(y) \\
&\overset{(a)}{\leq} \left[\iint_M \|k(\cdot,x) - k(\cdot,y)\|^2_{\mathcal{H}} d\mathbb{P}(x)\, d\mathbb{Q}(y)\right]^{\frac{1}{2}} \\
&\leq \left[\int_M k(x,x)\, d(\mathbb{P}+\mathbb{Q})(x) - 2\iint_M k(x,y)\, d\mathbb{P}(x)\, d\mathbb{Q}(y)\right]^{\frac{1}{2}} \\
&\leq \left[\gamma_k^2(\mathbb{P},\mathbb{Q}) + \iint_M (k(x,x) - k(x,y))\, d(\mathbb{P}\otimes\mathbb{P} + \mathbb{Q}\otimes\mathbb{Q})(x,y)\right]^{\frac{1}{2}} \\
&\leq \sqrt{\gamma_k^2(\mathbb{P},\mathbb{Q}) + 4C},
\end{aligned} \tag{36}$$

where we have used Jensen's inequality (Folland, 1999, p. 109) in (a).

(ii) Let $\mathcal{F} := \{f : \|f\|_{\mathcal{H}} < \infty\}$ and $\mathcal{G} := \{f : \|f\|_{BL} < \infty\}$. For $f \in \mathcal{F}$, we have

$$\begin{aligned}
\|f\|_{BL} &= \sup_{x \neq y} \frac{|f(x) - f(y)|}{\rho(x,y)} + \sup_{x \in M} |f(x)| = \sup_{x \neq y} \frac{|\langle f, k(\cdot,x) - k(\cdot,y)\rangle_{\mathcal{H}}|}{\|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}}} + \sup_{x \in M} |\langle f, k(\cdot,x)\rangle_{\mathcal{H}}| \\
&\leq (1 + \sqrt{C})\|f\|_{\mathcal{H}} < \infty,
\end{aligned}$$

which implies $f \in \mathcal{G}$ and, therefore, $\mathcal{F} \subset \mathcal{G}$. For any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$,

$$\begin{aligned}\gamma_k(\mathbb{P},\mathbb{Q}) &= \sup\{|\mathbb{P}f - \mathbb{Q}f| : f \in \mathcal{F}_k\} \\ &\leq \sup\{|\mathbb{P}f - \mathbb{Q}f| : \|f\|_{BL} \leq (1+\sqrt{C}), f \in \mathcal{F}\} \\ &\leq \sup\{|\mathbb{P}f - \mathbb{Q}f| : \|f\|_{BL} \leq (1+\sqrt{C}), f \in \mathcal{G}\} \\ &= (1+\sqrt{C})\beta(\mathbb{P},\mathbb{Q}).\end{aligned}$$

The upper bound is obtained as follows. For any $\mathbb{P}, \mathbb{Q} \in \mathscr{P}$, by Markov's inequality (Folland, 1999, Theorem 6.17), for all $\varepsilon > 0$, we have

$$\varepsilon^2 \mu(\|k(\cdot,X) - k(\cdot,Y)\|_{\mathcal{H}} > \varepsilon) \leq \iint_M \|k(\cdot,x) - k(\cdot,y)\|^2_{\mathcal{H}} d\mu(x,y),$$

where X and Y are distributed as $\mathbb{P}$ and $\mathbb{Q}$ respectively. Choose ε such that $\varepsilon^3 = \iint_M \|k(\cdot,x) - k(\cdot,y)\|^2_{\mathcal{H}} d\mu(x,y)$, such that $\mu(\|k(\cdot,X) - k(\cdot,Y)\|_{\mathcal{H}} > \varepsilon) \leq \varepsilon$. From the proof of Theorem 11.3.5 in Dudley (2002), when (M,ρ) is separable, we have

$$\mu(\rho(X,Y) \geq \varepsilon) < \varepsilon \Rightarrow \varsigma(\mathbb{P},\mathbb{Q}) \leq \varepsilon,$$

which implies that

$$\begin{aligned}\varsigma(\mathbb{P},\mathbb{Q}) &\leq \left(\inf_{\mu \in \mathcal{L}(\mathbb{P},\mathbb{Q})} \iint_M \|k(\cdot,x) - k(\cdot,y)\|^2_{\mathcal{H}} d\mu(x,y)\right)^{\frac{1}{3}} \\ &\leq \left(\iint_M \|k(\cdot,x) - k(\cdot,y)\|^2_{\mathcal{H}} d\mathbb{P}(x)\, d\mathbb{Q}(y)\right)^{\frac{1}{3}} \overset{(b)}{\leq} \left(\gamma_k^2(\mathbb{P},\mathbb{Q}) + 4C\right)^{\frac{1}{3}},\end{aligned}$$

where (b) follows from (36). The result follows from (28).

(iii) The proof of this result was presented in Sriperumbudur et al. (2009b) and is provided here for completeness. To prove the result, we use (30) and the coupling formulation for TV (Lindvall, 1992, p. 19), given as

$$\frac{1}{2} TV(\mathbb{P},\mathbb{Q}) = \inf_{\mu \in \mathcal{L}(\mathbb{P},\mathbb{Q})} \mu(X \neq Y),$$

where $\mathcal{L}(\mathbb{P},\mathbb{Q})$ is the set of all measures on $M \times M$ with marginals $\mathbb{P}$ and $\mathbb{Q}$. Here, X and Y are distributed as $\mathbb{P}$ and $\mathbb{Q}$ respectively. Consider

$$\|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}} \leq \mathbb{1}_{\{x \neq y\}} \|k(\cdot,x) - k(\cdot,y)\|_{\mathcal{H}} \leq 2\sqrt{C}\mathbb{1}_{\{x \neq y\}}. \tag{37}$$

Taking expectations w.r.t. μ and the infimum over $\mu \in \mathcal{L}(\mathbb{P},\mathbb{Q})$ on both sides of (37) gives the desired result, which follows from (30). ■

Proof (Theorem 23) We need to show that for measures $\mathbb{P}, \mathbb{P}_1, \mathbb{P}_2, \ldots \in \mathscr{P}$, $\mathbb{P}_n \xrightarrow{w} \mathbb{P}$ if and only if $\gamma_k(\mathbb{P}_n,\mathbb{P}) \to 0$ as $n \to \infty$. One direction is trivial as $\mathbb{P}_n \xrightarrow{w} \mathbb{P}$ implies $\gamma_k(\mathbb{P}_n,\mathbb{P}) \to 0$ as $n \to \infty$. We prove the other direction as follows. Since k is universal, $\mathcal{H}$ is dense in $C_b(M)$, the space of bounded continuous functions, w.r.t. the uniform norm, that is, for any $f \in C_b(M)$ and every $\varepsilon > 0$, there exists a $g \in \mathcal{H}$ such that $\|f - g\|_\infty \leq \varepsilon$. Therefore,

$$\begin{aligned}|\mathbb{P}_n f - \mathbb{P}f| &= |\mathbb{P}_n(f-g) + \mathbb{P}(g-f) + (\mathbb{P}_n g - \mathbb{P}g)| \\ &\leq \mathbb{P}_n|f-g| + \mathbb{P}|f-g| + |\mathbb{P}_n g - \mathbb{P}g| \\ &\leq 2\varepsilon + |\mathbb{P}_n g - \mathbb{P}g| \leq 2\varepsilon + \|g\|_{\mathcal{H}} \gamma_k(\mathbb{P}_n,\mathbb{P}).\end{aligned}$$

Since $\gamma_k(\mathbb{P}_n, \mathbb{P}) \to 0$ as $n \to \infty$ and ε is arbitrary, $|\mathbb{P}_n f - \mathbb{P} f| \to 0$ for any $f \in C_b(M)$. ■

Proof (Theorem 24) As mentioned in the proof of Theorem 23, one direction of the proof is straightforward: $\mathbb{P}_n \xrightarrow{w} \mathbb{P} \Rightarrow \gamma_k(\mathbb{P}_n, \mathbb{P}) \to 0$ as $n \to \infty$. Let us consider the other direction. Since $\psi \in C_0(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$ is a strictly positive definite function, any $f \in \mathcal{H}$ satisfies (Wendland, 2005, Theorem 10.12)

$$\int_{\mathbb{R}^d} \frac{|\widehat{f}(\omega)|^2}{\widehat{\psi}(\omega)} d\omega < \infty.$$

Assume that

$$\sup_{\omega \in \mathbb{R}^d} (1 + \|\omega\|_2)^l |\widehat{f}(\omega)|^2 < \infty,$$

for any $l \in \mathbb{N}$, which means $f \in \mathscr{S}_d$. Let (34) be satisfied for some $l = l_0$. Then,

$$\begin{aligned} \int_{\mathbb{R}^d} \frac{|\widehat{f}(\omega)|^2}{\widehat{\psi}(\omega)} d\omega &= \int_{\mathbb{R}^d} \frac{|\widehat{f}(\omega)|^2 (1 + \|\omega\|_2)^{l_0}}{\widehat{\psi}(\omega)(1 + \|\omega\|_2)^{l_0}} d\omega \\ &\leq \sup_{\omega \in \mathbb{R}^d} (1 + \|\omega\|_2)^{l_0} |\widehat{f}(\omega)|^2 \int_{\mathbb{R}^d} \frac{1}{\widehat{\psi}(\omega)(1 + \|\omega\|_2)^{l_0}} d\omega < \infty, \end{aligned}$$

which means $f \in \mathcal{H}$, that is, if $f \in \mathscr{S}_d$, then $f \in \mathcal{H}$, which implies $\mathscr{S}_d \subset \mathcal{H}$. Note that $\mathscr{S}(\mathbb{R}^d)$ is dense in $C_0(\mathbb{R}^d)$. Since $\psi \in C_0(\mathbb{R}^d)$, we have $\mathcal{H} \subset C_0(\mathbb{R}^d)$ (see the proof of Theorem 4.61 in Steinwart and Christmann, 2008) and, therefore, $\mathcal{H}$ is dense in $C_0(\mathbb{R}^d)$ w.r.t. the uniform norm. Suppose $\mathbb{P}, \mathbb{P}_1, \mathbb{P}_2, \ldots \in \mathscr{P}$. Using a similar analysis as in the proof of Theorem 23, it can be shown that for any $f \in C_0(\mathbb{R}^d)$ and every $\varepsilon > 0$, there exists a $g \in \mathcal{H}$ such that $|\mathbb{P}_n f - \mathbb{P} f| \leq 2\varepsilon + |\mathbb{P}_n g - \mathbb{P} g|$. Since ε is arbitrary and $\gamma_k(\mathbb{P}_n, \mathbb{P}) \to 0$ as $n \to \infty$, the result follows. ■

6. Conclusion and Discussion

We have studied various properties associated with a pseudometric γ_k on $\mathscr{P}$, which is based on the Hilbert space embedding of probability measures. First, we studied the conditions on the kernel (called the characteristic kernel) under which γ_k is a metric, and showed that apart from universal kernels, a large family of bounded continuous kernels induces a metric on $\mathscr{P}$: (a) integrally strictly pd kernels and (b) translation invariant kernels on $\mathbb{R}^d$ and $\mathbb{T}^d$ that have the support of their Fourier transform to be $\mathbb{R}^d$ and $\mathbb{Z}^d$ respectively. Next, we showed that there exist distinct distributions which will be considered close according to γ_k (whether or not the kernel is characteristic), and thus may be hard to distinguish based on finite samples. Finally, we compared γ_k to other metrics on $\mathscr{P}$ and explicitly presented the conditions under which it induces a weak topology on $\mathscr{P}$. These results together provide a strong theoretical foundation for using the γ_k metric in both statistics and machine learning applications.

We now discuss two topics related to γ_k, concerning the choice of kernel parameter and kernels defined on $\mathscr{P}$.

An important question not covered in the present paper is how to choose a characteristic kernel. Let us consider the following setting: $M = \mathbb{R}^d$ and $k_\sigma(x, y) = \exp(-\sigma \|x - y\|_2^2)$, $\sigma \in \mathbb{R}_+$, a Gaussian kernel with σ as the bandwidth parameter. $\{k_\sigma : \sigma \in \mathbb{R}_+\}$ is the family of Gaussian kernels and $\{\gamma_{k_\sigma} : \sigma \in \mathbb{R}_+\}$ is the associated family of distance measures indexed by the kernel parameter, σ. Note that k_σ is characteristic for any $\sigma \in \mathbb{R}_{++}$ and, therefore, γ_{k_σ} is a metric on $\mathscr{P}$ for any $\sigma \in \mathbb{R}_{++}$.

In practice, one would prefer a single number that defines the distance between $\mathbb{P}$ and $\mathbb{Q}$. The question therefore to be addressed is how to choose an appropriate σ. Note that as $\sigma \to 0$, $k_\sigma \to 1$ and as $\sigma \to \infty$, $k_\sigma \to 0$ a.e., which means $\gamma_{k_\sigma}(\mathbb{P},\mathbb{Q}) \to 0$ as $\sigma \to 0$ or $\sigma \to \infty$ for all $\mathbb{P},\mathbb{Q} \in \mathscr{P}$. This behavior is also exhibited by $k_\sigma(x,y) = \exp(-\sigma\|x-y\|_1)$, $\sigma > 0$ and $k_\sigma(x,y) = \sigma^2/(\sigma^2 + \|x-y\|_2^2)$, $\sigma > 0$, which are also characteristic. This means choosing *sufficiently small* or *sufficiently large* σ (depending on $\mathbb{P}$ and $\mathbb{Q}$) makes $\gamma_{k_\sigma}(\mathbb{P},\mathbb{Q})$ arbitrarily small. Therefore, σ must be chosen appropriately in applications to effectively distinguish between $\mathbb{P}$ and $\mathbb{Q}$.

To this end, one can consider the following modification to γ_k, which yields a pseudometric on $\mathscr{P}$,

$$\gamma(\mathbb{P},\mathbb{Q}) = \sup\{\gamma_k(\mathbb{P},\mathbb{Q}) : k \in \mathcal{K}\} = \sup\{\|\mathbb{P}k - \mathbb{Q}k\|_{\mathcal{H}} : k \in \mathcal{K}\}. \tag{38}$$

Note that γ is the maximal RKHS distance between $\mathbb{P}$ and $\mathbb{Q}$ over a family, $\mathcal{K}$ of measurable and bounded positive definite kernels. It is easy to check that, if any $k \in \mathcal{K}$ is characteristic, then γ is a metric on $\mathscr{P}$. Examples for $\mathcal{K}$ include:

1. $\mathcal{K}_g := \left\{e^{-\sigma\|x-y\|_2^2}, x,y \in \mathbb{R}^d : \sigma \in \mathbb{R}_+\right\}$.
2. $\mathcal{K}_l := \left\{e^{-\sigma\|x-y\|_1}, x,y \in \mathbb{R}^d : \sigma \in \mathbb{R}_+\right\}$.
3. $\mathcal{K}_\psi := \left\{e^{-\sigma\psi(x,y)}, x,y \in M : \sigma \in \mathbb{R}_+\right\}$, where $\psi : M \times M \to \mathbb{R}$ is a negative definite kernel (Berg et al., 1984, Chapter 3).
4. $\mathcal{K}_{rbf} := \left\{\int_0^\infty e^{-\lambda\|x-y\|_2^2} d\mu_\sigma(\lambda), x,y \in \mathbb{R}^d, \mu_\sigma \in \mathscr{M}^+ : \sigma \in \Sigma \subset \mathbb{R}^d\right\}$, where $\mathscr{M}^+$ is the set of all finite nonnegative Borel measures, μ_σ on $\mathbb{R}_+$ that are not concentrated at zero, etc.
5. $\mathcal{K}_{lin} := \left\{k_\lambda = \sum_{j=1}^l \lambda_j k_j \,|\, k_\lambda \text{ is pd}, \sum_{j=1}^l \lambda_j = 1\right\}$, which is the linear combination of pd kernels $\{k_j\}_{j=1}^l$.
6. $\mathcal{K}_{con} := \left\{k_\lambda = \sum_{j=1}^l \lambda_j k_j \,|\, \lambda_j \geq 0, \sum_{j=1}^l \lambda_j = 1\right\}$, which is the convex combination of pd kernels $\{k_j\}_{j=1}^l$.

The idea and validity behind the proposal of γ in (38) can be understood from a Bayesian perspective, where we define a non-negative finite measure λ over $\mathcal{K}$, and average γ_k over that measure, that is, $\alpha(\mathbb{P},\mathbb{Q}) := \int_{\mathcal{K}} \gamma_k(\mathbb{P},\mathbb{Q})\, d\lambda(k)$. This also yields a pseudometric on $\mathscr{P}$. That said, $\alpha(\mathbb{P},\mathbb{Q}) \leq \lambda(\mathcal{K})\gamma(\mathbb{P},\mathbb{Q})$, $\forall\, \mathbb{P},\mathbb{Q}$, which means that, if $\mathbb{P}$ and $\mathbb{Q}$ can be distinguished by α, then they can be distinguished by γ, but not vice-versa. In this sense, γ is stronger than α and therefore studying γ makes sense. One further complication with the Bayesian approach is in defining a sensible λ over $\mathcal{K}$. Note that γ_{k_0} can be obtained by defining $\lambda(k) = \delta(k-k_0)$ in $\alpha(\mathbb{P},\mathbb{Q})$. Future work will include analyzing γ and investigating its utility in applications compared to that of γ_k (with a fixed kernel, k). Sriperumbudur et al. (2009a) describes preliminary work, showing that $\gamma(\mathbb{P}_m,\mathbb{Q}_n)$ is a $\sqrt{mn/(m+n)}$-consistent estimator of $\gamma(\mathbb{P},\mathbb{Q})$, for families of kernels $\mathcal{K}$ including those mentioned above.

We now discuss how kernels on $\mathscr{P}$ can be obtained from γ_k. As noted by Gretton et al. (2007b, Section 4), and following Hein et al. (2004), γ_k is a *Hilbertian metric* on $\mathscr{P}$: the associated kernel can be easily computed using (33),

$$K(\mathbb{P},\mathbb{Q}) = \left\langle \int_M k(\cdot,x)\, d\mathbb{P}(x), \int_M k(\cdot,x)\, d\mathbb{Q}(x) \right\rangle_{\mathcal{H}} = \int\int_M k(x,y)\, d\mathbb{P}(x)\, d\mathbb{Q}(y),$$

where the positive definite kernel $K : \mathscr{P} \times \mathscr{P} \to \mathbb{R}$ is a dot-product kernel on $\mathscr{P}$. Using the results in Berg et al. (1984, Chapter 3, Theorems 2.2 and 2.3), Gaussian and inverse multi-quadratic kernels on $\mathscr{P}$ can be defined as

$$K(\mathbb{P},\mathbb{Q}) = \exp\left(-\sigma\gamma_k^2(\mathbb{P},\mathbb{Q})\right), \sigma > 0 \text{ and } K(\mathbb{P},\mathbb{Q}) = \left(\sigma+\gamma_k^2(\mathbb{P},\mathbb{Q})\right)^{-1}, \sigma > 0$$

respectively. Further work on Hilbertian metrics and positive definite kernels on probability measures has been carried out by Hein and Bousquet (2005) and Fuglede and Topsøe (2003).

Acknowledgments

The authors thank the editor and reviewers for their constructive comments. B. K. S. and G. R. G. L. wish to acknowledge support from the Max Planck Institute (MPI) for Biological Cybernetics, the National Science Foundation (grant DMS-MSPA 0625409), the Fair Isaac Corporation and the University of California MICRO program. Part of this work was done while B. K. S. was an intern at the MPI, and part was done while A. G. was a project scientist at CMU, under grants DARPA IPTO FA8750-09-1-0141, ONR MURI N000140710747, and NSF NeTS-NOSS CNS-0625518. This work is also supported by the IST Program of the EC, under the FP7 Network of Excellence, ICT-216886-NOE. B. K. S. wishes to thank Agnes Radl for her comments on an earlier version of the manuscript.

Appendix A. Supplementary Results

For completeness, we present the supplementary results that were used to prove the results in this paper. The following result is quoted from Folland (1999, Theorem 8.14).

Theorem 25 *Suppose $\phi \in L^1(\mathbb{R}^d)$, $\int_{\mathbb{R}^d} \phi(x)\,dx = a$ and $\phi_t(x) = t^{-d}\phi(t^{-1}x)$ for $t > 0$. If f is bounded and uniformly continuous on $\mathbb{R}^d$, then $f * \phi_t \to af$ uniformly as $t \to 0$.*

By imposing slightly stronger conditions on ϕ, the following result quoted from Folland (1999, Theorem 8.15) shows that $f * \phi_t \to af$ almost everywhere for $f \in L^r(\mathbb{R}^d)$.

Theorem 26 *Suppose $|\phi(x)| \leq C(1 + \|x\|_2)^{-d-\varepsilon}$ for some $C, \varepsilon > 0$, and $\int_{\mathbb{R}^d} \phi(x)\,dx = a$. If $f \in L^r(\mathbb{R}^d)$ $(1 \leq r \leq \infty)$, then $f * \phi_t(x) \to af(x)$ as $t \to 0$ for every x in the Lebesgue set of f—in particular, for almost every x, and for every x at which f is continuous.*

Theorem 27 (Fourier transform of a measure) *Let μ be a finite Borel measure on $\mathbb{R}^d$. The Fourier transform of μ is given by*

$$\widehat{\mu}(\omega) = \int_{\mathbb{R}^d} e^{-i\omega^T x}\,d\mu(x),\ \omega \in \mathbb{R}^d,$$

which is a bounded, uniformly continuous function on $\mathbb{R}^d$. In addition, $\widehat{\mu}$ satisfies the following properties:

(i) $\overline{\widehat{\mu}(\omega)} = \widehat{\mu}(-\omega), \forall\, \omega \in \mathbb{R}^d$, *that is, $\widehat{\mu}$ is conjugate symmetric,*

(ii) $\widehat{\mu}(0) = 1$.

The following result, called the Riemann-Lebesgue lemma, is quoted from Rudin (1991, Theorem 7.5).

Lemma 28 (Riemann-Lebesgue) *If $f \in L^1(\mathbb{R}^d)$, then $\widehat{f} \in C_0(\mathbb{R}^d)$, and $\|\widehat{f}\|_\infty \leq \|f\|_1$.*

The following theorem is a version of the *Paley-Wiener theorem* for distributions, and is proved in Rudin (1991, Theorem 7.23).

Theorem 29 (Paley-Wiener) *If $f \in \mathscr{D}'_d$ has compact support, then $\widehat{f}$ is the restriction to $\mathbb{R}^d$ of an entire function on $\mathbb{C}^d$.*

References

S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society, Series B (Methodological)*, 28:131–142, 1966.

N. Anderson, P. Hall, and D. Titterington. Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. *Journal of Multivariate Analysis*, 50:41–54, 1994.

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

A. D. Barbour and L. H. Y. Chen. *An Introduction to Stein's Method*. Singapore University Press, Singapore, 2005.

C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Spring Verlag, New York, 1984.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, London, UK, 2004.

K. M. Borgwardt, A. Gretton, M. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

P. Brémaud. *Mathematical Principles of Signal Processing*. Springer-Verlag, New York, 2001.

I. Csiszár. Information-type measures of difference of probability distributions and indirect observations. *Studia Scientiarium Mathematicarum Hungarica*, 2:299–318, 1967.

W. Dahmen and C. A. Micchelli. Some remarks on ridge functions. *Approx. Theory Appl.*, 3: 139–143, 1987.

E. del Barrio, J. A. Cuesta-Albertos, C. Matrán, and J. M. Rodríguez-Rodríguez. Testing of goodness of fit based on the L_2-Wasserstein distance. *Annals of Statistics*, 27:1230–1239, 1999.

L. Devroye and L. Györfi. No empirical probability measure can converge in the total variation sense for all distributions. *Annals of Statistics*, 18(3):1496–1499, 1990.

R. M. Dudley. *Real Analysis and Probability*. Cambridge University Press, Cambridge, UK, 2002.

G. B. Folland. *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, New York, 1999.

B. Fuglede and F. Topsøe. Jensen-Shannon divergence and Hilbert space embedding, 2003. Preprint.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 489–496, Cambridge, MA, 2008. MIT Press.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Kernel dimension reduction in regression. *Annals of Statistics*, 37(5):1871–1905, 2009a.

K. Fukumizu, B. K. Sriperumbudur, A. Gretton, and B. Schölkopf. Characteristic kernels on groups and semigroups. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 473–480, 2009b.

C. Gasquet and P. Witomski. *Fourier Analysis and Applications*. Springer-Verlag, New York, 1999.

A. L. Gibbs and F. E. Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435, 2002.

A. Gretton, R. Herbrich, A. Smola, O. Bousquet, and B. Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:2075–2129, December 2005a.

A. Gretton, A. Smola, O. Bousquet, R. Herbrich, A. Belitski, M. Augath, Y. Murayama, J. Pauls, B. Schölkopf, and N. Logothetis. Kernel constrained covariance for dependence measurement. In Z. Ghahramani and R. Cowell, editors, *Proc. 10^{th} International Workshop on Artificial Intelligence and Statistics*, pages 1–8, 2005b.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two sample problem. Technical Report 157, MPI for Biological Cybernetics, 2007a.

A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two sample problem. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, 2007b.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola. A kernel statistical test of independence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

M. Hein and O. Bousquet. Hilbertian metrics and positive definite kernels on probability measures. In Z. Ghahramani and R. Cowell, editors, *Proc. 10^{th} International Workshop on Artificial Intelligence and Statistics*, pages 136–143, 2005.

M. Hein, T.N. Lal, and O. Bousquet. Hilbertian metrics on probability measures and their application in SVMs. In *Proceedings of the 26th DAGM Symposium*, pages 270–277, Berlin, 2004. Springer.

E. L. Lehmann and J. P. Romano. *Testing Statistical Hypothesis*. Springer-Verlag, New York, 2005.

F. Liese and I. Vajda. On divergences and informations in statistics and information theory. *IEEE Trans. Information Theory*, 52(10):4394–4412, 2006.

T. Lindvall. *Lectures on the Coupling Method*. John Wiley & Sons, New York, 1992.

C. A. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667, 2006.

A. Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29:429–443, 1997.

A. Pinkus. Strictly positive definite functions on a real inner product space. *Adv. Comput. Math.*, 20:263–271, 2004.

S. T. Rachev. *Probability Metrics and the Stability of Stochastic Models*. John Wiley & Sons, Chichester, 1991.

S. T. Rachev and L. Rüschendorf. *Mass Transportation Problems. Vol. I Theory, Vol. II Applications*. Probability and its Applications. Springer-Verlag, Berlin, 1998.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

M. Reed and B. Simon. *Methods of Modern Mathematical Physics: Functional Analysis I*. Academic Press, New York, 1980.

M. Rosenblatt. A quadratic measure of deviation of two-dimensional density estimates and a test of independence. *Annals of Statistics*, 3(1):1–14, 1975.

W. Rudin. *Functional Analysis*. McGraw-Hill, USA, 1991.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

H. Shen, S. Jegelka, and A. Gretton. Fast kernel-based independent component analysis. *IEEE Transactions on Signal Processing*, 57:3498 – 3511, 2009.

G. R. Shorack. *Probability for Statisticians*. Springer-Verlag, New York, 2000.

A. J. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *Proc. 18th International Conference on Algorithmic Learning Theory*, pages 13–31. Springer-Verlag, Berlin, Germany, 2007.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, G. R. G. Lanckriet, and B. Schölkopf. Injective Hilbert space embeddings of probability measures. In R. Servedio and T. Zhang, editors, *Proc. of the 21st Annual Conference on Learning Theory*, pages 111–122, 2008.

B. K. Sriperumbudur, K. Fukumizu, A. Gretton, G. R. G. Lanckriet, and B. Schölkopf. Kernel choice and classifiability for RKHS embeddings of probability distributions. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1750–1758. MIT Press, 2009a.

B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. G. Lanckriet. On integral probability metrics, ϕ-divergences and binary classification. *http://arxiv.org/abs/0901.2698v4*, October 2009b.

B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet. On the relation between universality, characteristic kernels and RKHS embedding of measures. In Y. W. Teh and M. Titterington, editors, *Proc. 13th International Conference on Artificial Intelligence and Statistics*, volume 9 of *Workshop and Conference Proceedings*. JMLR, 2010a.

B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet. Universality, characteristic kernels and RKHS embedding of measures. *http://arxiv.org/abs/1003.0887*, March 2010b.

C. Stein. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proc. of the Sixth Berkeley Symposium on Mathematical Statistics and Probability*, 1972.

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 and A. Christmann. *Support Vector Machines*. Springer, 2008.

J. Stewart. Positive definite functions and generalizations, an historical survey. *Rocky Mountain Journal of Mathematics*, 6(3):409–433, 1976.

I. Vajda. *Theory of Statistical Inference and Information*. Kluwer Academic Publishers, Boston, 1989.

S. S. Vallander. Calculation of the Wasserstein distance between probability distributions on the line. *Theory Probab. Appl.*, 18:784–786, 1973.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer-Verlag, New York, 1996.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

N. Weaver. *Lipschitz Algebras*. World Scientific Publishing Company, 1999.

H. Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, UK, 2005.

Journal of Machine Learning Research 11 (2010) 1563-1600 Submitted 7/09; Revised 11/09; Published 4/10

Near-optimal Regret Bounds for Reinforcement Learning*

Thomas Jaksch TJAKSCH@UNILEOBEN.AC.AT
Ronald Ortner RORTNER@UNILEOBEN.AC.AT
Peter Auer AUER@UNILEOBEN.AC.AT
Chair for Information Technology
University of Leoben
Franz-Josef-Strasse 18
8700 Leoben, Austria

Editor: Sham Kakade

Abstract

For undiscounted reinforcement learning in Markov decision processes (MDPs) we consider the *total regret* of a learning algorithm with respect to an optimal policy. In order to describe the transition structure of an MDP we propose a new parameter: An MDP has *diameter D* if for any pair of states s, s' there is a policy which moves from s to s' in at most D steps (on average). We present a reinforcement learning algorithm with total regret $\tilde{O}(DS\sqrt{AT})$ after T steps for any unknown MDP with S states, A actions per state, and diameter D. A corresponding lower bound of $\Omega(\sqrt{DSAT})$ on the total regret of any learning algorithm is given as well.

These results are complemented by a sample complexity bound on the number of suboptimal steps taken by our algorithm. This bound can be used to achieve a (gap-dependent) regret bound that is logarithmic in T.

Finally, we also consider a setting where the MDP is allowed to change a fixed number of ℓ times. We present a modification of our algorithm that is able to deal with this setting and show a regret bound of $\tilde{O}(\ell^{1/3}T^{2/3}DS\sqrt{A})$.

Keywords: undiscounted reinforcement learning, Markov decision process, regret, online learning, sample complexity

1. Introduction

In a Markov decision process (MDP) M with finite state space $\mathcal{S}$ and finite action space $\mathcal{A}$, a learner in some state $s \in \mathcal{S}$ needs to choose an action $a \in \mathcal{A}$. When executing action a in state s, the learner receives a random reward r drawn independently from some distribution on $[0,1]$ with mean $\bar{r}(s,a)$. Further, according to the transition probabilities $p(s'|s,a)$, a random transition to a state $s' \in \mathcal{S}$ occurs.

Reinforcement learning of MDPs is a standard model for learning with delayed feedback. In contrast to important other work on reinforcement learning—where the performance of the *learned* policy is considered (see, e.g., Sutton and Barto 1998, Kearns and Singh 1999, and also the discussion and references given in the introduction of Kearns and Singh 2002)—we are interested in the performance of the learning algorithm *during learning*. For that, we compare the rewards collected by the algorithm during learning with the rewards of an optimal policy.

*. An extended abstract of this paper appeared in *Advances in Neural Information Processing Systems* 21 (2009), pp. 89–96.

An algorithm $\mathfrak{A}$ starting in an initial state s of an MDP M chooses at each time step t (possibly randomly) an action a_t. As the MDP is assumed to be unknown except the sets $\mathcal{S}$ and $\mathcal{A}$, usually an algorithm will map the history up to step t to an action a_t or, more generally, to a probability distribution over $\mathcal{A}$. Thus, an MDP M and an algorithm $\mathfrak{A}$ operating on M with initial state s constitute a stochastic process described by the states s_t visited at time step t, the actions a_t chosen by $\mathfrak{A}$ at step t, and the rewards r_t obtained ($t \in \mathbb{N}$). In this paper we will consider *undiscounted* rewards. Thus, the *accumulated reward* of an algorithm $\mathfrak{A}$ after T steps in an MDP M with initial state s, defined as

$$R(M,\mathfrak{A},s,T) := \sum_{t=1}^{T} r_t,$$

is a random variable with respect to the mentioned stochastic process. The value $\frac{1}{T}\mathbb{E}[R(M,\mathfrak{A},s,T)]$ then is the expected average reward of the process up to step T. The limit

$$\rho(M,\mathfrak{A},s) := \lim_{T\to\infty} \tfrac{1}{T}\mathbb{E}[R(M,\mathfrak{A},s,T)]$$

is called the *average reward* and can be maximized by an appropriate stationary *policy* $\pi : \mathcal{S} \to \mathcal{A}$ which determines an optimal action for each state (see Puterman, 1994). Thus, in what follows we will usually consider policies to be stationary.

The difficulty of learning an optimal policy in an MDP does not only depend on the MDP's size (given by the number of states and actions), but also on its transition structure. In order to measure this transition structure we propose a new parameter, the *diameter D* of an MDP. The diameter D is the time it takes to move from any state s to any other state s', using an appropriate policy for each pair of states s, s':

Definition 1 *Consider the stochastic process defined by a stationary policy $\pi : \mathcal{S} \to \mathcal{A}$ operating on an MDP M with initial state s. Let $T(s'|M,\pi,s)$ be the random variable for the first time step in which state s' is reached in this process. Then the* diameter *of M is defined as*

$$D(M) := \max_{s\neq s'\in\mathcal{S}} \min_{\pi:\mathcal{S}\to\mathcal{A}} \mathbb{E}\left[T(s'|M,\pi,s)\right].$$

In Appendix A we show that the diameter is at least $\log_{|\mathcal{A}|}|\mathcal{S}| - 3$. On the other hand, depending on the existence of states that are hard to reach under any policy, the diameter may be arbitrarily large. (For a comparison of the diameter to other mixing time parameters see below.)

In any case, a finite diameter seems necessary for interesting bounds on the *regret* of any algorithm with respect to an optimal policy. When a learner explores suboptimal actions, this may take him into a "bad part" of the MDP from which it may take up to D steps to reach again a "good part" of the MDP. Thus, compared to the simpler multi-armed bandit problem where each arm a is typically explored $\frac{\log T}{g}$ times (depending on the gap g between the optimal reward and the reward for arm a)—see, for example, the regret bounds of Auer et al. (2002a) for the UCB algorithms and the lower bound of Mannor and Tsitsiklis (2004)—the best one would expect for the general MDP setting is a regret bound of $\Theta\big(D|\mathcal{S}||\mathcal{A}|\log T\big)$. The alternative gap-independent regret bounds of $\tilde{O}(\sqrt{|\mathcal{B}|T})$ and $\Omega(\sqrt{|\mathcal{B}|T})$ for multi-armed bandits with $|\mathcal{B}|$ arms (Auer et al., 2002b) correspondingly translate into a regret bound of $\Theta(\sqrt{D|\mathcal{S}||\mathcal{A}|T})$ for MDPs with diameter D.

For MDPs with finite diameter (which usually are called *communicating*, see, e.g., Puterman 1994) the optimal average reward ρ^* does not depend on the initial state (cf. Puterman 1994, Section 8.3.3), and we set

$$\rho^*(M) := \rho^*(M,s) := \max_{\pi} \rho(M,\pi,s).$$

The optimal average reward is the natural benchmark[1] for a learning algorithm $\mathfrak{A}$, and we define the *total regret* of $\mathfrak{A}$ after T steps as

$$\Delta(M,\mathfrak{A},s,T) := T\rho^*(M) - R(M,\mathfrak{A},s,T).$$

In the following, we present our reinforcement learning algorithm UCRL2 (a variant of the UCRL algorithm of Auer and Ortner, 2007) which uses upper confidence bounds to choose an optimistic policy. We show that the total regret of UCRL2 after T steps is $\tilde{O}(D|\mathcal{S}|\sqrt{|\mathcal{A}|T})$. A corresponding lower bound of $\Omega(\sqrt{D|\mathcal{S}||\mathcal{A}|T})$ on the total regret of any learning algorithm is given as well. These results establish the diameter as an important parameter of an MDP. Unlike other parameters that have been proposed for various PAC and regret bounds, such as the *mixing time* (Kearns and Singh, 2002; Brafman and Tennenholtz, 2002) or the *hitting time* of an optimal policy (Tewari and Bartlett, 2008) (cf. the discussion below) the diameter only depends on the MDP's transition structure.

1.1 Relation to Previous Work

We first compare our results to the PAC bounds for the well-known algorithms E^3 of Kearns and Singh (2002), and R-Max of Brafman and Tennenholtz (2002) (see also Kakade, 2003). These algorithms achieve ε-optimal average reward with probability $1-\delta$ after time polynomial in $\frac{1}{\delta}$, $\frac{1}{\varepsilon}$, $|\mathcal{S}|$, $|\mathcal{A}|$, and the mixing time $T_\varepsilon^{\text{mix}}$ (see below). As the polynomial dependence on ε is of order $\frac{1}{\varepsilon^3}$, the PAC bounds translate into $T^{2/3}$ regret bounds at the best. Moreover, both algorithms need the ε-*return mixing time* $T_\varepsilon^{\text{mix}}$ of an optimal policy π^* as input parameter.[2] This parameter $T_\varepsilon^{\text{mix}}$ is the number of steps until the average reward of π^* over these $T_\varepsilon^{\text{mix}}$ steps is ε-close to the optimal average reward ρ^*. It is easy to construct MDPs of diameter D with $T_\varepsilon^{\text{mix}} \approx \frac{D}{\varepsilon}$. This additional dependence on ε further increases the exponent in the above mentioned regret bounds for E^3 and R-max. Also, the exponents of the parameters $|\mathcal{S}|$ and $|\mathcal{A}|$ in the PAC bounds of Kearns and Singh (2002) and Brafman and Tennenholtz (2002) are substantially larger than in our bound. However, there are algorithms with better dependence on these parameters. Thus, in the sample complexity bounds for the Delayed Q-Learning algorithm of Strehl et al. (2006) the dependence on states and actions is of order $|\mathcal{S}||\mathcal{A}|$, however at the cost of a worse dependence of order $\frac{1}{\varepsilon^4}$ on ε.

The MBIE algorithm of Strehl and Littman (2005, 2008)—similarly to our approach—applies confidence bounds to compute an optimistic policy. However, Strehl and Littman consider only a discounted reward setting. Their definition of regret measures the difference between the rewards[3] of an optimal policy and the rewards of the learning algorithm *along the trajectory taken by the learning algorithm*. In contrast, we are interested in the regret of the learning algorithm in respect to the rewards of the optimal policy *along the trajectory of the optimal policy*.[4] Generally, in discounted reinforcement learning only a finite number of steps is relevant, depending on the discount

1. It can be shown that $\max_{\mathfrak{A}} \mathbb{E}[R(M,\mathfrak{A},s,T)] = T\rho^*(M) + O(D(M))$ and $\max_{\mathfrak{A}} R(M,\mathfrak{A},s,T) = T\rho^*(M) + \tilde{O}(\sqrt{T})$ with high probability.
2. The knowledge of this parameter can be eliminated by guessing $T_\varepsilon^{\text{mix}}$ to be $1,2,\ldots$, so that sooner or later the correct $T_\varepsilon^{\text{mix}}$ will be reached (cf. Kearns and Singh 2002; Brafman and Tennenholtz 2002). However, since there is no condition on when to stop increasing $T_\varepsilon^{\text{mix}}$, the assumed mixing time eventually becomes arbitrarily large, so that the PAC bounds become exponential in the true $T_\varepsilon^{\text{mix}}$ (cf. Brafman and Tennenholtz, 2002).
3. Actually, the state values.
4. Indeed, one can construct MDPs for which these two notions of regret differ significantly. E.g., set the discount factor $\gamma = 0$. Then any policy which maximizes immediate rewards achieves 0 regret in the notion of Strehl and Littman. But such a policy may not move to states where the optimal reward is obtained.

factor. This makes discounted reinforcement learning similar to the setting with trials of constant length from a fixed initial state as considered by Fiechter (1994). For this case logarithmic online regret bounds in the number of trials have already been given by Auer and Ortner (2005). Also, the notion of regret is less natural than in undiscounted reinforcement learning: when summing up the regret in the individual visited states to obtain the total regret in the discounted setting, somehow contrary to the principal idea of discounting, the regret at each time step counts the same.

Tewari and Bartlett (2008) propose a generalization of the *index policies* of Burnetas and Katehakis (1997). These index policies choose actions optimistically by using confidence bounds only for the estimates in the current state. The regret bounds for the *index policies* of Burnetas and Katehakis (1997) and the OLP algorithm of Tewari and Bartlett (2008) are *asymptotically* logarithmic in T. However, unlike our bounds, these bounds depend on the gap between the "quality" of the best and the second best action, and these asymptotic bounds also hide an additive term which is exponential in the number of states. Actually, it is possible to prove a corresponding gap-dependent logarithmic bound for our UCRL2 algorithm as well (cf. Theorem 4 below). This bound holds uniformly over time and under weaker assumptions: While Tewari and Bartlett (2008) and Burnetas and Katehakis (1997) consider only *ergodic* MDPs in which *any* policy will reach every state after a sufficient number of steps, we make only the more natural assumption of a finite diameter.

Recently, Bartlett and Tewari (2009) have introduced the REGAL algorithm (inspired by our UCRL2 algorithm) and show—based on the methods we introduce in this paper—regret bounds where the diameter is replaced with a smaller transition parameter D_1 (that is basically an upper bound on the span of the *bias* of an optimal policy). Moreover, this bound also allows the MDP to have some *transient* states that are not reachable under any policy. However, the bound holds only when the learner knows an upper bound on this parameter D_1. In case the learner has no such upper bound, a doubling trick can be applied, but then the bound's dependence on $|\mathcal{S}|$ deteriorates from $|\mathcal{S}|$ to $|\mathcal{S}|^{3/2}$. Bartlett and Tewari (2009) also modify our lower bound example to obtain a lower bound of $\Omega(D_1\sqrt{|\mathcal{S}||\mathcal{A}|T})$ with respect to their new transition parameter D_1. Still, in the given example $D_1 = \sqrt{D}$, so that in this case their lower bound matches our lower bound.

2. Results

We summarize the results achieved for our algorithm UCRL2 (which will be described in the next section), and also state a corresponding lower bound. We assume an unknown MDP M to be learned, with $S := |\mathcal{S}|$ states, $A := |\mathcal{A}|$ actions, and finite diameter $D := D(M)$. Only $\mathcal{S}$ and $\mathcal{A}$ are known to the learner, and UCRL2 is run with confidence parameter δ.

Theorem 2 *With probability of at least* $1-\delta$ *it holds that for any initial state* $s \in \mathcal{S}$ *and any* $T > 1$*, the regret of* UCRL2 *is bounded by*

$$\Delta(M, \text{UCRL2}, s, T) \;\leq\; 34 \cdot DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}.$$

It is straightforward to obtain from Theorem 2 the following sample complexity bound.

Corollary 3 *With probability of at least* $1-\delta$ *the average per-step regret of* UCRL2 *is at most* ε *for any*

$$T \;\geq\; 4 \cdot 34^2 \cdot \frac{D^2S^2A}{\varepsilon^2}\log\left(\frac{34DSA}{\delta\varepsilon}\right)$$

steps.

It is also possible to give a sample complexity bound on the number of suboptimal steps UCRL2 takes, which allows to derive the following gap-dependent logarithmic bound on the expected regret.

Theorem 4 *For any initial state $s \in \mathcal{S}$, any $T \geq 1$ and any $\varepsilon > 0$, with probability of at least $1 - 3\delta$ the regret of* UCRL2 *is*

$$\Delta(M, \text{UCRL2}, s, T) \;\leq\; 34^2 \cdot \frac{D^2 S^2 A \log\left(\frac{T}{\delta}\right)}{\varepsilon} + \varepsilon T.$$

Moreover setting

$$g := \rho^*(M) - \max_{s \in \mathcal{S}} \max_{\pi : \mathcal{S} \to \mathcal{A}} \left\{ \rho(M, \pi, s) : \rho(M, \pi, s) < \rho^*(M) \right\}$$

to be the gap in average reward between best and second best policy in M, the expected regret of UCRL2 *(with parameter $\delta := \frac{1}{3T}$) for any initial state $s \in \mathcal{S}$ is*

$$\mathbb{E}\left[\Delta(M, \text{UCRL2}, s, T)\right] < 34^2 \cdot \frac{D^2 S^2 A \log(T)}{g} + 1 + \sum_{s,a} \left[1 + \log_2\Big(\max_{\pi : \pi(s) = a} T_\pi\Big)\right] \max_{\pi : \pi(s) = a} T_\pi,$$

where T_π is the smallest natural number such that for all $T \geq T_\pi$ the expected average reward after T steps is $\frac{g}{2}$-close to the average reward of π. Using the doubling trick to set the parameter δ, one obtains a corresponding bound (with larger constant) without knowledge of the horizon T.

These new bounds are improvements over the bounds that have been achieved by Auer and Ortner (2007) for the original UCRL algorithm in various respects: the exponents of the relevant parameters have been decreased considerably, the parameter D we use here is substantially smaller than the corresponding mixing time of Auer and Ortner (2007), and finally, the ergodicity assumption is replaced by the much weaker and more natural assumption that the MDP has finite diameter.

The following is an accompanying lower bound on the expected regret.

Theorem 5 *For any algorithm $\mathfrak{A}$, any natural numbers $S, A \geq 10$, $D \geq 20 \log_A S$, and $T \geq DSA$, there is an MDP M with S states, A actions, and diameter D,*[5] *such that for any initial state $s \in \mathcal{S}$ the expected regret of $\mathfrak{A}$ after T steps is*

$$\mathbb{E}\left[\Delta(M, \mathfrak{A}, s, T)\right] \geq 0.015 \cdot \sqrt{DSAT}\,.$$

Finally, we consider a modification of UCRL2 that is also able to deal with changing MDPs.

Theorem 6 *Assume that the MDP (i.e., its transition probabilities and reward distributions) is allowed to change $(\ell - 1)$ times up to step T, such that the diameter is always at most D. Restarting* UCRL2 *with parameter $\frac{\delta}{\ell^2}$ at steps $\left\lceil \frac{i^3}{\ell^2} \right\rceil$ for $i = 1, 2, 3 \ldots$, the regret (now measured as the sum of missed rewards compared to the ℓ optimal policies in the periods during which the MDP remains constant) is upper bounded by*

$$65 \cdot \ell^{1/3} T^{2/3} DS \sqrt{A \log\left(\tfrac{T}{\delta}\right)}$$

with probability of at least $1 - \delta$.

5. As already mentioned, the diameter of any MDP with S states and A actions is at least $\log_A S - 3$.

For the simpler multi-armed bandit problem, similar settings have already been considered by Auer et al. (2002b), and more recently by Garivier and Moulines (2008), and Yu and Mannor (2009). The achieved regret bounds are $O(\sqrt{\ell T \log T})$ in the first two mentioned papers, while Yu and Mannor (2009) derive regret bounds of $O(\ell \log T)$ for a setting with side observations on past rewards in which the number of changes ℓ need not be known in advance.

MDPs with a different model of changing rewards have already been considered by Even-Dar et al. (2005) and Even-Dar et al. (2009), respectively. There, the transition probabilities are assumed to be fixed and known to the learner, but the rewards are allowed to change at every step (however, independently of the history). In this setting, an upper bound of $O(\sqrt{T})$ on the regret against an optimal stationary policy (with the reward changes known in advance) is best possible and has been derived by Even-Dar et al. (2005). This setting recently has been further investigated by Yu et al. (2009), who also show that for achieving sublinear regret it is essential that the changing rewards are chosen obliviously, as an opponent who chooses the rewards depending on the learner's history may inflict linear loss on the learner. It should be noted that although the definition of regret in the nonstochastic setting looks the same as in the stochastic setting, there is an important difference to notice. While in the stochastic setting the average reward of an MDP is always maximized by a stationary policy $\pi : \mathcal{S} \to \mathcal{A}$, in the nonstochastic setting obviously a dynamic policy adapted to the reward sequence would in general earn more than a stationary policy. However, obviously no algorithm will be able to compete with the best dynamic policy for all possible reward sequences, so that—similar to the nonstochastic bandit problem, compare to Auer et al. (2002b)—one usually competes only with a finite set of experts, in the case of MDPs the set of stationary policies $\pi : \mathcal{S} \to \mathcal{A}$. For different notions of regret in the nonstochastic MDP setting see Yu et al. (2009).

Note that all our results scale linearly with the rewards. That is, if the rewards are not bounded in $[0,1]$ but taken from some interval $[r_{\min}, r_{\max}]$, the rewards can simply be normalized, so that the given regret bounds hold with additional factor $(r_{\max} - r_{\min})$.

3. The UCRL2 Algorithm

Our algorithm is a variant of the UCRL algorithm of Auer and Ortner (2007). As its predecessor, UCRL2 implements the paradigm of "optimism in the face of uncertainty". That is, it defines a set $\mathcal{M}$ of statistically *plausible* MDPs given the observations so far, and chooses an optimistic MDP $\tilde{M}$ (with respect to the achievable average reward) among these plausible MDPs. Then it executes a policy $\tilde{\pi}$ which is (nearly) optimal for the optimistic MDP $\tilde{M}$. More precisely, UCRL2 (see Figure 1) proceeds in episodes and computes a new policy $\tilde{\pi}_k$ only at the beginning of each episode k. The lengths of the episodes are not fixed a priori, but depend on the observations made. In Steps 2–3, UCRL2 computes estimates $\hat{r}_k(s,a)$ and $\hat{p}_k(s'|s,a)$ for the mean rewards and the transition probabilities from the observations made before episode k. In Step 4, a set $\mathcal{M}_k$ of plausible MDPs is defined in terms of confidence regions around the estimated mean rewards $\hat{r}_k(s,a)$ and transition probabilities $\hat{p}_k(s'|s,a)$. This guarantees that with high probability the true MDP M is in $\mathcal{M}_k$. In Step 5, *extended value iteration* (see below) is used to choose a near-optimal policy $\tilde{\pi}_k$ on an optimistic MDP $\tilde{M}_k \in \mathcal{M}_k$. This policy $\tilde{\pi}_k$ is executed throughout episode k (Step 6). Episode k ends when a state s is visited in which the action $a = \tilde{\pi}_k(s)$ induced by the current policy has been chosen *in* episode k equally often as *before* episode k. Thus, the total number of occurrences of

any state-action pair is at most doubled during an episode. The counts $v_k(s,a)$ keep track of these occurrences in episode k.[6]

3.1 Extended Value Iteration: Finding Optimistic Model and Optimal Policy

In Step 5 of the UCRL2 algorithm we need to find a near-optimal policy $\tilde{\pi}_k$ for an optimistic MDP $\tilde{M}_k$. While value iteration typically calculates an optimal policy for a fixed MDP, we also need to select an optimistic MDP $\tilde{M}_k$ that gives almost maximal optimal average reward among all plausible MDPs.

3.1.1 PROBLEM FORMULATION

We can formulate this as a general problem as follows. Let $\mathcal{M}$ be the set of all MDPs with (common) state space $\mathcal{S}$, (common) action space $\mathcal{A}$, transition probabilities $\tilde{p}(\cdot|s,a)$, and mean rewards $\tilde{r}(s,a)$ such that

$$\|\tilde{p}(\cdot|s,a) - \hat{p}(\cdot|s,a)\|_1 \leq d(s,a), \tag{1}$$

$$|\tilde{r}(s,a) - \hat{r}(s,a)| \leq d'(s,a) \tag{2}$$

for given probability distributions $\hat{p}(\cdot|s,a)$, values $\hat{r}(s,a)$ in $[0,1]$, $d(s,a) > 0$, and $d'(s,a) \geq 0$. Further, we assume that $\mathcal{M}$ contains at least one communicating MDP, that is, an MDP with finite diameter.

In Step 5 of UCRL2, the $d(s,a)$ and $d'(s,a)$ are obviously the confidence intervals as given by (4) and (3), while the communicating MDP assumed to be in $\mathcal{M}_k$ is the true MDP M. The task is to find an MDP $\tilde{M} \in \mathcal{M}$ and a policy $\tilde{\pi} : \mathcal{S} \to \mathcal{A}$ which maximize $\rho(\tilde{M}, \tilde{\pi}, s)$ for all states s.[7] This task is similar to *optimistic optimality* in *bounded parameter MDPs* as considered by Tewari and Bartlett (2007). A minor difference is that in our case the transition probabilities are bounded not individually but by the 1-norm. More importantly, while Tewari and Bartlett (2007) give a converging algorithm for computing the optimal value function, they do not bound the error when terminating their algorithm after finitely many steps. In the following, we will extend standard undiscounted value iteration (Puterman, 1994) to solve the set task.

First, note that we may combine all MDPs in $\mathcal{M}$ to get a single MDP with extended action set $\mathcal{A}'$. That is, we consider an MDP $\tilde{M}^+$ with continuous action space $\mathcal{A}'$, where for each action $a \in \mathcal{A}$, each admissible transition probability distribution $\tilde{p}(\cdot|s,a)$ according to (1) and each admissible mean reward $\tilde{r}(s,a)$ according to (2) there is an action in $\mathcal{A}'$ with transition probabilities $\tilde{p}(\cdot|s,a)$ and mean reward $\tilde{r}(s,a)$.[8] Then for each policy $\tilde{\pi}^+$ on $\tilde{M}^+$ there is an MDP $\tilde{M} \in \mathcal{M}$ and a policy $\tilde{\pi} : \mathcal{S} \to \mathcal{A}$ on $\tilde{M}$ such that the policies $\tilde{\pi}^+$ and $\tilde{\pi}$ induce the same transition probabilities and mean rewards on the respective MDP. (The other transition probabilities in $\tilde{M}$ can be set to $\hat{p}(\cdot|s,a)$.) On the other hand, for any given MDP $\tilde{M} \in \mathcal{M}$ and any policy $\tilde{\pi} : \mathcal{S} \to \mathcal{A}$ there is a policy $\tilde{\pi}^+$ on $\tilde{M}^+$ so that again the same transition probabilities and rewards are induced by $\tilde{\pi}$ on $\tilde{M}$ and $\tilde{\pi}^+$ on $\tilde{M}^+$. Thus, finding an MDP $\tilde{M} \in \mathcal{M}$ and a policy $\tilde{\pi}$ on $\tilde{M}$ such that $\rho(\tilde{M}, \tilde{\pi}, s) = \max_{M' \in \mathcal{M}, \pi, s'} \rho(M', \pi, s')$ for all initial states s, corresponds to finding an average reward optimal policy on $\tilde{M}^+$.

6. Since the policy $\tilde{\pi}_k$ is fixed for episode k, $v_k(s,a) \neq 0$ only for $a = \tilde{\pi}_k(s)$. Nevertheless, we find it convenient to use a notation which explicitly includes the action a in $v_k(s,a)$.
7. Note that, as we assume that $\mathcal{M}$ contains a communicating MDP, if an average reward of ρ is achievable in one state, it is achievable in all states.
8. Note that in $\tilde{M}^+$ the set of available actions now depends on the state.

Input: A confidence parameter $\delta \in (0,1)$, $\mathcal{S}$ and $\mathcal{A}$.

Initialization: Set $t := 1$, and observe the initial state s_1.

For episodes $k = 1,2,\ldots$ **do**

Initialize episode k:

1. Set the start time of episode k, $t_k := t$.
2. For all (s,a) in $\mathcal{S} \times \mathcal{A}$ initialize the state-action counts for episode k, $v_k(s,a) := 0$. Further, set the state-action counts prior to episode k,

$$N_k(s,a) := \#\{\tau < t_k : s_\tau = s, a_\tau = a\}.$$

3. For $s,s' \in \mathcal{S}$ and $a \in \mathcal{A}$ set the observed accumulated rewards and the transition counts prior to episode k,

$$R_k(s,a) := \sum_{\tau=1}^{t_k-1} r_\tau \mathbb{1}_{s_\tau=s, a_\tau=a},$$

$$P_k(s,a,s') := \#\{\tau < t_k : s_\tau = s, a_\tau = a, s_{\tau+1} = s'\}.$$

Compute estimates $\hat{r}_k(s,a) := \frac{R_k(s,a)}{\max\{1,N_k(s,a)\}}$, $\hat{p}_k(s'|s,a) := \frac{P_k(s,a,s')}{\max\{1,N_k(s,a)\}}$.

Compute policy $\tilde{\pi}_k$:

4. Let $\mathcal{M}_k$ be the set of all MDPs with states and actions as in M, and with transition probabilities $\tilde{p}(\cdot|s,a)$ close to $\hat{p}_k(\cdot|s,a)$, and rewards $\tilde{r}(s,a) \in [0,1]$ close to $\hat{r}_k(s,a)$, that is,

$$\left|\tilde{r}(s,a) - \hat{r}_k(s,a)\right| \leq \sqrt{\frac{7\log(2SAt_k/\delta)}{2\max\{1,N_k(s,a)\}}} \quad \text{and} \tag{3}$$

$$\left\|\tilde{p}(\cdot|s,a) - \hat{p}_k(\cdot|s,a)\right\|_1 \leq \sqrt{\frac{14S\log(2At_k/\delta)}{\max\{1,N_k(s,a)\}}}. \tag{4}$$

5. Use extended value iteration (see Section 3.1) to find a policy $\tilde{\pi}_k$ and an optimistic MDP $\tilde{M}_k \in \mathcal{M}_k$ such that

$$\tilde{\rho}_k := \min_s \rho(\tilde{M}_k, \tilde{\pi}_k, s) \geq \max_{M' \in \mathcal{M}_k, \pi, s'} \rho(M', \pi, s') - \frac{1}{\sqrt{t_k}}.$$

Execute policy $\tilde{\pi}_k$:

6. **While** $v_k(s_t, \tilde{\pi}_k(s_t)) < \max\{1, N_k(s_t, \tilde{\pi}_k(s_t))\}$ **do**
   (a) Choose action $a_t = \tilde{\pi}_k(s_t)$, obtain reward r_t, and observe next state s_{t+1}.
   (b) Update $v_k(s_t,a_t) := v_k(s_t,a_t) + 1$.
   (c) Set $t := t+1$.

Figure 1: The UCRL2 algorithm.

Input: Estimates $\hat{p}(\cdot|s,a)$ and distance $d(s,a)$ for a state-action pair (s,a), and the states in $\mathcal{S}$ sorted descendingly according to their u_i value. That is, let $\mathcal{S} := \{s'_1, s'_2, \ldots, s'_n\}$ with $u_i(s'_1) \geq u_i(s'_2) \geq \ldots \geq u_i(s'_n)$.

1. Set

$$\begin{aligned} p(s'_1) &:= \min\left\{1, \hat{p}(s'_1|s,a) + \tfrac{d(s,a)}{2}\right\}, \text{ and} \\ p(s'_j) &:= \hat{p}(s'_j|s,a) \text{ for all states } s'_j \text{ with } j > 1. \end{aligned}$$

2. Set $\ell := n$.
3. **While** $\sum_{s'_j \in \mathcal{S}} p(s'_j) > 1$ **do**
   (a) Reset $p(s'_\ell) := \max\{0, 1 - \sum_{s'_j \neq s'_\ell} p(s'_j)\}$.
   (b) Set $\ell := \ell - 1$.

Figure 2: Computing the inner maximum in the extended value iteration (5).

3.1.2 EXTENDED VALUE ITERATION

We denote the state values of the i-th iteration by $u_i(s)$. Then we get for undiscounted value iteration (Puterman, 1994) on $\tilde{M}^+$ for all $s \in \mathcal{S}$:

$$\begin{aligned} u_0(s) &= 0, \\ u_{i+1}(s) &= \max_{a \in \mathcal{A}} \left\{ \tilde{r}(s,a) + \max_{p(\cdot) \in \mathcal{P}(s,a)} \left\{ \sum_{s' \in \mathcal{S}} p(s') \cdot u_i(s') \right\} \right\}, \end{aligned} \tag{5}$$

where $\tilde{r}(s,a) := \hat{r}(s,a) + d'(s,a)$ are the maximal possible rewards according to condition (2), and $\mathcal{P}(s,a)$ is the set of transition probabilities $\tilde{p}\left(\cdot|s,a\right)$ satisfying condition (1).

While (5) is a step of value iteration with an infinite action space, $\max_p \boldsymbol{p} \cdot \boldsymbol{u}_i$ is actually a linear optimization problem over the convex polytope $\mathcal{P}(s,a)$. This implies that (5) can be evaluated considering only the finite number of vertices of this polytope.

Indeed, for a given state-action pair the inner maximum of (5) can be computed in $O(S)$ computation steps by an algorithm introduced by Strehl and Littman (2008). For the sake of completeness we display the algorithm in Figure 2. The idea is to put as much transition probability as possible to the state with maximal value $u_i(s)$ at the expense of transition probabilities to states with small values $u_i(s)$. That is, one starts with the estimates $\hat{p}(s'_j|s,a)$ for $p(s'_j)$ except for the state s'_1 with maximal $u_i(s)$, for which we set $p(s'_1) := \hat{p}(s'_1|s,a) + \frac{1}{2}d(s,a)$. In order to make $\boldsymbol{p}$ correspond to a probability distribution again, the transition probabilities from s to states with small $u_i(s)$ are reduced in total by $\frac{1}{2}d(s,a)$, so that $\|\boldsymbol{p} - \hat{p}\left(\cdot|s,a\right)\|_1 = d(s,a)$. This is done iteratively. Updating $\sum_{s'_j \in \mathcal{S}} p(s'_j)$ with every change of $\boldsymbol{p}$ for the computation of $\sum_{s'_j \neq s'_\ell} p(s'_j)$, this iterative procedure takes $O(S)$ steps. Thus, sorting the states according to their value $u_i(s)$ at each iteration i once, $\boldsymbol{u}_{i+1}$ can be computed from $\boldsymbol{u}_i$ in at most $O\left(S^2 A\right)$ steps.

3.1.3 CONVERGENCE OF EXTENDED VALUE ITERATION

We have seen that value iteration on the MDP $\tilde{M}^+$ with continuous action is equivalent to value iteration on an MDP with finite action set. Thus, in order to guarantee convergence, it is sufficient to assure that extended value iteration never chooses a policy with periodic transition matrix. (Intuitively, it is clear that optimal policies with periodic transition matrix do not matter as long as it is guaranteed that such a policy is not chosen by value iteration, compare to Sections 8.5, 9.4, and 9.5.3. of Puterman 1994. For a proof see Appendix B.) Indeed, extended value iteration always chooses a policy with aperiodic transition matrix: In each iteration there is a single fixed state s'_1 which is regarded as the "best" target state. For each state s, in the inner maximum an action with positive transition probability to s'_1 will be chosen. In particular, the policy chosen by extended value iteration will have positive transition probability from s'_1 to s'_1. Hence, this policy is aperiodic and has state independent average reward. Thus we obtain the following result.

Theorem 7 *Let $\mathcal{M}$ be the set of all MDPs with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probabilities $\tilde{p}(\cdot|s,a)$, and mean rewards $\tilde{r}(s,a)$ that satisfy* (1) *and* (2) *for given probability distributions $\hat{p}(\cdot|s,a)$, values $\hat{r}(s,a)$ in $[0,1]$, $d(s,a) > 0$, and $d'(s,a) \geq 0$. If $\mathcal{M}$ contains at least one communicating MDP, extended value iteration converges. Further, stopping extended value iteration when*

$$\max_{s\in S}\left\{u_{i+1}(s) - u_i(s)\right\} - \min_{s\in S}\left\{u_{i+1}(s) - u_i(s)\right\} < \varepsilon,$$

the greedy policy with respect to $\boldsymbol{u}_i$ is ε-optimal.

Remark 8 *When value iteration converges, a suitable transformation of $\boldsymbol{u}_i$ converges to the bias vector of an optimal policy. Recall that for a policy π the bias $\lambda(s)$ in state s is basically the expected advantage in total reward (for $T \to \infty$) of starting in state s over starting in the stationary distribution (the long term probability of being in a state) of π. For a fixed policy π, the Poisson equation*

$$\boldsymbol{\lambda} = \boldsymbol{r} - \rho\mathbf{1} + \boldsymbol{P}\boldsymbol{\lambda}$$

relates the bias vector $\boldsymbol{\lambda}$ to the average reward ρ, the mean reward vector $\boldsymbol{r}$, and the transition matrix $\boldsymbol{P}$. Now when value iteration converges, the vector $\boldsymbol{u}_i - \min_s u_i(s)\mathbf{1}$ converges to $\boldsymbol{\lambda} - \min_s \lambda(s)\mathbf{1}$. As we will see in inequality (11) *below, the so-called* span $\max_s u_i(s) - \min_s u_i(s)$ *of the vector $\boldsymbol{u}_i$ is upper bounded by the diameter D, so that this also holds for the span of the bias vector $\boldsymbol{\lambda}$ of the optimal policy found by extended value iteration, that is, $\max_s \lambda(s) - \min_s \lambda(s) \leq D$. Indeed, one can show that this holds for any optimal policy (cf. also Section 4 of Bartlett and Tewari, 2009).*

Remark 9 *We would like to note that the algorithm of Figure 2 can easily be adapted to the alternative setting of Tewari and Bartlett (2007), where each single transition probability $p(s'|s,a)$ is bounded as $0 \leq b^-(s',s,a) \leq p(s'|s,a) \leq b^+(s',s,a) \leq 1$. However, concerning convergence one needs to make some assumptions to exclude the possibility of choosing optimal policies with periodic transition matrices. For example, one may assume (apart from other assumptions already made by Tewari and Bartlett 2007) that for all s', s, a there is an admissible probability distribution $p(\cdot|s,a)$ with $p(s'|s,a) > 0$. Note that for Theorem 7 to hold, it is similarly essential that $d(s,a) > 0$. Alternatively, one may apply an aperiodicity transformation as described in Section 8.5.4 of Puterman (1994).*

Now returning to Step 5 of UCRL2, we stop value iteration when

$$\max_{s\in S}\{u_{i+1}(s)-u_i(s)\}-\min_{s\in S}\{u_{i+1}(s)-u_i(s)\}<\frac{1}{\sqrt{t_k}}, \tag{6}$$

which guarantees by Theorem 7 that the greedy policy with respect to $\boldsymbol{u}_i$ is $\frac{1}{\sqrt{t_k}}$-optimal.

4. Analysis of UCRL2 (Proofs of Theorem 2 and Corollary 3)

We start with a rough outline of the proof of Theorem 2. First, in Section 4.1, we deal with the random fluctuation of the rewards. Further, the regret is expressed as the sum of the regret accumulated in the individual episodes. That is, we set the *regret in episode k* to be

$$\Delta_k := \sum_{s,a} v_k(s,a)\big(\rho^* - \bar{r}(s,a)\big),$$

where $v_k(s,a)$ now denotes the final counts of state-action pair (s,a) in episode k. Then it is shown that the total regret can be bounded by

$$\sum_k \Delta_k + \sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)}$$

with high probability.

In Section 4.2, we consider the regret that is caused by failing confidence regions. We show that this term can be upper bounded by $\sqrt{T}$ with high probability. After this intermezzo, the regret of episodes for which the true MDP $M\in\mathcal{M}_k$ is examined in Section 4.3. Analyzing the extended value iteration scheme in Section 4.3.1 and using vector notation, we show that

$$\Delta_k \;\le\; \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k-\boldsymbol{I}\big)\boldsymbol{w}_k + 2\sum_{s,a} v_k(s,a)\sqrt{\tfrac{7\log(2SAt_k/\delta)}{2\max\{1,N_k(s,a)\}}} + 2\sum_{s,a}\frac{v_k(s,a)}{\sqrt{t_k}},$$

where $\tilde{\boldsymbol{P}}_k$ is the assumed transition matrix (in $\tilde{M}_k$) of the applied policy in episode k, $\boldsymbol{v}_k$ are the visit counts at the end of that episode, and $\boldsymbol{w}_k$ is a vector with $\|\boldsymbol{w}_k\|_\infty \le \frac{D(M)}{2}$. The last two terms in the above expression stem from the reward confidence intervals (3) and the approximation error of value iteration. These are bounded in Section 4.3.3 when summing over all episodes. The first term on the right hand side is analyzed further in Section 4.3.2 and split into

$$\begin{aligned}\boldsymbol{v}_k(\tilde{\boldsymbol{P}}_k-\boldsymbol{I})\boldsymbol{w}_k &= \boldsymbol{v}_k(\tilde{\boldsymbol{P}}_k-\boldsymbol{P}_k)\boldsymbol{w}_k + \boldsymbol{v}_k(\boldsymbol{P}_k-\boldsymbol{I})\boldsymbol{w}_k\\ &\le \big\|\boldsymbol{v}_k(\tilde{\boldsymbol{P}}_k-\boldsymbol{P}_k)\big\|_1\|\boldsymbol{w}_k\|_\infty + \boldsymbol{v}_k(\boldsymbol{P}_k-\boldsymbol{I})\boldsymbol{w}_k,\end{aligned}$$

where $\boldsymbol{P}_k$ is the true transition matrix (in M) of the policy applied in episode k. Substituting for $\tilde{\boldsymbol{P}}_k-\boldsymbol{P}_k$ the lengths of the confidence intervals as given in (4), the remaining term that needs analysis is $\boldsymbol{v}_k(\boldsymbol{P}_k-\boldsymbol{I})\boldsymbol{w}_k$. For the sum of this term over all episodes we obtain in Section 4.3.2 a high probability bound of

$$\sum_k \boldsymbol{v}_k(\boldsymbol{P}_k-\boldsymbol{I})\boldsymbol{w}_k \;\le\; D\sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)} + Dm,$$

where m is the number of episodes—a term shown to be logarithmic in T in Appendix C.2. Section 4.3.3 concludes the analysis of episodes with $M\in\mathcal{M}_k$ by summing the individual regret terms over all episodes k with $M\in\mathcal{M}_k$. In the final Section 4.4 we finish the proof by combining the results of Sections 4.1–4.3.

4.1 Splitting into Episodes

Recall that r_t is the (random) reward UCRL2 receives at step t when starting in some initial state s_1. For given state-action counts $N(s,a)$ after T steps, the r_t are independent random variables, so that by Hoeffding's inequality

$$\mathbb{P}\left\{ \sum_{t=1}^{T} r_t \le \sum_{s,a} N(s,a)\bar{r}(s,a) - \sqrt{\tfrac{5}{8}T\log\left(\tfrac{8T}{\delta}\right)} \,\middle|\, \big(N(s,a)\big)_{s,a} \right\} \le \left(\frac{\delta}{8T}\right)^{5/4} < \frac{\delta}{12T^{5/4}}. \tag{7}$$

Thus we get for the regret of UCRL2 (now omitting explicit references to M and UCRL2)

$$\Delta(s_1,T) \;=\; T\rho^* - \sum_{t=1}^{T} r_t \;<\; T\rho^* - \sum_{s,a} N(s,a)\bar{r}(s,a) + \sqrt{\tfrac{5}{8}T\log\left(\tfrac{8T}{\delta}\right)}$$

with probability at least $1-\frac{\delta}{12T^{5/4}}$. Denoting the number of episodes started up to step T by m, we have $\sum_{k=1}^{m} v_k(s,a) = N(s,a)$ and $\sum_{s,a} N(s,a) = T$. Therefore, writing $\Delta_k := \sum_{s,a} v_k(s,a)\big(\rho^* - \bar{r}(s,a)\big)$, it follows that

$$\Delta(s_1,T) \;\le\; \sum_{k=1}^{m} \Delta_k + \sqrt{\tfrac{5}{8}T\log\left(\tfrac{8T}{\delta}\right)} \tag{8}$$

with probability at least $1-\frac{\delta}{12T^{5/4}}$.

4.2 Dealing with Failing Confidence Regions

Let us now consider the regret of episodes in which the set of plausible MDPs $\mathcal{M}_k$ does not contain the true MDP M, $\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\notin\mathcal{M}_k}$. By the stopping criterion for episode k we have (except for episodes where $v_k(s,a)=1$ and $N_k(s,a)=0$, when $\sum_{s,a} v_k(s,a) = 1 \le t_k$ holds trivially)

$$\sum_{s,a} v_k(s,a) \;\le\; \sum_{s,a} N_k(s,a) = t_k - 1.$$

Hence, denoting $\mathcal{M}(t)$ to be the set of plausible MDPs as given by (3) and (4) using the estimates available at step t, we have due to $\rho^* \le 1$ that

$$\begin{aligned}
\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\notin\mathcal{M}_k} \;&\le\; \sum_{k=1}^{m}\sum_{s,a} v_k(s,a)\mathbb{1}_{M\notin\mathcal{M}_k} \;\le\; \sum_{k=1}^{m} t_k \mathbb{1}_{M\notin\mathcal{M}_k} \;=\; \sum_{t=1}^{T} t \sum_{k=1}^{m} \mathbb{1}_{t_k=t, M\notin\mathcal{M}_k} \\
&\le\; \sum_{t=1}^{T} t\mathbb{1}_{M\notin\mathcal{M}(t)} \;\le\; \sum_{t=1}^{\lfloor T^{1/4}\rfloor} t\mathbb{1}_{M\notin\mathcal{M}(t)} + \sum_{t=\lfloor T^{1/4}\rfloor+1}^{T} t\mathbb{1}_{M\notin\mathcal{M}(t)} \\
&\le\; \sqrt{T} + \sum_{t=\lfloor T^{1/4}\rfloor+1}^{T} t\mathbb{1}_{M\notin\mathcal{M}(t)}.
\end{aligned}$$

Now, $\mathbb{P}\left\{M\notin\mathcal{M}(t)\right\} \le \frac{\delta}{15t^6}$ (see Appendix C.1), and since

$$\sum_{t=\lfloor T^{1/4}\rfloor+1}^{T} \frac{1}{15t^6} \;\le\; \frac{1}{15T^{6/4}} + \int_{T^{1/4}}^{\infty} \frac{1}{15t^6}\,dt \;=\; \frac{1}{15T^{6/4}} + \frac{1}{75T^{5/4}} \;\le\; \frac{6}{75T^{5/4}} \;<\; \frac{1}{12T^{5/4}},$$

we have $\mathbb{P}\{\exists t : T^{1/4} < t \le T : M\notin\mathcal{M}(t)\} \le \frac{\delta}{12T^{5/4}}$. It follows that with probability at least $1-\frac{\delta}{12T^{5/4}}$,

$$\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\notin\mathcal{M}_k} \;\le\; \sqrt{T}. \tag{9}$$

4.3 Episodes with $M \in \mathcal{M}_k$

Now we assume that $M \in \mathcal{M}_k$ and start by considering the regret in a single episode k. The optimistic average reward $\tilde{\rho}_k$ of the optimistically chosen policy $\tilde{\pi}_k$ is essentially larger than the true optimal average reward ρ^*, and thus it is sufficient to calculate by how much the optimistic average reward $\tilde{\rho}_k$ overestimates the actual rewards of policy $\tilde{\pi}_k$. By the assumption $M \in \mathcal{M}_k$, the choice of $\tilde{\pi}_k$ and $\tilde{M}_k$ in Step 5 of UCRL2, and Theorem 7 we get that $\tilde{\rho}_k \geq \rho^* - \frac{1}{\sqrt{t_k}}$. Thus for the regret Δ_k accumulated in episode k we obtain

$$\Delta_k \;\leq\; \sum_{s,a} v_k(s,a)\big(\rho^* - \bar{r}(s,a)\big) \;\leq\; \sum_{s,a} v_k(s,a)\big(\tilde{\rho}_k - \bar{r}(s,a)\big) + \sum_{s,a} \frac{v_k(s,a)}{\sqrt{t_k}}\,. \tag{10}$$

4.3.1 EXTENDED VALUE ITERATION REVISITED

To proceed, we reconsider the extended value iteration of Section 3.1. As an important observation for our analysis, we find that for any iteration i the range of the state values is bounded by the diameter of the MDP M, that is,

$$\max_s u_i(s) - \min_s u_i(s) \leq D. \tag{11}$$

To see this, observe that $u_i(s)$ is the total expected i-step reward of an optimal non-stationary i-step policy starting in state s on the MDP $\tilde{M}^+$ with extended action set (as considered for extended value iteration). The diameter of this extended MDP is at most D as it contains by assumption the actions of the true MDP M. Now, if there were states s', s'' with $u_i(s'') - u_i(s') > D$, then an improved value for $u_i(s')$ could be achieved by the following nonstationary policy: First follow a policy which moves from s' to s'' most quickly, which takes at most D steps on average. Then follow the optimal i-step policy for s''. Since only D of the i rewards of the policy for s'' are missed, this policy gives $u_i(s') \geq u_i(s'') - D$, contradicting our assumption and thus proving (11).

It is a direct consequence of Theorem 8.5.6. of Puterman (1994), that when the convergence criterion (6) holds at iteration i, then

$$|u_{i+1}(s) - u_i(s) - \tilde{\rho}_k| \leq \frac{1}{\sqrt{t_k}} \tag{12}$$

for all $s \in \mathcal{S}$, where $\tilde{\rho}_k$ is the average reward of the policy $\tilde{\pi}_k$ chosen in this iteration on the optimistic MDP $\tilde{M}_k$.[9] Expanding $u_{i+1}(s)$ according to (5), we get

$$u_{i+1}(s) \;=\; \tilde{r}_k(s,\tilde{\pi}_k(s)) + \sum_{s'} \tilde{p}_k\left(s'|s,\tilde{\pi}_k(s)\right) \cdot u_i(s')$$

and hence by (12)

$$\left|\left(\tilde{\rho}_k - \tilde{r}_k(s,\tilde{\pi}_k(s))\right) - \left(\sum_{s'} \tilde{p}_k\left(s'|s,\tilde{\pi}_k(s)\right) \cdot u_i(s') - u_i(s)\right)\right| \leq \frac{1}{\sqrt{t_k}}. \tag{13}$$

Setting $\boldsymbol{r}_k := \big(\tilde{r}_k\big(s,\tilde{\pi}_k(s)\big)\big)_s$ to be the (column) vector of rewards for policy $\tilde{\pi}_k$, $\tilde{\boldsymbol{P}}_k := \big(\tilde{p}_k\left(s'|s,\tilde{\pi}_k(s)\right)\big)_{s,s'}$ the transition matrix of $\tilde{\pi}_k$ on $\tilde{M}_k$, and $\boldsymbol{v}_k := \big(v_k\big(s,\tilde{\pi}_k(s)\big)\big)_s$ the (row)

9. This is quite intuitive. We expect to receive average reward $\tilde{\rho}_k$ per step, such that the difference of the state values after $i+1$ and i steps should be about $\tilde{\rho}_k$.

vector of visit counts for each state and the corresponding action chosen by $\tilde{\pi}_k$, we can use (13)—recalling that $v_k(s,a) = 0$ for $a \neq \tilde{\pi}_k(s)$—to rewrite (10) as

$$\begin{aligned}\Delta_k &\leq \sum_{s,a} v_k(s,a)\big(\tilde{\rho}_k - \bar{r}(s,a)\big) + \sum_{s,a} \frac{v_k(s,a)}{\sqrt{t_k}} \\ &= \sum_{s,a} v_k(s,a)\big(\tilde{\rho}_k - \tilde{r}_k(s,a)\big) + \sum_{s,a} v_k(s,a)\big(\tilde{r}_k(s,a) - \bar{r}(s,a)\big) + \sum_{s,a} \frac{v_k(s,a)}{\sqrt{t_k}} \\ &\leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{u}_i + \sum_{s,a} v_k(s,a)\big(\tilde{r}_k(s,a) - \bar{r}(s,a)\big) + 2\sum_{s,a} \frac{v_k(s,a)}{\sqrt{t_k}}.\end{aligned}$$

Since the rows of $\tilde{\boldsymbol{P}}_k$ sum to 1, we can replace $\boldsymbol{u}_i$ by $\boldsymbol{w}_k$ where we set

$$w_k(s) := u_i(s) - \frac{\min_s u_i(s) + \max_s u_i(s)}{2},$$

such that it follows from (11) that $\|\boldsymbol{w}_k\| \leq \frac{D}{2}$. Further, since we assume $M \in \mathcal{M}_k$, $\tilde{r}_k(s,a) - \bar{r}(s,a) \leq |\tilde{r}_k(s,a) - \hat{r}_k(s,a)| + |\bar{r}(s,a) - \hat{r}_k(s,a)|$ is bounded according to (3), so that

$$\Delta_k \leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + 2\sum_{s,a} v_k(s,a)\sqrt{\tfrac{7\log(2SAt_k/\delta)}{2\max\{1,N_k(s,a)\}}} + 2\sum_{s,a} \frac{v_k(s,a)}{\sqrt{t_k}}. \tag{14}$$

Noting that $\max\{1,N_k(s,a)\} \leq t_k \leq T$ we get from (14) that

$$\Delta_k \leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + \left(\sqrt{14\log\left(\tfrac{2SAT}{\delta}\right)} + 2\right)\sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1,N_k(s,a)\}}}. \tag{15}$$

4.3.2 THE TRUE TRANSITION MATRIX

Now we want to replace the transition matrix $\tilde{\boldsymbol{P}}_k$ of the policy $\tilde{\pi}_k$ in the optimistic MDP $\tilde{M}_k$ by the transition matrix $\boldsymbol{P}_k := \big(p(s'|s,\tilde{\pi}_k(s))\big)_{s,s'}$ of $\tilde{\pi}_k$ in the true MDP M. Thus, we write

$$\begin{aligned}\boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k &= \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k + \boldsymbol{P}_k - \boldsymbol{I}\big)\boldsymbol{w}_k \\ &= \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k\big)\boldsymbol{w}_k + \boldsymbol{v}_k\big(\boldsymbol{P}_k - \boldsymbol{I}\big)\boldsymbol{w}_k.\end{aligned} \tag{16}$$

The first term. Since by assumption $\tilde{M}_k$ and M are in the set of plausible MDPs $\mathcal{M}_k$, the first term in (16) can be bounded using condition (4). Thus, also using that $\|\boldsymbol{w}_k\|_\infty \leq \frac{D}{2}$ we obtain

$$\begin{aligned}\boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k\big)\boldsymbol{w}_k &= \sum_s \sum_{s'} v_k\big(s,\tilde{\pi}_k(s)\big)\cdot\Big(\tilde{p}_k\left(s'|s,\tilde{\pi}_k(s)\right) - p\big(s'|s,\tilde{\pi}_k(s)\big)\Big)\cdot w_k(s') \\ &\leq \sum_s v_k\big(s,\tilde{\pi}_k(s)\big)\cdot\big\|\tilde{p}_k\left(\cdot\,|s,\tilde{\pi}_k(s)\right) - p(\cdot|s,\tilde{\pi}_k(s))\big\|_1\cdot\|\boldsymbol{w}_k\|_\infty \\ &\leq \sum_s v_k\big(s,\tilde{\pi}_k(s)\big)\cdot 2\sqrt{\frac{14S\log\left(2AT/\delta\right)}{\max\{1,N_k\left(s,\tilde{\pi}_k(s)\right)\}}}\cdot\frac{D}{2} \\ &\leq D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)}\sum_{s,a}\frac{v_k(s,a)}{\sqrt{\max\{1,N_k(s,a)\}}}.\end{aligned} \tag{17}$$

This term will turn out to be the dominating contribution in our regret bound.

The second term. The intuition about the second term in (16) is that the counts of the state visits $\boldsymbol{v}_k$ are relatively close to the stationary distribution $\boldsymbol{\mu}_k$ of the transition matrix $\boldsymbol{P}_k$, for which $\boldsymbol{\mu}_k \boldsymbol{P}_k = \boldsymbol{\mu}_k$, such that $\boldsymbol{v}_k\big(\boldsymbol{P}_k - \boldsymbol{I}\big)$ should be small. For the proof we define a suitable martingale and make use of the Azuma-Hoeffding inequality.

Lemma 10 (Azuma-Hoeffding inequality, Hoeffding 1963) *Let $X_1, X_2, \ldots$ be a martingale difference sequence with $|X_i| \leq c$ for all i. Then for all $\varepsilon > 0$ and $n \in \mathbb{N}$,*

$$\mathbb{P}\big\{ \textstyle\sum_{i=1}^n X_i \geq \varepsilon \big\} \;\leq\; \exp\big(-\frac{\varepsilon^2}{2nc^2}\big).$$

Denote the unit vectors with i-th coordinate 1 and all other coordinates 0 by $\boldsymbol{e}_i$. Let $s_1, a_1, s_2, \ldots, a_T$, s_{T+1} be the sequence of states and actions, and let $k(t)$ be the episode which contains step t. Consider the sequence $X_t := \big(p\,(\cdot|s_t,a_t) - \boldsymbol{e}_{s_{t+1}}\big)\,\boldsymbol{w}_{k(t)} \mathbb{1}_{M \in \mathcal{M}_{k(t)}}$ for $t = 1, \ldots, T$. Then for any episode k with $M \in \mathcal{M}_k$, we have due to $\|\boldsymbol{w}_k\|_\infty \leq \frac{D}{2}$ that

$$\begin{aligned}
\boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k &= \sum_{t=t_k}^{t_{k+1}-1} \big(p\,(\cdot|s_t,a_t) - \boldsymbol{e}_{s_t}\big)\boldsymbol{w}_k \\
&= \left(\sum_{t=t_k}^{t_{k+1}-1} p\,(\cdot|s_t,a_t) - \sum_{t=t_k}^{t_{k+1}-1} \boldsymbol{e}_{s_{t+1}} + \boldsymbol{e}_{s_{t_{k+1}}} - \boldsymbol{e}_{s_{t_k}}\right)\boldsymbol{w}_k \\
&= \sum_{t=t_k}^{t_{k+1}-1} X_t + w_k(s_{t_{k+1}}) - w_k(s_{t_k}) \\
&\leq \sum_{t=t_k}^{t_{k+1}-1} X_t + D.
\end{aligned}$$

Also due to $\|\boldsymbol{w}_k\|_\infty \leq \frac{D}{2}$, we have $|X_t| \leq (\|p\,(\cdot|s_t,a_t)\|_1 + \|\boldsymbol{e}_{s_{t+1}}\|_1)\frac{D}{2} \leq D$. Further, $\mathbb{E}\big[X_t \big| s_1, a_1, \ldots, s_t, a_t\big] = 0$, so that X_t is a sequence of martingale differences, and application of Lemma 10 gives

$$\mathbb{P}\left\{\sum_{t=1}^T X_t \geq D\sqrt{2T \cdot \tfrac{5}{4}\log\big(\tfrac{8T}{\delta}\big)}\right\} \leq \left(\frac{\delta}{8T}\right)^{5/4} < \frac{\delta}{12T^{5/4}}.$$

Since for the number of episodes we have $m \leq SA\log_2\big(\frac{8T}{SA}\big)$ as shown in Appendix C.2, summing over all episodes yields

$$\begin{aligned}
\sum_{k=1}^m \boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k \mathbb{1}_{M \in \mathcal{M}_k} &\leq \sum_{t=1}^T X_t + mD \\
&\leq D\sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)} + DSA\log_2\big(\tfrac{8T}{SA}\big) \qquad (18)
\end{aligned}$$

with probability at least $1 - \frac{\delta}{12T^{5/4}}$.

4.3.3 SUMMING OVER EPISODES WITH $M \in \mathcal{M}_k$

To conclude Section 4.3, we sum (15) over all episodes with $M \in \mathcal{M}_k$, using (16), (17), and (18), which yields that with probability at least $1 - \frac{\delta}{12T^{5/4}}$

$$\begin{aligned}
\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\in\mathcal{M}_k} &\leq \sum_{k=1}^{m} \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k\big)\boldsymbol{w}_k \mathbb{1}_{M\in\mathcal{M}_k} + \sum_{k=1}^{m} \boldsymbol{v}_k\big(\boldsymbol{P}_k - \boldsymbol{I}\big)\boldsymbol{w}_k \mathbb{1}_{M\in\mathcal{M}_k} \\
&\quad + \sum_{k=1}^{m} \left(\sqrt{14\log\left(\tfrac{2SAT}{\delta}\right)} + 2\right) \sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \\
&\leq D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} \cdot \sum_{k=1}^{m} \sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \\
&\quad + D\sqrt{\tfrac{5}{2}T\log\left(\tfrac{8T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right) \\
&\quad + \left(\sqrt{14\log\left(\tfrac{2SAT}{\delta}\right)} + 2\right) \sum_{k=1}^{m} \sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \,. \qquad (19)
\end{aligned}$$

Recall that $N(s,a) := \sum_k v_k(s,a)$ such that $\sum_{s,a} N(s,a) = T$ and $N_k(s,a) = \sum_{i<k} v_i(s,a)$. By the criterion for episode termination in Step 6 of the algorithm, we have that $v_k(s,a) \leq N_k(s,a)$. Using that for $Z_k = \max\left\{1, \sum_{i=1}^{k} z_i\right\}$ and $0 \leq z_k \leq Z_{k-1}$ it holds that (see Appendix C.3)

$$\sum_{k=1}^{n} \frac{z_k}{\sqrt{Z_{k-1}}} \leq \left(\sqrt{2}+1\right)\sqrt{Z_n},$$

we get

$$\sum_{s,a}\sum_{k} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \leq \left(\sqrt{2}+1\right)\sum_{s,a}\sqrt{N(s,a)}.$$

By Jensen's inequality we thus have

$$\sum_{s,a}\sum_{k} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \leq \left(\sqrt{2}+1\right)\sqrt{SAT}, \qquad (20)$$

and we get from (19) after some minor simplifications that with probability at least $1 - \frac{\delta}{12T^{5/4}}$

$$\begin{aligned}
\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\in\mathcal{M}_k} &\leq D\sqrt{\tfrac{5}{2}T\log\left(\tfrac{8T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right) \\
&\quad + \left(2D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} + 2\right)\left(\sqrt{2}+1\right)\sqrt{SAT} \,. \qquad (21)
\end{aligned}$$

4.4 Completing the Proof of Theorem 2

Finally, evaluating (8) by summing Δ_k over all episodes, we get by (9) and (21)

$$\begin{aligned}
\Delta(s_1, T) &\leq \sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\notin\mathcal{M}_k} + \sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\in\mathcal{M}_k} + \sqrt{\tfrac{5}{8}T\log\left(\tfrac{8T}{\delta}\right)} \\
&\leq \sqrt{\tfrac{5}{8}T\log\left(\tfrac{8T}{\delta}\right)} + \sqrt{T} + D\sqrt{\tfrac{5}{2}T\log\left(\tfrac{8T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right) \\
&\quad + \left(2D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} + 2\right)\left(\sqrt{2}+1\right)\sqrt{SAT} \qquad (22)
\end{aligned}$$

with probability at least $1-\frac{\delta}{12T^{5/4}}-\frac{\delta}{12T^{5/4}}-\frac{\delta}{12T^{5/4}}$. Further simplifications (given in Appendix C.4) yield that for any $T>1$ with probability at least $1-\frac{\delta}{4T^{5/4}}$

$$\Delta(s_1,T) \;\le\; 34DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}. \tag{23}$$

Since $\sum_{T=2}^{\infty}\frac{\delta}{4T^{5/4}}<\delta$ the statement of Theorem 2 follows by a union bound over all possible values of T. ■

4.5 Proof of Corollary 3

In order to obtain the PAC bound of Corollary 3 we simply have to find a sufficiently large T_0 such that for all $T\ge T_0$ the per-step regret is smaller than ε. By Theorem 2 this means that for all $T\ge T_0$ we shall have

$$\frac{34DS\sqrt{AT\log\left(\frac{T}{\delta}\right)}}{T}<\varepsilon, \text{ or equivalently} \quad T>\frac{34^2D^2S^2A\log\left(\frac{T}{\delta}\right)}{\varepsilon^2}. \tag{24}$$

Setting $T_0:=2\alpha\log\left(\frac{\alpha}{\delta}\right)$ for $\alpha:=\frac{34^2D^2S^2A}{\varepsilon^2}$ we have due to $x>2\log x$ (for $x>0$)

$$T_0 \;=\; \alpha\log\left(\frac{\alpha}{\delta}\cdot\frac{\alpha}{\delta}\right)>\alpha\log\left(2\frac{\alpha}{\delta}\log\left(\frac{\alpha}{\delta}\right)\right)=\alpha\log\left(\tfrac{T_0}{\delta}\right),$$

so that (24) as well as the corollary follow. ■

5. The Logarithmic Bound (Proof of Theorem 4)

To show the logarithmic upper bound on the expected regret, we start with a bound on the number of steps in suboptimal episodes (in the spirit of *sample complexity bounds* as given by Kakade, 2003). We say that an episode k is ε-*bad* if its average regret is more than ε, where the average regret of an episode of length ℓ_k is $\frac{\Delta_k}{\ell_k}$ with[10] $\Delta_k=\sum_{t=t_k}^{t_{k+1}-1}(\rho^*-r_t)$. The following result gives an upper bound on the number of steps taken in ε-bad episodes.

Theorem 11 *Let $L_\varepsilon(T)$ be the number of steps taken by* UCRL2 *in ε-bad episodes up to step T. Then for any initial state $s\in\mathcal{S}$, any $T>1$ and any $\varepsilon>0$, with probability of at least $1-3\delta$*

$$L_\varepsilon(T) \;\le\; 34^2\frac{D^2S^2A\log\left(\frac{T}{\delta}\right)}{\varepsilon^2}.$$

Proof The proof is an adaptation of the proof of Theorem 2 which gives an upper bound of $O\Big(DS\sqrt{L_\varepsilon A\log(AT/\delta)}\Big)$ on the regret $\Delta'_\varepsilon(s,T)$ in ε-bad episodes in terms of L_ε. The theorem then follows due to $\varepsilon L_\varepsilon\le\Delta'_\varepsilon(s,T)$.

Fix some $T>1$, and let K_ε and J_ε be two random sets that contain the indices of the ε-bad episodes up to step T and the corresponding time steps taken in these episodes, respectively. Then by an application of Hoeffding's inequality similar to (7) in Section 4.1 and a union bound over all possible values of L_ε, one obtains that with probability at least $1-\delta$,

$$\sum_{k\in K_\varepsilon}\sum_{t=t_k}^{t_{k+1}-1}r_t \;\ge\; \sum_{k\in K_\varepsilon}\sum_{s,a}v_k(s,a)\bar{r}(s,a)-\sqrt{2L_\varepsilon\log\left(\tfrac{T}{\delta}\right)}\,.$$

10. In the following we use the same notation as in the proof of Theorem 2.

Further, by summing up all error probabilities $\mathbb{P}\left\{M \notin \mathcal{M}(t)\right\} \leq \frac{\delta}{15t^6}$ for $t = 1, 2, \ldots$ one has

$$\mathbb{P}\left\{ \sum_{k \in K_\varepsilon} \Delta_k \mathbb{1}_{M \notin \mathcal{M}_k} > 0 \right\} \leq \delta \,.$$

It follows that with probability at least $1 - 2\delta$

$$\Delta'_\varepsilon(s,T) \;\leq\; \sqrt{2L_\varepsilon \log\left(\tfrac{T}{\delta}\right)} + \sum_{k \in K_\varepsilon} \Delta_k \mathbb{1}_{M \in \mathcal{M}_k}. \tag{25}$$

In order to bound the regret of a single episode with $M \in \mathcal{M}_k$ we follow the lines of the proof of Theorem 2 in Section 4.3. Combining (15), (16), and (17) we have that

$$\Delta_k \;\leq\; \boldsymbol{v}_k\big(\boldsymbol{P}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + \left(2D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} + 2\right) \sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \,. \tag{26}$$

In Appendix D we prove an analogon of (20), that is,

$$\sum_{k \in K_\varepsilon} \sum_{s,a} \frac{v_k(s,a)}{\sqrt{\max\{1, N_k(s,a)\}}} \;\leq\; \left(\sqrt{2}+1\right)\sqrt{L_\varepsilon SA} \,. \tag{27}$$

Then from (25), (26), and (27) it follows that with probability at least $1 - 2\delta$

$$\begin{aligned} \Delta'_\varepsilon(s,T) \;\;\leq\;\; & \sqrt{2L_\varepsilon \log\left(\tfrac{T}{\delta}\right)} + \left(2D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} + 2\right) \cdot \left(\sqrt{2}+1\right) \cdot \sqrt{L_\varepsilon SA} \\ & + \sum_{k \in K_\varepsilon} \boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k \mathbb{1}_{M \in \mathcal{M}_k} \,. \end{aligned} \tag{28}$$

For the regret term of $\sum_{k \in K_\varepsilon} \boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k \mathbb{1}_{M \in \mathcal{M}_k}$ we use an argument similar to the one applied to obtain (18) in Section 4.3.2. Here we have to consider a slightly modified martingale difference sequence

$$X_t = \big(p(\cdot|s_t, a_t) - \boldsymbol{e}_{s_{t+1}}\big)\boldsymbol{w}_{k(t)} \mathbb{1}_{M \in \mathcal{M}_{k(t)}} \mathbb{1}_{t \in J_\varepsilon}$$

for $t = 1, \ldots, T$ to get (using the bound on the number of episodes given in Appendix C.2)

$$\begin{aligned} \sum_{k \in K_\varepsilon} \boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k \mathbb{1}_{M \in \mathcal{M}_k} \;\;&\leq\;\; \sum_{t \in J_\varepsilon} X_t + DSA\log_2\left(\tfrac{8T}{SA}\right) \\ &\leq\;\; \sum_{t=1}^{T(L_\varepsilon)} X_t + DSA\log_2\left(\tfrac{8T}{SA}\right) \,, \end{aligned} \tag{29}$$

where we set $T(L) := \min\left\{t : \#\{\tau \leq t, \tau \in J_\varepsilon\} = L\right\}$. The application of the Azuma-Hoeffding inequality in Section 4.3.2 is replaced with the following consequence of Bernstein's inequality for martingales.

Lemma 12 (Freedman 1975) *Let $X_1, X_2, \ldots$ be a martingale difference sequence. Then*

$$\mathbb{P}\left\{ \sum_{i=1}^n X_i \geq \kappa,\; \sum_{i=1}^n X_i^2 \leq \gamma \right\} \;\leq\; \exp\left(-\frac{\kappa^2}{2\gamma + \frac{2\kappa}{3}}\right).$$

Application of Lemma 12 with $\kappa = 2D\sqrt{L\log(T/\delta)}$ and $\gamma = D^2 L$ yields that for $L \geq \frac{\log(T/\delta)}{D^2}$ it holds that

$$\mathbb{P}\left\{ \sum_{t=1}^{T(L)} X_t > 2D\sqrt{L\log\left(\tfrac{T}{\delta}\right)} \right\} < \frac{\delta}{T}. \tag{30}$$

On the other hand, if $L < \frac{\log(T/\delta)}{D^2}$, we have

$$\sum_{t=1}^{T(L)} X_t \;\leq\; DL \;=\; D\sqrt{L}\sqrt{L} \;<\; \sqrt{L}\sqrt{\log\left(\tfrac{T}{\delta}\right)} \;<\; 2D\sqrt{L\log\left(\tfrac{T}{\delta}\right)}. \tag{31}$$

Hence, (30) and (31) give by a union bound over all possible values of L_ε that with probability at least $1-\delta$

$$\sum_{t=1}^{T(L_\varepsilon)} X_t \;\leq\; 2D\sqrt{L_\varepsilon \log\left(\tfrac{T}{\delta}\right)}.$$

Together with (29) this yields that with probability at least $1-\delta$

$$\sum_{k\in K_\varepsilon} \boldsymbol{v}_k(\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k \mathbb{1}_{M\in\mathcal{M}_k} \;\leq\; 2D\sqrt{L_\varepsilon \log\left(\tfrac{T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right).$$

Thus by (28) we obtain that with probability at least $1-3\delta$

$$\begin{aligned}\Delta'_\varepsilon(s,T) \;\leq\; & \sqrt{2L_\varepsilon \log\left(\tfrac{T}{\delta}\right)} + \left(2D\sqrt{14S\log\left(\tfrac{2AT}{\delta}\right)} + 2\right)\cdot\left(\sqrt{2}+1\right)\cdot\sqrt{L_\varepsilon SA} \\ & + 2D\sqrt{L_\varepsilon \log\left(\tfrac{T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right).\end{aligned}$$

This can be simplified to

$$\Delta'_\varepsilon(s,T) \leq 34DS\sqrt{L_\varepsilon A\log\left(\tfrac{T}{\delta}\right)} \tag{32}$$

by similar arguments as given in Appendix C.4. Since $\varepsilon L_\varepsilon \leq \Delta'_\varepsilon(s,T)$, we get

$$L_\varepsilon \leq 34^2 \cdot \frac{D^2 S^2 A\log\left(\tfrac{T}{\delta}\right)}{\varepsilon^2}, \tag{33}$$

which proves the theorem. ■

Now we apply Theorem 11 to obtain the claimed logarithmic upper bound on the expected regret.

Proof of Theorem 4 Upper bounding L_ε in (32) by (33), we obtain for the regret $\Delta'_\varepsilon(s,T)$ accumulated in ε-bad episodes that

$$\Delta'_\varepsilon(s,T) \leq 34^2 \cdot \frac{D^2 S^2 A\log\left(\tfrac{T}{\delta}\right)}{\varepsilon}$$

with probability at least $1-3\delta$. Noting that the regret accumulated outside of ε-bad episodes is at most εT implies the first statement of the theorem.

For the bound on the expected regret, first note that the expected regret of each episode in which an optimal policy is executed is at most D, whereas due to Theorem 11 the expected regret in $\frac{g}{2}$-bad

episodes is upper bounded by $34^2 \cdot \frac{2 \cdot D^2 S^2 A \log(T)}{g} + 1$, as $\delta = \frac{1}{3T}$. What remains to do is to consider episodes k with expected average regret smaller than $\frac{g}{2}$ in which however a non-optimal policy $\tilde{\pi}_k$ was chosen.

First, note that for each policy π there is a T_π such that for all $T \geq T_\pi$ the expected average reward after T steps is $\frac{g}{2}$-close to the average reward of π. Thus, when a policy π is played in an episode of length $\geq T_\pi$ either the episode is $\frac{g}{2}$-bad (in expectation) or the policy π is optimal. Now we fix a state-action pair (s,a) and consider the episodes k in which the number of visits $v_k(s,a)$ in (s,a) is doubled. The corresponding episode lengths $\ell_k(s,a)$ are not necessarily increasing, but the $v_k(s,a)$ are monotonically increasing, and obviously $\ell_k(s,a) \geq v_k(s,a)$. Since the $v_k(s,a)$ are at least doubled, it takes at most $\lceil 1 + \log_2(\max_{\pi:\pi(s)=a} T_\pi) \rceil$ episodes until $\ell_k(s,a) \geq v_k(s,a) \geq \max_{\pi:\pi(s)=a} T_\pi$, when any policy π with $\pi(s) = a$ applied in episode k that is not $\frac{g}{2}$-bad (in expectation) will be optimal. Consequently, as only episodes of length smaller than $\max_{\pi:\pi(s)=a} T_\pi$ have to be considered, the regret of episodes k where $v_k(s,a) < \max_{\pi:\pi(s)=a} T_\pi$ is upper bounded by $\lceil 1 + \log_2(\max_{\pi:\pi(s)=a} T_\pi) \rceil \max_{\pi:\pi(s)=a} T_\pi$. Summing over all state-action pairs, we obtain an additional additive regret term of

$$\sum_{s,a} \left\lceil 1 + \log_2\Big(\max_{\pi:\pi(s)=a} T_\pi \Big) \right\rceil \max_{\pi:\pi(s)=a} T_\pi,$$

which concludes the proof of the theorem. ■

6. The Lower Bound (Proof of Theorem 5)

We first consider the two-state MDP depicted in Figure 3. That is, there are two states, the initial state $s_\circ$ and another state s_1, and $A' = \left\lfloor \frac{A-1}{2} \right\rfloor$ actions. For each action a, let the deterministic rewards be $r(s_\circ, a) = 0$ and $r(s_1, a) = 1$. For all but a single "good" action a^* let $p(s_1|s_\circ, a) = \delta := \frac{4}{D}$, whereas $p(s_1|s_\circ, a^*) = \delta + \varepsilon$ for some $0 < \varepsilon < \delta$ specified later in the proof. Further, let $p(s_\circ|s_1, a) = \delta$ for all a. The diameter of this MDP is $D' = \frac{1}{\delta} = \frac{D}{4}$. For the rest of the proof we assume that[11] $\delta \leq \frac{1}{3}$.

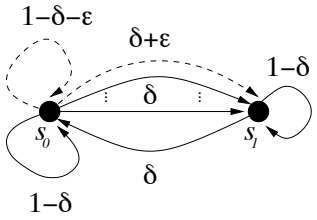


Figure 3: The MDP for the lower bound. The single action a^* with higher transition probability from state $s_\circ$ to state s_1 is shown as dashed line.

Consider $k := \left\lfloor \frac{S}{2} \right\rfloor$ copies of this MDP where only one of the copies has such a "good" action a^*. To complete the construction, we connect the k copies into a single MDP with diameter less than D,

11. Otherwise we have $D < 12$, so that due to the made assumptions $A > 2S$. In this case we employ a different construction: Using $S-1$ actions, we connect all states to get an MDP with diameter 1. With the remaining $A - S + 1$ actions we set up a bandit problem in each state as in the proof of the lower bound of Auer et al. (2002b) where only one state has a better action. This yields $\Omega(\sqrt{SAT})$ regret, which is sufficient, since D is bounded in this case.

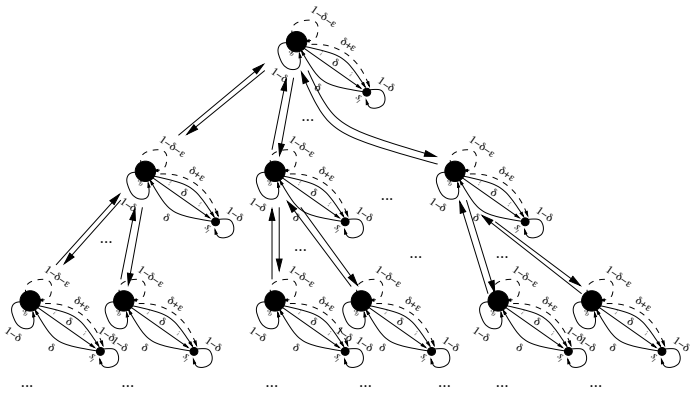


Figure 4: The composite MDP for the lower bound. Copies of the MDP of Figure 3 are arranged in an A'-ary tree, where the $s_\circ$-states are connected.

using at most $A - A'$ additional actions. This can be done by introducing $A'+1$ additional actions per state with deterministic transitions which do not leave the $s_|$-states and connect the $s_\circ$-states of the k copies by inducing an A'-ary tree structure on the $s_\circ$-states (one action for going toward the root, A' actions going toward the leaves—see Figure 4 for a schematic representation of the composite MDP). The reward for each of those actions is zero in any state. The diameter of the resulting MDP is at most $2(\frac{D}{4} + \lceil \log_{A'} k \rceil)$, which is twice the time it takes to travel to or from the root for any state in the MDP. Thus we have constructed an MDP M with $\leq S$ states, $\leq A$ actions, and diameter $\leq D$, for which we will show the claimed lower bound on the regret.

Actually, in the analysis we will consider the simpler MDP where all $s_\circ$-states are identified. We set this state to be the initial state. This MDP is equivalent to a single MDP M' like the one in Figure 3 with kA' actions which we assume in the following to be taken from $\{1,\ldots,kA'\}$. Note that learning this MDP is easier (as the learner is allowed to switch between different $s_\circ$-states without any cost for transition), while its optimal average reward is the same.

We prove the theorem by applying the same techniques as in the proof of the lower bound for the multi-armed bandit problem of Auer et al. (2002b). The pair $(s^*_\circ, a^*)$ identifying the copy with the better action and the better action is considered to be chosen uniformly at random from $\{1,\ldots,k\} \times \{1,\ldots,A'\}$, and we denote the expectation with respect to the random choice of $(s^*_\circ, a^*)$ as $\mathbb{E}_*[\cdot]$. We show that ε can be chosen such that M' and consequently also the composite MDP M forces regret $\mathbb{E}_*[\Delta(M, \mathfrak{A}, s_\circ, T)] \geq \mathbb{E}_*[\Delta(M', \mathfrak{A}, s^*_\circ, T)] > 0.015\sqrt{D'kA'T}$ on any algorithm $\mathfrak{A}$.

We write $\mathbb{E}_{\text{unif}}[\cdot]$ for the expectation when there is no special action (i.e., the transition probability from $s_\circ$ to $s_|$ is δ for all actions), and $\mathbb{E}_a[\cdot]$ for the expectation conditioned on a being the special action a^* in M'. As already argued by Auer et al. (2002b), it is sufficient to consider deterministic strategies for choosing actions. Indeed, any randomized strategy is equivalent to an (apriori) random choice from the set of all deterministic strategies. Thus, we may assume that any algorithm $\mathfrak{A}$ maps the sequence of observations up to step t to an action a_t.

Now we follow the lines of the proof of Theorem A.2 as given by Auer et al. (2002b). Let the random variables $N_|, N_\circ$ and $N^*_\circ$ denote the total number of visits to state $s_|$, the total number of visits to state $s_\circ$, and the number of times action a^* is chosen in state $s_\circ$, respectively. Further, write s_t as

usual for the state observed at step t. Then since $s_\circ$ is assumed to be the initial state, we have

$$\begin{aligned}
\mathbb{E}_a[N_\shortmid] &= \sum_{t=1}^{T} \mathbb{P}_a[s_t = s_\shortmid] = \sum_{t=2}^{T} \mathbb{P}_a[s_t = s_\shortmid] = \\
&= \sum_{t=2}^{T} \left(\mathbb{P}_a[s_t = s_\shortmid | s_{t-1} = s_\circ] \, \mathbb{P}_a[s_{t-1} = s_\circ] + \mathbb{P}_a[s_t = s_\shortmid | s_{t-1} = s_\shortmid] \, \mathbb{P}_a[s_{t-1} = s_\shortmid] \right) \\
&\leq \delta \sum_{t=2}^{T} \mathbb{P}_a[s_{t-1} = s_\circ, a_t \neq a^*] + (\delta + \varepsilon) \sum_{t=2}^{T} \mathbb{P}_a[s_{t-1} = s_\circ, a_t = a^*] + (1-\delta)\mathbb{E}_a[N_\shortmid] \\
&\leq \delta \mathbb{E}_a[N_\circ - N_\circ^*] + (\delta + \varepsilon)\mathbb{E}_a[N_\circ^*] + (1-\delta)\mathbb{E}_a[N_\shortmid].
\end{aligned}$$

Taking into account that choosing a^* instead of any other action in $s_\circ$ reduces the probability of staying in state $s_\circ$, it follows that (using $D' = \frac{1}{\delta}$)

$$\begin{aligned}
\mathbb{E}_a\left[R(M', \mathfrak{A}, s, T)\right] \leq \mathbb{E}_a[N_\shortmid] &\leq \mathbb{E}_a[N_\circ - N_\circ^*] + \tfrac{\delta+\varepsilon}{\delta}\mathbb{E}_a[N_\circ^*] \\
&= \mathbb{E}_a[N_\circ] + \mathbb{E}_a[N_\circ^*]\,\varepsilon D' \\
&\leq \mathbb{E}_{\text{unif}}[N_\circ] + \mathbb{E}_a[N_\circ^*]\,\varepsilon D' \\
&= \mathbb{E}_{\text{unif}}[T - N_\shortmid] + \mathbb{E}_a[N_\circ^*]\,\varepsilon D' \\
&= T - \mathbb{E}_{\text{unif}}[N_\shortmid] + \mathbb{E}_a[N_\circ^*]\,\varepsilon D'. \qquad (34)
\end{aligned}$$

Now denoting the step where the first transition from $s_\circ$ to $s_\shortmid$ occurs by $\tau_{\circ\shortmid}$, we may lower bound $\mathbb{E}_{\text{unif}}[N_\shortmid]$ by the law of total expectation as

$$\begin{aligned}
\mathbb{E}_{\text{unif}}[N_\shortmid] &= \sum_{t=1}^{T} \mathbb{E}_{\text{unif}}[N_\shortmid | \tau_{\circ\shortmid} = t] \, \mathbb{P}_{\text{unif}}[\tau_{\circ\shortmid} = t] = \sum_{t=1}^{T} \mathbb{E}_{\text{unif}}[N_\shortmid | \tau_{\circ\shortmid} = t] \, (1-\delta)^{t-1}\delta \\
&\geq \sum_{t=1}^{T-1} \frac{T-t}{2}(1-\delta)^{t-1}\delta = \frac{\delta T}{2}\sum_{t=1}^{T-1}(1-\delta)^{t-1} - \frac{\delta}{2}\sum_{t=1}^{T-1} t(1-\delta)^{t-1} \\
&= \frac{\delta T}{2} \cdot \frac{1-(1-\delta)^{T-1}}{\delta} - \frac{\delta}{2}\left(\frac{1-(1-\delta)^T}{\delta^2} - \frac{T(1-\delta)^{T-1}}{\delta} \right) \\
&= \frac{T - T(1-\delta)^{T-1}}{2} - \frac{1}{2\delta} + \frac{(1-\delta)^T}{2\delta} + \frac{T(1-\delta)^{T-1}}{2} \\
&= \frac{T}{2} - \frac{1}{2\delta} + \frac{(1-\delta)^T}{2\delta} \geq \frac{T}{2} - \frac{1}{2\delta} = \frac{T}{2} - \frac{D'}{2}. \qquad (35)
\end{aligned}$$

Therefore, combining (34) and (35) we obtain

$$\mathbb{E}_a\left[R(M', \mathfrak{A}, s, T)\right] \leq \frac{T}{2} + \mathbb{E}_a[N_\circ^*]\,\varepsilon D' + \frac{D'}{2}. \qquad (36)$$

As $\mathfrak{A}$ chooses its actions deterministically based on the observations so far, $N_\circ^*$ is a function of the observations up to step T, too. A slight difference to Auer et al. (2002b) is that in our setting the sequence of observations consists not just of the rewards but also of the next state, that is, upon playing action a_t the algorithm observes s_{t+1} and r_t. However, since the immediate reward is fully determined by the current state, $N_\circ^*$ is also a function of just the state sequence, and we may bound $\mathbb{E}_a[N_\circ^*]$ by the following lemma, adapted from Auer et al. (2002b).

Lemma 13 *Let $f : \{s_\circ, s_\downarrow\}^{T+1} \to [0,B]$ be any function defined on state sequences $\boldsymbol{s} \in \{s_\circ, s_\downarrow\}^{T+1}$ observed in the MDP M'. Then for any $0 \le \delta \le \frac{1}{2}$, any $0 \le \varepsilon \le 1-2\delta$, and any $a \in \{1,\ldots,kA'\}$,*

$$\mathbb{E}_a[f(\boldsymbol{s})] \;\le\; \mathbb{E}_{\text{unif}}[f(\boldsymbol{s})] + \frac{B}{2} \cdot \frac{\varepsilon}{\sqrt{\delta}} \sqrt{2\mathbb{E}_{\text{unif}}[N_\circ^*]}.$$

The proof of Lemma 13 is a straightforward modification of the respective proof given by Auer et al. (2002b). For details we refer to Appendix E.

Now let us assume that $\varepsilon \le \delta$. (Our final choice of ε below will satisfy this requirement.) By our assumption of $\delta \le \frac{1}{3}$ this yields that $\varepsilon \le \delta \le \frac{1}{3} \le 1-2\delta$. Then, since $N_\circ^*$ is a function of the state sequence with $N_\circ^* \in [0,T]$, we may apply Lemma 13 to obtain

$$\mathbb{E}_a[N_\circ^*] \;\le\; \mathbb{E}_{\text{unif}}[N_\circ^*] + \frac{T}{2}\varepsilon\sqrt{D'}\sqrt{2\mathbb{E}_{\text{unif}}[N_\circ^*]}. \tag{37}$$

An immediate consequence of (35) is that $\sum_{a=1}^{kA'} \mathbb{E}_{\text{unif}}[N_\circ^*] \le \frac{T}{2} + \frac{D'}{2}$, which yields by Jensen's inequality that $\sum_{a=1}^{kA'} \sqrt{2\mathbb{E}_{\text{unif}}[N_\circ^*]} \le \sqrt{kA'(T+D')}$. Thus we have from (37)

$$\begin{aligned}
\sum_{a=1}^{kA'} \mathbb{E}_a[N_\circ^*] &\le \frac{T}{2} + \frac{D'}{2} + \frac{\varepsilon T}{2}\sqrt{D'}\sqrt{kA'(T+D')} \\
&\le \frac{T}{2} + \frac{D'}{2} + \frac{\varepsilon T}{2}\sqrt{D'kA'T} + \frac{\varepsilon T D'}{2}\sqrt{kA'}.
\end{aligned}$$

Together with (36) this gives

$$\begin{aligned}
\mathbb{E}_*\left[R(M',\mathfrak{A},s,T)\right] &= \frac{1}{kA'}\sum_{a=1}^{kA'} \mathbb{E}_a[R(M,\mathfrak{A},s,T)] \\
&\le \frac{T}{2} + \frac{\varepsilon T D'}{2kA'} + \frac{\varepsilon D'^2}{2kA'} + \frac{\varepsilon^2 T D'}{2kA'}\sqrt{D'kA'T} + \frac{\varepsilon^2 T D'^2}{2kA'}\sqrt{kA'} + \frac{D'}{2}.
\end{aligned}$$

Calculating the stationary distribution, we find that the optimal average reward for the MDP M' is $\frac{\delta+\varepsilon}{2\delta+\varepsilon}$. Hence, the expected regret with respect to the random choice of a^* is at least

$$\begin{aligned}
&\mathbb{E}_*\left[\Delta(M',\mathfrak{A},s,T)\right] = \frac{\delta+\varepsilon}{2\delta+\varepsilon}T - \mathbb{E}_*[R(M,\mathfrak{A},s,T)] \\
&\ge \frac{\delta+\varepsilon}{2\delta+\varepsilon}T - \frac{T}{2} - \frac{\varepsilon T D'}{2kA'} - \frac{\varepsilon D'}{2kA'}\cdot D' - \frac{\varepsilon^2 T D'}{2kA'}\sqrt{D'kA'T} - \frac{\varepsilon^2 T D'}{2kA'}\sqrt{D'kA'}\cdot\sqrt{D'} - \frac{D'}{2}.
\end{aligned}$$

Since by assumption we have $T \ge DSA \ge 16D'kA'$ and thus $D' \le \frac{T}{16kA'}$, it follows that

$$\begin{aligned}
&\mathbb{E}_*\left[\Delta(M',\mathfrak{A},s,T)\right] \\
&\ge \frac{\varepsilon T}{4\delta+2\varepsilon} - \frac{\varepsilon T D'}{2kA'} - \frac{\varepsilon D'}{2kA'}\cdot\frac{T}{16kA'} - \frac{\varepsilon^2 T D'}{2kA'}\sqrt{D'kA'T} - \frac{\varepsilon^2 T D'}{2kA'}\sqrt{D'kA'}\sqrt{\frac{T}{16kA'}} - \frac{D'}{2} \\
&= \frac{\varepsilon T}{4\delta+2\varepsilon} - \varepsilon T D'\left(\frac{1}{2kA'} + \frac{1}{32k^2A'^2}\right) - \frac{\varepsilon^2 T D'}{kA'}\sqrt{D'kA'T}\left(\frac{1}{2} + \frac{1}{8\sqrt{kA'}}\right) - \frac{D'}{2}.
\end{aligned}$$

Now we choose $\varepsilon := c\sqrt{\frac{kA'}{TD'}}$, where $c := \frac{1}{5}$. Then because of $\frac{1}{\delta} = D' \leq \frac{T}{16kA'}$ it follows that $\varepsilon \leq c\frac{1}{4D'} = \frac{\delta}{20}$ (so that also $\varepsilon \leq \delta$ as needed to get (37)), and further $\frac{1}{4\delta+2\varepsilon} \geq \frac{1}{4+1/8}D'$. Hence we obtain

$$\mathbb{E}_*\left[\Delta(M',\mathfrak{A},s,T)\right] \;\geq\; \left(\frac{c}{4+\frac{1}{8}} - \frac{c}{2kA'} - \frac{c}{32k^2A'^2} - \frac{c^2}{2} - \frac{c^2}{8\sqrt{kA'}}\right)\sqrt{D'kA'T} - \frac{D'}{2}.$$

Finally, we note that

$$\frac{D'}{2} \leq \tfrac{1}{2}\sqrt{D'}\sqrt{\frac{T}{16kA'}} = \frac{1}{8kA'}\sqrt{D'kA'T},$$

and since by assumption $S,A \geq 10$ so that $kA' \geq 20$, it follows that

$$\mathbb{E}_*\left[\Delta(M',\mathfrak{A},s,T)\right] \;>\; 0.015\sqrt{D'kA'T},$$

which concludes the proof. ■

7. Regret Bounds for Changing MDPs (Proof of Theorem 6)

Consider the learner operates in a setting where the MDP is allowed to change ℓ times, such that the diameter never exceeds D (we assume an initial change at time $t = 1$). For this task we define the regret of an algorithm $\mathfrak{A}$ up to step T with respect to the average reward $\rho^*(t)$ of an optimal policy at step t as

$$\Delta'(\mathfrak{A},s,T) := \sum_{t=1}^{T} \rho^*(t) - r_t,$$

where r_t is as usual the reward received by $\mathfrak{A}$ at step t when starting in state s.

The intuition behind our approach is the following: When restarting UCRL2 every $\left(\frac{T}{\ell}\right)^{2/3}$ steps, the total regret for periods in which the MDP changes is at most $\ell^{1/3}T^{2/3}$. For each other period we have regret of $\tilde{O}\big((\frac{T}{\ell})^{1/3}\big)$ by Theorem 2. Since UCRL2 is restarted only $T^{1/3}\ell^{2/3}$ times, the total regret is $\tilde{O}\big(\ell^{1/3}T^{2/3}\big)$.

Because the horizon T is usually unknown, we use an alternative approach for restarting which however exhibits similar properties: UCRL2′ restarts UCRL2 with parameter $\frac{\delta}{\ell^2}$ at steps $\tau_i = \left\lceil \frac{i^3}{\ell^2} \right\rceil$ for $i = 1,2,3,\ldots$. Now we prove Theorem 6, which states that the regret of UCRL2′ is bounded by

$$\Delta'(\text{UCRL2}',s,T) \leq 65 \cdot \ell^{1/3}T^{2/3}DS\sqrt{A\log\left(\tfrac{T}{\delta}\right)}$$

with probability at least $1-\delta$ in the considered setting.

Let n be the largest natural number such that $\left\lceil \frac{n^3}{\ell^2} \right\rceil \leq T$, that is, n is the number of restarts up to step T. Then $\frac{n^3}{\ell^2} \leq \tau_n \leq T \leq \tau_{n+1} - 1 < \frac{(n+1)^3}{\ell^2}$ and consequently

$$\ell^{2/3}T^{1/3} - 1 \;\leq\; n \;\leq\; \ell^{2/3}T^{1/3}. \tag{38}$$

The regret Δ_c incurred due to changes of the MDP can be bounded by the number of steps taken in periods in which the MDP changes. This is maximized when the changes occur during the ℓ

longest periods, which contain at most $\tau_{n+1} - 1 - \tau_{n-\ell+1}$ steps. Hence we have

$$\begin{aligned}\Delta_c &\leq \tau_{n+1} - 1 - \tau_{n-\ell+1} \\ &\leq \tfrac{1}{\ell^2}(n+1)^3 - \tfrac{1}{\ell^2} - \tfrac{1}{\ell^2}(n-\ell+1)^3 \\ &= 3\tfrac{n^2}{\ell} + 6\tfrac{n}{\ell} - 3n - \tfrac{1}{\ell^2} + \ell - 3 + 3\tfrac{1}{\ell}. \end{aligned} \tag{39}$$

For $\ell \geq 2$ we get by (39) and (38) that

$$\Delta_c \leq 3\frac{n^2}{\ell} + \ell \leq 3\frac{\ell^{4/3}T^{2/3}}{\ell} + \ell = 3\ell^{1/3}T^{2/3} + \ell,$$

while for $\ell = 1$ we obtain also from (39) and (38) that

$$\Delta_c \leq 3n^2 + 3n \leq 3T^{2/3} + 3T^{1/3}.$$

Thus the contribution to the regret from changes of the MDP is at most

$$\begin{aligned}\Delta_c &\leq 3\ell^{1/3}T^{2/3} + 3T^{1/3} + \ell \\ &\leq 6\ell^{1/3}T^{2/3} + \ell^{1/3}\ell^{2/3} \\ &\leq 6\ell^{1/3}T^{2/3} + \ell^{1/3}T^{2/3} \\ &\leq 7\ell^{1/3}T^{2/3}. \end{aligned} \tag{40}$$

On the other hand, if the MDP does not change between the steps τ_i and $\min\{T, \tau_{i+1}\}$, the regret $\Delta(s_{\tau_i}, T_i)$ for these $T_i := \min\{T, \tau_{i+1}\} - \tau_i$ steps is bounded according to Theorem 2 (or more precisely (23)). Therefore, recalling that the confidence parameter is chosen to be $\frac{\delta}{\ell^2}$, this gives

$$\Delta(s_{\tau_i}, T_i) \leq 34DS\sqrt{T_i A \log\left(\tfrac{\ell^2 T_i}{\delta}\right)} \leq 34\sqrt{3}DS\sqrt{T_i}\sqrt{A\log\left(\tfrac{T}{\delta}\right)}$$

with probability $1 - \frac{\delta}{4\ell^2 T_i^{5/4}}$. As $\sum_{i=1}^n T_i = T$, we have by Jensen's inequality $\sum_{i=1}^n \sqrt{T_i} \leq \sqrt{n}\sqrt{T}$. Thus, summing over all $i = 1, \ldots, n$, the regret Δ_f in periods in which the MDP does not change is by (38)

$$\begin{aligned}\Delta_f &\leq \sum_{i=1}^n \Delta(s_{\tau_i}, T_i) \leq 34\sqrt{3}DS\sqrt{n}\sqrt{T}\sqrt{A\log\left(\tfrac{T}{\delta}\right)} \\ &\leq 34\sqrt{3}DS\,\ell^{1/3}\,T^{2/3}\,\sqrt{A\log\left(\tfrac{T}{\delta}\right)} \end{aligned} \tag{41}$$

with probability at least $1 - \sum_{i=1}^n \frac{\delta}{4\ell^2 T_i^{5/4}}$. We conclude the proof by bounding this latter probability. For $\left\lfloor \frac{\ell^2}{3} \right\rfloor < i < n$,

$$\begin{aligned}T_i &= \left\lceil \frac{(i+1)^3}{\ell^2} \right\rceil - \left\lceil \frac{i^3}{\ell^2} \right\rceil \\ &\geq \frac{(i+1)^3}{\ell^2} - \frac{i^3}{\ell^2} - \frac{\ell^2 - 1}{\ell^2} \\ &= \frac{3i^2}{\ell^2} + \frac{3i + 2 - \ell^2}{\ell^2} \geq \frac{3i^2}{\ell^2}, \end{aligned}$$

and consequently $\frac{1}{\ell^2 T_i^{5/4}} \leq \frac{1}{i^2}$. This together with $T_i \geq 1$ then yields

$$\begin{aligned} 1 - \sum_{i=1}^{n} \frac{\delta}{4\ell^2 T_i^{5/4}} &\geq 1 - \sum_{i=1}^{\lfloor \ell^2/3 \rfloor} \frac{\delta}{4\ell^2} - \sum_{i=\lfloor \ell^2/3 \rfloor + 1}^{n-1} \frac{\delta}{4i^2} - \frac{\delta}{4\ell^2} \\ &> 1 - \frac{\ell^2}{3} \cdot \frac{\delta}{4\ell^2} - \frac{\delta}{4} \sum_{i=1}^{\infty} \frac{1}{i^2} - \frac{\delta}{4} \\ &= 1 - \frac{\delta}{3} - \frac{\delta}{4} \cdot \frac{\pi^2}{6} > 1 - \delta. \end{aligned}$$

As $\Delta'(\text{UCRL2}', s, T) \leq \Delta_c + \Delta_f$, combining (40) and (41) yields

$$\Delta'(\text{UCRL2}', s, T) \leq 7\ell^{1/3} T^{2/3} + 34\sqrt{3} DS\, \ell^{1/3}\, T^{2/3} \sqrt{A \log\left(\tfrac{T}{\delta}\right)}$$

with probability at least $1 - \delta$, and Theorem 6 follows, since the claimed bound holds trivially for $A \log\left(\frac{T}{\delta}\right) < \log 4$. ■

8. Open Problems

There is still a gap between the upper bound on the regret of Theorem 2 and the lower bound of Theorem 5. We conjecture that the lower bound gives the right exponents for the parameters S and D (concerning the dependence on S compare also the sample complexity bounds of Strehl et al., 2006). The recent research of Bartlett and Tewari (2009) also poses the question whether the diameter in our bounds can be replaced by a smaller parameter, that is, by the span of the bias of an optimal policy. As the algorithm REGAL.C of Bartlett and Tewari (2009) demonstrates, this is at least true when this value is known to the learner. However, in the case of ignorance, currently this replacement of the diameter D can only be achieved at the cost of an additional factor of $\sqrt{S}$ in the regret bounds (Bartlett and Tewari, 2009). The difficulty in the proof is that while the span of an optimal policy's bias vector in the *assumed* optimistic MDP can be upper bounded by the diameter of the *true* MDP (cf. Remark 8), it is not clear how the spans of optimal policies in the assumed and the true MDP relate to each other.

A somehow related question is that of *transient* states, that is, the possibility that some of the states are not reachable under any policy. In this case the diameter is unbounded, so that our bounds become vacuous. Indeed, our algorithm cannot handle transient states: for any time step and any transient state s, UCRL2 optimistically assumes maximal possible reward in s and a very small but still positive transition probability to s from any other state. Thus insisting on the possibility of a transition to s, the algorithm fails to detect an optimal policy.[12] The assumption of having an upper bound on an optimal policy's bias resolves this problem, as this bound indirectly also gives some information on what the learner may expect from a state that has not been reached so far and thus may be transient. Consequently, with the assumed knowledge of such an upper bound, the REGAL.C algorithm of Bartlett and Tewari (2009) is also able to deal with transient states.

12. Actually, one can modify UCRL2 to deal with transient states by assuming transition probability 0 for all transitions not observed so far. This is complemented by an additional exploration phase between episodes where, for example, the state-action pair with the fewest number of visits is probed. While this algorithm gives asymptotically the same bounds, these however contain a large additive constant for all the episodes that occur before the transition structure assumed by the algorithm is correct.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work was supported in part by the Austrian Science Fund FWF (S9104-N13 SP4). The research leading to these results has also received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements n° 216886 (PASCAL2 Network of Excellence), and n° 216529 (Personal Information Navigator Adapting Through Viewing, PinView), and the Austrian Federal Ministry of Science and Research. This publication only reflects the authors' views.

Appendix A. A Lower Bound on the Diameter

We are going to show a more general result, from which the bound on the diameter follows. For a given MDP, let $T^*(s|s_0)$ be the minimal expected time it takes to move from state s_0 to state s.

Theorem 14 *Consider an MDP with state space $\mathcal{S}$ and A states. Let d_0 be an arbitrary distribution over $\mathcal{S}$, and $\mathcal{U} \subseteq \mathcal{S}$ be any subset of states. Then the sum of the minimal expected transition times to states in $\mathcal{U}$ when starting in an initial state distributed according to d_0 is bounded as follows:*

$$\mathcal{T}(\mathcal{U}|d_0) := \sum_{s\in\mathcal{U}} \sum_{s_0\in\mathcal{S}} d_0(s_0)\, T^*(s|s_0) \;\geq \min_{\substack{0\leq n_k\leq A^k, k\geq 0,\\ \sum_k n_k = |\mathcal{U}|}} \sum_k k\cdot n_k.$$

We think this bound is tight. The minimum on the right hand side is attained when the n_k are maximized for small k until $|\mathcal{U}|$ is exhausted. For $A \geq 2$, this gives an average (over the states in $\mathcal{U}$) expected transition time of at least $\log_A|\mathcal{U}| - 3$ to states in $\mathcal{U}$. Indeed, for $|\mathcal{U}| = \sum_{k=0}^{m-1} A^k + n_m$ we have $\frac{A^{m+1}-A}{(A-1)^2} < |\mathcal{U}|\left(1+\frac{1}{A-1}\right)$ as well as $m \geq \log_A\left(\frac{|\mathcal{U}|}{2}\right)$, so that

$$\begin{aligned}
\sum_{k=0}^{m-1} kA^k + m\cdot n_m &= m|\mathcal{U}| + \sum_{k=0}^{m-1}(k-m)A^k \\
&= m|\mathcal{U}| + \frac{m}{A-1} - \frac{A^{m+1}-A}{(A-1)^2} \\
&> |\mathcal{U}|\left(m-1-\tfrac{1}{A-1}\right) \\
&\geq |\mathcal{U}|\left(\log_A\left(\tfrac{|\mathcal{U}|}{2}\right) - 1 - \tfrac{1}{A-1}\right) \\
&\geq |\mathcal{U}|\left(\log_A|\mathcal{U}| - 3\right).
\end{aligned}$$

In particular, choosing $\mathcal{U} = \mathcal{S}$ gives the claimed lower bound on the diameter.

Corollary 15 *In any MDP with S states and $A \geq 2$ actions, the diameter D is lower bounded by $\log_A S - 3$.*

Remark 16 *For given S, A the minimal diameter is not always assumed by an MDP with deterministic transitions. Consider for example $S = 4$ and $A = 2$. Any deterministic MDP with four states and two actions has diameter at least 2. However, Figure 5 shows a corresponding MDP whose diameter is $\frac{3}{2}$.*

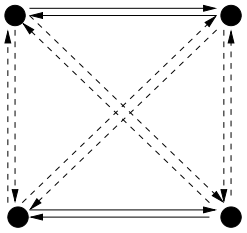


Figure 5: An MDP with four states and two actions whose diameter is $\frac{3}{2}$. In each state two actions are available. One action leads to another state deterministically, while the other action causes a random transition to each of the two other states with probability $\frac{1}{2}$ (indicated as dashed lines).

Proof of Theorem 14 Let $a^*(s_0,s)$ be the optimal action in state s_0 for reaching state s, and let $p(s|s_0,a)$ be the transition probability to state s when choosing action a in state s_0.

The proof is by induction on the size of $\mathcal{U}$. For $|\mathcal{U}| = 0,1$ the statement holds.

For $|\mathcal{U}| > 1$ we have

$$\begin{aligned}
\mathcal{T}(\mathcal{U}|d_0) &= \sum_{s_0\in\mathcal{S}}\sum_{s\in\mathcal{U}} d_0(s_0)\,T^*(s|s_0) \\
&= \sum_{s_0\in\mathcal{S}}\sum_{s\in\mathcal{U}\setminus\{s_0\}} d_0(s_0)\,T^*(s|s_0) \\
&= \sum_{s_0\in\mathcal{S}}\sum_{s\in\mathcal{U}\setminus\{s_0\}} d_0(s_0)\left(\sum_{s_1\in\mathcal{S}} p\big(s_1|s_0,a^*(s_0,s)\big)\,T^*(s|s_1)+1\right) \\
&= \sum_{s_0\in\mathcal{S}} d_0(s_0)\sum_a \sum_{\substack{s\in\mathcal{U}\setminus\{s_0\},\\ a^*(s_0,s)=a}} \left(\sum_{s_1\in\mathcal{S}} p(s_1|s_0,a)\,T^*(s|s_1)+1\right) \\
&= \sum_{s_0\in\mathcal{S}} d_0(s_0)\sum_a \left(\sum_{s_1\in\mathcal{S}}\sum_{s\in\mathcal{U}_{s_0,a}} p(s_1|s_0,a)\,T^*(s|s_1)+|\,\mathcal{U}_{s_0,a}|\right),
\end{aligned}$$

where $\mathcal{U}_{s_0,a} := \big\{s\in\mathcal{U}\setminus\{s_0\} : a^*(s_0,s)=a\big\}$.

If all $\mathcal{U}_{s_0,a}\subset\mathcal{U}$, we apply the induction hypothesis and obtain for suitable $n_k(s_0,a)$

$$\begin{aligned}
&\sum_{s_0\in\mathcal{S}} d_0(s_0)\sum_a \left(\sum_{s_1\in\mathcal{S}}\sum_{s\in\mathcal{U}_{s_0,a}} p(s_1|s_0,a)\,T^*(s|s_1)+|\,\mathcal{U}_{s_0,a}|\right) \\
&\quad\geq \sum_{s_0\in\mathcal{S}} d_0(s_0)\sum_a \left(\sum_k k\cdot n_k(s_0,a)+|\,\mathcal{U}_{s_0,a}|\right) \\
&\quad= \sum_{s_0\in\mathcal{S}} d_0(s_0)\sum_a\sum_k (k+1)\cdot n_k(s_0,a),
\end{aligned}$$

since $\sum_k n_k(s_0,a) = |\,\mathcal{U}_{s_0,a}|$. Furthermore, $n_k(s_0,a)\leq A^k$ and $|\mathcal{U}|-1\leq\sum_a|\,\mathcal{U}_{s_0,a}|\leq|\mathcal{U}|$. Thus setting $n'_k = \sum_{s_0} d_0(s_0)\sum_a n_{k-1}(s_0,a)$ for $k\geq 1$ and $n'_0 = |\mathcal{U}| - \sum_{k\geq 1} n'_k$ satisfies the conditions of the statement. This completes the induction step for this case.

If $\mathcal{U}_{s_0,a} = \mathcal{U}$ for some pair (s_0,a) (i.e., for all target states $s \in \mathcal{U}$ the same action is optimal in s_0), then we construct a modified MDP with shorter transition times. This is achieved by modifying one of the actions to give a deterministic transition from s_0 to some state in $\mathcal{U}$ (which is not reached deterministically by choosing action a). For the modified MDP the induction step works and the lower bound can be proven, which then also holds for the original MDP. ■

Appendix B. Convergence of Value Iteration (Proof of Theorem 7)

As sufficient condition for convergence of value iteration, Puterman (1994) assumes only that all optimal policies have aperiodic transition matrices. Actually, the proof of Theorem 9.4.4 of Puterman (1994)—the main result on convergence of value iteration—needs this assumption only at one step, that is, to guarantee that the optimal policy identified at the end of the proof has aperiodic transition matrix. In the following we give a proof sketch of Theorem 7 that concentrates on the differences to the convergence proof given by Puterman (1994).

Lemma 9.4.3 of Puterman (1994) shows that value iteration eventually chooses only policies π that satisfy $\boldsymbol{P}_\pi \boldsymbol{\rho}^* = \boldsymbol{\rho}^*$, where $\boldsymbol{P}_\pi$ is the transition matrix of π and $\boldsymbol{\rho}^*$ is the optimal average reward vector. More precisely, there is an i_0 such that for all $i \geq i_0$

$$\max_{\pi:\mathcal{S}\to\mathcal{A}} \{\boldsymbol{r}_\pi + \boldsymbol{P}_\pi \boldsymbol{u}_i\} = \max_{\pi\in E} \{\boldsymbol{r}_\pi + \boldsymbol{P}_\pi \boldsymbol{u}_i\},$$

where $\boldsymbol{r}_\pi$ is the reward vector of the policy π, and $E := \{\pi : \mathcal{S} \to \mathcal{A} \,|\, \boldsymbol{P}_\pi \boldsymbol{\rho}^* = \boldsymbol{\rho}^*\}$.

Unlike standard value iteration, extended value iteration always chooses policies with aperiodic transition matrix (cf. the discussion in Section 3.1.3). Thus when considering only aperiodic policies $F := \{\pi : \mathcal{S} \to \mathcal{A} | \, \boldsymbol{P}_\pi \text{ is aperiodic}\}$ in the proof of Lemma 9.4.3, the same argument shows that there is an i_0' such that for all $i \geq i_0'$

$$\max_{\pi\in F} \{\boldsymbol{r}_\pi + \boldsymbol{P}_\pi \boldsymbol{u}_i\} = \max_{\pi\in E\cap F} \{\boldsymbol{r}_\pi + \boldsymbol{P}_\pi \boldsymbol{u}_i\}. \tag{42}$$

Intuitively, (42) shows that extended value iteration eventually chooses only policies from $E \cap F$.

With (42) accomplished, the proof of Theorem 9.4.4, the main result on convergence of value iteration, can be rewritten word by word from Puterman (1994), with E replaced with $E \cap F$ and using (42) instead of Lemma 9.4.3. Thus, unlike in the original proof where the optimal policy π^* identified at the end of the proof is in E, in our case π^* is in $E \cap F$. Here Puterman (1994) uses the assumption that *all* optimal policies have aperiodic transition matrices to guarantee that π^* has aperiodic transition matrix. In our case, π^* has aperiodic transition matrix by definition, as it is in $E \cap F$.

Then by the aperiodicity of $\boldsymbol{P}_{\pi^*}$, the result of Theorem 9.4.4 follows, and one obtains analogously to Theorem 9.4.5 (a) of Puterman (1994) that

$$\lim_{i\to\infty} (\boldsymbol{u}_{i+1} - \boldsymbol{u}_i) = \boldsymbol{\rho}^*. \tag{43}$$

As the underlying MDP $\tilde{M}^+$ is assumed to be communicating (so that ρ^* is state-independent), analogously to Corollary 9.4.6 of Puterman (1994) convergence of extended value iteration follows from (43). Finally, with the convergence of extended value iteration established, the error bound for the greedy policy follows from Theorem 8.5.6 of Puterman (1994). ■

Appendix C. Technical Details for the Proof of Theorem 2

This appendix collects some technical details, starting with an error bound for our confidence intervals.

C.1 Confidence Intervals

Lemma 17 *For any $t \geq 1$, the probability that the true MDP M is not contained in the set of plausible MDPs $\mathcal{M}(t)$ at time t (as given by the confidence intervals in* (3) *and* (4)*) is at most $\frac{\delta}{15t^6}$, that is*

$$\mathbb{P}\left\{M \notin \mathcal{M}(t)\right\} \;<\; \frac{\delta}{15t^6}.$$

Proof Consider a fixed state-action pair (s,a) and assume some given number of visits $n>0$ in (s,a) before step t. Denote the estimates for transition probabilities and rewards obtained from these n observations by $\hat{p}\,(\cdot|s,a)$ and $\hat{r}\,(s,a)$, respectively. Let us first consider the probability with which a confidence interval for the transition probabilities fails. The random event observed for the transition probability estimates is the state to which the transition occurs. Generally, the L^1-deviation of the true distribution and the empirical distribution over m distinct events from n samples is bounded according to Weissman et al. (2003) by

$$\mathbb{P}\Big\{\left\|\hat{p}\,(\cdot)-p\,(\cdot)\right\|_1 \geq \varepsilon\Big\} \leq (2^m-2)\exp\left(-\tfrac{n\varepsilon^2}{2}\right). \tag{44}$$

Thus, in our case we have $m=S$ (for each possible transition there is a respective event), so that setting

$$\varepsilon = \sqrt{\frac{2}{n}\log\left(\tfrac{2^S 20SAt^7}{\delta}\right)} \leq \sqrt{\frac{14S}{n}\log\left(\tfrac{2At}{\delta}\right)},$$

we get from (44)

$$\begin{aligned}\mathbb{P}\left\{\left\|p\,(\cdot|s,a)-\hat{p}\,(\cdot|s,a)\right\|_1 \geq \sqrt{\frac{14S}{n}\log\left(\tfrac{2At}{\delta}\right)}\right\} \;&\leq\; 2^S\exp\left(-\frac{n}{2}\cdot\frac{2}{n}\log\left(\tfrac{2^S 20SAt^7}{\delta}\right)\right)\\ &=\; \frac{\delta}{20t^7SA}.\end{aligned}$$

For the rewards we observe real-valued, independent identically distributed (i.i.d.) random variables with support in $[0,1]$. Hoeffding's inequality gives for the deviation between the true mean $\bar{r}$ and the empirical mean $\hat{r}$ from n i.i.d. samples with support in $[0,1]$

$$\mathbb{P}\Big\{\left|\hat{r}-\bar{r}\right| \geq \varepsilon_r\Big\} \leq 2\exp\Big(-2n\varepsilon_r^2\Big).$$

Setting

$$\varepsilon_r = \sqrt{\frac{1}{2n}\log\left(\tfrac{120SAt^7}{\delta}\right)} \leq \sqrt{\frac{7}{2n}\log\left(\tfrac{2SAt}{\delta}\right)},$$

we get for state-action pair (s,a)

$$\begin{aligned}\mathbb{P}\left\{\left|\hat{r}\,(s,a)-\bar{r}(s,a)\right| \geq \sqrt{\frac{7}{2n}\log\left(\tfrac{2SAt}{\delta}\right)}\right\} \;&\leq\; 2\exp\left(-2n\cdot\frac{1}{2n}\log\left(\tfrac{120SAt^7}{\delta}\right)\right)\\ &=\; \frac{\delta}{60t^7SA}.\end{aligned}$$

Note that when there haven't been any observations, the confidence intervals trivially hold with probability 1 (for transition probabilities as well as for rewards). Hence a union bound over all possible values of $n = 1, \ldots, t-1$ gives (now writing $N(s,a)$ for the number of visits in (s,a))

$$\mathbb{P}\left\{ \left|\hat{r}(s,a) - \bar{r}(s,a)\right| \geq \sqrt{\frac{7\log\left(\frac{2SAt}{\delta}\right)}{2\max\{1,N(s,a)\}}} \right\} \;\leq\; \sum_{n=1}^{t-1} \frac{\delta}{60t^7SA} < \frac{\delta}{60t^6SA} \quad \text{and}$$

$$\mathbb{P}\left\{ \left\| p(\cdot|s,a) - \hat{p}(\cdot|s,a) \right\|_1 \geq \sqrt{\frac{14S\log\left(\frac{2At}{\delta}\right)}{\max\{1,N(s,a)\}}} \right\} \;\leq\; \sum_{n=1}^{t-1} \frac{\delta}{20t^7SA} < \frac{\delta}{20t^6SA}.$$

Summing these error probabilities over all state-action pairs we obtain the claimed bound $\mathbb{P}\{M \notin \mathcal{M}(t)\} < \frac{\delta}{15t^6}$. ■

C.2 A Bound on the Number of Episodes

Since in each episode the total number of visits to at least one state-action pair doubles, the number of episodes m is logarithmic in T. Actually, the number of episodes becomes maximal when all state-action pairs are visited equally often, which results in the following bound.

Proposition 18 *The number m of episodes of* UCRL2 *up to step $T \geq SA$ is upper bounded as*

$$m \;\leq\; SA\log_2\left(\tfrac{8T}{SA}\right).$$

Proof Let $N(s,a) := \#\{\tau < T+1 : s_\tau = s, a_\tau = a\}$ be the total number of observations of the state-action pair (s,a) up to step T. In each episode $k < m$ there is a state-action pair (s,a) with $v_k(s,a) = N_k(s,a)$ (or $v_k(s,a) = 1, N_k(s,a) = 0$). Let $K(s,a)$ be the number of episodes with $v_k(s,a) = N_k(s,a)$ and $N_k(s,a) > 0$. If $N(s,a) > 0$, then $v_k(s,a) = N_k(s,a)$ implies $N_{k+1}(s,a) = 2N_k(s,a)$, so that

$$N(s,a) \;=\; \sum_{k=1}^{m} v_k(s,a) \;\geq\; 1 + \sum_{k:v_k(s,a)=N_k(s,a)} N_k(s,a) \;\geq\; 1 + \sum_{i=1}^{K(s,a)} 2^{i-1} \;=\; 2^{K(s,a)}.$$

On the other hand, if $N(s,a) = 0$, then obviously $K(s,a) = 0$, so that generally, $N(s,a) \geq 2^{K(s,a)} - 1$ for any state-action pair (s,a). It follows that

$$T \;=\; \sum_{s,a} N(s,a) \;\geq\; \sum_{s,a} \left(2^{K(s,a)} - 1\right). \tag{45}$$

Now, in each episode a state-action pair (s,a) is visited for which either $N_k(s,a) = 0$ or $N_k(s,a) = v_k(s,a)$. Hence, $m \leq 1 + SA + \sum_{s,a} K(s,a)$, or equivalently $\sum_{s,a} K(s,a) \geq m - 1 - SA$. This implies

$$\sum_{s,a} 2^{K(s,a)} \;\geq\; SA\, 2^{\sum_{s,a} K(s,a)/SA} \;\geq\; SA\, 2^{\frac{m-1}{SA}-1}.$$

Together with (45) this gives

$$T \;\geq\; SA\left(2^{\frac{m-1}{SA}-1} - 1\right),$$

which yields

$$m \;\leq\; 1 + 2SA + SA\log_2\left(\tfrac{T}{SA}\right),$$

and the claimed bound on m follows for $T \geq SA$. ■

C.3 The Sum in (19)

Lemma 19 *For any sequence of numbers $z_1,\dots,z_n$ with $0 \le z_k \le Z_{k-1} := \max\left\{1, \sum_{i=1}^{k-1} z_i\right\}$*

$$\sum_{k=1}^{n} \frac{z_k}{\sqrt{Z_{k-1}}} \le \left(\sqrt{2}+1\right)\sqrt{Z_n}\,.$$

Proof We prove the statement by induction over n.
Base case: We first show that the lemma holds for all n with $\sum_{k=1}^{n-1} z_k \le 1$. Indeed, in this case $Z_k = 1$ for $k \le n-1$ and hence $z_n \le 1$. It follows that

$$\sum_{k=1}^{n} \frac{z_k}{\sqrt{Z_{k-1}}} = \sum_{k=1}^{n-1} z_k + z_n \le 1+1 < \left(\sqrt{2}+1\right) Z_n.$$

Note that this also shows that the lemma holds for $n=1$, since $\sum_{k=1}^{0} z_k = 0 \le 1$.
Inductive step: Now let us consider natural numbers n such that $\sum_{k=1}^{n-1} z_k > 1$. By the induction hypothesis we have

$$\sum_{k=1}^{n} \frac{z_k}{\sqrt{Z_{k-1}}} \le \left(\sqrt{2}+1\right)\sqrt{Z_{n-1}} + \frac{z_n}{\sqrt{Z_{n-1}}}\,.$$

Since $z_n \le Z_{n-1} = \sum_{k=1}^{n-1} z_k$ and $Z_{n-1} + z_n = Z_n$, we further have

$$\begin{aligned}
\left(\sqrt{2}+1\right)\sqrt{Z_{n-1}} + \frac{z_n}{\sqrt{Z_{n-1}}} &= \sqrt{\left(\sqrt{2}+1\right)^2 Z_{n-1} + 2\left(\sqrt{2}+1\right) z_n + \frac{z_n^2}{Z_{n-1}}} \\
&\le \sqrt{\left(\sqrt{2}+1\right)^2 Z_{n-1} + \left(2+2\sqrt{2}+1\right) z_n} \\
&= \sqrt{\left(\sqrt{2}+1\right)^2 Z_{n-1} + \left(\sqrt{2}+1\right)^2 z_n} \\
&= \left(\sqrt{2}+1\right)\sqrt{Z_{n-1}+z_n} = \left(\sqrt{2}+1\right)\sqrt{Z_n}\,,
\end{aligned}$$

which proves the lemma. ■

C.4 Simplifying (22)

Combining similar terms, (22) yields that with probability at least $1 - \frac{\delta}{4T^{5/4}}$

$$\begin{aligned}
\Delta(s_1,T) \;\le\; & DS\sqrt{AT}\left(\tfrac{3}{2}\sqrt{\tfrac{1}{A}\cdot\tfrac{5}{2}\log\left(\tfrac{8T}{\delta}\right)} + 2\left(\sqrt{2}+1\right)\sqrt{14\log\left(\tfrac{2AT}{\delta}\right)} + \sqrt{8} + 2 + \tfrac{1}{\sqrt{A}}\right) \\
& + DSA\log_2\left(\tfrac{8T}{SA}\right). \qquad (46)
\end{aligned}$$

We assume $A \ge 2$, since the bound is trivial otherwise. Also, for $1 < T \le 34^2 A\log\left(\frac{T}{\delta}\right)$ we have $\Delta(s_1,T) \le 34\sqrt{AT\log\left(\frac{T}{\delta}\right)}$ trivially. Considering $T > 34A\log\left(\frac{T}{\delta}\right)$ we have $A < \frac{1}{34\log\left(\frac{T}{\delta}\right)}\sqrt{AT\log\left(\frac{T}{\delta}\right)}$ and also $\log_2(8T) < 2\log(T)$, so that

$$DSA\log_2\left(\tfrac{8T}{SA}\right) \;<\; \tfrac{2}{34}DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}\,.$$

Further, $T > 34A\log\left(\frac{T}{\delta}\right)$ also implies $\log\left(\frac{2AT}{\delta}\right) \le 2\log\left(\frac{T}{\delta}\right)$ and $\log\left(\frac{8T}{\delta}\right) \le 2\log\left(\frac{T}{\delta}\right)$. Thus, we have by (46) that for any $T > 1$ with probability at least $1 - \frac{\delta}{4T^{5/4}}$

$$\begin{aligned}\Delta(s_1,T) &\le DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}\left(\tfrac{3}{2}\sqrt{\tfrac{5}{2}} + 2\left(\sqrt{2}+1\right)\sqrt{28} + \sqrt{8} + 2 + \tfrac{1}{\sqrt{2}} + \tfrac{2}{34}\right)\\ &\le 34DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}.\end{aligned}$$

Appendix D. Technical Details for the Proof of Theorem 4: Proof of (27)

For a given index set K_ε of episodes we would like to bound the sum

$$\sum_{k\in K_\varepsilon}\sum_{s,a}\frac{v_k(s,a)}{\sqrt{\max\{1,N_k(s,a)\}}} = \sum_{s,a}\sum_{k=1}^{m}\frac{v_k(s,a)}{\sqrt{\max\{1,N_k(s,a)\}}}\mathbb{1}_{k\in K_\varepsilon}.$$

We will do this by modifying the sum so that Lemma 19 becomes applicable. Compared to the setting of Lemma 19 there are some "gaps" in the sum caused by episodes $\notin K_\varepsilon$. In the following we show that the contribution of episodes that occur after step $L_\varepsilon := \sum_{k\in K_\varepsilon}\sum_{s,a} v_k(s,a)$ is not larger than the missing contributions of the episodes $\notin K_\varepsilon$. Intuitively speaking, one may fill the episodes that occur after step L_ε into the gaps of episodes $\notin K_\varepsilon$ as Figure 6 suggests.

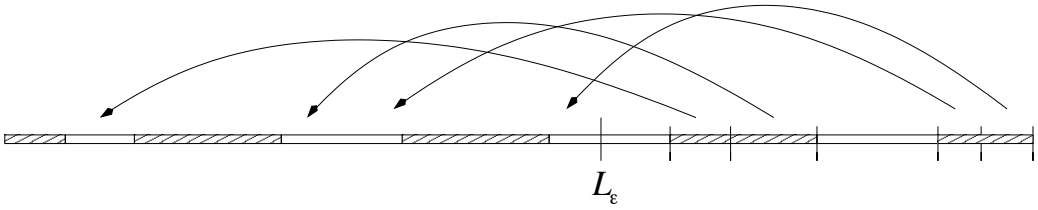


Figure 6: Illustration of the proof idea. Shaded boxes stand for episodes $\in K_\varepsilon$, empty boxes for episodes $\notin K_\varepsilon$. The contribution of episodes after step L_ε can be "filled into the gaps" of episodes $\notin K_\varepsilon$ before step L_ε.

Let $\ell_\varepsilon(s,a) := \sum_{k\in K_\varepsilon} v_k(s,a)$, so that $\sum_{s,a}\ell_\varepsilon(s,a) = L_\varepsilon$. We consider a fixed state-action pair (s,a) and skip the reference to it for ease of reading, so that N_k refers to the number of visits to (s,a) up to episode k, and N denotes the total number of visits to (s,a). Further, we abbreviate $d_k := \sqrt{\max\{1,N_k\}}$, and let $m_\varepsilon := \max\{k : N_k < \ell_\varepsilon\}$ be the episode containing the ℓ_ε-th visit to (s,a). Due to $v_k = N_{k+1} - N_k$ we have

$$v_{m_\varepsilon} = (N_{m_\varepsilon+1} - \ell_\varepsilon) + (\ell_\varepsilon - N_{m_\varepsilon}). \tag{47}$$

Since $N_{m_\varepsilon} = \sum_{k=1}^{m_\varepsilon-1} v_k$, this yields

$$\begin{aligned}\ell_\varepsilon - N_{m_\varepsilon} + \sum_{k=1}^{m_\varepsilon-1} v_k &= \ell_\varepsilon = \sum_{k=1}^{m} v_k\mathbb{1}_{k\in K_\varepsilon}\\ &= \sum_{k=1}^{m_\varepsilon-1} v_k\mathbb{1}_{k\in K_\varepsilon} + \left(N_{m_\varepsilon+1} - \ell_\varepsilon\right)\mathbb{1}_{m_\varepsilon\in K_\varepsilon} + \left(\ell_\varepsilon - N_{m_\varepsilon}\right)\mathbb{1}_{m_\varepsilon\in K_\varepsilon} + \sum_{k=m_\varepsilon+1}^{m} v_k\mathbb{1}_{k\in K_\varepsilon},\end{aligned}$$

or equivalently,

$$\left(N_{m_\varepsilon+1}-\ell_\varepsilon\right)\mathbb{1}_{m_\varepsilon\in K_\varepsilon}+\sum_{k=m_\varepsilon+1}^{m} v_k\mathbb{1}_{k\in K_\varepsilon} \;=\; \left(\ell_\varepsilon-N_{m_\varepsilon}\right)\mathbb{1}_{m_\varepsilon\notin K_\varepsilon}+\sum_{k=1}^{m_\varepsilon-1} v_k\mathbb{1}_{k\notin K_\varepsilon}. \tag{48}$$

By (47) and due to $d_k \geq d_{m_\varepsilon}$ for $k \geq m_\varepsilon$ we have

$$\begin{aligned}\sum_{k=1}^{m}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon} \;\leq\; & \sum_{k=1}^{m_\varepsilon-1}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon}+\frac{\ell_\varepsilon-N_{m_\varepsilon}}{d_{m_\varepsilon}}\mathbb{1}_{m_\varepsilon\in K_\varepsilon}\\ & +\frac{1}{d_{m_\varepsilon}}\left(\left(N_{m_\varepsilon+1}-\ell_\varepsilon\right)\mathbb{1}_{m_\varepsilon\in K_\varepsilon}+\sum_{k=m_\varepsilon+1}^{m} v_k\mathbb{1}_{k\in K_\varepsilon}\right).\end{aligned}$$

Hence, we get together with (48), using that $d_k \leq d_{m_\varepsilon}$ for $k \leq m_\varepsilon$

$$\begin{aligned}\sum_{k=1}^{m}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon} \;\leq\; & \sum_{k=1}^{m_\varepsilon-1}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon}+\frac{\ell_\varepsilon-N_{m_\varepsilon}}{d_{m_\varepsilon}}\mathbb{1}_{m_\varepsilon\in K_\varepsilon}\\ & +\frac{1}{d_{m_\varepsilon}}\left(\left(\ell_\varepsilon-N_{m_\varepsilon}\right)\mathbb{1}_{m_\varepsilon\notin K_\varepsilon}+\sum_{k=1}^{m_\varepsilon-1} v_k\mathbb{1}_{k\notin K_\varepsilon}\right)\\ \leq\; & \sum_{k=1}^{m_\varepsilon-1}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon}+\frac{\ell_\varepsilon-N_{m_\varepsilon}}{d_{m_\varepsilon}}\mathbb{1}_{m_\varepsilon\in K_\varepsilon}+\frac{\ell_\varepsilon-N_{m_\varepsilon}}{d_{m_\varepsilon}}\mathbb{1}_{m_\varepsilon\notin K_\varepsilon}+\sum_{k=1}^{m_\varepsilon-1}\frac{v_k}{d_k}\mathbb{1}_{k\notin K_\varepsilon}\\ =\; & \sum_{k=1}^{m_\varepsilon-1}\frac{v_k}{d_k}+\frac{\ell_\varepsilon-N_{m_\varepsilon}}{d_{m_\varepsilon}}.\end{aligned}$$

Now define v'_k as follows: let $v'_k := v_k$ for $k < m_\varepsilon$ and $v'_{m_\varepsilon} := \ell_\varepsilon - N_{m_\varepsilon}$. Then we have just seen that

$$\sum_{k=1}^{m}\frac{v_k}{d_k}\mathbb{1}_{k\in K_\varepsilon} \;\leq\; \sum_{k=1}^{m_\varepsilon}\frac{v'_k}{d_k}.$$

Since further $\sum_{k=1}^{m_\varepsilon} v'_k = \ell_\varepsilon$ we get by Lemma 19 that

$$\sum_{k=1}^{m_\varepsilon}\frac{v'_k}{d_k} \;\leq\; \left(\sqrt{2}+1\right)\sqrt{\ell_\varepsilon}\,.$$

As $\sum_{s,a}\ell_\varepsilon(s,a) = L_\varepsilon$, we finally obtain by Jensen's inequality

$$\sum_{k\in K_\varepsilon}\sum_{s,a}\frac{v_k(s,a)}{\sqrt{\max\{1,N_k(s,a)\}}} \;\leq\; \left(\sqrt{2}+1\right)\sqrt{L_\varepsilon SA}\,,$$

as claimed. ■

Appendix E. Proof of Lemma 13

Let us first recall some notation from Section 6. Thus $\mathbb{P}_a[\cdot]$ denotes the probability conditioned on a being the "good" action, while the probability with respect to a setting where all actions in state $s_\circ$ are equivalent (i.e., $\varepsilon = 0$) is denoted by $\mathbb{P}_{\text{unif}}[\cdot]$. Let $\mathcal{S} := \{s_\circ, s_1\}$ and denote the state

observed at step τ by s_τ and the state-sequence up to step τ by $s^\tau = s_1, \ldots, s_\tau$. Basically, the proof follows along the lines of the proof of Lemma A.1 of Auer et al. (2002b). The first difference is that our observations now consist of the sequence of $T+1$ states instead of a sequence of T observed rewards. Still it is straightforward to get analogously to the proof of Auer et al. (2002b), borrowing the notation, that for any function f from $\{s_\circ, s_1\}^{T+1}$ to $[0,B]$,

$$\mathbb{E}_a[f(s)] - \mathbb{E}_{\text{unif}}[f(s)] \;\leq\; \frac{B}{2}\sqrt{2\log(2)\,\mathrm{KL}\left(\mathbb{P}_{\text{unif}} \middle\| \mathbb{P}_a\right)}, \tag{49}$$

where $\mathrm{KL}(P\|Q)$ denotes for two distributions P, Q the *Kullback-Leibler divergence* defined as $\mathrm{KL}(P\|Q) := \sum_{s\in\mathcal{S}^{T+1}} P\{s\}\log_2\left(\frac{P\{s\}}{Q\{s\}}\right)$. It holds that (cf. Auer et al., 2002b)

$$\mathrm{KL}\left(\mathbb{P}_{\text{unif}} \middle\| \mathbb{P}_a\right) = \sum_{t=1}^{T} \mathrm{KL}\left(\mathbb{P}_{\text{unif}}\left[s_{t+1}\middle|s^t\right] \middle\| \mathbb{P}_a\left[s_{t+1}\middle|s^t\right]\right), \tag{50}$$

where $\mathrm{KL}(P\{s_{t+1}|s^t\}\|Q\{s_{t+1}|s^t\}) := \sum_{s^{t+1}\in\mathcal{S}^{t+1}} P\{s^{t+1}\}\log_2\left(\frac{P\{s_{t+1}|s^t\}}{Q\{s_{t+1}|s^t\}}\right)$. By the Markov property and the fact that the action a_t is determined by a sequence $s^t \in \mathcal{S}^t$ we have (similar to Auer et al., 2002b)

$$\begin{aligned}
&\mathrm{KL}\left(\mathbb{P}_{\text{unif}}\left[s_{t+1}\middle|s^t\right] \middle\| \mathbb{P}_a\left[s_{t+1}\middle|s^t\right]\right) = \sum_{s^{t+1}\in\mathcal{S}^{t+1}} \mathbb{P}_{\text{unif}}\left[s^{t+1}\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s_{t+1}|s^t\right]}{\mathbb{P}_a\left[s_{t+1}|s^t\right]}\right)\\
&= \sum_{s^t\in\mathcal{S}^t} \mathbb{P}_{\text{unif}}\left[s^t\right] \sum_{s'\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s_{t+1}=s'|s^t\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s'|s^t\right]}{\mathbb{P}_a\left[s'|s^t\right]}\right)\\
&= \sum_{s^{t-1}\in\mathcal{S}^{t-1}} \mathbb{P}_{\text{unif}}\left[s^{t-1}\right] \sum_{(s'',a')\in\mathcal{S}\times\mathcal{A}} \mathbb{P}_{\text{unif}}\left[s_t=s'', a_t=a'|s^{t-1}\right]\\
&\qquad \cdot \sum_{s'\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s_{t+1}=s'|s^{t-1}, s_t=s'', a_t=a'\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s'|s^{t-1}, s_t=s'', a_t=a'\right]}{\mathbb{P}_a\left[s'|s^{t-1}, s_t=s'', a_t=a'\right]}\right)\\
&= \sum_{s^{t-1}\in\mathcal{S}^{t-1}} \mathbb{P}_{\text{unif}}\left[s^{t-1}\right] \sum_{a'=1}^{kA'} \sum_{s''\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s_t=s'', a_t=a'|s^{t-1}\right]\\
&\qquad\qquad \cdot \sum_{s'\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s'|s'', a'\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s'|s'', a'\right]}{\mathbb{P}_a\left[s'|s'', a'\right]}\right).
\end{aligned}$$

Since $\log_2\left(\frac{\mathbb{P}_{\text{unif}}[s'|s'',a']}{\mathbb{P}_a[s'|s'',a']}\right) \neq 0$ only for $s'' = s_\circ$ and a' being the special action a, we get

$$\begin{aligned}
&\mathrm{KL}\left(\mathbb{P}_{\text{unif}}\left[s_{t+1}\middle|s^t\right] \middle\| \mathbb{P}_a\left[s_{t+1}\middle|s^t\right]\right) =\\
&= \sum_{s^{t-1}\in\mathcal{S}^{t-1}} \mathbb{P}_{\text{unif}}\left[s^{t-1}\right]\mathbb{P}_{\text{unif}}\left[s_t=s_\circ, a_t=a|s^{t-1}\right] \cdot \sum_{s'\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s'|s_\circ,a\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s'|s_\circ,a\right]}{\mathbb{P}_a\left[s'|s_\circ,a\right]}\right)\\
&= \mathbb{P}_{\text{unif}}[s_t=s_\circ, a_t=a] \sum_{s'\in\mathcal{S}} \mathbb{P}_{\text{unif}}\left[s'|s_\circ,a\right]\log_2\left(\frac{\mathbb{P}_{\text{unif}}\left[s'|s_\circ,a\right]}{\mathbb{P}_a\left[s'|s_\circ,a\right]}\right)\\
&= \mathbb{P}_{\text{unif}}[s_t=s_\circ, a_t=a]\left(\delta\log_2\left(\frac{\delta}{\delta+\varepsilon}\right) + (1-\delta)\log_2\left(\frac{1-\delta}{1-\delta-\varepsilon}\right)\right).
\end{aligned} \tag{51}$$

To complete the proof we use the following lemma.

Lemma 20 *For any* $0 \le \delta \le \frac{1}{2}$ *and* $\varepsilon \le 1-2\delta$ *we have*

$$\delta \log_2\left(\frac{\delta}{\delta+\varepsilon}\right) + (1-\delta)\log_2\left(\frac{1-\delta}{1-\delta-\varepsilon}\right) \;\le\; \frac{\varepsilon^2}{\delta \log(2)}\,.$$

Indeed, application of Lemma 20 together with (50) and (51) gives that

$$\begin{aligned} \mathrm{KL}\left(\mathbb{P}_{\mathrm{unif}} \middle\| \mathbb{P}_a\right) &= \sum_{t=1}^{T} \mathrm{KL}\left(\mathbb{P}_{\mathrm{unif}}\left[s_{t+1}\middle|\boldsymbol{s}^t\right] \middle\| \mathbb{P}_a\left[s_{t+1}\middle|\boldsymbol{s}^t\right]\right) \\ &\le \sum_{t=1}^{T} \mathbb{P}_{\mathrm{unif}}[s_t = s_\circ, a_t = a]\,\frac{\varepsilon^2}{\delta\log(2)} \;=\; \mathbb{E}_{\mathrm{unif}}[N_\circ^*]\,\frac{\varepsilon^2}{\delta\log(2)}, \end{aligned}$$

which together with (49) yields

$$\mathbb{E}_a[f(\boldsymbol{s})] - \mathbb{E}_{\mathrm{unif}}[f(\boldsymbol{s})] \le \frac{B}{2}\cdot\frac{\varepsilon}{\sqrt{\delta}}\sqrt{2\mathbb{E}_{\mathrm{unif}}[N_\circ^*]},$$

as claimed by Lemma 13.

Proof of Lemma 20 Consider

$$h_\delta(\varepsilon) := \frac{\varepsilon^2}{\delta} - \delta\log\left(\frac{\delta}{\delta+\varepsilon}\right) - (1-\delta)\log\left(\frac{1-\delta}{1-\delta-\varepsilon}\right).$$

We show that $h_\delta(\varepsilon) \ge 0$ for $\delta \le \frac{1}{2}$ and $0 \le \varepsilon \le \varepsilon_0$, where

$$\varepsilon_0 := \frac{1}{2} - \delta + \frac{1}{2}\sqrt{1-2\delta}.$$

Indeed, $h_\delta(0) = 0$ for all δ, while for the first derivative

$$h'_\delta(\varepsilon) := \frac{\partial}{\partial\varepsilon} h_\delta(\varepsilon) \;=\; 2\cdot\frac{\varepsilon}{\delta} + \frac{\delta}{\delta+\varepsilon} - \frac{1-\delta}{1-\delta-\varepsilon}$$

we have $h'_\delta(\varepsilon) \ge 0$ for $\delta \le \frac{1}{2}$ and $0 \le \varepsilon \le \varepsilon_0$. It remains to show that $\delta \le \frac{1}{2}$ and $\varepsilon \le 1-2\delta$ imply $\varepsilon \le \varepsilon_0$. Indeed, for $\delta \le \frac{1}{2}$ and $\varepsilon \le 1-2\delta$ we have

$$\varepsilon - \varepsilon_0 \;\le\; 1-2\delta-\varepsilon_0 = \frac{1}{2} - \delta - \frac{1}{2}\sqrt{1-2\delta} \;=\; \frac{1}{2}\left((1-2\delta) - \sqrt{1-2\delta}\right) \;\le\; 0.$$

■

References

Peter Auer and Ronald Ortner. Logarithmic online regret bounds for reinforcement learning. In *Advances in Neural Information Processing Systems 19 (NIPS 2006)*, pages 49–56. MIT Press, 2007.

Peter Auer and Ronald Ortner. Online regret bounds for a new reinforcement learning algorithm. In *Proceedings 1st Austrian Cognitive Vision Workshop (ACVW 2005)*, pages 35–42. ÖCG, 2005.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multi-armed bandit problem. *Mach. Learn.*, 47:235–256, 2002a.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multi-armed bandit problem. *SIAM J. Comput.*, 32:48–77, 2002b.

Peter L. Bartlett and Ambuj Tewari. REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2009)*, 2009.

Ronen I. Brafman and Moshe Tennenholtz. R-max – a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.*, 3:213–231, 2002.

Apostolos N. Burnetas and Michael N. Katehakis. Optimal adaptive policies for Markov decision processes. *Math. Oper. Res.*, 22(1):222–255, 1997.

Eyal Even-Dar, Sham M. Kakade, and Yishay Mansour. Experts in a Markov decision process. In *Advances in Neural Information Processing Systems 17 (NIPS 2004)*, pages 401–408. MIT Press, 2005.

Eyal Even-Dar, Sham M. Kakade, and Yishay Mansour. Online Markov decision processes. *Math. Oper. Res.*, 34(3):726–736, 2009.

Claude-Nicolas Fiechter. Efficient reinforcement learning. In *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory (COLT 1994)*, pages 88–97. ACM, 1994.

David A. Freedman. On tail probabilities for martingales. *Ann. Probab.*, 3:100–118, 1975.

Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems. Preprint, 2008. URL `http://arxiv.org/pdf/0805.3415`.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.*, 58:13–30, 1963.

Sham M. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.

Michael J. Kearns and Satinder P. Singh. Near-optimal reinforcement learning in polynomial time. *Mach. Learn.*, 49:209–232, 2002.

Michael J. Kearns and Satinder P. Singh. Finite-sample convergence rates for Q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems 11 (NIPS 1998)*, pages 996–1002. MIT Press, 1999.

Shie Mannor and John N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *J. Mach. Learn. Res.*, 5:623–648, 2004.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1994.

Alexander L. Strehl and Michael L. Littman. A theoretical analysis of model-based interval estimation. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005)*, pages 857–864. ACM, 2005.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for Markov decision processes. *J. Comput. System Sci.*, 74(8):1309–1331, 2008.

Alexander L. Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L. Littman. PAC model-free reinforcement learning. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006)*, pages 881–888. ACM, 2006.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Ambuj Tewari and Peter Bartlett. Optimistic linear programming gives logarithmic regret for irreducible MDPs. In *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, pages 1505–1512. MIT Press, 2008.

Ambuj Tewari and Peter L. Bartlett. Bounded parameter Markov decision processes with average reward criterion. In *Learning Theory, 20th Annual Conference on Learning Theory (COLT 2007)*, pages 263–277, 2007.

Tsachy Weissman, Erik Ordentlich, Gadiel Seroussi, Sergio Verdu, and Marco L. Weinberger. Inequalities for the L1 deviation of the empirical distribution. Technical Report HPL-2003-97, HP Laboratories Palo Alto, 2003. URL `www.hpl.hp.com/techreports/2003/HPL-2003-97R1.pdf`.

Jia Yuan Yu and Shie Mannor. Piecewise-stationary bandit problems with side observations. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009)*, pages 1177–1184, 2009.

Jia Yuan Yu, Shie Mannor, and Nahum Shimkin. Markov decision processes with arbitrary reward processes. *Math. Oper. Res.*, 34(3):737–757, 2009.

Journal of Machine Learning Research 11 (2010) 1601-1604 Submitted 11/09; Published 4/10

MOA: Massive Online Analysis

Albert Bifet ABIFET@CS.WAIKATO.AC.NZ
Geoff Holmes GEOFF@CS.WAIKATO.AC.NZ
Richard Kirkby RKIRKBY@CS.WAIKATO.AC.NZ
Bernhard Pfahringer BERNHARD@CS.WAIKATO.AC.NZ
Department of Computer Science
University of Waikato
Hamilton, New Zealand

Editor: Mikio Braun

Abstract

Massive **O**nline **A**nalysis (MOA) is a software environment for implementing algorithms and running experiments for online learning from evolving data streams. MOA includes a collection of offline and online methods as well as tools for evaluation. In particular, it implements boosting, bagging, and Hoeffding Trees, all with and without Naïve Bayes classifiers at the leaves. MOA supports bi-directional interaction with WEKA, the Waikato Environment for Knowledge Analysis, and is released under the GNU GPL license.

Keywords: data streams, classification, ensemble methods, java, machine learning software

1. Introduction

Green computing is the study and practice of using computing resources efficiently. A main approach to green computing is based on algorithmic efficiency. In the data stream model, data arrive at high speed, and an algorithm must process them under very strict constraints of space and time.

MOA is an open-source framework for dealing with massive evolving data streams. MOA is related to WEKA, the Waikato Environment for Knowledge Analysis, which is an award-winning open-source workbench containing implementations of a wide range of batch machine learning methods.

A data stream environment has different requirements from the traditional batch learning setting. The most significant are the following:

Requirement 1 Process an example at a time, and inspect it only once (at most)

Requirement 2 Use a limited amount of memory

Requirement 3 Work in a limited amount of time

Requirement 4 Be ready to predict at any time

Figure 1 illustrates the typical use of a data stream classification algorithm, and how the requirements fit in a repeating cycle:

1. The algorithm is passed the next available example from the stream (Requirement 1).

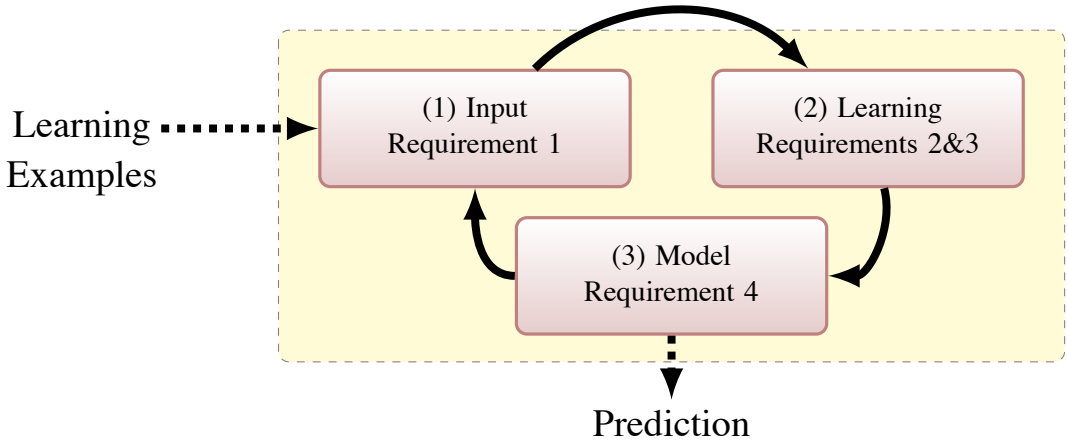


Figure 1: The data stream classification cycle

2. The algorithm processes the example, updating its data structures. It does so without exceeding the memory bounds set on it (requirement 2), and as quickly as possible (Requirement 3).

3. The algorithm is ready to accept the next example. On request it is able to predict the class of unseen examples (Requirement 4).

In traditional batch learning the problem of limited data is overcome by analyzing and averaging multiple models produced with different random arrangements of training and test data. In the stream setting the problem of (effectively) unlimited data poses different challenges. One solution involves taking snapshots at different times during the induction of a model to see how much the model improves.

The evaluation procedure of a learning algorithm determines which examples are used for training the algorithm, and which are used to test the model output by the algorithm.When considering what procedure to use in the data stream setting, one of the unique concerns is how to build a picture of accuracy over time. Two main approaches arise:

- **Holdout**: When traditional batch learning reaches a scale where cross-validation is too time consuming, it is often accepted to instead measure performance on a single holdout set. This is most useful when the division between train and test sets has been pre-defined, so that results from different studies can be directly compared.

- **Interleaved Test-Then-Train or Prequential**: Each individual example can be used to test the model before it is used for training, and from this the accuracy can be incrementally updated. When intentionally performed in this order, the model is always being tested on examples it has not seen. This scheme has the advantage that no holdout set is needed for testing, making maximum use of the available data. It also ensures a smooth plot of accuracy over time, as each individual example will become increasingly less significant to the overall average (Gama et al., 2009).

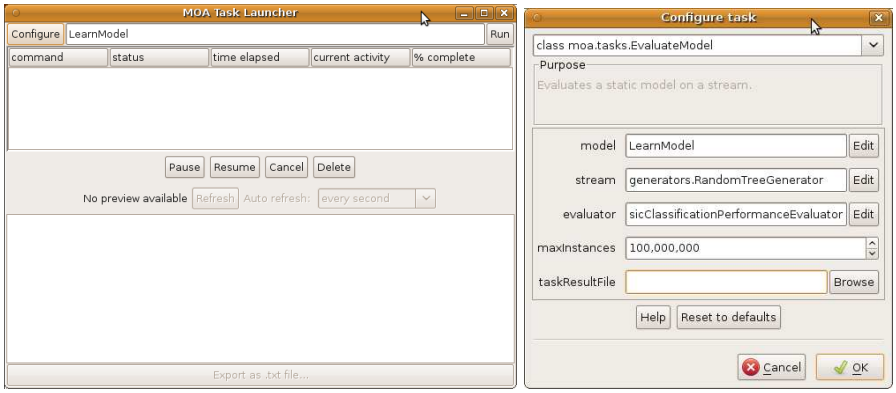


Figure 2: MOA Graphical User Interface

As data stream classification is a relatively new field, such evaluation practices are not nearly as well researched and established as they are in the traditional batch setting. The majority of experimental evaluations use less than one million training examples. In the context of data streams this is disappointing, because to be truly useful at data stream classification the algorithms need to be capable of handling very large (potentially infinite) streams of examples. Demonstrating systems only on small amounts of data does not build a convincing case for capacity to solve more demanding stream applications (Kirkby, 2007).

MOA permits evaluation of data stream classification algorithms on large streams, in the order of tens of millions of examples where possible, and under explicit memory limits. Any less than this does not actually test algorithms in a realistically challenging setting.

2. Experimental Framework

MOA is written in Java. The main benefits of Java are portability, where applications can be run on any platform with an appropriate Java virtual machine, and the strong and well-developed support libraries. Use of the language is widespread, and features such as automatic garbage collection help to reduce programmer burden and error.

MOA contains stream generators, classifiers and evaluation methods. Figure 2 shows the MOA graphical user interface. However, a command line interface is also available.

Considering data streams as data generated from pure distributions, MOA models a concept drift event as a weighted combination of two pure distributions that characterizes the target concepts before and after the drift. Within the framework, it is possible to define the probability that instances of the stream belong to the new concept after the drift. It uses the sigmoid function, as an elegant and practical solution (Bifet et al., 2009a,b).

MOA contains the data generators most commonly found in the literature. MOA streams can be built using generators, reading ARFF files, joining several streams, or filtering streams. They allow for the simulation of a potentially infinite sequence of data. The following generators are currently available: Random Tree Generator, SEA Concepts Generator, STAGGER Concepts Generator, Rotating Hyperplane, Random RBF Generator, LED Generator, Waveform Generator, and Function Generator.

MOA contains several classifier methods such as: Naive Bayes, Decision Stump, Hoeffding Tree, Hoeffding Option Tree (Pfahringer et al., 2008), Bagging, Boosting, Bagging using ADWIN, and Bagging using Adaptive-Size Hoeffding Trees (Bifet et al., 2009b).

2.1 Website, Tutorials, and Documentation

MOA can be found at: http://moa.cs.waikato.ac.nz/.

The website includes a tutorial, an API reference, a user manual, and a manual about mining data streams. Several examples of how the software can be used are available. For example, a non-trivial example of the EvaluateInterleavedTestThenTrain task creating a comma separated values file, training the HoeffdingTree classifier on the WaveformGenerator data, training and testing on a total of 100 million examples, and testing every one million examples, is encapsulated by the following commandline:

```
java -cp .:moa.jar:weka.jar -javaagent:sizeofag.jar moa.DoTask \
  "EvaluateInterleavedTestThenTrain -l HoeffdingTree \
  -s generators.WaveformGenerator \
  -i 100000000 -f 1000000" > htresult.csv
```

MOA is easy to use and extend. A simple approach to writing a new classifier is to extend moa.classifiers.AbstractClassifier, which will take care of certain details to ease the task. Although the current focus in MOA is on classification, we plan to extend the framework to include data stream clustering, regression, and frequent pattern learning (Bifet, 2010).

References

Albert Bifet. *Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams*. IOS Press, 2010.

Albert Bifet, Geoff Holmes, Bernhard Pfahringer, and Ricard Gavaldà. Improving adaptive bagging methods for evolving data streams. In *First Asian Conference on Machine Learning, ACML 2009*, 2009a.

Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Richard Kirkby, and Ricard Gavaldà. New ensemble methods for evolving data streams. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009b.

João Gama, Raquel Sebastião, and Pedro Pereira Rodrigues. Issues in evaluation of stream learning algorithms. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

Richard Kirkby. *Improving Hoeffding Trees*. PhD thesis, University of Waikato, November 2007.

Bernhard Pfahringer, Geoff Holmes, and Richard Kirkby. Handling numeric attributes in hoeffding trees. In *PAKDD Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 296–307, 2008.

Journal of Machine Learning Research 11 (2010) 1605-1641 Submitted 10/09; Revised 4/10; Published 5/10

On the Foundations of Noise-free Selective Classification

Ran El-Yaniv RANI@CS.TECHNION.AC.IL
Yair Wiener WYAIR@TX.TECHNION.AC.IL
Computer Science Department
Technion – Israel Institute of Technology
Haifa 32000, Israel

Editor: Gabor Lugosi

Abstract

We consider *selective classification*, a term we adopt here to refer to 'classification with a reject option.' The essence in selective classification is to trade-off classifier coverage for higher accuracy. We term this trade-off the *risk-coverage (RC) trade-off*. Our main objective is to characterize this trade-off and to construct algorithms that can optimally or near optimally achieve the best possible trade-offs in a controlled manner. For noise-free models we present in this paper a thorough analysis of selective classification including characterizations of RC trade-offs in various interesting settings.

Keywords: classification with a reject option, selective classification, perfect learning, high performance classification, risk-coverage trade-off

1. Introduction

In this paper we study the trade-off between coverage and accuracy of classifiers with a reject option, a trade-off we refer to as the *risk-coverage (RC) trade-off*. Our main goal is to characterize this trade-off and to construct algorithms that can optimally or near optimally control it. Throughout the paper we use the term *selective classification* to refer to 'classification with a reject option.' Selective classification was introduced a number of decades ago and among the earliest studies are papers authored by Chow (1957, 1970), focusing on Bayesian solutions for the case where the underlying distributions are fully known. Through the years, selective classification continued to draw attention and numerous papers have been published. The attraction of effective selective classification is rather obvious in applications where one is not concerned with, or can afford partial coverage of the domain, and/or in cases where extremely low risk is a must but is not achievable in standard classification frameworks. Classification problems in medical diagnosis and in bioinformatics are often instances of such applications (Meltzer et al., 2001; Hanczar and Dougherty, 2008).

Despite the relatively large number of research publications on selective classification, the vast majority of these works have been concerned with implementing a reject option within specific learning schemes, by endowing a learning scheme (e.g., neural networks, SVMs) with a reject mechanism. Most of the reject mechanisms were based on "ambiguity" or (lack of) "confidence" principles: "when confused or when in doubt, refuse to classify." While there are many convincing accounts for the potential effectiveness of selective classification in reducing the risk, we are not familiar with a thorough or conclusive discussions on the relative power of the numerous rejection mechanisms that have been considered so far. The very few theoretical works that considered se-

lective classification (see Section 10) do provide some risk or coverage bounds for specific schemes (e.g., ensemble methods) or learning principles (e.g., ERMs), but altogether characterizations of achievable (or non-achievable) RC trade-offs are absent in the current literature. In particular, the work done so far has not facilitated formal discussions of RC trade-off *optimality*.

A thorough understanding and effective use of selective classification requires characterization of the theoretical and practical boundaries of RC trade-offs, which are essential elements in any discussion of *optimality* in selective classification. These missing elements in the current literature are critical when constructing and exploring selective classification schemes and selective classification algorithms that aim at achieving optimality in controlling the RC trade-off.

One of our longer term goals is to provide such characterizations and introduce a notion of optimality for selective classification in the most general agnostic model. As a first step, however, we focus in this work on noiseless settings whereby a perfect hypothesis for the problem at hand exists (the so called "realizable case"). Moreover, we place special emphasis on the extreme case where zero risk has to be guaranteed. For this extreme case, which we call "perfect learning," we provide a thorough analysis that includes tight positive and negative results for the most general types of realizable settings (distribution independent, infinite hypothesis spaces). We also discuss some specific settings (linear classifiers, specific distribution families) and show an efficient algorithm for linear classifiers that achieves "perfect learning" with guaranteed coverage. Our results on "perfect learning" are instrumental in exploring entire RC trade-offs. Recalling known results on optimal standard realizable learning (no rejection is allowed), we show how to "interpolate" bounds and strategies for these two extreme cases (perfect learning and standard learning) so as to reveal upper and lower envelopes of optimal RC trade-offs.

2. Selective Classification: Preliminary Definitions

Let $\mathcal{X}$ be some feature space, for example, d-dimensional vectors in $\mathbb{R}^d$. In standard binary classification, the goal is to learn a binary classifier $f : \mathcal{X} \to \{\pm 1\}$, using a finite training sample of m labeled examples, $S_m = \{(x_i, y_i)\}_{i=1}^m$, assumed to be sampled i.i.d. from some *unknown* underlying distribution $P(X,Y)$ over $\mathcal{X} \times \{\pm 1\}$. We assume that the classifier is to be selected from a hypothesis space $\mathcal{F}$ and focus on the *realizable* setting where the labels are determined by some *unknown target hypothesis* $f^* \in \mathcal{F}$. Thus, it is assumed that P satisfies $\mathrm{Pr}_P(Y = f^*(X)|X) = 1$.

In *selective classification* the learner should output a binary *selective classifier* defined to be a pair (f,g), with f being a standard binary classifier, and $g : \mathcal{X} \to [0,1]$ a *selection function* whose meaning is as follows. When applying the selective classifier to a sample x, its output is:

$$(f,g)(x) \triangleq \begin{cases} \mathit{reject}, & \text{w.p. } 1-g(x); \\ f(x), & \text{w.p. } g(x). \end{cases} \tag{1}$$

Thus, in its most general form, the selective classifier is *randomized*. Whenever the selection function is a zero-one rule, $g : \mathcal{X} \to \{0,1\}$, we say that the selective classifier is deterministic. Note that "standard learning" (i.e., no rejection is allowed) is the special case of selective classification where $g(x)$ selects all points (i.e., $g(x) \equiv 1$).

The two main characteristics of a selective classifier are its *coverage* and its *risk* (or "true error").

Definition 1 (coverage) *The* coverage *of a selective classifier (f,g) is the mean value of the selection function $g(X)$ taken over the underlying distribution P,*

$$\Phi(f,g) \triangleq \mathbf{E}\left[g(X)\right].$$

Definition 2 (risk) *For a bounded loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0,1]$, we define the risk of a selective classifier (f,g) as the average loss on the accepted samples,*

$$R(f,g) \triangleq \frac{\mathbf{E}\left[\ell(f(X),Y)\cdot g(X)\right]}{\Phi(f,g)}.$$

This risk definition clearly reduces to the standard definition of risk if $g(x) \equiv 1$. Note that (at the outset) both the coverage and risk are *unknown quantities* because they are defined in terms of the unknown underlying distribution P.

We define a learning algorithm ALG to be a (random) function that, given a sample S_m, chooses a selective classifier (f,g). We evaluate learners with respect to their coverage and risk and derive both positive and negative results on achievable risk and coverage. Our model is a slight extension of the standard minimax model for standard statistical learning as described, for example, by Antos and Lugosi (1998). Thus, we consider the following game between the learner and an adversary. The parameters of the game are a domain $\mathcal{X}$ and an hypothesis class $\mathcal{F}$.

1. A tolerance level δ and a training sample size m are given.
2. The learner chooses a learning algorithm ALG.
3. With full knowledge of the learner's choice, the adversary chooses a distribution $P(X)$ over $\mathcal{X}$, and a target hypothesis $f^* \in \mathcal{F}$ (or a distribution over $\mathcal{F}$ according to which f^* is selected).
4. A training sample S_m is drawn i.i.d. according to P and f^*.
5. ALG is applied on S_m and outputs a selective classifier (f,g).

The result of the game is evaluated in terms of the risk and coverage obtained by the chosen selective classifier and clearly, these are random quantities that trade-off each other. A *positive result* in this model is a pair of bounds, $B_R = B_R(\mathcal{F},\delta,m)$ and $B_\Phi = B_\Phi(\mathcal{F},\delta,m)$, for risk and coverage, respectively, that for any δ and m, hold with high probability, of at least $1-\delta$ for *any* distribution P; namely,

$$\Pr\{R(f,g) \le B_R \ \wedge \ \Phi(f,g) \ge B_\Phi\} \ge 1-\delta.$$

The probability is taken w.r.t. the random choice of training samples S_m, as well as w.r.t. all other random choices introduced, such as a random choice of f^* by the adversary (if applicable), a random choice of (f,g) by ALG (if applicable), and the randomized selection function (Equation (1)).

A *negative result* is a probabilistic statement on the impossibility of any positive result. Thus, in its most general form a negative result is a pair of bounds B_R and B_Φ that, for any δ, satisfy

$$\Pr\{R(f,g) \ge B_R \ \vee \ \Phi(f,g) \le B_\Phi\} \ge \delta,$$

for *some* probability P. Here again, probability is taken w.r.t. the random choice of the training samples S_m, as well as w.r.t. all other random choices.

For a selective classifier (f,g) with coverage $\Phi(f,g)$ we can specify a Risk-Coverage (RC) trade-off as a bound on the risk $R(f,g)$, expressed in terms of $\Phi(f,g)$. Thus, a *positive result on the RC trade-off* is a probabilistic statement of the following form

$$\Pr\{R(f,g) \le B(\Phi(f,g),\delta,m)\} \ge 1-\delta.$$

Similarly, a *negative result on the RC trade-off* is a statement of the form,

$$\Pr\{R(f,g) \geq B(\Phi(f,g),\delta,m)\} \geq \delta.$$

Clearly, all results (positive and negative) are qualified by the model parameters, namely the domain $\mathcal{X}$ and the hypothesis space $\mathcal{F}$, and the quality/generality of a result should be assessed w.r.t. generality of these parameters. An additional major consideration is, of course, the computational complexity of the learning algorithm.

Finally, in the sequel we rely on the following standard definition of the version space (Mitchell, 1977).

Definition 3 (version space) *Given an hypothesis class $\mathcal{F}$ and a training sample S_m, the version space $VS_{\mathcal{F},S_m}$ is the set of all hypotheses in $\mathcal{F}$ that classify S_m correctly.*

3. Contributions

The purpose of this section is to provide a high level technical overview of our contributions. Using a training sample S_m, the goal in selective classification is to output a selective classifier (f,g) that has sufficiently low risk with sufficiently high coverage. Obviously, these two quantities trade-off each other. We call the trade-off between risk and coverage the *risk-coverage (RC) trade-off*. The best way to benefit from selective classification is to *control* the creation of the classifier so as to meet a prescribed error/coverage specification along the RC trade-off. For example, it might be desirable to devise a learning system that will receive as input an error constraint (say, 2% error) and, based on a finite (and small) training sample, will be capable of generating a classifier whose ensured test error (w.h.p.) is not larger than 2%, while having the maximum possible coverage of the domain. If the RC trade-off is revealed, it is possible to know if the 2% error constraint can be met and what would be the corresponding coverage.

In Figure 1 we schematically depict elements of the RC trade-off. The x-axis measures risk (error in the case of the $0/1$ loss) and the y-axis is coverage. The entire region depicted, called the *RC plane*, consisting of all (r,c) points in the rectangle of interest, where r is a risk (error) coordinate and c is a coverage coordinate. Assume a fixed problem setting (including an unknown underlying distribution P, m training examples drawn i.i.d. from P, an hypothesis space $\mathcal{F}$ and a tolerance parameter δ). To fully characterize the RC trade-off we need to determine for each point (r,c) on the RC plane if it is (efficiently) "achievable." We say that (r,c) is (efficiently) *achievable* if there is an (efficient) learning algorithm that will output a selective classifier (f,g) such that with probability of at least $1-\delta$, its coverage is at least c and its risk is at most r.

Notice that point r^* (the coordinate $(r^*,1)$) where the coverage is 1 represents "standard learning." At this point we require full coverage with certainty and the achievable risk represents the lowest possible risk in our fixed setting (which should be achievable with probability of at least $1-\delta$). Point r^* represent one extreme of the RC trade-off. The other extreme of the RC trade-off is point c^*, where we require zero risk *with certainty*. The coverage at c^* is the optimal (highest possible) in our setting when zero error is required. We call point c^* *perfect learning* because achievable perfect learning means that we can generate a classifier that never errs with certainty for the problem at hand. Note that at the outset, it is not at all clear if non-trivial perfect learning (with guaranteed positive coverage) can be accomplished.

The full RC trade-off is some (unknown) curve connecting points c^* and r^*. This curve passes somewhere in the zone labeled with a question mark and represents optimal selective classification.

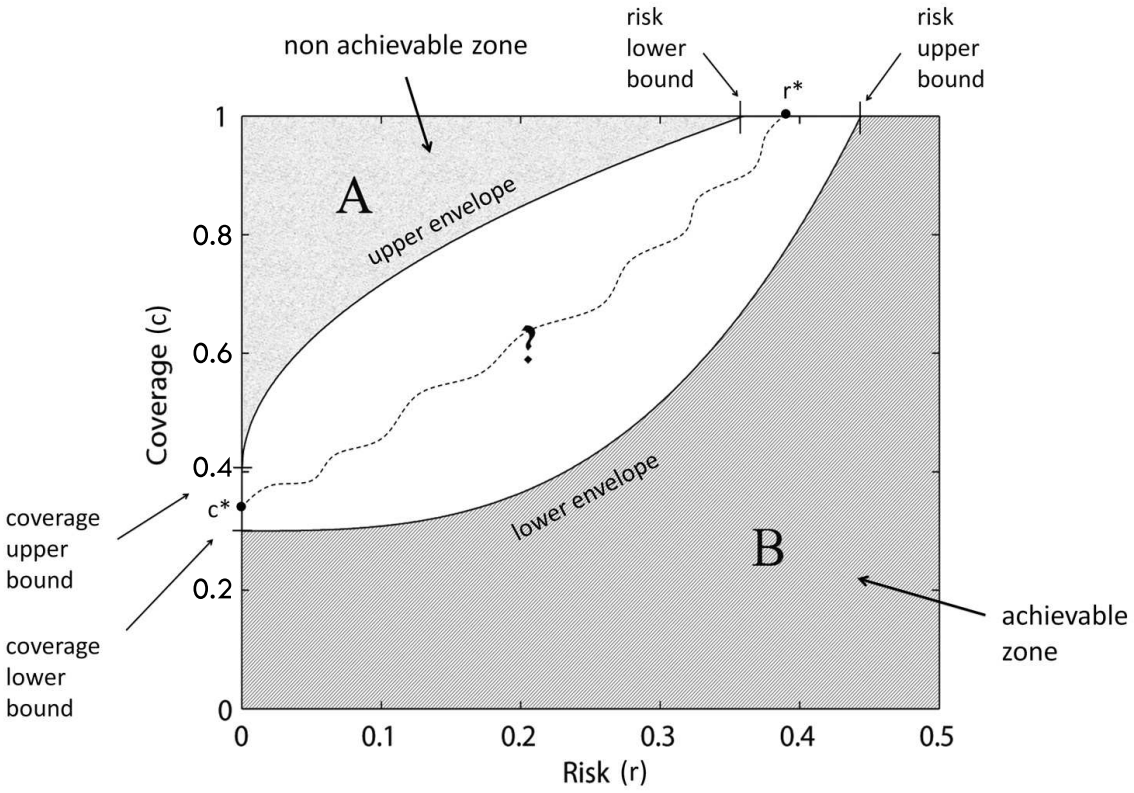


Figure 1: The RC plane and RC trade-off

Points above this curve (e.g., at zone *A*) are not achievable. Points below this curve (e.g., at zone *B*) are achievable. One of the main goals of this paper is to study the RC curve and provide as tight as possible boundaries for it. To this end we characterize upper and lower envelopes of the RC curve as schematically depicted in Figure 1. The upper envelop is a boundary of a "non-achievable zone" (zone A) and therefore we consider any upper envelop as a "negative result." The lower envelop is a boundary of an "achievable zone" (zone B) and is therefore considered as a "positive result." Note that upper and lower envelopes, as depicted in the figure, represent two different things, which are formally defined in Section 2 as probabilistic statements on possibility and impossibility.

Point r^* on the RC curve ("standard learning") was extensively studied in the literature. Perfect learning (point c^*) was never considered. For the most part, the existing work on selective classification exhibited (either empirically or theoretically) specific but anecdotal points or curves in the achievable zone (B) but, to the best of our knowledge no systematic attempts were ever made to characterize the RC-curve, which corresponds to *optimal selective classification*. In particular, there are currently no "negative" results attempting to characterize non achievable zones in the RC plane.

Our technical exposition begins by focusing on perfect learning (point c^* in the RC plane). Given the training set S_m, we are required to generate a "perfect" selective classifier (f,g) for which

it is known *with certainty* that $R(f,g) = 0$.[1] Obviously, zero risk is trivially achieved by taking g that rejects the entire input space $\mathcal{X}$. But is it possible to achieve perfect learning on a guaranteed fraction of the effective volume of $\mathcal{X}$?

Our first observation is Theorem 8, stating that for any finite hypothesis class $\mathcal{F}$, perfect learning with guaranteed coverage is achievable by a particular selective classification strategy. For any tolerance δ, with probability of at least $1-\delta$, it is guaranteed that the coverage achieved by this strategy will be at least

$$1 - \frac{1}{m} O(|\mathcal{F}| + \ln(1/\delta)). \tag{2}$$

The learning strategy that achieves this performance is simple and natural and can be termed *consistent selective strategy (CSS)*: take f to be any hypothesis from the version space (with respect to S_m), and construct a g that deterministically rejects any point that is not classified unanimously by all version space hypotheses. This CSS strategy is optimal for perfect learning. We show in Theorem 7 that any other strategy that achieves perfect learning cannot have larger coverage than CSS. It is interesting to note that the optimal selection function is not obtained by thresholding soft classification values, which is the commonly used heuristic.

It is easy to see why the classifier (f,g) selected by CSS has zero risk with certainty. Since f^* is assumed to be in the version space, and since g rejects all instances that are not classified unanimously by all the hypotheses in the version space, any selection of f from the version space will have identical classification to f^*. Nonetheless, it is surprising at the outset that the selection function g doesn't reject a lot and in fact, its rejection rate can be very small for sufficiently large m as it decreases at rate $1/m$.

This distribution-free coverage guarantee (2) is proven to be nearly tight for CSS and therefore, it is the best possible bound for any selective learner. Specifically, as shown in Theorem 11, there exist a particular finite hypothesis class and a particular underlying distribution for which a matching negative result (up to multiplicative constants) holds for any *consistent selective* learner. This result is readily extended to any selective learner by the CSS coverage optimality of Theorem 7.

What about infinite hypothesis spaces? We show in Theorem 14 that it is impossible to provide any coverage guarantees for perfect learning, in the general case. Specifically, for linear classifiers, we show a bad distribution for which any selective learner ensuring zero risk will be forced to reject the entire volume of $\mathcal{X}$, thus failing to guarantee more than zero coverage. Thus, in the general case, point c^* is simply the coordinate $(0,0)$ on the RC plane. The implication of this result is that when aiming at very small risks, the rejection rate might in general be very high (very small coverage), which may be unacceptable in many applications.

So the bad news is that perfect learning with guaranteed coverage cannot in general be achieved if the hypothesis space is infinite. Fortunately, however, this observation does not preclude non-trivial perfect learning in less adverse situations. What can be accomplished are both data-dependent and distribution-dependent guarantees. For any selective hypothesis (f,g), that is consistent with a sample S_m, Theorem 21 ensures perfect learning with a high probability coverage guarantee of the following form:

$$\Phi(f,g) \geq 1 - \frac{1}{m} O\left(\gamma(\mathcal{F},\hat{n}) \ln \frac{m}{\gamma(\mathcal{F},\hat{n})} + \ln \frac{m}{\delta}\right), \tag{3}$$

1. The requirement that in perfect learning the risk is zero *with certainty* is dual to the requirement that the coverage is 100% *with certainty* in standard learning.

where $\hat{n}$ is a new empirical quantity measuring the version space compression set size (see Definition 15), and $\gamma(\mathcal{F},k)$ is a new complexity measure of the set of all possible version spaces generated by training samples of size k. We call $\gamma(\mathcal{F},k)$ the "order-k characterizing set complexity" of $\mathcal{F}$, and it is derived using VC-dimension arguments (see Definition 18).

This general data-dependent bound is then applied to linear classifiers. Relying on a classical result in combinatorial geometry that bounds the number of facets of polytopes, we derive in Theorem 27 the following upper bound on the order-k characterizing set complexity of linear classifiers in $\mathbb{R}^d$,

$$\gamma(\mathcal{F},k) \leq O(d^3(k/d)^{d/2}\log k).$$

Plugging this bound to (3) results in a data-dependent compression bound for linear classifiers in terms of $\hat{n}$, the size of the compression set of the version space.

We then consider the evaluation of the compression set size $\hat{n}$ for specific distributions. Using a classical result in geometric probability theory on the average number of maximal random vectors, we show in Lemma 32 that if the underlying distribution is any (unknown) finite mixture of arbitrary multi-dimensional Gaussians in $\mathbb{R}^d$, then the compression set size of the version space obtained using m labeled examples satisfies, with probability of at least $1-\delta$,

$$\hat{n} = O\left((\log m)^d/\delta\right).$$

This bound immediately yields a coverage guarantee for perfect learning of linear classifiers, as stated in Corollary 33. This is a powerful result providing strong indication on the potential effectiveness of perfect learning with guaranteed coverage in a variety of applications.

In Section 7 we derive upper and lower envelopes for the RC curve. Our results on perfect learning described above play a major role in the derivation of these envelopes. We generalize the CSS strategy and define a "controllable selective strategy" (Definition 34). This strategy is parameterized by a number $\alpha \in [0,1]$ which controls the rejection rate by interpolating perfect learning and optimal standard learning. In particular, this strategy, applied with $\alpha = 0$ is perfect learning, and with $\alpha = 1$ it is optimal standard learning (full coverage), which in the realizable case is known to be achieved by any consistent learner. For any finite hypothesis space, the lower envelop we present in Theorem 36 is

$$R_\alpha(f,g) \leq \left(\frac{1-\Phi_0/\Phi_\alpha(f,g)}{1-\Phi_0}\right)\cdot\frac{1}{m}\left(\ln|\mathcal{F}|+\ln\frac{2}{\delta}\right),$$

where Φ_0 is the coverage guarantee of perfect learning in Equation (2), $R_\alpha(f,g)$ is the risk of the "controllable selective strategy" with control parameter α, and $\Phi_\alpha(f,g)$ is the matching coverage.

The upper envelop on the RC curve is then derived in Theorem 37 for any selective classifier (f,g) by constructing a particular bad distribution for which

$$R(f,g) \geq \frac{1}{4\Phi}\cdot\min\left(2\Phi-1, 2\Phi-2+\frac{1}{4m}\cdot\left[VCdim(\mathcal{F})-\frac{16}{3}\ln\frac{1}{1-2\delta}\right]\right).$$

An exact implementation of the CSS strategy appears as if it should be computationally difficult. Given a particular training set, CSS must reject a point iff it is not classified the same by all hypotheses in the current version space. In Section 8 we show an efficient algorithm that implements CSS of linear classifiers. The main idea leading to this construction is the following observation. Given a test point x we examine if the inclusion of x with either positive or negative labels in the

training sets results in linearly separable sets. Clearly, CSS must reject x iff both these augmented sets are linearly separable. Thus, the construction of the CSS selection function can be reduced to two tests of linear separability, which can be efficiently accomplished using known techniques for testing linear separability. Note that we do not construct the selection function explicitly during the training process, a task that may indeed require intensive computation. Rather, we benefit from a "lazy learning" approach whereby the selection function is constructed at test time per each example (as in nearest neighbor algorithms).

4. Perfect Learning with Finite Hypothesis Spaces

In this section we consider the simplest case of realizable learning with a finite hypothesis space $\mathcal{F}$. We show that perfect selective classification with guaranteed coverage is achievable (from a learning-theoretic perspective) by a learning strategy termed *consistent selective strategy (CSS)*. Moreover, CSS is shown to be optimal in its coverage rate, which is fully characterized by providing lower and upper bounds that match in their asymptotic behavior in the sample size m. We start by defining a region in $\mathcal{X}$, which is termed the "maximal agreement set." Any hypothesis that is consistent with the sample S_m is guaranteed to be consistent with the target hypothesis f^* on this entire region.

Definition 4 (agreement set) *Let $\mathcal{G} \subseteq \mathcal{F}$. A subset $\mathcal{X}' \subseteq \mathcal{X}$ is an* agreement set *with respect to $\mathcal{G}$ if all hypotheses in $\mathcal{G}$ agree on every instance in $\mathcal{X}'$, namely,*

$$\forall \ g_1, g_2 \in \mathcal{G}, \ \ x \in \mathcal{X}', \quad g_1(x) = g_2(x).$$

Definition 5 (maximal agreement set) *Let $\mathcal{G} \subseteq \mathcal{F}$. The* maximal agreement set *with respect to $\mathcal{G}$ is the union of all agreement sets with respect to $\mathcal{G}$.*

Recall that the version space $VS_{\mathcal{F},S_m} \subseteq \mathcal{F}$ is the set of all hypotheses that classify S_m correctly (Definition 3).

Definition 6 (consistent selective strategy (CSS)) *Given S_m, a* consistent selective strategy (CSS) *is a selective classification strategy that takes f to be any hypothesis in $VS_{\mathcal{F},S_m}$ (i.e., a consistent learner), and takes a (deterministic) selection function g that equals one for all points in the maximal agreement set with respect to $VS_{\mathcal{F},S_m}$, and zero otherwise.*

Recall that the (unknown) labeling hypothesis f^* is in $VS_{\mathcal{F},S_m}$. Thus, CSS simply rejects all points that might incur an error with respect to f^*. An immediate consequence is that any CSS selective hypothesis (f,g) always satisfies $R(f,g) = 0$. The main concern, however, is whether its coverage $\Phi(f,g)$ can be bounded from below and whether any other strategy that achieves perfect learning with certainty can achieve better coverage. The following theorem proves that CSS has the largest possible coverage among all strategies.

Theorem 7 (CSS coverage optimality) *Given S_m, let (f,g) be a selective classifier chosen by any strategy that ensures zero risk with certainty for* any *unknown distribution P and* any *target concept $f^* \in \mathcal{F}$. Let (f_c,g_c) be a selective classifier selected by CSS using S_m. Then, $\Phi(f,g) \leq \Phi(f_c,g_c)$.*

Proof For the sake of simplicity we limit the discussion to deterministic strategies. The extension to stochastic strategies is omitted but is straightforward. Given a hypothetical sample $\tilde{S}_m$ of size m, let $(\tilde{f}_c, \tilde{g}_c)$ be the selective classifier chosen by CSS and let $(\tilde{f}, \tilde{g})$ be the selective classifier chosen by any competing strategy. Assume that there exists $x_0 \in \mathcal{X}$ ($x_0 \notin \tilde{S}_m$) such that $\tilde{g}(x_0) = 1$ and $\tilde{g}_c(x_0) = 0$. According to the CSS construction of $\tilde{g}_c$, since $\tilde{g}_c(x_0) = 0$, there are at least two hypotheses $h_1, h_2 \in VS_{\mathcal{F},\tilde{S}_m}$ such that $h_1(x_0) \neq h_2(x_0)$. Assume, without loss of generality, that $h_1(x_0) = \tilde{f}(x_0)$. We will now construct a new "imaginary" classification problem and show that, under the above assumption, the competing strategy fails to guarantee zero risk with certainty. Let the imaginary target concept f'^* be h_2 and the imaginary underlying distribution P' be

$$P'(x) = \begin{cases} (1-\varepsilon)/m, & \text{if } x \in \tilde{S}_m; \\ \varepsilon, & \text{if } x = x_0; \\ 0, & \text{otherwise.} \end{cases}$$

Imagine a random sample S'_m drawn i.i.d from P'. There is a positive (perhaps small) probability that S'_m will equal $\tilde{S}_m$, in which case $(f', g') = (\tilde{f}, \tilde{g})$. Since $g'(x_0) = \tilde{g}(x_0) = 1$ and $f^*(x_0) \neq f'(x_0)$, with positive probability $R(f', g') = \varepsilon > 0$. Contradiction to the assumption that the competing strategy achieves perfect learning with certainty. It follows that for any sample $\tilde{S}_m$ and for any $x \in \mathcal{X}$, if $\tilde{g}(x) = 1$ then $\tilde{g}_c(x) = 1$. Consequently, for any unknown distribution P, $\Phi(\tilde{f}, \tilde{g}) \leq \Phi(\tilde{f}_c, \tilde{g}_c)$. ■

The next result establishes the existence of perfect learning with guaranteed coverage in the finite case.

Theorem 8 (guaranteed coverage) *Assume a finite $\mathcal{F}$ and let (f, g) be a selective classifier selected by CSS. Then, $R(f, g) = 0$ and for any $0 \leq \delta \leq 1$, with probability of at least $1 - \delta$,*

$$\Phi(f, g) \geq 1 - \frac{1}{m}\left((\ln 2)\min\{|\mathcal{F}|, |\mathcal{X}|\} + \ln\frac{1}{\delta}\right). \tag{4}$$

Proof For any ε, let $G_1, G_2, \ldots, G_k$, be all the hypothesis subsets of $\mathcal{F}$ with corresponding maximal agreement sets, $\lambda_1, \lambda_2, \ldots, \lambda_k$, such that each λ_i has volume of at most $1 - \varepsilon$ with respect to P. For any $1 \leq i \leq k$, the probability that a single point will be randomly drawn from λ_i is thus at most $1 - \varepsilon$. The probability that all training points will be drawn from λ_i is therefore at most $(1-\varepsilon)^m$. If a training point x is in $\mathcal{X} \setminus \lambda_i$, then there are at least two hypotheses $f_1, f_2 \in G_i$ that do not agree on x. Hence,

$$\Pr_P(G_i \subseteq VS_{\mathcal{F},S_m}) \leq (1-\varepsilon)^m.$$

We note that

$$k \leq 2^{\min\{|\mathcal{F}|, |\mathcal{X}|\}},$$

and by the union bound,

$$\Pr_P(\exists G_i \quad G_i \subseteq VS_{\mathcal{F},S_m}) \leq k \cdot (1-\varepsilon)^m \leq 2^{\min\{|\mathcal{F}|, |\mathcal{X}|\}} \cdot (1-\varepsilon)^m.$$

Therefore, with probability of at least $1 - 2^{\min\{|\mathcal{F}|, |\mathcal{X}|\}} \cdot (1-\varepsilon)^m$, the version space $VS_{\mathcal{F},S_m}$ differs from any subset G_i, and hence it has a maximal agreement set with volume of *at least* $1 - \varepsilon$. Using the inequality $1 - \varepsilon \leq \exp(-\varepsilon)$, we have

$$2^{\min\{|\mathcal{F}|, |\mathcal{X}|\}} \cdot (1-\varepsilon)^m \leq 2^{\min\{|\mathcal{F}|, |\mathcal{X}|\}} \cdot \exp(-m\varepsilon).$$

Equating the right-hand side to δ and solving for ε completes the proof. ■

A leading term in the coverage guarantee (4) is $|\mathcal{F}|$. In corresponding results in standard consistent learning (Haussler, 1988) the corresponding term is $\log|\mathcal{F}|$. This may raise a concern on the tightness of (4). However, as shown in Corollary 13, this bound is tight (up to multiplicative constants). To prove the Corollary we will require the following two definitions.

Definition 9 (binomial tail distribution) *Let $Z_1, Z_2, \ldots Z_m$ be m independent Bernoulli random variables each with a success probability p. Then for any $0 \leq k \leq m$ we define*

$$Bin(m,k,p) \triangleq \Pr\left(\sum_{i=1}^{m} Z_i \leq k\right).$$

Definition 10 (binomial tail inversion, Langford, 2005) *For any $0 \leq \delta \leq 1$ we define*

$$\overline{Bin}(m,k,\delta) \triangleq \max_p \{p : Bin(m,k,p) \geq \delta\}.$$

Theorem 11 (non-achievable coverage, implicit bound) *Let $0 \leq \delta \leq \frac{1}{2}$, m, and $n > 1$ be given. There exist a distribution P, that depends on m and n, and a finite hypothesis class $\mathcal{F}$ of size n, such that for any selective classifier (f,g), chosen from $\mathcal{F}$ by CSS (so $R(f,g) = 0$) using a training sample S_m drawn i.i.d. according to P, with probability of at least δ,*

$$\Phi(f,g) \leq 1 - \frac{1}{2} \cdot \overline{Bin}\left(m, \frac{|\mathcal{F}|}{2}, 2\delta\right).$$

Proof Let $\mathcal{X} \triangleq \{e_1, e_2, \ldots e_{n+1}\}$ be the standard (vector) basis of $\mathbb{R}^{n+1}$, $\mathcal{X}' \triangleq \mathcal{X} \setminus \{e_{n+1}\}$ and P be the source distribution over $\mathcal{X}$ satisfying

$$P(e_i) \triangleq \begin{cases} \overline{Bin}\left(m, \frac{n}{2}, 2\delta\right)/n, & \text{if } i \leq n; \\ 1 - \overline{Bin}\left(m, \frac{n}{2}, 2\delta\right), & \text{otherwise;} \end{cases}$$

where $\overline{Bin}(m,k,\delta)$ is the binomial tail inversion (Definition 10). Since

$$\overline{Bin}\left(m, \frac{n}{2}, 2\delta\right) \triangleq \max_p \left\{p : Bin\left(m, \frac{n}{2}, p\right) \geq 2\delta\right\},$$

and S_m is drawn i.i.d. according to P, we get that with probability of at least 2δ,

$$\left|\left\{x \in S_m : x \in \mathcal{X}'\right\}\right| \leq \frac{n}{2}.$$

Let $\mathcal{F}$ be the class of singletons such that

$$f_i(e_j) \triangleq \begin{cases} 1, & \text{if } i = j; \\ -1, & \text{otherwise.} \end{cases}$$

Taking $f^* \triangleq f_{i^*}$, for some $1 \leq i^* \leq n$, we have,

$$\begin{aligned} & \Pr\left(e_{i^*} \notin S_m, \left|\left\{x \in S_m : x \in \mathcal{X}'\right\}\right| \leq \frac{n}{2}\right) \\ = \; & \Pr\left(e_{i^*} \notin S_m \,\middle|\, \left|\left\{x \in S_m : x \in \mathcal{X}'\right\}\right| \leq \frac{n}{2}\right) \cdot \Pr\left(\left|\left\{x \in S_m : x \in \mathcal{X}'\right\}\right| \leq \frac{n}{2}\right) \\ \geq \; & \left(1 - \frac{1}{n}\right)^{\frac{n}{2}} \cdot 2\delta \geq \delta. \end{aligned}$$

If $e_{i^*} \notin S_m$ then all samples in S_m are negative, so each sample in $\mathcal{X}'$ can reduce the version space $VS_{\mathcal{F},S_m}$ by at most one hypothesis. Hence, with probability of at least δ,

$$|VS_{\mathcal{F},S_m}| \geq |\mathcal{F}| - \frac{n}{2} = \frac{n}{2}.$$

Since the coverage $\Phi(f,g)$ is the volume of the maximal agreement set with respect to the version space $VS_{\mathcal{F},S_m}$, it follows that

$$\Phi(f,g) \;\; = \;\; 1 - |VS_{\mathcal{F},S_m}| \cdot \frac{\overline{Bin}\left(m, \frac{n}{2}, 2\delta\right)}{n} \leq 1 - \frac{1}{2} \cdot \overline{Bin}\left(m, \frac{|\mathcal{F}|}{2}, 2\delta\right).$$

■

Remark 12 *The result of Theorem 11 is based on the use of the class of singletons. Augmenting this class by the empty set and choosing a uniform distribution over $\mathcal{X}$ results in a tighter bound. However, the bound will be significantly less general as it will hold only for a single hypothesis in $\mathcal{F}$ and not for any hypothesis in $\mathcal{F}$.*

Corollary 13 (non-achievable coverage, explicit bound) *Let $0 \leq \delta \leq \frac{1}{4}$, m, and $n > 1$ be given. There exist a distribution P, that depends on m and n, and a finite hypothesis class $\mathcal{F}$ of size n, such that for any selective classifier (f,g), chosen from $\mathcal{F}$ by CSS (so $R(f,g) = 0$) using a training sample S_m drawn i.i.d. according to P, with probability of at least δ,*

$$\Phi(f,g) \leq \max\left\{0, 1 - \frac{1}{8m}\left(|\mathcal{F}| - \frac{16}{3}\ln\frac{1}{1-2\delta}\right)\right\}.$$

Proof Applying Lemma 43 we get

$$\overline{Bin}\left(m, \frac{|\mathcal{F}|}{2}, 2\delta\right) \geq \min\left\{1, \frac{|\mathcal{F}|}{4m} - \frac{4}{3m}\ln\frac{1}{1-2\delta}\right\}.$$

Applying Theorem 11 completes the proof. ■

5. Consistent Selective Classification Over Infinite Hypothesis Spaces

In this section we consider an infinite hypothesis space $\mathcal{F}$. We show that in the general case, perfect selective classification with guaranteed (non-zero) coverage is not achievable even when $\mathcal{F}$ has a finite VC-dimension. We then derive a meaningful coverage guarantee using posterior information on the source distribution (data-dependent bound).

We start this section with a negative result that precludes non-trivial perfect learning when $\mathcal{F}$ is the set of linear classifiers. The result is obtained by constructing a particularly bad distribution.

Theorem 14 (non-achievable coverage) *Let m and $d > 2$ be given. There exist a distribution P, an infinite hypothesis class $\mathcal{F}$ with a finite VC-dimension d, and a target hypothesis in $\mathcal{F}$, such that $\Phi(f,g) = 0$ for any selective classifier (f,g), chosen from $\mathcal{F}$ by CSS using a training sample S_m drawn i.i.d. according to P.*

Proof Let $\mathcal{F}$ be the class of all linear classifiers in $\mathbb{R}^2$ and let P be a uniform distribution over the arcs,

$$(x-2)^2 + y^2 = 2, \quad x < 1,$$

and

$$(x+2)^2 + y^2 = 2, \quad x > -1.$$

Figure 2 depicts this construction. The training set S_m consists of points on these arcs, labeled by any linear classifier that passes between the arcs. The maximal agreement set, A, with respect to the version space $VS_{\mathcal{F},S_m}$ is partitioned into two subsets A^+ and A^- according to the labels obtained by hypotheses in the version space. Clearly, A^+ is confined by a polygon whose vertices lie on the right-hand side arc. Since P is concentrated on the arc, the probability volume of A^+ is exactly zero for any finite m. The same analysis holds for A^-, and therefore the coverage is forced to be zero. The VC-dimension of the class of all linear classifiers in $\mathbb{R}^2$ is 3. Embedding the distribution P in a higher dimensional space $\mathbb{R}^d$ and using the class of all linear classifiers in $\mathbb{R}^d$ completes the proof. ■

A direct corollary of Theorem 14 is that, in the general case, perfect selective classification with distribution-free guaranteed coverage is not achievable for infinite hypothesis spaces. However, this is certainly not the end of the story for perfect learning. In the remainder of this paper we derive meaningful coverage guarantees using posterior or prior information on the source distribution (data- and distribution-dependent bounds).

In order to guarantee meaningful coverage we first need to study the complexity of the selection function $g(x)$ chosen by CSS. The complexity of the classification function $f(x)$ is determined only by the hypothesis class $\mathcal{F}$ and it is independent of the sample size itself. However, the complexity of $g(x)$ (the maximal agreement set) chosen by CSS generally depends on the sample size. Therefore, increasing the training sample size does not necessarily guarantee non-trivial coverage. Our main task is to find the complexity class of the family of maximal agreement sets from which $g(x)$ is chosen. Let us define the family of all maximal agreement sets as $\mathcal{H} = \bigcup \mathcal{H}_n$ such that $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \mathcal{H}_3 \subset \ldots$. We can now exploit the fact that CSS chooses a maximal agreement set that belongs to a specific subclass $\mathcal{H}_n$ with a complexity measured in terms of the VC dimension of $\mathcal{H}_n$. We term this approach *Structural Coverage Maximization (SCM)* following the analogous and familiar *Structural*

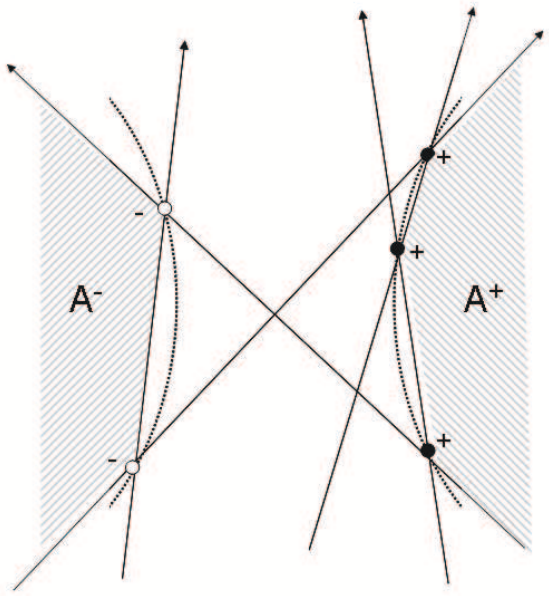


Figure 2: A worst-case distribution for linear classifiers: points are drawn uniformly at random on the two arcs and labeled by a linear classifier that passes between these arcs. The probability volume of the maximal agreement set is zero.

Risk Minimization (SRM) approach (Vapnik, 1998). A useful way to parameterize $\mathcal{H}$ is to use the size of the "version space compression set" (Definition 15).

Definition 15 (version space compression set) *Let S_m be a labeled sample of m points and let $VS_{\mathcal{F},S_m}$ be the induced version space. The* version space compression set, *$S_{\hat{n}} \subseteq S_m$ is a smallest subset of S_m satisfying $VS_{\mathcal{F},S_m} = VS_{\mathcal{F},S_{\hat{n}}}$. Note that for any given $\mathcal{F}$ and S_m, the size of the version space compression set, denoted $\hat{n} = \hat{n}(\mathcal{F}, S_m)$, is unique.*

Since a maximal agreement set is a region in $\mathcal{X}$, rather than an hypothesis, we formally define the dual hypothesis that matches every maximal agreement set.

Definition 16 (characterizing hypothesis) *Let $\mathcal{G} \subseteq \mathcal{F}$ and let $A_{\mathcal{G}}$ be the maximal agreement set with respect to $\mathcal{G}$. The* characterizing hypothesis *of $\mathcal{G}$, $f_{\mathcal{G}}(x)$ is a binary hypothesis over $\mathcal{X}$ obtaining positive values over $A_{\mathcal{G}}$ and zero otherwise.*

We are now ready to formally define $\mathcal{H}_n$, a class we term "order-n characterizing set."

Definition 17 (order-n characterizing set) *For each n, let $\mathcal{S}_n$ be the set of all possible labeled samples of size n (all n-subsets, each with all possible labelings). The order-n* characterizing set *of $\mathcal{F}$, denoted $\mathcal{H}_n$, is the set of all characterizing hypotheses $f_{\mathcal{G}}(x)$, where $\mathcal{G} \subseteq \mathcal{F}$ is a version space induced by some member of $\mathcal{S}_n$.*

Definition 18 (characterizing set complexity) *Let $\mathcal{H}_n$ be the order-n characterizing set of $\mathcal{F}$. The order-n* characterizing set complexity *of $\mathcal{F}$, denoted $\gamma(\mathcal{F}, n)$, is the VC-dimension of $\mathcal{H}_n$.*

Lemma 19 *The characterizing hypothesis $f_{VS_{\mathcal{F},S_m}}(x)$ belongs to the order-$\hat{n}$ characterizing set of $\mathcal{F}$, where $\hat{n} = \hat{n}(\mathcal{F}, S_m)$ is the size of the version space compression set.*

Proof According to Definition 15, there exists a subset $S_{\hat{n}} \subset S_m$ of size $\hat{n}$ such that $VS_{\mathcal{F},S_m} = VS_{\mathcal{F},S_{\hat{n}}}$. The rest of the proof follows immediately from Definition 17. ■

Before stating the main result of this section, we state a classical result that will be used later.

Theorem 20 (Vapnik and Chervonenkis, 1971; Anthony and Bartlett, 1999, p.53) *Let $\mathcal{F}$ be a hypothesis space with VC-dimension h. For any probability distribution P on $\mathcal{X} \times \{\pm 1\}$, with probability of at least $1-\delta$ over the choice of S_m from P^m, any hypothesis $f \in \mathcal{F}$ consistent with S_m satisfies*

$$R(f) \leq \varepsilon(h, m, \delta) = \frac{2}{m}\left[h \ln \frac{2em}{h} + \ln \frac{2}{\delta}\right], \tag{5}$$

where $R(f) \triangleq \mathbf{E}\left[\mathbb{I}(f(x) \neq f^*(x))\right]$ is the risk of f.

We note that inequality (5) actually holds only for $h \leq m$. For any $h > m$ it is clear that no meaningful upper bound on the risk can be achieved. It is easy to fix the inequality for the general case by replacing $\ln\left(\frac{2em}{h}\right)$ by $\ln_+\left(\frac{2em}{h}\right)$, where $ln_+(x) \triangleq \max(\ln(x), 1)$.

Theorem 21 (data-dependent coverage guarantee) *For any m, let $a_1, a_2, \ldots, a_m \in \mathbb{R}$ be given, such that $a_i \geq 0$ and $\sum_{i=1}^m a_i \leq 1$. Let (f,g) be a selective CSS classifier. Then, $R(f,g) = 0$, and for any $0 \leq \delta \leq 1$, with probability of at least $1-\delta$,*

$$\Phi(f,g) \geq 1 - \frac{2}{m}\left[\gamma(\mathcal{F}, \hat{n}) \ln_+\left(\frac{2em}{\gamma(\mathcal{F}, \hat{n})}\right) + \ln \frac{2}{a_{\hat{n}}\delta}\right],$$

where $\hat{n}$ is the size of the version space compression set, $\gamma(\mathcal{F}, \hat{n})$ is the order-$\hat{n}$ characterizing set complexity of $\mathcal{F}$.

Proof Given our sample $S_m = \{(x_i, f^*(x_i))\}_{i=1}^m$ (labeled by the unknown target function f^*), we define the "synthetic" sample $S'_m = \{(x_i, 1)\}_{i=1}^m$. S'_m can be assumed to have been sampled i.i.d from the marginal distribution of X with positive labels (P').

Theorem 20 can now be applied on the synthetic problem with the training sample S'_m, the distribution P', and the hypothesis space taken to be $\mathcal{H}_i$, the order-i characterizing set of $\mathcal{F}$. It follows that for all $f \in VS_{\mathcal{H}_i, S'_m}$, with probability of at least $1 - a_i\delta$ over choices of S'_m from $(P')^m$,

$$\Pr_{P'}(f(x) \neq 1) \quad \leq \quad \frac{2}{m}\left[h_i \ln\left(\frac{2em}{h_i}\right) + \ln \frac{2}{a_i\delta}\right], \tag{6}$$

where h_i is the VC-dimension of $\mathcal{H}_i$. Then, applying the union bound yields, with probability of at least $1-\delta$, that inequality (6) holds simultaneously for all $1 \leq i \leq m$.

All hypotheses in the version space $VS_{\mathcal{F},S_m}$ agree on all samples in S_m. Hence, the characterizing hypothesis $f_{VS_{\mathcal{F},S_m}}(x) = 1$ for any point $x \in S_m$. Let $\hat{n}$ be the size of the version space compression set. According to Lemma 19, $f_{VS_{\mathcal{F},S_m}}(x) \in \mathcal{H}_{\hat{n}}$. Noting that $f_{VS_{\mathcal{F},S_m}}(x) = 1$ for any $x \in S'_m$, we learn that $f_{VS_{\mathcal{F},S_m}}(x) \in VS_{\mathcal{H}_{\hat{n}}, S'_m}$. Therefore, with probability of at least $1-\delta$ over choices of S_m,

$$\Pr_P(f_{VS_{\mathcal{F},S_m}}(x) \neq 1) \leq \frac{2}{m}\left[h_{\hat{n}} \ln\left(\frac{2em}{h_{\hat{n}}}\right) + \ln \frac{2}{a_{\hat{n}}\delta}\right].$$

Since $\Phi(f,g) = \Pr_P(f_{VS_{\mathcal{F},S_m}}(x) = 1)$, and $h_{\hat{n}}$ is the order-$\hat{n}$ characterizing set complexity of $\mathcal{F}$, the proof is complete. ■

6. Consistent Selective Classification With Linear Classifiers

The data dependent bound in Theorem 21 is stated in terms of a new complexity measure (the "characterizing set complexity" of Definition 18). Can this measure be explicitly evaluated or bounded for some interesting hypothesis classes? In this section we consider the class of linear classifiers in $\mathbb{R}^d$. Relying on a classical result from combinatorial geometry, we infer an explicit upper bound on the characterizing set complexity for linear classifiers. Combining this bound with Theorem 21, we immediately obtain a data-dependent compression coverage guarantee, as stated in Corollary 28. We then show that if the unknown underlying distribution is a finite mixture of Gaussians, then CSS will ensure perfect learning with guaranteed coverage. This powerful result, which is stated in Corollary 33, indicates that consistent selective classification might be relevant in various applications of interest.

Fix any positive integer d, and let $\mathcal{F} \triangleq \{f_{\bar{w},\phi}(\bar{x})\}$ be the class of all linear binary classifiers in $\mathbb{R}^d$, where $\bar{w}$ are d-dimensional real vectors, ϕ are scalars, and

$$f_{\bar{w},\phi}(\bar{x}) = \begin{cases} +1, & \bar{w}^T\bar{x} - \phi \geq 0; \\ -1, & \bar{w}^T\bar{x} - \phi < 0. \end{cases}$$

Given a binary labeled training sample S_m, define $R^+ \triangleq R^+(S_m) \subseteq \mathbb{R}^d$ to be the subset of the maximal agreement set with respect to the version space $VS_{\mathcal{F},S_m}$, consisting of all points with positive labels. R^+ is called the 'maximal positive agreement set.' The 'maximal negative agreement set', $R^- \triangleq R^-(S_m)$, is defined similarly. Before continuing, we define a new symmetric hypothesis class $\tilde{\mathcal{F}}$ that allows for a simpler analysis. Let $\tilde{\mathcal{F}} \triangleq \{f_{\bar{w},\phi}(\bar{x})\}$ be the function class

$$\tilde{f}_{\bar{w},\phi}(\bar{x}) = \begin{cases} +1, & \text{if } \bar{w}^T\bar{x} - \phi > 0; \\ 0, & \text{if } \bar{w}^T\bar{x} - \phi = 0; \\ -1, & \text{if } \bar{w}^T\bar{x} - \phi < 0, \end{cases}$$

where we interpret 0 as a classification that agrees with both $+1$ and -1. Given a sample S_m, we define $\tilde{R}^+ \subseteq \mathbb{R}^d$ to be the region in $\mathbb{R}^d$ for which any hypothesis in the version space[2] $VS_{\tilde{\mathcal{F}},S_m}$ classifies either $+1$ or 0 (i.e., this is the maximal positive agreement set). We define $\tilde{R}^-$ analogously with respect to negative or zero classifications. While $\mathcal{F}$ and $\tilde{\mathcal{F}}$ are not identical, the maximal agreement sets they induce are identical. This is stated in the following technical lemma whose proof appears in the appendix.

Lemma 22 (maximal agreement set equivalence) *For any linearly separable sample S_m, $R^+ = \tilde{R}^+$ and $R^- = \tilde{R}^-$.*

The next technical lemma, whose proof also appears in the appendix, provides useful information on the geometry of the maximal agreement set for the class of linear classifiers.

2. Any hypothesis in $\tilde{\mathcal{F}}$ that classifies every sample in S_m correctly or as 0 belongs to the version space.

Lemma 23 (maximal agreement set geometry I) *Let S_m be a linearly separable labeled sample that is a spanning set of $\mathbb{R}^d$. Then the regions R^+ and R^- are each an intersection of a finite number of half-spaces, with at least d samples on the boundary of each half-space.*

Our goal is to bound the characterizing set complexity of $\mathcal{F}$. As we show below, this complexity measure is directly related to the number of facets of the convex hull of n points in $\mathbb{R}^d$. The following classical combinatorial geometry theorem by Klee (see Preparata and Shamos, 1990, page 98) is thus particularly useful. The statement of Klee's theorem provided here is readily obtained from the original by using the Stirling approximation of the binomial coefficient.

Theorem 24 (Klee, 1966) *The number of facets of a d-polytope with n vertices is at most*

$$2 \cdot \left(\frac{en}{\lfloor d/2 \rfloor} \right)^{\lfloor d/2 \rfloor}. \tag{7}$$

An immediate conclusion is that (7) upper bounds the number of facets of the convex hull of n points in $\mathbb{R}^d$ (which is of course a d-polytope).

Lemma 25 (maximal agreement set geometry II) *Let S_n be a linearly separable sample consisting of $n \geq d+1$ labeled points. Then the regions $R^+(S_n)$ and $R^-(S_n)$ are each an intersection of at most*

$$2(d+1) \cdot \left(\frac{2en}{d} \right)^{\lfloor \frac{d+1}{2} \rfloor}$$

half-spaces in $\mathbb{R}^d$.

Proof For the sake of clarity, we limit the analysis to a sample S_n in general position; that is, we assume that no more than d points lie on a $(d-1)$-dimensional plane. Handling a sample S_n in arbitrary position can be straightforwardly treated by including an appropriate infinitesimal displacement of the points (the technical proof is omitted).

By Lemma 22, we can limit our discussion to the hypothesis space $\tilde{\mathcal{F}}$ (rather than $\mathcal{F}$). Since S_n includes more than d samples in general position it is a spanning set of $\mathbb{R}^d$. According to Lemma 23, R^+ is an intersection of a finite number of half-spaces, with at least d samples on the boundary of each half-space (and *exactly* d in the general position). Let $S^+ \subseteq S_n$ be the subset of all positive samples in S_n, and $S^- \subseteq S_n$, the negative ones. Let $\tilde{f}_{\bar{w},\phi}$ be one of the half-spaces defining R^+. Then,

$$\forall \bar{x} \in S_n \quad \begin{cases} \bar{w}^T \bar{x} - \phi \geq 0, & \text{if } \bar{x} \in S^+; \\ \bar{w}^T \bar{x} - \phi \leq 0, & \text{if } \bar{x} \in S^-. \end{cases}$$

Also, exactly d samples, $\bar{x}$, satisfy $\bar{w}\bar{x} - \phi = 0$.

We now embed the samples in $\mathbb{R}^{d+1}$ using the following transformation, $\bar{x} \to \bar{x}'$:

$$\bar{x}' \triangleq \begin{cases} (0, \bar{x}), & \text{if } \bar{x} \in S^+; \\ (1, -\bar{x}), & \text{if } \bar{x} \in S^-. \end{cases}$$

For each half-space $(\bar{w}, \phi)$ in $\mathbb{R}^d$ we define a unique half-space, $(\bar{w}', \phi')$, in $\mathbb{R}^{d+1}$,

$$\bar{w}' \triangleq (2\phi, \bar{w}), \quad \phi' \triangleq \phi.$$

We observe that

$$\bar{w}'^T\bar{x}'-\phi'=\begin{cases}\bar{w}^T\bar{x}-\phi\geq 0, & \text{if } \bar{x}\in S^+;\\ 2\phi-\bar{w}^T\bar{x}-\phi=-(\bar{w}^T\bar{x}-\phi)\geq 0, & \text{if } \bar{x}\in S^-,\end{cases}$$

and for exactly d samples we have

$$\bar{w}'^T\bar{x}'-\phi'=\begin{cases}\bar{w}^T\bar{x}-\phi=0, & \text{if } \bar{x}\in S^+;\\ 2\phi-\bar{w}^T\bar{x}-\phi=-(\bar{w}^T\bar{x}-\phi)=0, & \text{if } \bar{x}\in S^-.\end{cases}$$

Let $\bar{v}$ be any orthogonal vector to the d samples on the boundary of the half-space. Defining

$$\bar{w}''\triangleq\bar{w}'+\alpha\bar{v},\quad \phi''\triangleq\phi',$$

with an appropriate choice of α we have,

$$\forall\bar{x}'\in S_n\quad \bar{w}''^T\bar{x}'-\phi''=\bar{w}'^T\bar{x}'-\phi'+\alpha\bar{v}'^T\bar{x}'\geq 0,$$

and for exactly $d+1$ samples (including the original d samples),

$$\bar{w}''\bar{x}'-\phi''=0.$$

We observe that $\tilde{f}_{\bar{w}'',\phi''}$ is a facet of the convex hull of the samples in $\mathbb{R}^{d+1}$. Up to $d+1$ different half-spaces in $\mathbb{R}^d$ can be transformed into a single half-space in $\mathbb{R}^{d+1}$ (the number of combinations of choosing d samples out of $d+1$ samples on the boundary). Using Theorem 24, we bound the number $F(d)$ of facets of the convex hull of the points in $\mathbb{R}^{d+1}$ as follows:

$$F(d)\leq 2\cdot\left(\frac{en}{\lfloor\frac{d+1}{2}\rfloor}\right)^{\lfloor\frac{d+1}{2}\rfloor}\leq 2\cdot\left(\frac{2en}{d}\right)^{\lfloor\frac{d+1}{2}\rfloor}.$$

Since up to $d+1$ half-spaces in $\mathbb{R}^d$ can be mapped onto a single facet of the convex hull in $\mathbb{R}^{d+1}$, we can bound the number of half-spaces in $\mathbb{R}^d$ by

$$(d+1)\cdot F(d)\leq 2(d+1)\cdot\left(\frac{2en}{d}\right)^{\lfloor\frac{d+1}{2}\rfloor}.$$

■

Lemma 26 (*Blumer et al., 1989, Lemma 3.2.3*) *Let $\mathcal{F}$ be a binary hypothesis class of finite VC dimension $h\geq 1$. For all $k\geq 1$, define the k-fold intersection,*

$$\mathcal{F}_{k\cap}\triangleq\left\{\cap_{i=1}^k f_i : f_i\in\mathcal{F}, 1\leq i\leq k\right\},$$

and the k-fold union,

$$\mathcal{F}_{k\cup}\triangleq\left\{\cup_{i=1}^k f_i : f_i\in\mathcal{F}, 1\leq i\leq k\right\}.$$

Then, for all $k\geq 1$,

$$VC(\mathcal{F}_{k\cap}), VC(\mathcal{F}_{k\cap})\leq 2hk\log(3k).$$

Lemma 27 (characterizing set complexity) *Fix $d \geq 2$ and $n > d$. Let $\mathcal{F}$ be the class of all linear binary classifiers in $\mathbb{R}^d$. Then, the order-n characterizing set complexity of $\mathcal{F}$ satisfies*

$$\gamma(\mathcal{F},n) \leq 83 \cdot (d+1)^3 \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \log n.$$

Proof Let $\mathcal{G} = \mathcal{F}_{k\cap}$ be the class of k-fold intersections of half-spaces in $\mathbb{R}^d$. Since the VC dimension of the class of all half-spaces in $\mathbb{R}^d$ is $d+1$, we obtain, using Lemma 26, that the VC dimension of $\mathcal{G}$ satisfies

$$VC(\mathcal{G}) \leq 2k\log(3k)(d+1).$$

Let $\mathcal{H}_n$ be the order-n characterizing set of $\mathcal{F}$. From Lemma 25 we know that any hypothesis $f \in \mathcal{H}_n$ is a union of two regions, where each region is an intersection of no more than

$$k = 2(d+1) \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor}$$

half-spaces in $\mathbb{R}^d$. Therefore, $\mathcal{H}_n \subset \mathcal{G}_{2\cup}$. Using Lemma 26, we get

$$\begin{aligned} VC(\mathcal{H}_n) &\leq VC(\mathcal{G}_{2\cup}) \leq 4\log(6) \cdot VC(\mathcal{G}) \leq 8k\log(6)\log(3k)(d+1) \\ &\leq 16(d+1)^2 \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \log(6) \cdot \log\left(6(d+1) \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor}\right). \end{aligned}$$

For $n > d \geq 2$ we get

$$\begin{aligned} &\log\left(6(d+1) \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor}\right) \leq \log(6n) + \left\lfloor \frac{d+1}{2} \right\rfloor \cdot \log\frac{2en}{d} \\ \leq\ & 3 \cdot \log n + \left\lfloor \frac{d+1}{2} \right\rfloor \cdot \log n^2 \leq (d+4) \cdot \log n \leq 2 \cdot (d+1) \cdot \log n. \end{aligned}$$

Therefore,

$$VC(\mathcal{H}_n) \leq 83 \cdot (d+1)^3 \cdot \left(\frac{2en}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \log n$$

■

Corollary 28 (data-dependent coverage guarantee) *Let $\mathcal{F}$ be the class of linear binary classifiers in $\mathbb{R}^d$ and assume that the conditions of Theorem 21 hold. Then, $R(f,g) = 0$, and for any $0 \leq \delta \leq 1$, with probability of at least $1-\delta$,*

$$\Phi(f,g) \geq 1 - \frac{2}{m}\left[83(d+1)^3 \Lambda_{\hat{n},d} \ln_+\left(\frac{2em}{\Lambda_{\hat{n},d}}\right) + \ln\frac{2}{a_{\hat{n}}\delta}\right],$$

where $\hat{n}$ is the size of the empirical version space compression set, and

$$\Lambda_{\hat{n},d} = \left(\frac{2e\hat{n}}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \log\hat{n}.$$

Proof Define

$$\Psi(\gamma(\mathcal{F},n)) \triangleq 1-\frac{2}{m}\left[\gamma(\mathcal{F},n)\ln_+\left(\frac{2em}{\gamma(\mathcal{F},n)}\right)+\ln\frac{2}{a_n\delta}\right].$$

We note that $\Psi(\gamma(\mathcal{F},n))$ is a continues function. For any $\gamma(\mathcal{F},n)<2m$

$$\frac{\partial\Psi(\gamma(\mathcal{F},n))}{\partial\gamma(\mathcal{F},n)}=-\frac{2}{m}\ln\frac{2em}{\gamma(\mathcal{F},n)}+\frac{2}{m}<0,$$

and for any $\gamma(\mathcal{F},n)>2m$

$$\frac{\partial\Psi(\gamma(\mathcal{F},n))}{\partial\gamma(\mathcal{F},n)}=-\frac{2}{m}<0.$$

Thus, $\Psi(\gamma(\mathcal{F},n))$ is monotonically decreasing. Noting that $\ln_+(x)$ is monotonically increasing, by applying Theorem 21 together with Lemma 27 the proof is complete. ■

As long as the empirical version space compression set size $\hat{n}$ is sufficiently small compared to m, Corollary 28 provides a meaningful coverage guarantee. Since $\hat{n}$ might depend on m, it is hard to analyze the effective rate of the bound. To further explore this guarantee, we now bound $\hat{n}$ in terms of m for a specific family of source distributions and derive a distribution-dependent coverage guarantee.

Theorem 29 (Bentley, Kung, Schkolnick, and Thompson, 1978) *If m points in d dimensions have their components chosen independently from any set of continuous distributions (possibly different for each component), then the expected number of convex hull vertices v is*

$$\mathbf{E}[v]=O\left((\log m)^{d-1}\right).$$

Definition 30 (sliced multivariate Gaussian distribution) *A* sliced multivariate Gaussian *distribution, $\mathcal{N}(\Sigma,\mu,w,\phi)$, is a multivariate Gaussian distribution restricted by a half space in $\mathbb{R}^d$. Thus, if Σ is a non-singular covariance matrix, the pdf of the sliced Gaussian is*

$$\frac{1}{C}e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}\cdot\mathbb{I}(w^Tx-\phi\geq 0),$$

where $\mu=(\mu_1,\ldots,\mu_d)^T$, $\mathbb{I}$ is the indicator function and C is an appropriate normalization factor.

Lemma 31 *Let P be a sliced multivariate Gaussian distribution. If m points are chosen independently from P, then the expected number of convex hull vertices is $O\left((\log m)^{d-1}\right)$.*

Proof Let $X\sim\mathcal{N}(\Sigma,\mu,w,\phi)$ and $Y\sim\mathcal{N}(\Sigma,\mu)$. There is a random vector Z, whose components are independent standard normal random variables, a vector μ, and a matrix A such that $Y=AZ+\mu$. Since

$$w^Ty-\phi=w^T(Az+\mu)-\phi=w^TAz+w^T\mu-\phi,$$

we get that $X=AZ_0+\mu$, where $Z_0\sim\mathcal{N}(I,0,w^TA,\phi-w^T\mu)$. Due to the spherical symmetry of Z, we can choose the half-space $(w^TA,\phi-w^T\mu)$ to be axis-aligned by rotating the axes. We note that the d components of Z are chosen independently and that the axis-aligned half-space enforces

restriction only on one of the axes. Therefore, the components of Z_0 are chosen independently as well. Applying Theorem 29, we get that if m points are chosen independently from Z_0, then the expected number of convex hull vertices is $O\left((\log m)^{d-1}\right)$. The proof is complete by noting that the number of convex hull vertices is preserved under affine transformations. ■

Lemma 32 (version space compression set size) *Let $\mathcal{F}$ be the class of all linear binary classifiers in $\mathbb{R}^d$. Assume that the underlying distribution P is a mixture of a fixed number of Gaussians. Then, for any $0 \leq \delta \leq 1$, with probability of at least $1-\delta$, the empirical version space compression set size is*

$$\hat{n} = O\left(\frac{(\log m)^{d-1}}{\delta}\right).$$

Proof Let S_n be a version space compression set. Consider $\bar{x}_0 \in S_n$. Since S_n is a compression set there is a half-space, $(\bar{w}, \phi)$, such that $f_{\bar{w},\phi} \in VS_{\mathcal{F},S_n \setminus \{\bar{x}_0\}}$ and $f_{\bar{w},\phi} \notin VS_{\mathcal{F},S_n}$. W.l.o.g. assume that $\bar{x}_0 \in S_n$ is positive; thus $\bar{w}^T \bar{x}_0 - \phi < 0$, and for any other positive point $\bar{x} \in S_n$, $\bar{w}^T \bar{x} - \phi \geq 0$. For an appropriate $\phi' < \phi$, there exists a half-space $(\bar{w}, \phi')$ such that $\bar{w}^T \bar{x}_0 - \phi' = 0$, and for any other positive point $\bar{x} \in S_n$, $\bar{w}^T \bar{x} - \phi' > 0$. Therefore, $\bar{x}_0$ is a convex hull vertex. It follows that we can bound the number of positive samples in S_n by the number of vertices of the convex hull of all the positive points. Defining v as the number of convex hull vertices and using Markov's inequality, we get that for any $\varepsilon > 0$,

$$\Pr(v \geq \varepsilon) \leq \frac{\mathbf{E}[v]}{\varepsilon}.$$

Since f^* is a linear classifier, the underlying distribution of the positive points is a mixture of sliced multivariate Gaussians. Using Lemmas 31 and 44, we get that with probability of at least $1-\delta$,

$$v \leq \frac{\mathbf{E}[v]}{\delta} = O\left(\frac{(\log m)^{d-1}}{\delta}\right).$$

Repeating the same arguments for the negative points completes the proof. ■

Corollary 33 (distribution-dependent coverage guarantee) *Let $\mathcal{F}$ be the class of all linear binary classifiers in $\mathbb{R}^d$, and let P be a mixture of a fixed number of Gaussians. Then, $R(f,g) = 0$, and for any $0 \leq \delta \leq 1$, with probability of at least $1-\delta$,*

$$\Phi(f,g) \geq 1 - O\left(\frac{(\log m)^{d^2}}{m} \cdot \frac{1}{\delta^{(d+3)/2}}\right).$$

Proof

$$\Lambda_{\hat{n},d} = \left(\frac{2e\hat{n}}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \log \hat{n} \leq \left(\frac{2e}{d}\right)^{\left\lfloor \frac{d+1}{2} \right\rfloor} \cdot \hat{n}^{\frac{d+3}{2}}.$$

Applying Lemma 32,

$$\Lambda_{\hat{n},d} = O\left(\frac{(\log m)^{d^2}}{\delta^{(d+3)/2}}\right).$$

The proof is complete by noting that $\Lambda_{\hat{n},d} \geq 1$ and using Corollary 28 with $a_i = 2^{-i}$. ■

7. Risk-coverage Trade-off Envelopes

In previous sections we have shown that by compromising the coverage we can achieve zero risk. This is in contrast to the classical setting, where we compromise risk to achieve full coverage. Is it possible to learn a selective classifier with full control over this trade-off? What are the performance limitations of this trade-off control?

In this section we present some answers to these questions thus deriving lower and upper envelopes for the risk-coverage (RC) trade-off. These results heavily rely on the previous results on perfect learning and on classical results on standard learning without rejection. The envelopes are obtained by interpolating bounds on these two extreme types of learning. We begin this section by deriving a lower envelop; that is, we introduce a strategy that can control the RC trade-off.

7.1 Lower Envelop: Controlling the Coverage-risk Trade-off

Our lower RC envelop is facilitated by the following strategy, which is a generalization of the consistent selective classification strategy (CSS) of Definition 6.

Definition 34 (controllable selective strategy) *Given a* mixing *parameter* $0 \leq \alpha \leq 1$, *the* controllable selective strategy *chooses a selective classifier* (f,g) *such that* f *is in the version space* $VS_{\mathcal{F},S_m}$ *(as in CSS), and* g *is defined as follows:* $g(x) = 1$ *for any* x *in the maximal agreement set,* A*, with respect to* $VS_{\mathcal{F},S_m}$*, and* $g(x) = \alpha$ *for any* $x \in \mathcal{X} \setminus A$.

Clearly, CSS is a special case of the controllable selective strategy obtained with $\alpha = 0$. Standard consistent learning (in the classical setting) is the special case obtained with $\alpha = 1$. We now state a well known (and elementary) upper bound for classical realizable learning.

Theorem 35 (Haussler, 1988) *Let* $\mathcal{F}$ *be any finite hypothesis class. Let* $f \in VS_{\mathcal{F},S_m}$ *be a classifier chosen by any consistent learner. Then, for any* $0 \leq \delta \leq 1$, *with probability of at least* $1-\delta$,

$$R(f) \leq \frac{1}{m}\left(\ln|\mathcal{F}| + \ln\frac{1}{\delta}\right),$$

where $R(f)$ *is standard risk (true error) of the classifier* f.

The following result provides a distribution independent upper bound on the risk of the controllable selective strategy as a function of its coverage.

Theorem 36 (lower envelop) *Let* $\mathcal{F}$ *be any finite hypothesis class. Let* (f,g) *be a selective classifier chosen by a controllable selective learner after observing a training sample* S_m*. Then, for any* $0 \leq \delta \leq 1$, *with probability of at least* $1-\delta$,

$$R(f,g) \leq \left(\frac{1-\Phi_0/\Phi(f,g)}{1-\Phi_0}\right) \cdot \frac{1}{m}\left(\ln|\mathcal{F}| + \ln\frac{2}{\delta}\right),$$

where

$$\Phi_0 \triangleq 1 - \frac{1}{m}\left((\ln 2)|\mathcal{F}| + \ln\frac{2}{\delta}\right).$$

Proof For any controllable selective learner with a mixing parameter α we have,

$$\Phi(f,g) \quad = \quad \mathbf{E}[g(X)] = \mathbf{E}[\mathbb{I}(g(X)=1)] + \alpha \mathbf{E}[\mathbb{I}(g(X)\neq 1)].$$

By Theorem 8, with probability of at least $1-\frac{\delta}{2}$,

$$\mathbf{E}[\mathbb{I}(g(X)=1)] \geq 1 - \frac{1}{m}\left((\ln 2)|\mathcal{F}| + \ln\frac{2}{\delta}\right) \triangleq \Phi_0.$$

Therefore, since $\Phi(f,g) \leq 1$,

$$\alpha = \frac{\Phi(f,g) - \mathbf{E}[\mathbb{I}(g(X)=1)]}{1-\mathbf{E}[\mathbb{I}(g(X)=1)]} \leq \frac{\Phi(f,g)-\Phi_0}{1-\Phi_0}.$$

Using the law of total expectation we get

$$\begin{aligned}
\mathbf{E}[\ell(f(X),Y)\cdot g(X)] &= \overbrace{\mathbf{E}[\ell(f(X),Y)\cdot g(X) \mid g(x)=1]}^{0} \cdot \Pr(g(X)=1) \\
&+ \mathbf{E}[\ell(f(X),Y)\cdot g(X) \mid g(x)=\alpha] \cdot \Pr(g(X)=\alpha) \\
&= \alpha \cdot \mathbf{E}[\ell(f(X),Y) \mid g(x)=\alpha] \cdot \Pr(g(X)=\alpha) \\
&= \alpha \cdot \mathbf{E}[\ell(f(X),Y)].
\end{aligned}$$

According to Definition 2.2

$$R(f,g) \quad = \quad \frac{\mathbf{E}[\ell(f(X),Y)\cdot g(X)]}{\Phi(f,g)} = \frac{\alpha\cdot\mathbf{E}[\ell(f(X),Y)]}{\Phi(f,g)} = \frac{\alpha\cdot R(f,g)}{\Phi(f,g)}.$$

Applying Theorem 35 together with the union bound completes the proof. ■

7.2 Upper Envelop: Trade-off Control Limitation

We now present a negative result which identifies a region of non-achievable coverage-risk trade-off on the RC plane. The statement is a probabilistic lower bound on the risk of *any* selective classifier expressed as a function of the coverage. It negates any high probability upper bound on the risk of the classifier (where the probability is over choice of S_m and the target hypothesis).

Theorem 37 (non-achievable coverage-risk trade-off) *Let $\mathcal{F}$ be any hypothesis class and let $0 \leq \delta \leq \frac{1}{4}$ and m be given. There exists a distribution P (that depends on $\mathcal{F}$), such that for any selective classifier (f,g), chosen using a training sample S_m drawn i.i.d. according to P, with probability of at least δ,*

$$R(f,g) \geq \min\left(\frac{1}{2} - \frac{1}{4\Phi(f,g)}, \frac{1}{2} - \frac{1}{2\Phi(f,g)} + \frac{1}{16m\cdot\Phi(f,g)}\cdot\left[VCdim(\mathcal{F}) - \frac{16}{3}\ln\frac{1}{1-2\delta}\right]\right)$$

Proof If η is the VC-dimension of hypothesis class $\mathcal{F}$, there exists a set of data points $\mathcal{X}' = \{e_1, e_2, \ldots e_\eta\}$ shattered by $\mathcal{F}$. Let $\mathcal{X} \triangleq \mathcal{X}' \cup \{e_{\eta+1}\}$. The bad distribution is constructed as follows. Define $\overline{Bin}(m,k,\delta)$, the binomial tail inversion,

$$\overline{Bin}\left(m, \frac{\eta}{2}, 2\delta\right) \triangleq \max_p \left\{p : Bin\left(m, \frac{\eta}{2}, p\right) \geq 2\delta\right\},$$

where $Bin(m,k,p)$ is the binomial tail. Define P to be the source distribution over $\mathcal{X}$ satisfying

$$P(e_i) \triangleq \begin{cases} \overline{Bin}\left(m, \frac{\eta}{2}, 2\delta\right)/\eta, & \text{if } i \leq \eta; \\ 1 - \overline{Bin}\left(m, \frac{\eta}{2}, 2\delta\right), & \text{otherwise,} \end{cases}$$

Assuming that the training sample is selected i.i.d. from P, it follows that with probability of at least 2δ,

$$\left|\left\{x \in S_m : x \in \mathcal{X}'\right\}\right| \leq \frac{\eta}{2}.$$

$\mathcal{F}$ shatters $\mathcal{X}'$ thus inducing all dichotomies over $\mathcal{X}'$. Every sample from $\mathcal{X}'$ can reduce the version space by half, so with probability of at least 2δ, the version space $VS_{\mathcal{F},S_m}$ includes all dichotomies over at least $\frac{\eta}{2}$ instances. Therefore, over these instances (referred to as $x_1, x_2, \ldots, x_{\eta/2}$), with probability of $1/2$ the error is at least $\frac{1}{2}$.[3]

$$\begin{aligned} \Phi(f,g) &= \sum_{i=1}^{\eta+1} \{P(e_i)\cdot g(e_i)\} = P(e_1)\cdot\sum_{i=1}^{\eta} g(e_i) + P(e_{\eta+1})\cdot g(e_{\eta+1}) \\ &\leq P(e_1)\cdot\sum_{i=1}^{\frac{\eta}{2}} g(x_i) + \frac{\eta}{2}\cdot P(e_1) + P(e_{\eta+1}) = P(e_1)\cdot\sum_{i=1}^{\frac{\eta}{2}} g(x_i) + 1 - \frac{\eta}{2}\cdot P(e_1) \\ \Longrightarrow\ & P(e_1)\cdot\sum_{i=1}^{\frac{\eta}{2}} g(x_i) \geq \Phi(f,g) + \frac{\eta}{2}\cdot P(e_1) - 1. \end{aligned}$$

$$\begin{aligned} \Phi(f,g)\cdot R(f,g) &= \sum_{i=1}^{\eta+1} \{P(e_i)\cdot g(e_i)\cdot \mathbb{I}(f(e_i)\neq f^*(e_i))\} \geq \sum_{i=1}^{\frac{\eta}{2}} \left\{P(x_i)\cdot g(x_i)\cdot\frac{1}{2}\right\} \\ &\geq \frac{\Phi(f,g)-1}{2} + \frac{\eta}{4}\cdot P(e_1) = \frac{\Phi(f,g)-1}{2} + \frac{1}{4}\cdot\overline{Bin}\left(m,\frac{\eta}{2},2\delta\right). \end{aligned}$$

Applying Lemma 43 we get

$$R(f,g) \geq \min\left(\frac{1}{2} - \frac{1}{4\Phi(f,g)}, \frac{1}{2} - \frac{1}{2\Phi(f,g)} + \frac{1}{16m\cdot\Phi(f,g)}\cdot\left[VCdim(\mathcal{F}) - \frac{16}{3}\ln\frac{1}{1-2\delta}\right]\right)$$

■

Corollary 38 *Let $0 \leq \delta \leq \frac{1}{4}$, m, and $n > 1$ be given. There exist a distribution P, that depends on m and n, and a finite hypothesis class $\mathcal{F}$ of size n, such that for any selective classifier (f,g), chosen using a training sample S_m drawn i.i.d. according to P, with probability of at least δ, if*

$$\Phi(f,g) \geq max\left\{\frac{3}{4}, 1 - \frac{1}{16m}\cdot\left[VCdim(\mathcal{F}) - \frac{16}{3}\ln\frac{1}{1-2\delta}\right]\right\}$$

then

$$R(f,g) \geq \frac{1}{16m}\cdot\left[VCdim(\mathcal{F}) - \frac{16}{3}\ln\frac{1}{1-2\delta}\right].$$

3. According to the game theoretic setting the adversary can choose a distribution over $\mathcal{F}$. In this case the expectation in the risk is averaged over random instances and random labels. Therefore, the error over the instances $x_1, x_2, \ldots, x_{\eta/2}$ is exactly $1/2$ and we can replace the term 2δ with δ.

Proof Assuming

$$\Phi(f,g) \geq max\left\{\frac{3}{4}, 1-\frac{1}{16m}\cdot\left[VCdim(\mathcal{F})-\frac{16}{3}\ln\frac{1}{1-2\delta}\right]\right\},$$

we apply Theorem 37 to complete the proof. ■

8. CSS Implementation: Lazy CSS

In previous sections we analyzed the performance of CSS and proved that (in the realizable case) it can achieve sharp coverage rates under reasonable assumptions on the source distribution while guaranteeing zero error on the accepted samples. However, it remains unclear whether an efficient implementation of CSS is at reach. In this section we propose an algorithm for CSS and show that it can be efficiently implemented for linear classifiers.

The following method, which we term *lazy CSS*, is very similar to the implicit selective sampling algorithm of Cohn et al. (1994). Instead of explicitly constructing the CSS selection function g during training (which indeed can be a very complex task), we adapt a "lazy learning" approach that can potentially facilitate an efficient CSS implementation during test time. In particular, we propose to evaluate $g(x)$ at any given test point x during the classification process. For the training set S_m and a test point x we define the following two sets:

$$S^+_{m,x} \triangleq S_m \cup \{(x,+1)\}, \quad S^-_{m,x} \triangleq S_m \cup \{(x,-1)\};$$

that is, $S^+_{m,x}$ is the (labeled) training set S_m augmented by the test point x labeled positively, and $S^-_{m,x}$ is S_m augmented by x labeled negatively. The selection value $g(x)$ is determined as follows: $g(x)=0$ (i.e., x is rejected) iff there exist hypotheses $f^+, f^- \in \mathcal{F}$ that are consistent with $S^+_{m,x}$ and $S^-_{m,x}$, respectively.

The following lemma states that the selection function $g(x)$ constructed by lazy CSS is a precise implementation of CSS.

Lemma 39 *Let $\mathcal{F}$ be any hypothesis class, S_m a labeled training set, and x, a test point. Then x belongs to the maximal agreement set of $VS_{\mathcal{F},S_m}$ iff there is no hypothesis $f \in \mathcal{F}$ that is consistent with either $S^+_{m,x}$ or $S^-_{m,x}$.*

Proof If there exist hypotheses $f^+, f^- \in \mathcal{F}$ that are consistent with $S^+_{m,x}$ and $S^-_{m,x}$, then there exist two hypotheses in $\mathcal{F}$ that correctly classify S_m (therefore they belong to $VS_{\mathcal{F},S_m}$) but disagree on x. Hence, x does not belong to the maximal agreement set of $VS_{\mathcal{F},S_m}$. Conversely, if x does not belong to the maximal agreement set of $VS_{\mathcal{F},S_m}$, then there are two hypotheses, f_1 and f_2, which correctly classify S_m but disagree on x. Let's assume, without loss of generality, that f_1 classifies x positively. Then, f_1 is consistent with $S^+_{m,x}$ and f_2 is consistent with $S^-_{m,x}$. Thus there exist hypotheses $f^+, f^- \in \mathcal{F}$ that are consistent with $S^+_{m,x}$ and $S^-_{m,x}$. ■

For the case of linear classifiers it follows that computing the lazy CSS selection function for any test point is reduced to two applications of a linear separability test. Yogananda et al. (2007) recently presented a fast linear separability test with a worst case time complexity of $O(mr^3)$ and space complexity of $O(md)$, where m is the number of points, d is the dimension and $r \leq \min(m, d+1)$.

Remark 40 *For the realizable case we can modify* any *rejection mechanism by restricting rejection* only *to the region chosen for rejection by CSS. Since CSS accepts only samples that are guaranteed to have zero test error, the overall performance of the modified rejection mechanism is guaranteed to be at least as good as the original mechanism. Using this technique we were able to improve the performance (RC curve) of the most commonly used rejection mechanism for linear classifiers, which rejects samples according to a simple symmetric distance from the decision boundary (a "margin").*

9. Which Rejection Model?

In classical classification and Bayes decision theory the goal is to minimize a cost function (or a loss function), where the cost is specified by a $K \times K$ cost matrix ($K = 2$ for the binary case). Given the cost matrix, the objective is to select a classifier that minimizes the average weighted cost (over unobserved instances) as specified by this matrix. When introducing rejection it is necessary to introduce a suitable optimization criterion (which is referred here also as a 'rejection model'). Obviously, the desired criterion should take into account both the risk of the classifier and its coverage. The question we discuss in this section is: what would be an appropriate optimization criterion for selective classification?

A very common rejection model in the literature is the *cost model*, whereby a specific cost d is associated with rejection (see, e.g., Tortorella, 2001) and the objective is to minimize the generalized *rejective risk function*,

$$\ell_c(f,g) \quad \triangleq \quad d \cdot \mathbf{E}\left[1 - g(X)\right] + \mathbf{E}\left[\mathbb{I}(f(X) \neq Y) \cdot g(X)\right]. \tag{8}$$

Given our definitions of risk and coverage, the function (8) can be easily expressed as a function over the RC plane of Figure 1,

$$\ell_c(R,\Phi) = d\,(1 - \Phi(f,g)) + R(f,g)\Phi(f,g). \tag{9}$$

For any fixed d, Equation (9) defines level sets (or elevation contour lines) over the RC plane. For example, Figure 3(a) depicts elevation contour lines induced by (9) with a rejection cost $d = 0.3$. The thick line in this figure represent our knowledge of the optimal RC trade-off. Thus, an optimal classifier, according to this cost model, has a risk-coverage profile that minimizes the cost (9) with respect to all choices on the RC trade-off curve. This optimal choice is depicted in Figure 3(a) by the black dot. This popular cost model was refined to accommodate differentiation between the cost of false positive and false negative as well as different costs for rejection of positive and negative samples (Herbei and Wegkamp, 2006; Pietraszek, 2005; Tortorella, 2001; Santos-Pereira and Pires, 2005). Such extensions or refinements are appealing because they allow for additional control and more flexibility in modeling the problem at hand. Nevertheless, these cost models are often criticized for lack of usability in applications where it is impossible or hard to precisely quantify the cost of rejection. It is interesting to note that for an ideal Bayesian setting, where the underlying distribution is completely known, Chow showed (Chow, 1970) that the cost d upper bounds the probability of misclassification. In this case one can control the classification error by specifying a matching rejection cost.

In Pietraszek (2005) two additional optimization models are introduced. The first, *bounded-improvement* model, is depicted as contour elevation lines over the RC plane in Figure 3(c). In this

model, given a constraint on the misclassification cost, the classifier should reject as few samples as possible. The constraint is specified by the ∞ symbol in the RC plane, which is the cost defined for the entire rectangle containing all risk-coverage profiles having risk larger than the constraint (0.3 in this example). In the second, *bounded-abstention* model (depicted in Figure 3(b)), given a constraint on the coverage (0.5 in this example), the classifier should have the lowest misclassification cost. It is argued in Pietraszek (2005) that these models are more suitable than the above cost model in many applications, for instance, when a classifier with limited classification throughput (e.g., a human expert) should handle the rejected instances, and in a medical and quality assurance applications, where the goal is to reduce the misclassification cost to a user-defined value.

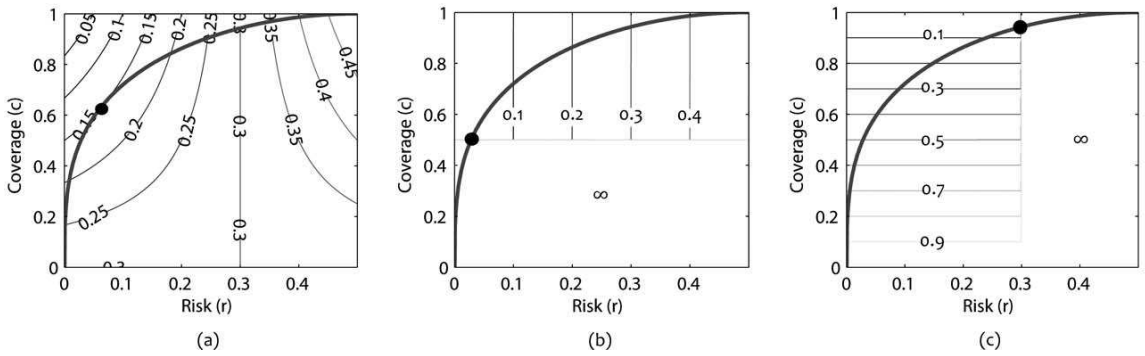


Figure 3: Rejection models: (a) cost (b) bounded-abstention (c) bounded-improvement

Which cost model among the above three is the right model? This question is, obviously, ill-defined and the answer depends on the application. Thus, when deriving bounds for a specific generalized rejective risk function the results are limited to only one specific model. Instead, one can handle *any* rejective risk function over the RC plane by identifying the RC trade-off. Specifically, by bounding the coverage and the risk separately (as we do in this paper) we can in principle optimize *any* generalized rejective risk function according to any desired rejection model including the cost, the bounded-improvement and bounded-abstention models.

10. Related Work

The idea of classification with a reject option dates back to Chow's seminal papers (Chow, 1957, 1970). These papers analyzed both the Bayes-optimal reject decision and the reject-rate vs. error trade-off. This is done under the 0-1 loss function, assuming that the underlying distribution is completely known. The Bayes-optimal rejection policy is based, as in standard classification, on maximum a posteriori probabilities. Instances should be rejected whenever none of the posteriori probabilities are sufficiently distinct. This type of rejection can be termed *ambiguity-based* rejection. Referring to the diagram in Figure 1, one of Chow's main results (for the case of complete probabilistic knowledge), is that the optimal RC trade-off (depicted by the dotted line) is monotonically increasing.

While the optimal decision can be identified in the case of complete probabilistic knowledge, it was argued (Fumera et al., 2000) that when the a posteriori probabilities are estimated with errors, Chow's rule (Chow, 1970) does not provide the optimal error-reject trade-off. Tortorella (2001) and Santos-Pereira and Pires (2005) discussed Bayesian-optimal decisions in the case of arbitrary cost

matrices. In these papers the optimal reject rule was chosen based on the ROC curve evaluated on a subset of the training data. As in most papers on the subject emerging from the engineering community (see, e.g., Fumera et al., 2000; Fumera and Roli, 2002; Pietraszek, 2005; Bounsiar et al., 2006; Landgrebe et al., 2006) no probabilistic or other guarantees are provided for the misclassification error.

Very few studies have focused on error bounds for classifiers with a reject option. Hellman (1970) proposed and analyzed two rejection rules for the nearest neighbor algorithm. Extending Cover and Hart's classic result for the 1-nearest neighbor algorithm (Cover and Hart, 1967), Hellman showed that the test error (over non-rejected points) of a nearest neighbor algorithm with a reject option can be bounded *asymptotically* (as the sample size approaches infinity) by some factor of the Bayes error (with reject). To the best of our knowledge, this excess risk bound is the first that has been introduced in the context of classification with a reject option.

Herbei and Wegkamp (2006) developed excess risk bounds for the classification with a reject option setting where the loss function is the 0-1 loss, extended such that the cost of each reject point is $0 \leq d \leq 1/2$ (cost model; see Section 9). This result generalizes the excess risk bounds of Tsybakov (2004) for standard binary classification without reject (which is equivalent to the case $d = 1/2$). The bound applies to any empirical error minimization technique. This result is further extended in Bartlett and Wegkamp (2007) and Wegkamp (2007) in various ways, including the use of the hinge loss function for efficient optimization. The main results of Herbei and Wegkamp (both for plug-in rules and empirical risk minimization) degenerate, in the realizable case, to a meaningless bound, where classification with a reject option is not guaranteed to be any better than classification without reject. These results are also limited only to the cost model (see discussion on Section 9). Saying that, we must also note that comparing bounds that were derived for the agnostic setting with our results can be misleading or "unfair" since the agnostic setting is much more difficult. The only purpose of this comparison is to clarify that the results here are not special cases of any of the currently known agnostic bounds.

Freund et al. (2004) studied a simple ensemble method for binary classification. Given an hypothesis class $\mathcal{F}$, the method outputs a weighted average of all the hypotheses in $\mathcal{F}$ such that the weight of each hypothesis exponentially depends on its individual training error. Their algorithm abstains from prediction whenever the weighted average of all individual predictions is inconclusive (i.e., sufficiently close to zero). Two regret bounds for this algorithm were derived. The first bounds the probability of error when the classifier decides not to reject. If ε is the error of the best hypothesis in $\mathcal{F}$, the error of the aggregating algorithm is bounded above (w.h.p.) by $2\varepsilon + O(\frac{1}{m^{1/2-\theta}})$, where $0 < \theta < 1/2$ is an hyperparameter. The authors also proved that for a sufficiently large training sample size, $m = \Omega((\sqrt{ln(1/\delta)ln(|\mathcal{F}|)})^{1/\theta})$, the probability that the algorithms will abstain from prediction is bounded above by $5\varepsilon + O(\frac{\ln|\mathcal{F}|}{\sqrt{m^{1/2-\theta}}})$. To the best of our knowledge, these bounds are the first to provide some guarantee for both the error of the classifier and the coverage. Therefore, these results are related to the bounded-improvement and bounded-abstention models (see Section 9). As was rightfully stated by the authors, the final aggregated hypothesis can significantly outperform the best base-hypothesis in $\mathcal{F}$ in some favorable situations. Unfortunately, the regret bound provided does not exploit these situations, as it is bounded by twice the generalization error of the best hypothesis. Referring to the diagram in Figure 1, The results of Freund et al. can be depicted as a curve in region *B* (thus characterizing some achievable zone). For the realizable case, the bounds of Freund et al. achieve much slower rates than those we derive in this paper.

Selective classification is related to selective sampling (Atlas et al., 1990). In selective sampling the learner sequentially processes unlabeled examples, and for each one determines whether or not to request a label. One of the earliest active learning algorithms for the realizable case (termed "mellow active learner") was proposed by Cohn et al. (1994). Their well motivated approach is to request labels only for samples that belong to the region of disagreement (the complement of our maximal agreement set). As mentioned in Section 8, this is very similar to our CSS. Hanneke studied the rate for which the region of disagreement collapses as the algorithm processes examples (Hanneke, 2007, 2009). He introduced the notion of *disagreement coefficient* and derived upper bounds on the label complexity in active learning expressed in terms of this coefficient. While his results capture the convergence rate of the region of disagreement as a function of the number of *label requests*, in selective classification we are interested in convergence rates as a function of the size of the *training set* (in selective sampling the number of labels does not necessarily match the number of samples). Specific disagreement coefficient values were recently derived for some interesting hypothesis classes including homogeneous linear classifiers in $\mathbb{R}^d$ under uniform data distribution (Hanneke, 2007) and linear classifiers in $\mathbb{R}^d$ under smooth data density bounded away from zero (Friedman, 2009). While coverage bounds and label complexity bounds cannot be directly compared, we conjecture that formal connections between these two settings exist because the disagreement region plays a key role in both. The precise relation between these two settings is yet to be discovered.

11. Concluding Remarks

Selective classification is well recognized as a very attractive technique for improving classification accuracy. In fact, it is among very few methods that can help in practical applications where sufficiently low error cannot be achieved in the standard model. Nevertheless, not enough is known about selective classification in order to harness its power in a controlled, optimal way, or to avoid its use in cases where it cannot sufficiently help.

In this work we made a first step toward a rigorous analysis of selective classification by revealing properties of the risk-coverage trade-off, which represents optimal selective classification. By focusing on the extreme case of perfect learning we were able to derive initial results for entire risk-coverage trade-offs.

Many interesting questions are left open. Among the most important open questions are the following. What would be an analogous concept to perfect learning in the fully agnostic (non-realizable) setting? What is the precise relation between selective classification and selective sampling? Is it possible to implement efficiently the CSS strategy and prove useful bounds for other natural hypothesis classes? Can selective classification be rigorously analyzed in transductive, semi-supervised or active settings? With respect to agnostic extensions, while it doesn't make much sense to talk about "perfect learning" in a noisy setting, it is meaningful and interesting to consider the analogous concept to regret (or excess risk) bounds. Here we could employ a selective strategy aiming at achieving the error rate of the best hypothesis in the class precisely (and perhaps with certainty).

Acknowledgments

We thank Charles Elkan for useful discussions and the anonymous referees for excellent comments.

Appendix A.

Lemma 41 *For $u > \sqrt{2}v > 0$,*

$$\frac{u-v}{u+v} \geq 1 - 4 \cdot \frac{v}{u}.$$

Proof

$$\frac{u-v}{u+v} = 1 - \frac{2v}{u+v} = 1 - 2v\frac{u-v}{u^2-v^2} \geq 1 - 2v\frac{u}{u^2-v^2}.$$

Since $u > \sqrt{2}v$, we have

$$u^2 - v^2 > \frac{u^2}{2}.$$

Applying to the previous inequality completes the proof. ■

Lemma 42 (Bernstein's inequality Hoeffding, 1963) *Let $X_1, \ldots, X_n$ be independent zero-mean random variables. Suppose that $|X_i| \leq M$ almost surely, for all i. Then for all positive t,*

$$\Pr\left(\sum_{i=1}^{n} X_i > t\right) \leq \exp\left\{-\frac{t^2/2}{\sum \mathbf{E}\left[X_j^2\right] + Mt/3}\right\}.$$

Lemma 43 (binomial tail inversion lower bound) *For $k > 0$ and $\delta \leq \frac{1}{2}$,*

$$\overline{Bin}(m,k,\delta) \geq \min\left(1, \frac{k}{2m} - \frac{4}{3m}\ln\frac{1}{1-\delta}\right).$$

Proof Let $Z_1, \ldots Z_m$ be independent Bernoulli random variables each with a success probability $0 \leq p \leq 1$. Setting $W_i \triangleq Z_i - p$,

$$\begin{aligned} Bin(m,k,p) &= \Pr_{Z_1,\ldots,Z_m \sim B(p)^m}\left(\sum_{i=1}^{m} Z_i \leq k\right) = 1 - \Pr\left(\sum_{i=1}^{m} Z_i > k\right) \\ &= 1 - \Pr\left(\sum_{i=1}^{m} W_i > k - mp\right). \end{aligned}$$

Clearly, $\mathbf{E}[W_i] = 0$, $|W_i| \leq 1$, and $\mathbf{E}\left[W_i^2\right] = p \cdot (1-p)^2 + (1-p) \cdot p^2 = p \cdot (1-p)$. Using Lemma 42 (Bernstein's inequality) we thus obtain,

$$Bin(m,k,p) \geq 1 - \exp\left(-\frac{(k-mp)^2/2}{mp(1-p) + (k-mp)/3}\right).$$

Since $(1-p) \leq 1$,

$$\begin{aligned}\frac{(k-mp)^2/2}{mp(1-p)+(k-mp)/3} &\geq \frac{(k-mp)^2}{2mp+\frac{2}{3}\cdot(k-mp)} = \frac{(k-mp)^2}{\frac{4}{3}mp+\frac{2}{3}\cdot k} \\ &\geq \frac{k^2-2mpk}{\frac{4}{3}mp+\frac{2}{3}\cdot k}.\end{aligned}$$

Therefore,

$$Bin(m,k,p) \geq 1-e^{-\frac{k^2-2mpk}{\frac{4}{3}mp+\frac{2}{3}\cdot k}}.$$

Equating the right-hand side to δ and solving for p, we have

$$p \leq \frac{k}{2m}\cdot\frac{k-\frac{2}{3}t}{k+\frac{2}{3}\cdot t},$$

where $t \triangleq \ln\frac{1}{(1-\delta)}$. Choosing

$$p = \min\left\{1, \frac{1}{2m}\left(k-\frac{8}{3}\ln\frac{1}{1-\delta}\right)\right\} = \min\left\{1, \frac{k}{2m}\cdot\left(1-4\cdot\frac{\frac{2}{3}t}{k}\right)\right\},$$

using the fact that $k \geq 1 > \frac{2\sqrt{2}}{3}\ln\frac{1}{1/2} \geq \frac{2\sqrt{2}}{3}t$, and applying Lemma 41, we get

$$p \leq \frac{k}{2m}\cdot\frac{k-\frac{2}{3}t}{k+\frac{2}{3}\cdot t}.$$

Therefore, $Bin(m,k,p) \geq \delta$. Since $\overline{Bin}(m,k,\delta) = \max_p\{p : Bin(m,k,p) \geq \delta\}$, we conclude that

$$\overline{Bin}(m,k,\delta) \geq p = \min\left(1, \frac{k}{2m} - \frac{4}{3m}\ln\frac{1}{1-\delta}\right).$$

■

Lemma 44 *Let S_1 and S_2 be two sets in $\mathbb{R}^d$. Then,*

$$H(S_1 \cup S_2) \leq H(S_1) + H(S_2),$$

where $H(S)$ is the number of convex hull vertices of S.

Proof Assume $x \in S_1 \cup S_2$ is a convex hull vertex of $S_1 \cup S_2$. Then, there is a half-space (w,ϕ) such that, $w\cdot x - \phi = 0$, and any other $y \in S_1 \cup S_2$ satisfies $w\cdot y - \phi > 0$. Assume w.l.o.g. that $x \in S_1$. Then it is clear that any $y \in S_1$ satisfies $w\cdot y - \phi > 0$. Therefore, x is a convex hull vertex of S_1. ■

Proof of Lemma 22 Let $S^+ \subseteq S_m$ be the set of all positive samples in S_m, and $S^- \subseteq S_m$ be the set of all negative samples. Let $\bar{x}_0 \in R^+$. There exists a hypothesis $f_{\bar{w},\phi}(\bar{x})$ such that

$$\begin{aligned}&\forall \quad \bar{x} \in S^+, \quad \bar{w}^T\bar{x} - \phi \geq 0; \\ &\forall \quad \bar{x} \in S^-, \quad \bar{w}^T\bar{x} - \phi < 0,\end{aligned}$$

and

$$\bar{w}^T \bar{x}_0 - \phi \geq 0.$$

Let's assume that $\bar{x}_0 \notin \tilde{R}^+$. Then, there exists a hypothesis $\tilde{f}_{\bar{w}',\phi'}(\bar{x})$ such that

$$\begin{aligned} \forall \quad \bar{x} \in S^+, \quad & \bar{w}'^T \bar{x} - \phi' \geq 0; \\ \forall \quad \bar{x} \in S^-, \quad & \bar{w}'^T \bar{x} - \phi' \leq 0, \end{aligned}$$

and

$$\bar{w}'^T \bar{x}_0 - \phi' < 0.$$

Defining

$$\bar{w}_0 \triangleq \bar{w} + \alpha \bar{w}', \qquad \phi_0 \triangleq \phi + \alpha \phi',$$

where

$$\alpha > \left| \frac{\bar{w}^T \bar{x}_0 - \phi}{\bar{w}'^T \bar{x}_0 - \phi'} \right|,$$

we deduce that there exists a hypothesis $f_{\bar{w}_0,\phi_0}(\bar{x})$ such that

$$\begin{aligned} \forall \quad \bar{x} \in S^+, \quad & \bar{w}_0^T \bar{x} - \phi_0 \geq 0; \\ \forall \quad \bar{x} \in S^-, \quad & \bar{w}_0^T \bar{x} - \phi_0 < 0, \end{aligned}$$

and

$$\begin{aligned} \bar{w}_0^T \bar{x}_0 - \phi_0 &= \bar{w}^T \bar{x}_0 - \phi + \alpha \left[\bar{w}'^T \bar{x}_0 - \phi' \right] = \bar{w}^T \bar{x}_0 - \phi - \alpha \left| \bar{w}'^T \bar{x}_0 - \phi' \right| \\ &< \bar{w}^T \bar{x}_0 - \phi - \left| \bar{w}^T \bar{x}_0 - \phi \right| = 0. \end{aligned}$$

Therefore, $\bar{x}_0 \notin R^+$. Contradiction. Hence, $\bar{x}_0 \in \tilde{R}^+$ and $R^+ \subseteq \tilde{R}^+$. The proof that $R^- \subseteq \tilde{R}^-$ follows the same argument.

To prove that $\tilde{R}^+ \subseteq R^+$, we look at $VS_{\tilde{\mathcal{F}},S_m}$:

$$\forall \tilde{f}_{\bar{w},\phi} \in VS_{\tilde{\mathcal{F}},S_m}, \bar{x} \in \tilde{R}^+ \quad \bar{w}^T \bar{x} - \phi \geq 0.$$

We observe that if $f_{\bar{w},\phi} \in VS_{\mathcal{F},S_m}$, then $\tilde{f}_{\bar{w},\phi} \in VS_{\tilde{\mathcal{F}},S_m}$. Therefore,

$$\forall f_{\bar{w},\phi} \in VS_{\mathcal{F},S_m}, \bar{x} \in \tilde{R}^+ \quad \bar{w}^T \bar{x} - \phi \geq 0.$$

Hence, $\tilde{R}^+ \subseteq R^+$.

It remains to prove that $\tilde{R}^- \subseteq R^-$. Assuming $\bar{x}_0 \notin R^-$ implies that there exists a hypothesis $f_{\bar{w},\phi}(\bar{x})$ such that

$$\begin{aligned} \forall \quad \bar{x} \in S^+, \quad & \bar{w}^T \bar{x} - \phi \geq 0; \\ \forall \quad \bar{x} \in S^-, \quad & \bar{w}^T \bar{x} - \phi < 0, \end{aligned}$$

and

$$\bar{w}^T \bar{x}_0 - \phi \geq 0.$$

Defining[4]

$$\bar{w}_0 \triangleq \bar{w}, \quad \phi_0 \triangleq \phi - \left| \max_{\bar{x} \in S^-} \left(\bar{w}^T \bar{x} - \phi \right) \right|,$$

we conclude that there exists a hypothesis $\tilde{f}_{\bar{w}_0,\phi_0}(\bar{x})$ such that

$$\begin{aligned} &\forall \quad \bar{x} \in S^+ \quad \bar{w}_0^T \bar{x} - \phi_0 \geq 0; \\ &\forall \quad \bar{x} \in S^- \quad \bar{w}_0^T \bar{x} - \phi_0 \leq \max_{\bar{x} \in S^-} \left(\bar{w}^T \bar{x} - \phi \right) + \left| \max_{\bar{x} \in S^-} \left(\bar{w}^T \bar{x} - \phi \right) \right| = 0, \end{aligned}$$

and

$$\bar{w}_0^T \bar{x}_0 - \phi_0 > 0.$$

Therefore, $\bar{x}_0 \notin \tilde{R}^-$, so $\tilde{R}^- \subseteq R^-$. ■

Proof of Lemma 23 According to Lemma 22, $R^+ = \tilde{R}^+$ and $R^- = \tilde{R}^-$. Therefore, we can restrict our discussion to the hypothesis class $\tilde{\mathcal{F}}$. Due to the symmetry of the hypothesis class $\tilde{\mathcal{F}}$ we will concentrate only on the positive region R^+. Set $G \triangleq VS_{\tilde{\mathcal{F}},S_m}$. By definition,

$$\tilde{R}^+ = \bigcap_{f'_{\bar{w},\phi} \in G} f'_{\bar{w},\phi},$$

where $f'_{\bar{w},\phi}$ denotes the region in $\mathcal{X}$ for which the linear classifier $f'_{\bar{w},\phi}$ obtains the value one or zero. Let $f_{\bar{w},\phi} \in G$ be a half-space with $k < d$ points on its boundary. We will prove that there exist two half-spaces in G ($f_{\bar{w}_1,\phi_1}, f_{\bar{w}_2,\phi_2}$) such that each has at least $k+1$ samples on its boundary and

$$f_{\bar{w},\phi} \bigcap f_{\bar{w}_1,\phi_1} \bigcap f_{\bar{w}_2,\phi_2} = f_{\bar{w}_1,\phi_1} \bigcap f_{\bar{w}_2,\phi_2}.$$

Therefore,

$$\tilde{R}^+ = \bigcap_{f'_{\bar{w},\phi} \in G \setminus \{f_{\bar{w},\phi}\}} f'_{\bar{w},\phi}.$$

Repeating this process recursively with every half-space in G, with less than d points on its boundary, completes the proof.

Before proceeding with the rigorous analysis let's review the main idea behind the proof. If a half-space in $\mathbb{R}^d$ has less than d points on its boundary, it has at least one degree of freedom. Rotating the half-space clockwise or counterclockwise around a specific axis (defined by the points on the boundary) by sufficiently small angles will maintain correct classification over S_m. We will rotate the half-space clockwise and counterclockwise until "touching" the first point in S_m on each direction. This operation will maintain correct classification but will result in having one additional point on the boundary. Then we only have to show that the intersection of the three half-spaces (original and two rotated ones) is the same as the intersection of the two rotated ones.

4. If S^- is an empty set we can arbitrarily define $\phi_0 \triangleq \phi - 1$.

Let $f_{\bar{w},\phi} \in G$ be a half-space with $k < d$ points on its boundary. Without loss of generality assume that these points are $S_m^0 \triangleq \{\bar{x}_1, \bar{x}_2, ..., \bar{x}_k\}$. For the sake of simplicity we will first *translate* the space such that $\bar{x}_1$ will lie on the origin. Since $\bar{x}_1$ is on the boundary of the half-space we get

$$\bar{w}^T \bar{x}_1 - \phi = 0 \quad \Longrightarrow \quad \phi = \bar{w}^T \bar{x}_1.$$

Therefore,

$$\forall \bar{x} \in S_m^0 \quad 0 = \bar{w}^T \bar{x} - \phi = \bar{w}^T \bar{x} - \bar{w}^T \bar{x}_1 = \bar{w}^T (\bar{x} - \bar{x}_1).$$

Hence, the weight vector $\bar{w}$ is orthogonal to all the translated samples $(\bar{x}_1 - \bar{x}_1), \ldots, (\bar{x}_k - \bar{x}_1)$. We now have $k < d$ vectors in $\mathbb{R}^d$ (including the weight vector) so we can always find at least one vector $\bar{v}$ which is orthogonal to all the rest. We now *rotate* the translated samples around the origin so as to align the vector $\bar{w}$ with the first axis, and align the vector $\bar{v}$ with the second axis. From now on all translated and rotated coordinates and vectors will be marked with prime. Define the following rotation matrix in $\mathbb{R}^d$,

$$R_\theta \triangleq \begin{pmatrix} \cos\theta & \sin\theta & 0 & 0 & \ldots \\ -\sin\theta & \cos\theta & 0 & 0 & \ldots \\ 0 & 0 & 1 & 0 & \ldots \\ 0 & 0 & 0 & 1 & \\ \vdots & \vdots & \vdots & & \ddots \end{pmatrix}.$$

We can now define two new half-spaces in the translated and rotated space, $f_{R_\alpha \bar{w}',0}$ and $f_{R_{-\beta} \bar{w}',0}$, where

$$\alpha = \max_{0 < \alpha' \leq \pi} \{\alpha' \quad | \quad \forall \bar{x}' \in S_m \quad (\bar{w}'^T \bar{x}') \cdot (R_{\alpha'} \bar{w}')^T \bar{x}' \geq 0\}, \tag{10}$$

and

$$\beta = \max_{0 < \beta' \leq \pi} \{\beta' \quad | \quad \forall \bar{x}' \in S_m \quad (\bar{w}'^T \bar{x}') \cdot (R_{-\beta'} \bar{w}')^T \bar{x}' \geq 0\}. \tag{11}$$

According to Claim 45, both $f_{R_\alpha \bar{w}',0}$ and $f_{R_{-\beta} \bar{w}',0}$ correctly classify S_m and have at least $k+1$ samples on their boundaries.

Now we examine the intersection of $f_{R_\alpha \bar{w}',0}$ and $f_{R_{-\beta} \bar{w}',0}$. According to Claim 46, if $(R_\alpha \bar{w}')^T \bar{x}' \geq 0$ and $(R_{-\beta} \bar{w}')^T \bar{x}' \geq 0$, then $\bar{w}'^T \bar{x}' \geq 0$. The intersection of $f_{R_\alpha \bar{w}',0}$, $f_{R_{-\beta} \bar{w}',0}$ and $f_{\bar{w}',0}$ thus equals the intersection of $f_{R_\alpha \bar{w}',0}$ and $f_{R_{-\beta} \bar{w}',0}$, as required. ■

Claim 45 *Both $f_{R_\alpha \bar{w}',0}$ and $f_{R_{-\beta} \bar{w}',0}$ correctly classify S_m and have at least $k+1$ samples on their boundaries.*

Proof We note that after translation all half-spaces pass through the origin, so $\phi' = 0$. Recall the definitions of α and β as maximums (Equations (10) and (11), respectively). We show that the maximums over α' and β' are well defined. Let $\bar{x}' = (x'_1, x'_2, \cdots x'_d)^T$. Since $\bar{w}' = (1, 0, 0, \cdots)^T$ we get that $R_\alpha \bar{w}' = (\cos\alpha, -\sin\alpha, 0, \ldots)^T$ and

$$(\bar{w}'^T \bar{x}') \cdot (R_{\alpha'} \bar{w}')^T \bar{x}' = {x'_1}^2 \cos\alpha - x'_1 \cdot x'_2 \cdot \sin\alpha.$$

Since S_m is a spanning set of $\mathbb{R}^d$, at least one sample has a vector with component $x'_1 \neq 0$. As all components are finite and ${x'_1}^2 > 0$, we can always find a sufficiently small α' such that ${x'_1}^2 \cos\alpha' -$

$x'_1 \cdot x'_2 \cdot \sin\alpha' > 0$. Hence, the maximum exists. Furthermore, for $\alpha' = \pi$ we have ${x'_1}^2 \cos\pi - x'_1 \cdot x'_2 \cdot \sin\pi = -{x'_1}^2 < 0$. Noticing that ${x'_1}^2 \cos\alpha - x'_1 \cdot x'_2 \cdot \sin\alpha$ is continuous in α, and applying the intermediate value theorem, we know that $0 < \alpha < \pi$ and ${x'_1}^2 \cos\alpha - x'_1 \cdot x'_2 \cdot \sin\alpha = 0$. Therefore, there exists a sample in S_m that is not on the boundary of $f_{\bar{w}',0}$ (since $x'_1 \neq 0$) but on the boundary of $f_{R_\alpha \bar{w}',0}$. Recall that all points in S_m^0 are orthogonal to $\bar{w}' = (1,0,0,\cdots)^T$ and $\bar{v} = (0,1,0,\cdots)^T$. Therefore,

$$\forall \bar{x}' \in S_m^0 \quad (R_\alpha \bar{w}')^T \bar{x}' = x'_1 \cdot \cos\alpha - x'_2 \cdot \sin\alpha = \bar{w}'^T \bar{x} \cdot \cos\alpha - \bar{v}'^T \bar{x} \cdot \sin\alpha = 0,$$

and they reside on the boundary of $f_{R_\alpha \bar{w}',0}$. Overall, $f_{R_\alpha \bar{w}',0}$ correctly classifies S_m and has at least $k+1$ samples on its boundary. The same argument applies for β by symmetry. ■

Claim 46 *Using the notation introduced in the proof of Lemma 23, if* $(R_\alpha \bar{w}')^T \bar{x}' \geq 0$ *and* $(R_{-\beta} \bar{w}')^T \bar{x}' \geq 0$*, then*

$$\bar{w}'^T \bar{x}' \geq 0.$$

Proof If $(R_\alpha \bar{w}')^T \bar{x}' \geq 0$ and $(R_{-\beta} \bar{w}')^T \bar{x}' \geq 0$, then

$$\begin{cases} x'_1 \cos\alpha - x'_2 \cdot \sin\alpha \geq 0, \\ x'_1 \cos\beta + x'_2 \cdot \sin\beta \geq 0. \end{cases}$$

Multiplying the first inequality by $\sin\beta > 0$, the second inequality by $\sin\alpha > 0$, and adding the two we have

$$\sin(\alpha+\beta) \cdot x'_1 \geq 0.$$

According to Claim 47 below, $\sin(\alpha+\beta) \geq 0$. If $\sin(\alpha+\beta) = 0$, then $(\alpha+\beta) = \pi$ and $\cos(\alpha+\beta) = -1$. Using the trigonometric identities

$$\begin{aligned} \cos(\alpha-\beta) &= \cos\alpha\cos\beta + \sin\alpha\sin\beta; \\ \sin(\alpha-\beta) &= \sin\alpha\cos\beta - \cos\alpha\sin\beta, \end{aligned}$$

we get that

$$\cos\beta = \cos(\beta+\alpha-\alpha) = \cos(\alpha+\beta) \cdot \cos\alpha + \sin(\alpha+\beta) \cdot \sin\alpha = -\cos\alpha,$$

and

$$\sin\beta = \sin(\beta+\alpha-\alpha) = \sin(\alpha+\beta) \cdot \cos\alpha - \cos(\alpha+\beta) \cdot \sin\alpha = \sin\alpha.$$

Therefore, for any $x'_1 \cos\alpha - x'_2 \cdot \sin\alpha > 0$, it holds that $x'_1 \cos\beta + x'_2 \cdot \sin\beta < 0$ and $\tilde{R}^+$ is degenerated. Contradiction to the fact that S_m is a spanning set of the $\mathbb{R}^d$. Therefore, $\sin(\alpha+\beta) > 0$, $x'_1 \geq 0$ and $\bar{w}'^T \bar{x}' \geq 0$. ■

Claim 47 *Using the notation introduced in the proof of Lemma 23,*

$$\sin(\alpha+\beta) \geq 0.$$

Proof By definition we get that for all samples in S_m,

$$\begin{cases} {x'_1}^2 \cos\alpha - x'_1 \cdot x'_2 \cdot \sin\alpha \geq 0, \\ {x'_1}^2 \cos\beta + x'_1 \cdot x'_2 \cdot \sin\beta \geq 0. \end{cases}$$

Multiplying the first inequality by $\sin\beta > 0$ ($0 < \beta < \pi$), the second inequality by $\sin\alpha > 0$, adding the two, and using the trigonometric identity

$$\sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta,$$

we have

$$\sin(\alpha+\beta) \cdot {x'_1}^2 \geq 0.$$

Since there is a sample in S_m with a vector component $x'_1 \neq 0$, we conclude that $\sin(\alpha+\beta) \geq 0$. ■

References

M. Anthony and P.L. Bartlett. *Neural Network Learning; Theoretical Foundations*. Cambridge University Press, 1999.

A. Antos and G. Lugosi. Strong minimax lower bounds for learning. *Machine Learning*, 30(1): 31–56, 1998.

L. Atlas, D. Cohn, R. Ladner, A.M. El-Sharkawi, and R.J. Marks. Training connectionist networks with queries and selective sampling. In *Neural Information Processing Systems (NIPS)*, pages 566–573, 1990.

P.L. Bartlett and M.H. Wegkamp. Classification with a reject option using a hinge loss. Technical report M980, Department of Statistics, Florida State University, 2007.

J.L. Bentley, H.T. Kung, M. Schkolnick, and C.D. Thompson. On the average number of maxima in a set of vectors and applications. *Journal of the ACM*, 25, 1978.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M.K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36, 1989.

A. Bounsiar, E. Grall, and P. Beauseroy. A kernel based rejection method for supervised classification. *International Journal of Computational Intelligence*, 3:312–321, 2006.

C.K. Chow. An optimum character recognition system using decision function. *IEEE Transactions on Computers*, 6(4):247–254, 1957.

C.K. Chow. On optimum recognition error and reject trade-off. *IEEE Transactions on Information Theory*, 16:41–36, 1970.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

T.M. Cover and P. Hart. Neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

Y. Freund, Y. Mansour, and R.E. Schapire. Generalization bounds for averaged classifiers. *Annals of Statistics*, 32(4):1698–1722, 2004.

E. Friedman. Active learning for smooth problems. *In Proceedings of the* 22^{nd} *Annual Conference on Learning Theory*, 2009.

G. Fumera and F. Roli. Support vector machines with embedded reject option. In *Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines*, pages 811–919, 2002.

G. Fumera, F. Roli, and G. Giacinto. Multiple reject thresholds for improving classification reliability. In *Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*, pages 863–871, 2000.

B. Hanczar and E.R. Dougherty. Classification with reject option in gene expression data. *Bioinformatics*, 24(17):1889–1895, 2008.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 353–360, 2007.

S. Hanneke. *Theoretical Foundations of Active Learning*. PhD thesis, Carnegie Mellon University, 2009.

D. Haussler. Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 36:177–221, 1988.

M.E. Hellman. The nearest neighbor classification rule with a reject option. *IEEE Transactions on Systems, Man and Cybernetics*, 6:179–185, 1970.

R. Herbei and M.H. Wegkamp. Classification with reject option. *The Canadian Journal of Statistics*, 34(4):709–721, 2006.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.

T.C.W. Landgrebe, D.M.J. Tax, P. Paclík, and R.P.W. Duin. The interaction between classification and reject performance for distance-based reject-option classifiers. *Pattern Recognition Letters*, 27(8):908–917, 2006.

J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6:273–306, 2005.

P. S. Meltzer, J. Khan, J. S. Wei, M Ringnér, L. H. Saal, M Ladanyi, F Westermann, F Berthold, M Schwab, C. R. Antonescu, and C Peterson. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6), June 2001.

T. Mitchell. Version spaces: a candidate elimination approach to rule learning. In *IJCAI'77: Proceedings of the 5th international joint conference on Artificial Intelligence*, pages 305–310, 1977.

T. Pietraszek. Optimizing abstaining classifiers using ROC analysis. In *Proceedings of the Twenty-Second International Conference on Machine Learning(ICML)*, pages 665–672, 2005.

F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1990.

C.M. Santos-Pereira and A.M. Pires. On optimal reject rules and ROC curves. *Pattern Recognition Letters*, 26(7):943–952, 2005.

F. Tortorella. An optimal reject rule for binary classifiers. *Lecture Notes in Computer Science*, 1876: 611–620, 2001.

A.B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32(1): 135–166, 2004.

V. Vapnik. *Statistical Learning Theory*. Wiley Interscience, New York, 1998.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

M.H. Wegkamp. Lasso type classifiers with a reject option. *Electronic Journal of Statistics*, 1: 155–168, 2007.

A.P. Yogananda, M.N. Murthy, and G. Lakshmi. A fast linear separability test by projection of positive points on subspaces. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 713–720, 2007.

Journal of Machine Learning Research 11 (2010) 1643-1662 Submitted 2/10; Published 5/10

Introduction to Causal Inference

Peter Spirtes PS7Z@ANDREW.CMU.EDU
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Editor: Lawrence Saul

Abstract

The goal of many sciences is to understand the mechanisms by which variables came to take on the values they have (that is, to find a generative model), and to predict what the values of those variables would be if the naturally occurring mechanisms were subject to outside manipulations. The past 30 years has seen a number of conceptual developments that are partial solutions to the problem of causal inference from observational sample data or a mixture of observational sample and experimental data, particularly in the area of graphical causal modeling. However, in many domains, problems such as the large numbers of variables, small samples sizes, and possible presence of unmeasured causes, remain serious impediments to practical applications of these developments. The articles in the Special Topic on Causality address these and other problems in applying graphical causal modeling algorithms. This introduction to the Special Topic on Causality provides a brief introduction to graphical causal modeling, places the articles in a broader context, and describes the differences between causal inference and ordinary machine learning classification and prediction problems.

Keywords: Bayesian networks, causation, causal inference

1. Introduction

The goal of many sciences is to understand the mechanisms by which variables came to take on the values they have (that is, to find a generative model), and to predict what the values of those variables would be if the naturally occurring mechanisms were subject to outside manipulations. For example, a randomized experiment is one kind of manipulation that substitutes the outcome of a randomizing device to set the value of a variable (for example, whether or not a particular new medication is given to a patient who has agreed to participate in a drug trial) in place of the naturally occurring mechanism that determines the variable's value. In non-experimental settings, biologists gather data about the gene activation levels in normally functioning systems in order to understand which genes affect the activation levels of which other genes, and to predict what the effects of manipulating the system to turn some genes on or off would be. Epidemiologists gather data about dietary habits and life expectancy in the general population and seek to find what dietary factors affect life expectancy and predict the effects of advising people to change their diets. Finding answers to questions about the mechanisms by which variables come to take on values, or predicting the value of a variable after some other variable has been manipulated, is characteristic of causal inference. If only non-experimental data are available, predicting the effects of manipulations typically involves drawing samples from one probability density (in the unmanipulated population)

and making inferences about the values of a variable in a population that has a different probability density (in the manipulated population).

The rapid spread of interest in the last three decades in principled methods of search or estimation of causal relations has been driven in part by technological developments, especially the changing nature of modern data collection and storage techniques, and the increases in the processing power and storage capacities of computers. Statistics books from 30 years ago often presented examples with fewer than 10 variables, in domains where some background knowledge was plausible. In contrast, in new domains such as climate research (where satellite data now provide daily quantities of data unthinkable a few decades ago), fMRI brain imaging, and microarray measurements of gene expression, the number of variables can range into the tens of thousands, and there is often limited background knowledge to reduce the space of alternative causal hypotheses. Even when experimental interventions are possible, performing the many thousands of experiments that would be required to discover causal relationships between thousands or tens of thousands of variables is often not practical. In such domains, non-automated causal discovery techniques from sample data, or sample data together with a limited number of experiments, appears to be hopeless, while the availability of computers with increased processing power and storage capacity allow for the practical implementation of computationally intensive automated search algorithms over large search spaces.

The past 30 years has also seen a number of conceptual developments that are partial solutions to these causal inference problems, particularly in the area of graphical causal modeling. Sections 3 and 4 of this paper describe some of these developments: a variety of well defined mathematical objects to represent causal relations (for example, directed acyclic graphs); well defined connections between aspects of these objects and sample data (for example, the Causal Markov and Causal Faithfulness Assumptions); ways to compute those connections (for example, d-separation); and a theory of representation and calculation of the effects of manipulations (for example, by breaking edges in a graph); and search algorithms (for example, the PC algorithm). However, in many domains, problems such as the large numbers of variables, small samples sizes, and possible presence of unmeasured causes, remain serious impediments to practical applications of these developments.

The articles in the Special Topic on Causality (containing articles from 2007 to 2009) address these and other problems in making causal inferences. Although there are some superficial similarities between traditional supervised machine learning problems and causal inference (for example, both employ model search and feature selection, the kinds of models employed overlap, some model scores can be used for both purposes), these similarities can mask some very important differences between the two kinds of problems. This introduction to the Special Topic on Causality provides a brief introduction to graphical causal modeling, places the articles in a broader context, and describes the differences between causal inference and ordinary machine learning classification or prediction problems; it is not intended to provide a broad overview or a tutorial surveying all methods of causal inference.

Section 2 describes the problem of causal inference in more detail, and differentiates it from the typical machine learning supervised classification or prediction problem; Section 3 describes several different kinds of causal models; Section 4 describes some problems associated with search for causal models, and why algorithms appropriate for the discovery of good classification or prediction models in machine learning are not always appropriate for the discovery of good causal models; and Section 5 describes some major open problems in the field. The various articles in the Special Topic on Causality are described throughout this article, depending upon which topic they address.

2. Manipulating Versus Conditioning

This section will describe three different kinds of problems (one typical machine learning or statistical problem, and two kinds of causal problems), and three different kinds of probability densities (conditional, manipulated, and counterfactual) that are useful for solving the problems.

2.1 Conditional Probabilities

Suppose that there is a population of individuals with the following random variables at time t: rw_t is the average number of glasses of red wine consumed per day in the 5 years prior to t, bmi_t is the body mass index of a person at time t, sex_t is the person's sex (0 = male, 1 = female) at time t, and ha_t is whether or not an individual had a heart attack in the 5 years prior to t. Since sex_t is rarely time-dependent, it will be replaced simply by sex.

Suppose an insurance company at time t wants to determine what rates to charge an individual for health insurance who has $rw_t = 1$, $bmi_t = 25$, and $sex = 0$, and that this rate is partly based on the probability of the individual having a heart attack in the next 5 years. This can be estimated by using the rate of heart attacks among the subpopulation matching the subject, that is $rw_t = 1$, $bmi_t = 25$, $sex = 0$. It is impossible to measure the values of ha_{t+5} at time t, because they haven't occurred yet, but if the probability density is stable across time, the density of ha_{t+5} among the subset of the population with $rw_t = 1$, $bmi_t = 25$, and $sex = 0$ will be the same as the density of ha_t among the subpopulation for which $rw_{t-5} = 1$, $bmi_{t-5} = 25$, and $sex = 0$. The density in a subpopulation is a conditional density, in this case $P(ha_t \mid rw_{t-5} = 1, bmi_{t-5} = 25, sex = 0)$.

Conditioning maps a given joint density, and a given subpopulation (typically specified by a set of values for random variables) into a new density. The conditional density is a function of the joint density over the random variables, and a set of values for a set of random variables.[1] The estimation of a conditional probability is often non-trivial because the number of people with $rw_{t-5} = 1$, $bmi_{t-5} = 25$, $sex = 0$ might be small. A large part of statistics and machine learning is devoted to estimating conditional probabilities from realistic sample sizes under a variety of assumptions.

If the insurance company is not attempting to change anyone's behavior then the question of whether drinking the right amount of red wine *prevents* heart attacks is irrelevant to their concerns; the only relevant question is whether the amount of red wine that someone drinks *predicts* heart attack rates. It is possible that people who drink an average of between 1 and 2 glasses of red wine per day for 5 years have lowered rates of heart attacks because of socio-economic factors that both cause average daily consumption of red wine and other life-style factors that prevent heart attacks. But even if moderate red wine consumption does not prevent heart attacks, the insurance company can still use the conditional probability to help determine the rates to charge.

If $\mathbf{X}$ is a set of measured variables, the conditional probability density $P(\mathbf{Y} \mid \mathbf{X})$ is not only useful for predicting future values of $\mathbf{Y}$, it is also useful for predicting current unmeasured values of $\mathbf{Y}$, and for classifying individuals in cases where $\mathbf{Y}$ is categorical.

Problem 1: Predictive Modeling

Input: Samples from a density $P(\mathbf{O})$ (where $\mathbf{O}$ is a set of observed random variables), and two sets of variables $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{O}$.

Output: A consistent, efficient estimate of $P(\mathbf{Y} \mid \mathbf{X})$.

1. In order to avoid technicalities, I will assume that the set of values conditioned on do not have measure 0.

2.2 Manipulated Probabilities

In contrast to the previous case, suppose that an epidemiologist is deciding whether or not to recommend providing very strong incentives for adults to drink an average of 1 to 2 glasses of red wine per day in order to prevent heart attacks. Suppose further that if adopted the incentives will be very widely effective. The density of heart attacks observationally *conditional* on drinking an average of 1 to 2 glasses of red wine per day is not the density relevant to answering this question, and the question of whether drinking red wine prevents heart attacks is crucial. Suppose drinking red wine does not prevent heart attacks, but the heart attack rate is lower among moderate red wine drinkers because some socio-economic variable causes both moderate red wine drinking and other healthy life-styles choices that prevent heart attacks. In that case, after the incentives to drink red wine are in place, the density of socioeconomic status among red wine drinkers will be different than prior to the incentives, and the conditional density of heart attacks among moderate red wine drinkers will not be the same after the incentives were adopted as prior to their adoption. Thus, using observational conditional densities to predict heart attacks after the incentives are in place will lead to incorrect predictions.

The density that is relevant to determining whether or not to recommend drinking a moderate amount of red wine is not the density of heart attacks among people who have chosen to drink red wine (choice being the mechanism for determining red wine consumption in the unmanipulated population), but the density of heart attacks among people who would drink red wine after the incentives are in place. If the incentives are very effective, the density of heart attacks among people who would drink red wine after the incentives are in place is approximately equal to the density of heart attacks among people who are assigned to drink moderate amounts of red wine in an experimental study.

The density of heart attacks among people who have been *assigned* to drink red wine (as opposed to those who have *chosen* to drink red wine, as is currently the case) is a *manipulated* density, that results from taking action on a given population - it may or may not be equal to any observational conditional density, depending upon what the causal relations between variables are. Manipulated probability densities are the appropriate probability densities to use when making predictions about the effects of taking actions ("manipulating" or "doing") on a given population (for example, assigning red wine drinking), rather than observing ("seeing") the values of given variables. Manipulated probabilities are the probabilities that are implicitly used in decision theory, where the different actions under consideration are manipulations.[2]

A simple form of manipulation specifies what new density P' is assigned to some variable in a population at a given time. For example, forcing everyone in an (adult) population to drink an average of 1 glass of red wine daily from t–10 to t–5, assigns $P'(rw_{t-5} = 1) = 1$. (Since rw_{t-5} measures red wine drinking for the past 5 years, an intervention on rw_{t-5} begins at t–10.) After this density has been assigned, there is a resulting joint density for the random variables at time t, denoted by $P(sex, bmi_{t-5}, ha_{t-5}, rw_{t-5}, bmi_t, ha_t, rw_t \,||\, P'(rw_{t-5} = 1) = 1)$, where the double bar indicates the density that has been assigned to rw_{t-5}, in this case that everyone has been assigned the value $rw_{t-5} = 1$.[3] This is in contrast to the conditional density $P(sex, bmi_{t-5}, ha_{t-5}, rw_{t-5}, bmi_t, ha_t,$

2. The use of manipulated probability densities in decision theory is often not explicit. The assumption that the density of states of nature are independent of the actions taken (act-state independence) is one way to ensure that the manipulated densities that are needed are equal to observed conditional densities that can be measured.
3. There is no completely standard notation for denoting a manipulated density. This notation is adapted from Lauritzen (1999).

$rw_t \mid rw_{t-5} = 1)$, which is the density of the variables in the subpopulation where $rw_{t-5} = 1$ because people have been observed to drink that amount of red wine, as in the unmanipulated population.

$P(sex, bmi_{t-5}, ha_{t-5}, rw_{t-5}, bmi_t, ha_t, rw_t \mid\mid P'(rw_{t-5} = 1) = 1)$ is a density, so it is possible to form marginal and conditional probability densities from it. For example, $P(ha_t \mid bmi_{t-5} = 25 \mid\mid P'(rw_{t-5} = 1) = 1)$ is the probability of having had a heart attack between t–5 and t among people who have a *bmi* of 25 at t–5, everyone having been assigned to drink an average of 1 glass of red wine daily between t–10 and t–5. In this paper, in order to simplify the exposition, it will be assumed that all attempted manipulations are successful; that is, if $P'(rw_{t-5} = 1) = x$ then $P(rw_{t-5} = 1 \mid\mid P'(rw_{t-5} = 1) = x) = x$ (that is, if rw_{t-5} is manipulated to have value 1 with probability x, then in the manipulated population, rw_{t-5} has value 1 with probability x.) For example, if it is assumed that $P'(rw_{t-5} = 1) = 1$ then $P(rw_{t-5} = 1 \mid\mid P'(rw_{t-5} = 1) = 1) = 1$, that is if everyone has been assigned to drink an average of 1 glass of red wine per day for 5 years (denoted $P'(rw_{t-5} = 1) = 1$), that everyone has done so.

In a randomized trial, a manipulation could set $P'(rw_{t-5} = 1) = 0.5$ and $P'(rw_{t-5} = 0) = 0.5$, in which case the resulting density is $P(sex, bmi_{t-5}, ha_{t-5}, rw_{t-5}, bmi_t, ha_t, rw_t \mid\mid \{P'(rw_{t-5} = 1) = 0.5, P'(rw_{t-5} = 0) = 0.5\})$.

In more complex manipulations, different probabilities can be assigned to different subpopulations. For example, the amount of red wine someone is assigned to drink could be based on *sex*: $P'(rw_{t-5} = 0 \mid sex = 0) = 0.25, P'(rw_{t-5} = 1 \mid sex = 0) = 0.75, P'(rw_{t-5} = 0 \mid sex = 1) = 0.\ 5, P'(rw_{t-5} = 2 \mid sex = 1) = 0.5$. The resulting density is $P(sex, bmi_{t-5}, ha_{t-5}, rw_{t-5}, bmi_t, ha_t, rw_t \mid\mid \{P'(rw_{t-5} = 0 \mid sex = 0) = 0.25, P'(rw_{t-5} = 1 \mid sex = 0) = 0.75, P'(rw_{t-5} = 0 \mid sex = 1) = 0.5, P'(rw_{t-5} = 2 \mid sex = 1) = 0.5\})$. In general, which manipulations are performed on which subpopulations can be a function both of the values of various random variables, and of what other past manipulations have been performed.

In many cases the values of some variables in the pre-manipulation density are stable, and the temporal indices on those variables are omitted. Similarly, if it is assumed that variables in the post-manipulation population eventually stabilize to fixed values, the time indices of those variables are omitted in the post-manipulation density, and the time-independent variables refer to the stable values. Both of these kinds of omissions of time indices are illustrated by the use of *sex* in the example.

In contrast to conditional probabilities, which can be estimated from samples from a population, typically the gold standard for estimating manipulated densities is an experiment, often a randomized trial. However, in many cases experiments are too expensive, too difficult, or not ethical to carry out. This raises the question of what can be determined about manipulated probability densities from samples from a population, possibly in combination with a limited number of randomized trials. The problem is even more difficult because the inference is made from a set of measured random variables $\mathbf{O}$ from samples that might not contain variables that are causes of multiple variables in $\mathbf{O}$.

Problem 2: Causal Predictive Modeling

Input: Samples from a population with density $P(\mathbf{O})$, and a (possibly empty) set of manipulated densities $P(\mathbf{O} \mid\mid M_1), \ldots\ P(\mathbf{O} \mid\mid M_n)$, a manipulation M, and sets $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{O}$.
Output: A consistent, efficient estimate of $P(\mathbf{Y} \mid \mathbf{X} \mid\mid M)$ if possible, and an output of "not possible" otherwise.

With causal inference, as with statistical inference, it is generally the case that in order to make inference tractable both computationally and statistically, simplifying assumptions are made. One kind of simplifying assumption common to both statistical and causal inference is the assumption that the population distribution lies in some parametric family (for example, Gaussian) or that relationships between variables are exactly linear. An example of a simplifying assumption unique to causal inference is that multiple causal mechanisms relating variables do not exactly cancel (Section 3). So, although the goal of Problem 2 is stated as finding a consistent estimate of a manipulated density, it is more realistic to state the goal as finding a sufficiently good estimate of a manipulated density when the sample size is large enough.

Problem 2 is usually broken into two parts: finding a set of causal models from sample data, some manipulations (experiments) and background assumptions (Sections 3 and 4), and predicting the effects of a manipulation from a set of causal models (Section 3). Here, a "causal model" (Section 3) specifies for each possible manipulation that can be performed on the population (including the manipulation that does nothing to a population) a post-manipulation density over a given set of variables. In some cases, the inferred causal models may contain unmeasured variables as well as measured variables.

Problem 3: Constructing Causal Models from Sample Data

Input: Samples from a population with density $P(\mathbf{O})$, a (possibly empty) set of manipulated densities $P(\mathbf{O}||M_1), \ldots\ P(\mathbf{O}||M_n)$, and background assumptions.

Output: A set of causal models that is as small as possible, and contains a true causal model that contains at least the variables in $\mathbf{O}$.

Problem 4: Predicting the Effects of Manipulations from Causal Models

Input: An unmanipulated density $P(\mathbf{O})$, a set $\mathbf{C}$ of causal models that contain at least the variables in $\mathbf{O}$, a manipulation M, and sets $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{O}$.

Output: A function g such that $P(\mathbf{Y} \mid \mathbf{X} \,||\, M) = g(P(\mathbf{O}), \mathbf{C}, M, \mathbf{X}, \mathbf{Y})$ if one exists, and an output of "no function" otherwise.

In analogy to the goals of statistical modeling, it would be more accurate but much more vague to state that the goal in Problem 3 is to find a useful (for example, sufficiently simple, sufficiently accurate, etc.) causal model, rather than a true causal model.

The reason that the stated goal for the output of Problem 3 is a set of causal models, is that it is generally not possible to reliably find a true causal model given the inputs. Furthermore, in contrast to predictive models, even if a true causal model can be inferred from a sample from the unmanipulated population, it generally cannot be validated on a sample from the unmanipulated population, because a causal model contains predictions about a manipulated population that might not actually exist. This has been a serious impediment to the improvement of algorithms for constructing causal models, because it makes evaluating the performance of such algorithms difficult. It is possible to evaluate causal inference algorithms on simulated data, to employ back-

ground knowledge to check the performance of algorithms, and to conduct limited (due to expense, time, and ethical constraints) experiments, but these serve as only partial checks how algorithms perform on real data in a wide variety of domains. For examples, see the Causality Challenge (http://www.causality.inf.ethz.ch/challenge.php).

In the Special Topic on Causality in this journal, Shpitser and Pearl (2008) and Zhang (2008) address Problem 4. Bromberg and Margaritis (2009), Pellet and Elisseeff (2008), He and Geng (2009), and (indirectly) Kang and Tian (2009), Aliferis et al. (2010a), and Aliferis et al. (2010b) address Problem 3. Both the problems and the papers will be described in more detail in subsequent sections.

2.3 Effects of Counterfactual Manipulations

There are cases in ethics, the law, and epidemiology in which there are questions about applying a manipulation to a subpopulation whose membership cannot be measured at the time that the manipulation is applied. For example, epidemiologists sometimes want to know what would the effect on heart attacks have been, if a manipulation such as assigning moderate drinking of red wine from t–10 to t–5, had been applied to the subpopulation which has *not* moderately drunk red wine from t–10 to t–5. When the manipulation under consideration assigns a value to a random variable to a subpopulation with a different actual value of the random variable, the probability in question is a *counterfactual* probability. If the subpopulation that did not moderately drink red wine between t–10 and t–5 differs systematically from the rest of the population with respect to causes of heart attacks, the subpopulations' response to being assigned to drink red wine would be different than the rest of the population.

Questions about counterfactual probabilities arise naturally in assigning blame in ethics or in the law. For example, the question of whether tobacco companies were negligent in the case of someone who smoked and developed lung cancer depends upon the probability that person would not have gotten lung cancer if they had not smoked.

A counterfactual probability cannot be estimated directly from a randomized experiment, because it is impossible to perform a randomized experiment that assigns moderate red wine drinking between t–10 to t–5 to a group of people who already have not been moderate wine drinkers between t–10 and t–5. This raises the question of how counterfactual probabilities can be estimated. One general approach is to assume that the value of red wine drinking between t–10 and t–5 contains information about hidden causes of red wine drinking that are also causes of heart attacks.

Problem 5: Counterfactual predictive modeling
Input: An unmanipulated density $P(\mathbf{O})$, a set $\mathbf{C}$ of causal models that contain at least the variables in $\mathbf{O}$, a counterfactual manipulation M, and sets $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{O}$.
Output: A function g such that $P(\mathbf{Y} \mid \mathbf{X} \,||\, M) = g(P(\mathbf{O}), \mathbf{C}, M, \mathbf{X}, \mathbf{Y})$ if one exists, and an output of "no function" otherwise.

In the Special Topic on Causality in this journal, Shpitser and Pearl (2008) describes a solution to Problem 5 in the case where the causal graph is known, but may contain unmeasured common causes.

3. Causal Models

This section describes several different kinds of commonly used causal models, and how to use them to calculate the effects of manipulations. The next section describes search algorithms for discovering causal models.

A (parametric) *statistical model* (with free parameters) is a set of probability densities that can be mapped into a single density by specifying the values of the free parameters (for example, a family of multivariate Gaussian densities).[4] For example, a Hidden Markov Model with a fixed structure but free parameters is a statistical model that represents a certain set of probability densities. A *causal model with free parameters* also specifies a set of probability densities over a given set of variables; however, in addition, for each manipulation that can be performed on the population it also specifies a set of post-manipulation probability densities over a given set of variables. A causal model with free parameters together with the values of the free parameters is a *causal model with fixed parameters*; a causal model with fixed parameters is mapped to a single density given a specification of a manipulation.

Often, a causal model is specified in two parts: a statistical model, and a causal graph that describes the causal relations between variables. The most frequently used causal models belong to two broad families: (1) causal Bayesian networks, (2) structural equation models. Causal Bayesian networks (and related models), specify a density for a variable as a function of the values of its causes. Structural equation models (SEMs) specify the value of a variable as a function of the values of its causes (typically including some unmeasured noise terms.) However, not surprisingly, the two kinds of models are closely linked, as explained in Section 3.2.

The statistical setup for both causal Bayesian networks and structural equation models is a standard one. There is a population of units, where depending upon the problem, the units could be people, cities, cells, genes, etc. It is assumed that there is a density over the population, which assigns probabilities to each measurable subset (event) of the population. Each unit also has a set of properties at a time, where the properties are represented by random variables, which are functions from the units to real numbers. The following sections describe the causal part of the model.

3.1 Causal Bayesian Networks

A *Bayesian network* is a pair $\langle G, P \rangle$, where G is a directed acyclic graph (DAG) whose vertices are random variables, and P is a density such that each variable V in G is independent of variables that are neither descendants nor parents of V in G,[5] conditional on the parents of V in G. In this case P is said to satisfy the *local directed Markov condition* for G.

There are two conditions that are equivalent to the local directed Markov condition described below that are useful in causal inference: the global directed Markov condition, and factorization according to G, both of which are described next.

The conditional independence relations specified by satisfying the local directed Markov condition for DAG G might also entail other conditional independence relations. There is a fast algorithm for determining from G whether a given conditional independence relation is entailed by satisfying the local directed Markov condition for G, that uses the d-separation relation, a relation among the

4. In the nomenclature of machine learning, what this article calls a "model (with free parameters)" is often called a "model family" or "learning machine" and a "model (with fixed parameter values)" is often called a "model instance" or "model".

5. X is a *parent* of Y if the graph contains the edge $X \rightarrow Y$. Y is a *descendant* of X if there is a directed path from X to Y.

vertices of G. A variable B is a *collider* (*v-structure*) *on a path* U if and only if U contains a subpath $A \rightarrow B \leftarrow C$. For disjoint sets of vertices $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ in a DAG G, $\mathbf{X}$ is *d-connected* to $\mathbf{Y}$ given $\mathbf{Z}$ if and only if there is an acyclic path U between some member X of $\mathbf{X}$, and some member Y of $\mathbf{Y}$, such that every collider on U is either a member of $\mathbf{Z}$ or an ancestor of a member of $\mathbf{Z}$, and every non-collider on U is not in $\mathbf{Z}$.[6] For disjoint sets of vertices, $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, $\mathbf{X}$ is *d-separated* from $\mathbf{Y}$ given $\mathbf{Z}$ if and only if $\mathbf{X}$ is not d-connected to $\mathbf{Y}$ given $\mathbf{Z}$. $\mathbf{X}$ is d-separated from $\mathbf{Y}$ conditional on $\mathbf{Z}$ in DAG G if and only if $\mathbf{X}$ is independent of $\mathbf{Y}$ conditional on $\mathbf{Z}$ in every density that satisfies the local directed Markov condition for G (Pearl, 1988). If $\mathbf{X}$ is independent of $\mathbf{Y}$ conditional on $\mathbf{Z}$ in P whenever $\mathbf{X}$ is d-separated from $\mathbf{Y}$ conditional on $\mathbf{Z}$ in G, then P satisfies the *global directed Markov condition* for G.

For the set of random variables $\mathbf{V}$ in G, a density $P(\mathbf{V})$ *factors according to* DAG G iff

$$P(\mathbf{V}) = \prod_{V \in \mathbf{V}} P(V|\mathbf{Parents}(V,G))$$

where **Parents**(V,G) is the set of parents of V in G.

The local directed Markov condition, the global directed Markov condition, and factorization according to a DAG G are all equivalent under mild regularity assumptions (Lauritzen et al., 1990).

A DAG can also be used to represent causal relations between variables. A is a *direct cause* of B relative to a set of variables $\mathbf{V}$ in a population when there exist two manipulations of $\mathbf{V}\backslash\{B\}$ (that is, all the variables in $\mathbf{V}$, except B, are manipulated to specific values) that differ only in the values assigned to A and that produce different probability densities of B. A *causal DAG* G for a population contains an edge $A \rightarrow B$ iff A is a direct cause of B in the specified population.

In order to use samples from probability densities to make causal inferences, some assumptions relating causal relations to probability densities need to be made. The following Causal Markov Assumption is commonly made, if only implicitly. A set of variables $\mathbf{V}$ is *causally sufficient* iff there is no variable C not in $\mathbf{V}$ that is a direct cause of more than one variable in $\mathbf{V}$ (relative to $\mathbf{V} \cup \{C\}$).

Causal Markov Assumption: For a causally sufficient set of variables $\mathbf{V}$ in a population N with density $P(\mathbf{V})$, $P(\mathbf{V})$ satisfies the local directed Markov condition for the causal DAG of N.

Under the Causal Markov Assumption, in a causal Bayesian network a manipulation of X to $P'(X \mid \mathbf{Y})$ (where $\mathbf{Y}$ is assumed to contain only non-descendants of X in a causal DAG G) simply replaces the term $P(X \mid \mathbf{Parents}(X,G))$ in the factorization of the joint density by the manipulated density $P'(X \mid \mathbf{Y})$:

$$P(\mathbf{V}||P'(X|\mathbf{Y})) = P'(X|\mathbf{Y}) \prod_{V \in \mathbf{V}\backslash\{X\}} P(V|\mathbf{Parents}(V,G)).$$

This is called the *manipulation rule*. The importance of the manipulation rule is that if the causal DAG is known, and the unmanipulated density can be estimated from a sample, it allows the prediction of the effect of an unobserved manipulation. Hence the manipulation rule is the solution to Problem 4, in the special case where the observed variables are causally sufficient, and the unique correct causal DAG is known.

6. For both the d-separation relation and the independence relation, if $\mathbf{X}$ contains a single vertex X, then X will be written instead of $\{X\}$, and similarly for $\mathbf{Y}$ and $\mathbf{Z}$. D-connection can also be defined for cyclic graphs and graphs with double-headed arrows (Spirtes, 1995; Koster, 1999; Cox and Wermuth, 1996).

The solution to Problem 4 is more difficult when the set of observed variables is not causally sufficient. There are sufficient and (almost) necessary rules for determining which manipulated conditional probability densities are invariant under a given manipulation (that is, which densities are the same in the unmanipulated population and the manipulated population) and rules for how to express some non-invariant conditional densities as functions of observed densities (Spirtes et al., 1993). Pearl's do-calculus extended the sufficient and (almost) necessary conditions for determining which conditional densities were invariant from single manipulations to sequences of manipulations, and showed how a broader range of non-invariant manipulated densities could be expressed in terms of observed densities (Pearl, 1995). In the Special Topic on Causality of this journal, Shpitser and Pearl (2008) describe an algorithm that has recently been developed and show that it is a complete solution to Problem 4 in the special case where a unique causal DAG is known (Shpitser and Pearl, 2006a,b; Huang and Valtorta, 2006).

Calculation of the effect of a counterfactual manipulation when causal sufficiency does not hold among the observed variables is a complex operation that requires several copies of the causal graph in order to keep track both of the actual value of the variable being manipulated, and the counterfactual value of the variable being manipulated. In the Special Topic on Causality, Shpitser and Pearl (2008) describe for the first time an algorithm that is a complete solution to Problem 5 in the special case where a unique causal DAG is known, even if the set of observed variables is not causally sufficient.

3.2 Structural Equation Models (SEMs)

Structural equation models are widely used in the social sciences (Bollen, 1989) and in some natural sciences. The set of random variables in a structural equation model (SEM) can be divided into two subsets, the "error variables" or "error terms," and the substantive variables (for which there is no standard terminology in the literature). The substantive variables are the variables of interest, but they are not necessarily all observed. Which variables are substantive, and which variables are error terms can vary with the analysis of the problem. Each substantive variable is a function of other substantive variables and a unique error term. The joint density over the substantive variables is a function of the density over the error terms and of the functions relating each variable to its causes. There is an edge $A \rightarrow B$ in the graph ("path diagram") of a SEM when A is a non-trivial argument in the function for B. A manipulation of a variable B to a constant c is represented in a SEM by replacing the equation for B with $B = c$.

In general, the graph of a SEM may have cycles (that is, directed paths from a variable to itself), and may explicitly include error terms with double-headed arrows between them to represent that the error terms are dependent (for example, $\varepsilon_A \leftrightarrow \varepsilon_B$); if no such double-headed edge exists in the graph, the error terms are assumed to be independent of each other. An error term is not explicitly included in the graph unless it is the endpoint of a double-headed arrow; otherwise, an error term occurs in the SEM model, but is not shown in the graph. If the graph has no directed cycles and no double-headed arrows, then the graph is a DAG and the SEM is said to be *recursive*; otherwise it is said to be *non-recursive*.

In a recursive SEM, if the marginal density over the substantive variables is $P(\mathbf{V})$, then $\langle G, P(\mathbf{V})\rangle$ is a Bayesian network (Spirtes et al., 2001; Pearl, 2000); for short, say that a SEM with an associated graph that is a DAG is a Bayesian network (although the SEM contains some extra structure in that it

entails that any non-determinism among the substantive variables is only due to the marginalization of the error terms.)

Non-recursive SEMs are of interest because they allow for the representation of feedback (with cycles) or unmeasured common causes (represented by double-headed arrows.) In the case of linear non-recursive SEMs, it is still possible to deduce the conditional independencies (or more generally the zero partial correlations) entailed for all Gaussian SEMs (or more generally linear SEMs) from the graph G of a non-recursive linear SEM using a minor modification of the d-separation relation (Koster, 1999; Spirtes, 1995).

For both theoretical interest and for the purposes of efficient (constraint-based) search of the space of linear non-recursive SEMs without cycles (Section 4.2), it is of interest to find some proper subset of the set of all conditional independence relations entailed by the (modified) d-separation which entail all the rest, that is, a modified form of the local directed Markov condition. (In contrast to the recursive case, where such a subset is given by the independencies entailed by the local directed Markov condition, in the non-recursive case SEMs do not generally satisfy the local directed Markov condition). One such subset of conditional independencies was described by Richardson (2003). In this special issue, the paper by Kang and Tian (2009) describes another such subset, which is often smaller than the one described by Richardson, and hence might be more useful for the purposes of search.

4. Model Search

Traditionally, there have been a number of different approaches to causal discovery. The gold standard of causal discovery has typically been to perform planned or randomized experiments (Fisher, 1971). There are obvious practical and ethical considerations that limit the application of experiments in many instances, particularly on human beings. Moreover, recent data collection techniques and causal inference problems raise several practical difficulties regarding the number of experiments that need to be performed in order to answer all of the outstanding questions.

In the absence of experiments, in practice (particularly in the social sciences) search for causal models is often informal, and based on a combination of background assumptions about causal relations together with statistical tests of the causal models. If a model is rejected by a statistical test, the researcher looks for a modification of the original hypothesized model that will pass a statistical test. The search typically halts when a model that is compatible with background knowledge does not fail a statistical test (Rodgers and Maranto, 1989). Often, the final model is presented, and the search itself is not described. Informal searches of this kind fail to account for multiple testing problems, and can potentially lead to severe overfitting problems. The reliability of such a search depends upon the correctness of the background assumptions, and the extent to which the space of alternatives compatible with the background assumptions was searched. Furthermore, unless the background assumptions are very extensive, or the number of variables is tiny, it is not feasible to estimate and test all of the models compatible with background assumptions. This is further complicated by the fact that, as explained below, for reliable causal inference it is not sufficient to find one model that passes a statistical test; instead it is necessary to find all such models. Recent developments in automated model search have attempted to address these problems with traditional methods of search.

There are several major differences between model search in the case of predicting the unmanipulated value of Y, and model search in the case of predicting the post-manipulation value of Y, based on the different uses of statistical models and causal models described in the following section.

4.1 Underdetermination of Causal Models by Data

Causal model (with fixed parameter) search is often broken into two parts: search for a causal graph, and estimation of the free parameters from sample data and the causal graph. (In some cases, the prediction of the effects of manipulations does not require estimating all of the free parameters, but does require estimating functionals of the free parameters.) Generally, the estimation of the free parameters employs standard statistical methods. For example, in a linear SEM with a recursive DAG, no unmeasured variables, and Gaussian errors, the maximum likelihood estimate of the edge coefficients is given by regressing each variable on its parents in the DAG. This section concentrates on the search for causal graphs, because the search for causal graphs is significantly different than the search for graphs that are to be used only as statistical models.

In causal model search based on unmanipulated data, if no preference for simpler models over more complex models is made, then the causal models are underdetermined to such an extent that useful causal inference is impossible for many important parametric families (for example, Gaussian or multinomial) or unrestricted probability densities. There are a variety of simplicity assumptions that select simpler models over more complex models that can be made. In the case of search based upon maximizing some model score given sample data (such as the Bayesian Information Criterion), the simplicity assumption is a penalty for complexity built into the score (Chickering, 2002). For search that is not based upon model scores, the following simplicity assumption is often, if implicitly made:

Causal Faithfulness Assumption: For a causally sufficient set of variables **V** in a population N, every conditional independence relation true in the density over **V** is entailed by the local directed Markov condition for the causal DAG of N.

There are several other versions of the assumption that are considerably weaker than the one stated here (and more intuitively justifiable) but still permit reliable causal inference, at the cost of requiring more complicated algorithms with more complex and somewhat less informative output (Ramsey et al., 2006).

However, even given the Causal Markov and Faithfulness Assumptions and the assumption that the observed variables are causally sufficient, the true causal model is underdetermined by the available evidence and background assumptions, because of the hierarchy of equivalence relations described below.

Two different DAGs G and G' that have the same set of d-separation relations are said to be *Markov* (*conditional independence*, *d-separation*) *equivalent*.

For each DAG G, there is a set **P** of probability densities that satisfy the local directed Markov condition for G, denoted $\mathbf{P}(G)$ that are said to be *represented* by G. In many cases, some subset of **P** that belongs to a parametric or semi-parametric family **F** is of interest; for example, the Gaussian subset of **P**. Two DAGs G and G' are *statistically equivalent with respect to* **F** iff $\mathbf{P}(G) \cap \mathbf{F} = \mathbf{P}(G') \cap \mathbf{F}$. Two DAGs that are statistically equivalent with respect to **F** are the same statistical model with respect to **F**.

Two DAGs are *causally equivalent* (with respect to a family of densities **F**) iff they represent the same set of probability densities (in family **F**) for every manipulation (including the null ma-

nipulation.) It is easy to see that no pair of DAGs that differ in their structure can be causally equivalent.

As an example, $A \rightarrow B \leftarrow C \leftarrow D$ and $A \rightarrow B \leftarrow C \rightarrow D$ are Markov equivalent, but not causally equivalent. They are statistically equivalent with respect to Gaussian SEMs, but they are not statistically equivalent with respect to linear SEMs with at most one Gaussian error term, and no determinism among the substantive variables (Shimizu et al., 2006).[7]

In the absence of further information (for example, samples from manipulated densities or background domain knowledge) all of the DAGs in a statistical equivalence class fit the data and the background assumptions equally well, and are equally simple. Hence standard scores such as Bayesian Information Criterion, Minimum Description Length, chi-squared statistics, etc. all produce equal scores for the alternative DAGs in a statistical equivalence class for all data sets -- in general, there is no one DAG with the highest score, but rather, there is a set of DAGs with equally high scores. Furthermore, for computational and statistical reasons, it is sometimes easier to search for the Markov equivalence class of DAGs, even if it is known that the statistical equivalence class is a proper subset of the Markov equivalence class.

If the DAG is to be used to estimate observational (not manipulated) conditional densities, this is not a problem, because all of the statistically equivalent models will produce the same estimate. However, if the DAG is to be used to predict the effects of manipulations, then the different models will make different predictions about at least some manipulations. So in the case of causal modeling, unlike observational statistical modeling, it is not enough to simply output one arbitrarily selected DAG from a set of highest scoring DAGs -- it is important to output the entire set, so that all of the different answers given by the different models can be taken into account. Once the set of highest scoring DAGs is found, the problem of dealing with the underdetermination of the effects of manipulations must also be dealt with. These problems are described in more detail in the next two subsections.

If the assumption of causal sufficiency of the observed variables is not made, all three kinds of equivalence classes have corresponding equivalence classes over the set of observed variables, and the problem of causal underdetermination becomes much more severe. For example, for a given set of observed variables **O**, the Markov equivalence class relative to **O** consists of the set of all DAGs (possibly containing variables not in **O**) that have the same set of d-separation relations among the variables in **O**; this might be much larger than the Markov equivalence class that consists of the set of DAGs (containing only variables in **O**) that have the same set of d-separation relations among the variables in **O**.

4.2 Constraint-based Search

First, the problem where only sample data from the unmanipulated population density is available will be considered. The number of DAGs grows super-exponentially with the number of vertices, so even for modest numbers of variables it is not possible to examine each DAG to determine whether it is compatible with the population density given the Causal Markov and Faithfulness Assumptions. Constraint based search algorithms, given as input an oracle that returns answers about conditional independence in the population and optional background knowledge about orientations of edges, return a representation of a Markov equivalence class (or if there is background knowl-

7. In a linear SEM it is assumed that each variable is a linear function of its causal parents and a unique error term; in a Gaussian SEM it is assumed in addition that the errors term are Gaussian.

edge, a subset of a Markov equivalence class) on the basis of oracle queries. One example of a constraint-based algorithm is the PC algorithm (Spirtes and Glymour, 1991). If the oracle always gives correct answers, and the Causal Markov and Causal Faithfulness Assumptions hold, then the PC algorithm always outputs a Markov equivalence class that contains the true causal model, even though the algorithm does not check each directed acyclic graph. In the worse case, it is exponential in the number of variables, but for sparse graphs it can run on hundreds of variables in an acceptable amount of time (Spirtes and Glymour, 1991; Spirtes et al., 1993; Meek, 1995). Kalisch and Buhlmann (2007) showed that under a strengthened version of the Causal Faithfulness Assumption, the PC algorithm is uniformly consistent for very high-dimensional, sparse DAGs where the number of nodes is allowed to quickly grow with sample size n, as fast as $O(n^a)$ for any $0 < a < \infty$. In practice, the judgments about conditional independence are made by performing (fallible) statistical tests. A number of other variants of constraint-based algorithms have been proposed that improve on either the accuracy or speed of the PC algorithm, or to weaken the assumptions under which it is guaranteed to be correct.

There are both advantages and disadvantages of constraint based searches as compared to either a Bayesian approach to the problem of causal discovery (Heckerman and Geiger, 1995), or an approach based upon assigning a score to each causal model for a given data set (for example, Bayesian information criterion) and searching for the set of causal models that maximize the score (Chickering, 2002).

The disadvantages of constraint-based search include that the output of constraint-based searches give no indication of how much better the best set of output models is compared to the next best set of models; at small sample sizes tests of conditional independence have low power, particularly when many variables are conditioned on; mistakes made early in a constraint based searches can lead to later mistakes; and if the only constraints used are conditional independence constraints, as is often but not always the case, then at best the search outputs a Markov equivalence class, rather than a statistical equivalence class.[8] In addition, constraint-based methods have the problem of multiple testing. If no control is made for multiple testing, the models may overfit the data. However, adjustments to control for overfitting, such as the Bonferroni correction, are often too conservative and as a result the corrected statistical tests are not very powerful. The issue of multiple testing appears in Bayesian approaches to causal discovery as multiple causal model scoring. The issue is handled automatically by Bayesian methods by their use of prior probabilities (Heckerman et al., 1999).

The advantages of constraint-based algorithms are that they are easier to generalize to the case where the observed variables are not causally sufficient, they are generally fast, and given recent developments of non-parametric conditional independence tests, they are applicable without parametric assumptions (Tillman et al., 2009).

In the Special Topic on Causation, Bromberg and Margaritis (2009) models the problem of low power of statistical tests as a knowledge base containing a set of independence facts related through conditional independence axioms that may contain errors due to errors in the tests of conditional independence. The inconsistencies are resolved through the use of a defeasible logic called argumentation that is augmented with a preference function. The logic is used to reason about and possibly correct errors in these tests. Experimental evaluation shows significant improvements in the accuracy of argumentative over purely statistical tests, and improvements on the accuracy of causal

8. For searches that use non-conditional independence constraints see Silva et al. (2006) and Shpitser et al. (2009).

structure discovery from sampled data from randomly generated causal models and on real-world data sets.

The contributions to the Special Topic on Causality by Aliferis et al. (2010a) and Aliferis et al. (2010b) show that a general framework for localized causal membership structure learning is very accurate even in small sample situations and can thus be used as a first step for efficient global structure learning, as well as accurate prediction and feature selection. It also provides extensive empirical comparisons of state of the art causal learning methods with non-causal methods for the above tasks. In addition, they show that unexpectedly some constraint-based methods are self-correcting with respect to multiple testing, and this may constitute a new methodology for control of multiple statistical testing.

Another problem with constraint-based algorithms is to make them feasible for even higher dimensional data sets. In the Special Topic on Causality, Pellet and Elisseeff (2008) link the causal model search problem to a classic machine learning prediction problem. They show how a generic feature-selection algorithm returning strongly relevant variables can be turned into a causal model search algorithm. Under the Causal Markov and Causal Faithfulness Assumptions, the smallest set of features relevant to predicting a vertex V is the set of parents, children, and parents of children of V. Ideally, the variables returned by a feature-selection algorithm identify those features of the causal graph. Then further processing removes the extra edges (between V and those variables that are parents of children of V but that are neither parents nor children of V) and provides as many orientations as possible. This algorithm is more accurate than PC and other constraint-based algorithms, and has the advantage that it can use arbitrary feature-selection techniques developed for high-dimensional data sets under different assumptions to provide causal model learning algorithms for high-dimensional data under those assumptions.

4.3 Dealing with Underdetermination

One possibility for dealing with the underdetermination of causal models by observational data is to strengthen the available information by sampling from manipulated densities, or in other words, performing experiments.

In the Special Topic on Causality, He and Geng (2009) propose an algorithm for distinguishing between members of a Markov equivalence class by a set of optimally designed experiments. They consider several kinds of experiments, and both a batch-design and a sequential design to minimize the required number of manipulations using both minimax and maximum entropy criteria.

If some members of the Markov equivalence class cannot be eliminated through experimentation, there are several different approaches to using the entire Markov equivalence class to predict the effects of manipulations. (This is Problem 4 in the case where the predictions are made from a set of causal models $\mathbf{C}$ rather than a single causal model, and the set of observed variables may not be causally sufficient.) One possibility is to predict an interval for the potential effects of the manipulated quantity, instead of a point value. Theoretically, an interval could be obtained by calculating the manipulated quantity for each DAG G in the Markov equivalence class, and taking the lower and upper limits. Depending upon how many different SEMs there are in the output, this is sometimes computationally feasible (Maathuis et al., 2009).

A second possibility is to use a Bayesian approach, and perform model averaging. That is, a prior probability is placed over each causal DAG G, and a posterior probability for each G is calculated. Then the manipulated quantity is calculated for each G in the output of the search,

and the results are averaged together. This requires putting a prior probability over each graph; in addition, if there are many graphs in the output, then this may not be computationally feasible (Hoeting et al., 1999).

A third alternative is to have an algorithm that determines whether each DAG in the Markov equivalence class predicts the same effect of a given manipulation. For example, if the Markov equivalence class contains $A \rightarrow B \leftarrow C \rightarrow D$ and $A \rightarrow B \leftarrow C \leftarrow D$, then the two causal DAGs disagree about the effect of manipulating D on C, but agree about the effect of manipulating A on B. Even when the observed variables are not causally sufficient there is an algorithm (the Prediction Algorithm) for determining when all of the DAGs in a Markov equivalence class relative to the observed variables agree about the effect of a particular manipulation, and returns the common value of the predicted manipulation when they do all agree (Spirtes et al., 1993). However, this algorithm is known to be correct but incomplete (that is, it sometimes returns "don't know" even when all models in the equivalence class agree on the effect of a particular manipulation). In this special issue, Zhang (2008) provides a modified version of Pearl's do-calculus that is more complete than the Prediction algorithm.

5. Open Questions

The following is an overview of important problems that remain in the domain of causal modeling.

1. Matching causal models and search algorithms to causal problems. There are a wide variety of causal models that have been employed in different disciplines. What new models and search algorithms are appropriate for different domains such as feedback or reversible systems (Richardson, 1996)? What search algorithms are appropriate for different combinations of kinds of data, such as experimental and observational data (Eberhardt et al., 2005; Cooper and Yoo, 1999; Yoo and Cooper, 2004; He and Geng, 2009)? What search algorithms are appropriate for different kinds of background knowledge, and different families of probability densities?

2. Model selection, and prior knowledge. What kind of scores can be used to best evaluate causal models from various kinds of data? In a related vein, what are good families of prior densities that capture various kinds of background knowledge?

3. Improve efficiency and efficacy of search algorithms. How can search algorithms be improved to incorporate different kinds of background knowledge, search over different classes of causal models, run faster, handle more variables and larger sample sizes, be more reliable at small sample sizes, and produce output that is as informative as possible?

4. Characterization of search algorithms. For causal search algorithms, what are their semantic and syntactic properties (for example, soundness, consistency, maximum informativeness)? What are their statistical properties (pointwise consistency, uniform consistency, sample efficiency)?[9] What are their computational properties (computational complexity)?

5. Adding or relaxing simplifying assumptions. What plausible alternatives are there to the Causal Markov and Faithfulness Assumptions? Are there other assumptions that might be weaker and hold in more domains and applications without much loss about what can be reliably inferred?

9. Intuitively, an estimator is pointwise consistent when as the sample size increases without limit, regardless of the true value, with probability 1 the absolute value of the difference between the estimator and the true value approaches zero. An estimator is uniformly consistent if for any given ε and δ, there is a single sample size such that in the worst case, the probability is less than ε that the absolute value of the difference between the estimator and the true value is greater than δ. For precise definitions in the causal context, see Robins et al. (2003).

Are there stronger assumptions that are plausible for some domains that might allow for stronger causal inferences? How often are these assumptions violated, and how much do violations of these assumptions lead to incorrect inferences? Can various statistical assumptions be relaxed? For example, what if the sample selection process is not i.i.d., but may be causally affected by variables of interest?

6. Derivation of consequences from causal graph and unmanipulated densities. Shpitser and Pearl have given complete algorithms for deriving the consequences of various causal models with hidden common causes in terms of the unmanipulated density and the given manipulation (Shpitser and Pearl, 2008). Partial extensions of these results to deriving consequences from sets of causal models have been given (Zhang, 2008); are there further extensions to derivations from sets of causal models?

7. New constraints for structure learning. The Causal Markov and Causal Faithfulness Assumptions, in addition to entailing conditional independence constraints on densities, also entail other constraints on densities. For example, in a linear SEM, if an unobserved variable T causes observed variables X_1, X_2, X_3, X_4, and there are no other causal relations among these variables, then there are no entailed conditional independence relations among just the observed variables X_1, X_2, X_3, X_4. However, the SEM entails $\text{cov}(X_1,X_2)\ \text{cov}(X_3,X_4) = \text{cov}(X_1,X_3)\ \text{cov}(X_2,X_4) = \text{cov}(X_1,X_4)\ \text{cov}(X_2,X_3)$ regardless of the values of the free parameters. This information is useful in finding causal structure with unmeasured variables. In addition, there are sometimes constraints that are not conditional independence constraints on the density of the observed variables that do not depend upon any parametric assumptions (Shpitser et al., 2009). How can these non-parametric constraints be incorporated into search algorithms?

8. Find variable definitions. In many domains, such as fMRI research, there are thousands of variables, but the measured variables do not correspond to functional units of the brain. How is it possible to define new variables that are functions of the measured variables, but more useful for causal inference and more meaningful?

9. Find new applications of causal inference. Applications of causal inference algorithms in many domains (Cooper and Glymour, 1999) help test and improve causal inference algorithms, suggest new problems, and produce domain knowledge.

10. Creating good benchmarks. What are the most appropriate performance measures for causal inference algorithms? What benchmarks can be established? What is the best research design for testing causal inference algorithms?

11. Formal connections between different causal modeling approaches. Many different fields have studied causal discovery including Artificial Intelligence, Econometrics, Operations Research, Control Theory, and Statistics. What are the formal connections between the different models, assumptions, and algorithms used in each of these fields? What can each of these domains learn from the others?

Acknowledgments

I would like to thank Isabelle Guyon, Constantin Aliferis, Greg Cooper, and Rob Tillman for many helpful comments.

References

Constantin Aliferis, Alexander Statnikov, Ionnis Tsamardinos, Subramani Mani, and Xenophon Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification, Part I: Algorithms and empirical evaluation. *Journal of Machine Learning Research*, 11:171–234, 2010a.

Constantin Aliferis, Alexander Statnikov, Ionnis Tsamardinos, Subramani Mani, and Xenophon Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification, Part II: Analysis and extensions. *Journal of Machine Learning Research*, 11:235–284, 2010b.

Kenneth A. Bollen. *Structural Equations with Latent Variables*. Wiley-Interscience, 1989.

Facundo Bromberg and Dimitris Margaritis. Improving the reliability of causal discovery from small data sets using argumentation. *Journal of Machine Learning Research*, 10:301–340, 2009.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

Greg Cooper and Clark Glymour. *Computation, Causation, and Discovery*. AAAI Press, 1999.

Gregory Cooper and Changwon Yoo. Causal discovery from a mixture of experimental and observational data. In Kathryn Laskey and Henri Prade, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 116–125, San Francisco, CA, 1999. Morgan Kauffman.

David Cox and Nanny Wermuth. *Multivariate Dependencies: Models, Analysis and Interpretation (Monographs on Statistics and Applied Probability)*. Chapman and Hall, 1996.

Frederick Eberhardt, Richard Scheines, and Clark Glymour. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In Fahiem Bacchus and Tommi Jaakkola, editors, *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 178–184, Arlington, VA, 2005. AUAI Press.

Ronald Fisher. *The Design of Experiments*. Macmillan Pub Co, 1971.

Yang-Bo He and Zhi Geng. Active learning of causal networks with intervention experiments and optimal designs. *Journal of Machine Learning Research*, 10:2523–2547, 2009.

David Heckerman and Dan Geiger. Learning Bayesian networks: a unification for discrete and Gaussian domains. In Philippe Besnard and Steve Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 274–282. Morgan Kaufman, 1995.

David Heckerman, Chris Meek, and Gregory Cooper. A Bayesian approach to causal discovery. In Greg Cooper and Clark Glymour, editors, *Computation, Causation, and Discovery*, pages 141–165. MIT Press, Cambridge, MA, 1999.

Jennifer Hoeting, David Madigan, Adrian Raftery, and Chris Volinsky. Bayesian model averaging: A tutorial. *Statistical Science*, 14(4):382–401, 1999.

Yimin Huang and Marco Valtorta. Identifiability in causal Bayesian networks: A sound and complete algorithm. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pages 1149–1154, Edinboro, Scotland, 2006. AAAI Press.

Markus Kalisch and Peter Buhlmann. Estimating high dimensional directed acyclic graphs with the PC algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.

Changsung Kang and Jin Tian. Markov properties for linear causal models with correlated errors. *Journal of Machine Learning Research*, 10:41–70, 2009.

Jan Koster. On the validity of the Markov interpretation of path diagrams of Gaussian structural equation models with correlated errors. *Scandinavian Journal of Statistics*, pages 413–431, 1999.

Steffen Lauritzen. Causal inference from graphical models. In D. Barnsdorf-Nielsen and C. Kluppenberg, editors, *Complex Stochastic Systems*, pages 141–165. Chapman and Hall, Baton Rouge, LA, 1999.

Steffen Lauritzen, Phil Dawid, B. Larsen, and H. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

Marloes Maathuis, Markus Kalisch, and Peter Buhlmann. Estimating high-dimensional intervention effects from observational data. *Annals of Statistics*, 37(6A):3133–3164, 2009.

Chris Meek. Strong completeness and faithfulness in Bayesian networks. In Phillipe Besnard and Steve Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 411–419, Montreal, Quebec, 1995. Morgan Kaufman.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

Jean-Philippe Pellet and Andre Elisseeff. Using Markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9:1295–1342, 2008.

Joseph Ramsey, Peter Spirtes, and Jiji Zhang. Adjacency-faithfulness and conservative causal inference. In Rina Dechter and Thomas Richardson, editors, *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 401–408, Cambridge, MA, 2006. AUAI Press.

Thomas Richardson. A discovery algorithm for directed cyclic graphs. In Eric Horvitz and Finn Jensen, editors, *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pages 454–462, Cambridge, MA, 1996. Morgan Kaufmann.

Thomas Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, 30:145–157, 2003.

James Robins, Richard Scheines, Peter Spirtes, and Larry Wasserman. Uniform consistency in causal inference. *Biometrika*, 90(3):491–515, 2003.

R. Rodgers and C. Maranto. Causal-models of publishing productivity in psychology. *J Appl Psychol*, 74(4):636–649, 1989.

Shohei Shimizu, Aapo Hyvarinen, Patrick Hoyer, and Yutaku Kano. Finding a causal ordering via independent component analysis. *Comput Stat Data An*, 50(11):3278–3293, 2006.

Ilya Shpitser and Judea Pearl. Identification of conditional intervention distributions. In Rina Dechter and Thomas Richardson, editors, *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 437–444, Cambridge, MA, 2006a. AUAI Press.

Ilya Shpitser and Judea Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pages 1219–1226, Menlo Park, California, 2006b. AAAI Press.

Ilya Shpitser and Judea Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9:1941–1979, 2008.

Ilya Shpitser, Thomas Richardson, and James Robins. Testing edges by truncation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1957–1963. AAAI Press, 2009.

Ricardo Silva, Richard Scheines, Clark Glymour, and Peter Spirtes. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7:191–246, 2006.

Peter Spirtes. Directed cyclic graphical representations of feedback models. In Phillipe Besnard and Steve Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 491–499, Montreal, Canada, 1995. Morgan Kaufmann.

Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):67–72, 1991.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Spring-Verlag Lectures in Statistics, 1993.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search, Second Edition (Adaptive Computation and Machine Learning)*. The MIT Press, 2001.

Robert Tillman, Arthur Gretton, and Peter Spirtes. Nonlinear directed acyclic structure learning with weakly additive noise models. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Proceedings of Advances in Neural Processing Information Systems 22*, pages 1847–1855, Vancouver, BC, 2009. Curran Associates, Inc.

Changwon Yoo and Gregory Cooper. An evaluation of a system that recommends microarray experiments to perform to discover gene-regulation pathways. *Artificial Intelligence in Medicine*, 31(2):169–182, 2004.

Jiji Zhang. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9: 1437–1474, 2008.

Journal of Machine Learning Research 11 (2010) 1663-1707 Submitted 6/09; Revised 1/10; Published 5/10

Consensus-Based Distributed Support Vector Machines

Pedro A. Forero FORER002@UMN.EDU
Alfonso Cano ALFONSO@UMN.EDU
Georgios B. Giannakis GEORGIOS@UMN.EDU
Department of Electrical and Computer Engineering
University of Minnesota
Minneapolis, MN 55455, USA

Editor: Sathiya Keerthi

Abstract

This paper develops algorithms to train support vector machines when training data are distributed across different nodes, and their communication to a centralized processing unit is prohibited due to, for example, communication complexity, scalability, or privacy reasons. To accomplish this goal, the centralized linear SVM problem is cast as a set of decentralized convex optimization subproblems (one per node) with consensus constraints on the wanted classifier parameters. Using the alternating direction method of multipliers, fully distributed training algorithms are obtained without exchanging training data among nodes. Different from existing incremental approaches, the overhead associated with inter-node communications is fixed and solely dependent on the network topology rather than the size of the training sets available per node. Important generalizations to train nonlinear SVMs in a distributed fashion are also developed along with sequential variants capable of online processing. Simulated tests illustrate the performance of the novel algorithms.[1]

Keywords: support vector machine, distributed optimization, distributed data mining, distributed learning, sensor networks

1. Introduction

Problems calling for *distributed learning* solutions include those arising when training data are acquired by different nodes, and their communication to a central processing unit, often referred to as fusion center (FC), is costly or even discouraged due to, for example, scalability, communication overhead, or privacy reasons. Indeed, in applications involving wireless sensor networks (WSNs) with battery-operated nodes, transferring each sensor's data to the FC maybe prohibited due to power limitations. In other cases, nodes gathering sensitive or private information needed to design the classifier may not be willing to share their training data.

For *centralized learning* on the other hand, the merits of support vector machines (SVMs) have been well documented in various supervised classification tasks emerging in applications such as medical imaging, bio-informatics, speech, and handwriting recognition, to name a few (Vapnik, 1998; Schölkopf and Smola, 2002; El-Naqa et al., 2002; Liang et al., 2007; Ganapathiraju et al., 2004; Li, 2005; Markowska-Kaczmar and Kubacki, 2005). Centralized SVMs are maximum-margin

1. Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2- 0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

linear classifiers designed based on a centrally available training set comprising multidimensional data with corresponding classification labels. Training an SVM requires solving a quadratic optimization problem of dimensionality dependent on the cardinality of the training set. The resulting linear SVM discriminant depends on a subset of elements from the training set, known as support vectors (SVs). Application settings better suited for nonlinear discriminants have been also considered by mapping vectors at the classifier's input to a higher dimensional space, where linear classification is performed. In either linear or nonlinear SVMs designed with centrally available training data, the decision on new data to be classified is based solely on the SVs.

For this reason, recent designs of *distributed SVM* classifiers rely on SVs obtained from local training sets (Flouri et al., 2006, 2008; Lu et al., 2008). These SVs obtained locally per node are incrementally passed on to neighboring nodes, and further processed at the FC to obtain a discriminant function approaching the centralized one obtained as if all training sets were centrally available. Convergence of the *incremental* distributed (D) SVM to the centralized SVM requires multiple SV exchanges between the nodes and the FC (Flouri et al., 2006); see also Flouri et al. (2008), where convergence of a gossip-based DSVM is guaranteed when classes are linearly separable. Without updating local SVs through node-FC exchanges, DSVM schemes can approximate but not ensure the performance of centralized SVM classifiers (Navia-Vazquez et al., 2006).

Another class of DSVMs deals with *parallel* designs of centralized SVMs—a direction well motivated when training sets are prohibitively large (Chang et al., 2007; Do and Poulet, 2006; Graf et al., 2005; Bordes et al., 2005). Partial SVMs obtained using small training subsets are combined at a central processor. These parallel designs can be applied to distributed networked nodes, only if a central unit is available to judiciously combine partial SVs from intermediate stages. Moreover, convergence to the centralized SVM is generally not guaranteed for any partitioning of the aggregate data set (Graf et al., 2005; Bordes et al., 2005).

The novel approach pursued in the present paper trains an SVM in a *fully distributed* fashion that does not require a central processing unit. The centralized SVM problem is cast as a set of coupled decentralized convex optimization subproblems with consensus constraints imposed on the desired classifier parameters. Using the alternating direction method of multipliers (ADMoM), see, for example, Bertsekas and Tsitsiklis (1997), distributed training algorithms that are provably convergent to the centralized SVM are developed based solely on message exchanges among neighboring nodes. Compared to existing alternatives, the novel DSVM classifier offers the following distinct features.

- **Scalability and reduced communication overhead**. Compared to approaches having distributed nodes communicate training samples to an FC, the DSVM approach here relies on *in-network* processing with messages exchanged only among single-hop neighboring nodes. This keeps the communication overhead per node at an affordable level within its neighborhood, even when the network scales to cover a larger geographical area. In FC-based approaches however, nodes consume increased resources to reach the FC as the coverage area grows. Different from, for example, Lu et al. (2008), and without exchanging SVs, the novel DSVM incurs a fixed overhead for inter-node communications per iteration regardless of the size of the local training sets.

- **Robustness to isolated point(s) of failure**. If the FC fails, an FC-based SVM design will fail altogether—a critical issue in tactical applications such as target classification. In contrast, if a single node fails while the network remains connected, the proposed algorithm will converge

to a classifier trained using the data of nodes that remain operational. But even if the network becomes disconnected, the proposed algorithm will stay operational with performance dependent on the number of training samples per connected sub-network.

- **Fully decentralized network operation**. Alternative distributed approaches include incremental and parallel SVMs. Incremental passing of local SVs requires identification of a Hamiltonian cycle (going through all nodes once) in the network (Lu et al., 2008; Flouri et al., 2006). And this is needed not only in the deployment stage, but also every time a node fails. However, Hamiltonian cycles do not always exist, and if they do, finding them is an NP-hard task (Papadimitriou, 2006). On the other hand, parallel SVM implementations assume full (any-to-any) network connectivity, and require a central unit defining how SVs from intermediate stages/nodes are combined, along with predefined inter-node communication protocols; see, for example, Chang et al. (2007), Do and Poulet (2006) and Graf et al. (2005).

- **Convergence guarantees to centralized SVM performance**. For linear SVMs, the novel DSVM algorithm is *provably convergent* to a centralized SVM classifier, as if all distributed samples were available centrally. For nonlinear SVMs, it converges to the solution of a modified cost function whereby nodes agree on the classification decision for a subset of points. If those points correspond to a classification query, the network "agrees on" the classification of these points with performance identical to the centralized one. For other classification queries, nodes provide classification results that closely approximate the centralized SVM.

- **Robustness to noisy inter-node communications and privacy preservation**. The novel DSVM scheme is robust to various sources of disturbance that maybe present in message exchanges. Those can be due to, for example, quantization errors, additive Gaussian receiver noise, or, Laplacian noise intentionally added for privacy (Dwork et al., 2006; Chaudhuri and Monteleoni, 2008).

The rest of this paper is organized as follows. To provide context, Section 2 outlines the centralized linear and nonlinear SVM designs. Section 3 deals with the novel fully distributed linear SVM algorithm. Section 3.1 is devoted to an online DSVM algorithm for synchronous and asynchronous updates, while Section 4 generalizes the DSVM formulation to allow for nonlinear classifiers using kernels. Finally, Sections 5 and 6 present numerical results and concluding remarks.

General notational conventions are as follows. Upper (lower) bold face letters are used for matrices (column vectors); $(\cdot)^T$ denotes matrix and vector transposition; the ji-th entry of a matrix (j-th entry of a vector) is denoted by $[\cdot]_{ji}$ ($[\cdot]_j$); diag($\mathbf{x}$) denotes a diagonal matrix with $\mathbf{x}$ on its main diagonal; diag$\{\cdot\}$ is a (block) diagonal matrix with the elements in $\{\cdot\}$ on its diagonal; $|\cdot|$ denotes set cardinality; $\succeq$ ($\preceq$) element-wise $\geq$ ($\leq$); $\{\cdot\}$ ($[\cdot]$) a set (matrix) of variables with appropriate elements (entries); $\|\cdot\|$ the Euclidean norm; $\mathbf{1}_j$ ($\mathbf{0}_j$) a vector of all ones (zeros) of size N_j; $\mathbf{I}_M$ stands for the $M \times M$ identity matrix; $\mathbb{E}\{\cdot\}$ denotes expected value; and $\mathcal{N}(m,\Sigma)$ for the multivariate Gaussian distribution with mean m, and covariance matrix Σ.

2. Preliminaries and Problem Statement

With reference to Figure 1, consider a network with J nodes modeled by an undirected graph $\mathcal{G}(\mathcal{J},\mathcal{E})$ with vertices $\mathcal{J} := \{1,\ldots,J\}$ representing nodes, and edges $\mathcal{E}$ describing links among com-

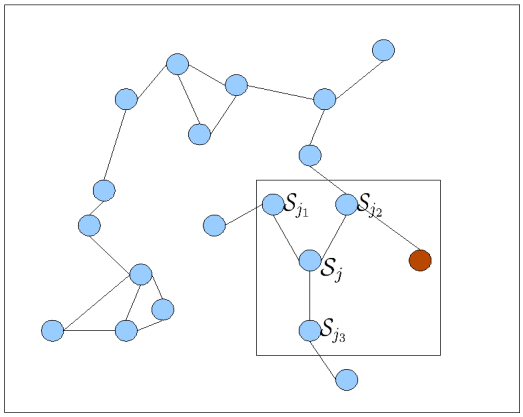


Figure 1: Network example where connectivity among nodes, represented by colored circles, is denoted by a line joining them.

municating nodes. Node $j \in \mathcal{J}$ only communicates with nodes in its one-hop neighborhood (ball) $\mathcal{B}_j \subseteq \mathcal{J}$. The graph $\mathcal{G}$ is assumed connected, that is, any two nodes in $\mathcal{G}$ are connected by a (perhaps multihop) path in $\mathcal{G}$. Notice that nodes do not have to be fully connected (any-to-any), and $\mathcal{G}$ is allowed to contain cycles. At every node $j \in \mathcal{J}$, a labeled training set $\mathcal{S}_j := \{(\mathbf{x}_{jn}, y_{jn}) : n = 1, \ldots, N_j\}$ of size N_j is available, where $\mathbf{x}_{jn} \in \mathcal{X}$ is a $p \times 1$ data vector belonging to the input space $\mathcal{X} \subseteq \mathbb{R}^p$, and $y_{jn} \in \mathcal{Y} := \{-1, 1\}$ denotes its corresponding class label.[2]

Given $\mathcal{S}_j$ per node j, the *goal* is to find a maximum-margin linear discriminant function $g(\mathbf{x})$ in a distributed fashion, and thus enable each node to classify any new input vector $\mathbf{x}$ to one of the two classes $\{-1, 1\}$ without communicating $\mathcal{S}_j$ to other nodes $j' \neq j$. Potential application scenarios include but are not limited to the following ones.

Example 1 *(Wireless sensor networks).* Consider a set of wireless sensors deployed to infer the presence or absence of a pollutant over a geographical area at any time t_n. Sensor j measures and forms a local binary decision variable $y_{jn} \in \{1, -1\}$, where $y_{jn} = 1(-1)$ indicates presence (absence) of the pollutant at the position vector $\mathbf{x}_j := [x_{j1},\ x_{j2},\ x_{j3}]^T$. (Each sensor knows its position $\mathbf{x}_j$ using existing self-localization algorithms Langendoen and Reijers, 2003.) The goal is to have each low-cost sensor improve the performance of local detection achieved based on $\mathcal{S}_j = \{([\mathbf{x}_j^T, t_n]^T, y_{jn}) : n = 1, \ldots, N_j\}$, and through collaboration with other sensors approach the global performance attainable if each sensor had available all other sensors data. Stringent power limitations prevent sensor j to send its set $\mathcal{S}_j$ to all other sensors or to an FC, if the latter is available. If these sensors are acoustic and are deployed to classify underwater unmanned vehicles, divers or submarines, then low transmission bandwidth and multipath effects further discourage incremental communication of the local data sets to an FC (Akyildiz et al., 2005).

Example 2 *(Distributed medical databases).* Suppose that $\mathcal{S}_j$ are patient data records stored at a hospital j. Each $\mathbf{x}_{jn}$ here contains patient descriptors (e.g., age, sex or blood pressure), and y_{jn} is a particular diagnosis (e.g., the patient is diabetic or not). The objective is to automatically diagnose

2. Although not included in this model for simplicity, K-ary alphabets $\mathcal{Y}$ with $K > 2$ can be considered as well.

(classify) a patient arriving at hospital j with descriptor $\mathbf{x}$, using all available data $\{\mathcal{S}_j\}_{j=1}^J$, rather than $\mathcal{S}_j$ alone. However, a nonchalant exchange of database entries $(\mathbf{x}_{jn}^T, y_{jn})$ can pose a privacy risk for the information exchanged. Moreover, a large percentage of medical information may require exchanging high resolution images. Thus, communicating and processing large amounts of high-dimensional medical data at an FC may be computationally prohibitive.

Example 3 *(Collaborative data mining).* Consider two different government agencies, a local agency A and a nation-wide agency B, with corresponding databases $\mathcal{S}_A$ and $\mathcal{S}_B$. Both agencies are willing to collaborate in order to classify jointly possible security threats. However, lower clearance level requirements at agency A prevents agency B from granting agency A open access to $\mathcal{S}_B$. Furthermore, even if an agreement granting temporary access to agency A were possible, databases $\mathcal{S}_A$ and $\mathcal{S}_B$ are confined to their current physical locations due to security policies.

If $\{\mathcal{S}_j\}_{j=1}^J$ were all centrally available at an FC, then the global variables $\mathbf{w}^*$ and b^* describing the *centralized* maximum-margin linear discriminant function $g^*(\mathbf{x}) = \mathbf{x}^T\mathbf{w}^* + b^*$ could be obtained by solving the convex optimization problem; see, for example, Schölkopf and Smola (2002, Ch. 7)

$$\begin{aligned} \{\mathbf{w}^*, b^*\} = \arg \min_{\mathbf{w}, b, \{\xi_{jn}\}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{j=1}^{J}\sum_{n=1}^{N_j}\xi_{jn} \\ \text{s.t.} \quad & y_{jn}(\mathbf{w}^T\mathbf{x}_{jn} + b) \geq 1 - \xi_{jn} \quad \forall j \in \mathcal{J},\ n = 1, \ldots, N_j \\ & \xi_{jn} \geq 0 \qquad\qquad\qquad\qquad \forall j \in \mathcal{J},\ n = 1, \ldots, N_j \end{aligned} \tag{1}$$

where the slack variables ξ_{jn} account for non-linearly separable training sets, and C is a tunable positive scalar.

Nonlinear discriminant functions $g(\mathbf{x})$ can also be found along the lines of (1) after mapping vectors $\mathbf{x}_{jn}$ to a higher dimensional space $\mathcal{H} \subseteq \mathbb{R}^P$, with $P > p$, via a nonlinear transformation $\phi : \mathcal{X} \to \mathcal{H}$. The generalized maximum-margin linear classifier in $\mathcal{H}$ is then obtained after replacing $\mathbf{x}_{jn}$ with $\phi(\mathbf{x}_{jn})$ in (1), and solving the following optimization problem

$$\begin{aligned} \{\mathbf{w}^*, b^*\} = \arg \min_{\mathbf{w}, b, \{\xi_{jn}\}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{j=1}^{J}\sum_{n=1}^{N_j}\xi_{jn} \\ \text{s.t.} \quad & y_{jn}(\mathbf{w}^T\phi(\mathbf{x}_{jn}) + b) \geq 1 - \xi_{jn} \quad \forall j \in \mathcal{J},\ n = 1, \ldots, N_j \\ & \xi_{jn} \geq 0 \qquad\qquad\qquad\qquad \forall j \in \mathcal{J},\ n = 1, \ldots, N_j. \end{aligned} \tag{2}$$

Problem (2) is typically tackled by solving its dual. Letting λ_{jn} denote the Lagrange multiplier corresponding to the constraint $y_{jn}(\mathbf{w}^T\phi(\mathbf{x}_{jn}) + b) \geq 1 - \xi_{jn}$, the dual problem of (2) is:

$$\begin{aligned} \max_{\{\lambda_{jn}\}} \quad & -\frac{1}{2}\sum_{j=1}^{J}\sum_{i=1}^{J}\sum_{n=1}^{N_j}\sum_{m=1}^{N_i}\lambda_{jn}\lambda_{im}y_{jn}y_{im}\phi^T(\mathbf{x}_{jn})\phi(\mathbf{x}_{im}) + \sum_{j=1}^{J}\sum_{n=1}^{N_j}\lambda_{jn} \\ \text{s.t.} \quad & \sum_{j=1}^{J}\sum_{n=1}^{N_j}\lambda_{jn}y_{jn} = 0 \\ & 0 \leq \lambda_{jn} \leq C \qquad\qquad \forall j \in \mathcal{J},\ n = 1, \ldots, N_j. \end{aligned} \tag{3}$$

Using the Lagrange multipliers λ_{jn}^* optimizing (3), and the Karush-Kuhn-Tucker (KKT) optimality conditions, the optimal classifier parameters can be expressed as

$$\begin{aligned} \mathbf{w}^* &= \sum_{j=1}^{J} \sum_{n=1}^{N_j} \lambda_{jn}^* y_{jn} \phi(\mathbf{x}_{jn}), \\ b^* &= y_{jn} - \mathbf{w}^{*T} \phi(\mathbf{x}_{jn}) \end{aligned} \tag{4}$$

with $\mathbf{x}_{jn}$ in (4) satisfying $\lambda_{jn}^* \in (0, C)$. Training vectors corresponding to non-zero λ_{jn}^*'s constitute the SVs. Once the SVs are identified, all other training vectors with $\lambda_{jn}^* = 0$ can be discarded since they do not contribute to $\mathbf{w}^*$. From this vantage point, SVs are the most informative elements of the training set. Solving (3) does not require knowledge of ϕ but only inner product values $\phi^T(\mathbf{x}_{jn})\phi(\mathbf{x}_{im}) := K(\mathbf{x}_{jn}, \mathbf{x}_{im})$, which can be computed through a pre-selected positive semi-definite kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$; see, for example, Schölkopf and Smola (2002, Ch. 2). Although not explicitly given, the optimal slack variables ξ_{jn}^* can be found through the KKT conditions of (2) in terms of λ_{jn}^* (Schölkopf and Smola, 2002). The optimal discriminant function can be also expressed in terms of kernels as

$$g^*(\mathbf{x}) = \sum_{j=1}^{J} \sum_{n=1}^{N_j} \lambda_{jn}^* y_{jn} K(\mathbf{x}_{jn}, \mathbf{x}) + b^* \tag{5}$$

where $b^* = y_{jn} - \sum_{i=1}^{J} \sum_{m=1}^{N_i} \lambda_{im}^* y_{im} K(\mathbf{x}_{im}, \mathbf{x}_{jn})$ for any SV $\mathbf{x}_{jn}$ with $\lambda_{jn}^* \in (0, C)$. This so-called kernel trick allows finding maximum-margin linear classifiers in higher dimensional spaces without explicitly operating in such spaces (Schölkopf and Smola, 2002).

The objective here is to develop *fully distributed* solvers of the centralized problems in (1) and (2) while guaranteeing performance approaching that of a centralized equivalent SVM. Although incremental solvers are possible, the size of information exchanges required might be excessive, especially if the number of SVs per node is large (Flouri et al., 2008; Lu et al., 2008). Recall that exchanging all local SVs among all nodes in the network several times is necessary for incremental DSVMs to approach the optimal centralized solution. Moreover, incremental schemes require a Hamiltonian cycle in the network to be identified in order to minimize the communication overhead. Computing such a cycle is an NP-hard task and in most cases a sub-optimal cycle is used at the expense of increased communication overhead. In other situations, communicating SVs directly might be prohibited because of the sensitivity of the information bore, as already mentioned in Examples 2 and 3.

3. Distributed Linear Support Vector Machine

This section presents a reformulation of the maximum-margin linear classifier problem in (1) to an equivalent distributed form, which can be solved using the alternating direction method of multipliers (ADMoM) outlined in Appendix A. (For detailed exposition of the ADMoM, see, for example, Bertsekas and Tsitsiklis, 1997.)

To this end, consider replacing the common (coupling) variables $(\mathbf{w}, b)$ in (1) with auxiliary per-node variables $\{(\mathbf{w}_j, b_j)\}_{j=1}^{J}$, and adding consensus constraints to force these variables to agree across neighboring nodes. With proper scaling of the cost by J, the proposed *consensus-based*

reformulation of (1) becomes

$$\begin{aligned} \min_{\{\mathbf{w}_j, b_j, \xi_{jn}\}} \quad & \frac{1}{2}\sum_{j=1}^{J} \|\mathbf{w}_j\|^2 + JC\sum_{j=1}^{J}\sum_{n=1}^{N_j} \xi_{jn} \\ \text{s.t.} \quad & y_{jn}(\mathbf{w}_j^T \mathbf{x}_{jn} + b_j) \geq 1 - \xi_{jn} \qquad \forall j \in \mathcal{J},\ n = 1,\ldots,N_j \\ & \xi_{jn} \geq 0 \qquad \forall j \in \mathcal{J},\ n = 1,\ldots,N_j \\ & \mathbf{w}_j = \mathbf{w}_i,\ b_j = b_i \qquad \forall j \in \mathcal{J},\ i \in \mathcal{B}_j. \end{aligned} \tag{6}$$

From a high-level view, problem (6) can be solved in a distributed fashion because each node j can optimize only the j-dependent terms of the cost, and also meet all the consensus constraints $\mathbf{w}_j = \mathbf{w}_i,\ b_j = b_i$, by exchanging messages only with nodes i in its neighborhood $\mathcal{B}_j$. What is more, network connectivity ensures that consensus in neighborhoods enables network-wide consensus. And thus, as the ensuing lemma asserts, solving (6) is equivalent to solving (1) so long as the network remains connected.

Lemma 1 *If $\{(\mathbf{w}_j, b_j)\}_{j=1}^{J}$ denotes a feasible solution of* (6)*, and the graph $\mathcal{G}$ is connected, then problems* (1) *and* (6) *are equivalent, that is, $\mathbf{w}_j = \mathbf{w}$ and $b_j = b\ \forall j = 1,\ldots,J$, where $(\mathbf{w}, b)$ is a feasible solution of* (1)*.*

Proof See Appendix B. ■

To specify how (6) can be solved using the ADMoM, define for notational brevity the augmented vector $\mathbf{v}_j := [\mathbf{w}_j^T,\ b_j]^T$, the augmented matrix $\mathbf{X}_j := [[\mathbf{x}_{j1},\ldots,\mathbf{x}_{jN_j}]^T, \mathbf{1}_j]$, the diagonal label matrix $\mathbf{Y}_j := \text{diag}([y_{j1},\ldots,y_{jN_j}])$, and the vector of slack variables $\xi_j := [\xi_{j1},\ldots,\xi_{jN_j}]^T$. With these definitions, it follows readily that $\mathbf{w}_j = (\mathbf{I}_{p+1} - \Pi_{p+1})\mathbf{v}_j$, where Π_{p+1} is a $(p+1)\times(p+1)$ matrix with zeros everywhere except for the $(p+1, p+1)$-st entry, given by $[\Pi_{p+1}]_{(p+1)(p+1)} = 1$. Thus, problem (6) can be rewritten as

$$\begin{aligned} \min_{\{\mathbf{v}_j, \xi_j, \omega_{ji}\}} \quad & \frac{1}{2}\sum_{j=1}^{J} \mathbf{v}_j^T(\mathbf{I}_{p+1} - \Pi_{p+1})\mathbf{v}_j + JC\sum_{j=1}^{J} \mathbf{1}_j^T \xi_j \\ \text{s.t.} \quad & \mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j \succeq \mathbf{1}_j - \xi_j \qquad \forall j \in \mathcal{J} \\ & \xi_j \succeq \mathbf{0}_j \qquad \forall j \in \mathcal{J} \\ & \mathbf{v}_j = \omega_{ji},\ \omega_{ji} = \mathbf{v}_i \qquad \forall j \in \mathcal{J},\ \forall i \in \mathcal{B}_j \end{aligned} \tag{7}$$

where the redundant variables $\{\omega_{ji}\}$ will turn out to facilitate the decoupling of the classifier parameters $\mathbf{v}_j$ at node j from those of their neighbors at neighbors $i \in \mathcal{B}_j$.

As in the centralized case, problem (7) will be solved through its dual. Toward this objective, let α_{ji1} (α_{ji2}) denote the Lagrange multipliers corresponding to the constraint $\mathbf{v}_j = \omega_{ji}$ (respectively $\omega_{ji} = \mathbf{v}_i$), and consider what we term surrogate augmented Lagrangian function

$$\begin{aligned} \mathcal{L}(\{\mathbf{v}_j\},\{\xi_j\},\{\omega_{ji}\},\{\alpha_{jik}\}) = & \frac{1}{2}\sum_{j=1}^{J} \mathbf{v}_j^T(\mathbf{I}_{p+1} - \Pi_{p+1})\mathbf{v}_j + JC\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j + \sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ji1}^T(\mathbf{v}_j - \omega_{ji}) \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ji2}^T(\omega_{ji} - \mathbf{v}_i) + \frac{\eta}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\|\mathbf{v}_j - \omega_{ji}\|^2 + \frac{\eta}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\|\omega_{ji} - \mathbf{v}_i\|^2 \end{aligned} \tag{8}$$

where the adjective "surrogate" is used because $\mathcal{L}$ does not include the set of constraints $\mathcal{W} := \{\mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j \succeq \mathbf{1}_j - \xi_j,\ \xi_j \succeq \mathbf{0}_j\}$, and the adjective "augmented" because $\mathcal{L}$ includes two quadratic terms (scaled by the tuning constant $\eta > 0$) to further regularize the equality constraints in (7). The role of these quadratic terms $||\mathbf{v}_j - \omega_{ji}||^2$ and $||\omega_{ji} - \mathbf{v}_i||^2$ is twofold: (a) they effect *strict* convexity of $\mathcal{L}$ with respect to (w.r.t.) ω_{ji}, and thus ensure convergence to the unique optimum of the global cost (whenever possible), even when the local costs are convex but not strictly so; and (b) through the scalar η, they allow one to trade off speed of convergence for steady-state approximation error (Bertsekas and Tsitsiklis, 1997, Ch. 3).

Consider now solving (7) iteratively by minimizing $\mathcal{L}$ in a cyclic fashion with respect to one set of variables while keeping all other variables fixed. The multipliers $\{\alpha_{ji1}, \alpha_{ji2}\}$ must be also updated per iteration using gradient ascent. The iterations required per node j are summarized in the following lemma.

Lemma 2 *The distributed iterations solving* (7) *are*

$$\{\mathbf{v}_j(t+1), \xi_j(t+1)\} = \arg \min_{\{\mathbf{v}_j, \xi_j\} \in \mathcal{W}} \mathcal{L}(\{\mathbf{v}_j\}, \{\xi_j\}, \{\omega_{ji}(t)\}, \{\alpha_{jik}(t)\}), \tag{9}$$

$$\{\omega_{ji}(t+1)\} = \arg \min_{\{\omega_{ji}\}} \mathcal{L}(\{\mathbf{v}_j(t+1)\}, \{\xi_j(t+1)\}, \{\omega_{ji}\}, \{\alpha_{jik}(t)\}), \tag{10}$$

$$\alpha_{ji1}(t+1) = \alpha_{ji1}(t) + \eta(\mathbf{v}_j(t+1) - \omega_{ji}(t+1)) \qquad \forall j \in \mathcal{J},\ \forall i \in \mathcal{B}_j, \tag{11}$$

$$\alpha_{ji2}(t+1) = \alpha_{ji2}(t) + \eta(\omega_{ji}(t+1) - \mathbf{v}_i(t+1)) \qquad \forall j \in \mathcal{J},\ \forall i \in \mathcal{B}_j. \tag{12}$$

and correspond to the ADMoM solver reviewed in Appendix A.

Proof See Appendix C. ■

Lemma 2 links the proposed DSVM design with the convergent ADMoM solver, and thus ensures convergence of the novel MoM-DSVM to the centralized SVM classifier. However, for the particular problem at hand it is possible to simplify iterations (9)-(12). Indeed, simple inspection of (8) confirms that with all other variables fixed, the cost in (10) is linear-quadratic in ω_{ji}; hence, $\omega_{ji}(t+1)$ can be found in closed form per iteration, and the resulting closed-form expression can be substituted back to eliminate ω_{ji} from $\mathcal{L}$. Furthermore, Appendix D shows that the two sets of multipliers α_{ji1} and α_{ji2} can be combined into one set α_j after appropriate initialization of the iterations (11) and (12), as asserted by the following lemma.

Lemma 3 *Selecting* $\alpha_{ji1}(0) = \alpha_{ji2}(0) = 0_{(p+1)\times 1}$ *as initialization* $\forall j \in \mathcal{J}$, $\forall i \in \mathcal{B}_j$, *iterations* (9)-(12) *reduce to*

$$\{\mathbf{v}_j(t+1), \xi_j(t+1)\} = \arg \min_{\{\mathbf{v}_j, \xi_j\} \in \mathcal{W}} \mathcal{L}'(\{\mathbf{v}_j\}, \{\xi_j\}, \{\mathbf{v}_j(t)\}, \{\alpha_j(t)\}), \tag{13}$$

$$\alpha_j(t+1) = \alpha_j(t) + \frac{\eta}{2} \sum_{i \in \mathcal{B}_j} [\mathbf{v}_j(t+1) - \mathbf{v}_i(t+1)] \quad \forall j \in \mathcal{J} \tag{14}$$

where $\alpha_j(t) := \sum_{i\in\mathcal{B}_j} \alpha_{ji1}(t)$, *and* $\mathcal{L}'$ *is given by*

$$\mathcal{L}'(\{\mathbf{v}_j\},\{\xi_j\},\{\mathbf{v}_j(t)\},\{\alpha_j(t)\}) = \frac{1}{2}\sum_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j + JC\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j + 2\sum_{j=1}^{J}\alpha_j^T(t)\mathbf{v}_j + \eta\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{v}_j - \frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2. \quad (15)$$

Proof See Appendix D. ■

The optimization problem in (13) involves the reduced Lagrangian $\mathcal{L}'$ in (15), which is linear-quadratic in $\mathbf{v}_j$ and ξ_j. In addition, the constraint set $\mathcal{W}$ is linear in these variables. To solve this constrained minimization problem through its dual, let $\lambda_j := [\lambda_{j1},\ldots,\lambda_{jN_j}]^T$ denote Lagrange multipliers per node corresponding to the constraints $\mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j \succeq \mathbf{1}_j - \xi_j$. Solving the dual of (13) yields the optimal λ_j at iteration $t+1$, namely $\lambda_j(t+1)$, as a function of $\mathbf{v}_j(t)$ and $\alpha_j(t)$; while the KKT conditions provide expressions for $\mathbf{v}_j(t+1)$ as a function of $\alpha_j(t)$, and the optimal dual variables $\lambda_j(t+1)$. Notwithstanding, the resultant iterations are decoupled across nodes. These iterations and the associated convergence guarantees can be summarized as follows.

Proposition 1 *Consider the per node iterates* $\lambda_j(t)$, $\mathbf{v}_j(t)$ *and* $\alpha_j(t)$, *given by*

$$\lambda_j(t+1) = \arg\max_{\lambda_j:\,\mathbf{0}_j\preceq\lambda_j\preceq JC\mathbf{1}_j} -\frac{1}{2}\lambda_j^T\mathbf{Y}_j\mathbf{X}_j\mathbf{U}_j^{-1}\mathbf{X}_j^T\mathbf{Y}_j\lambda_j + \left(\mathbf{1}_j + \mathbf{Y}_j\mathbf{X}_j\mathbf{U}_j^{-1}\mathbf{f}_j(t)\right)^T\lambda_j, \quad (16)$$

$$\mathbf{v}_j(t+1) = \mathbf{U}_j^{-1}\left[\mathbf{X}_j^T\mathbf{Y}_j\lambda_j(t+1) - \mathbf{f}_j(t)\right], \quad (17)$$

$$\alpha_j(t+1) = \alpha_j(t) + \frac{\eta}{2}\sum_{i\in\mathcal{B}_j}[\mathbf{v}_j(t+1) - \mathbf{v}_i(t+1)] \quad (18)$$

where $\mathbf{U}_j := (1+2\eta|\mathcal{B}_j|)\mathbf{I}_{p+1} - \Pi_{p+1}$, $\mathbf{f}_j(t) := 2\alpha_j(t) - \eta\sum_{i\in\mathcal{B}_j}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]$, $\eta > 0$, *and arbitrary initialization vectors* $\lambda_j(0)$, $\mathbf{v}_j(0)$, *and* $\alpha_j(0) = \mathbf{0}_{(p+1)\times 1}$. *The iterate* $\mathbf{v}_j(t)$ *converges to the solution of* (7), *call it* $\mathbf{v}^*$, *as* $t\to\infty$; *that is,* $\lim_{t\to\infty}\mathbf{v}_j(t) = \mathbf{v}^*$.

Proof See Appendix E. ■

Similar to the centralized SVM algorithm, if $[\lambda_j(t)]_n \neq 0$, then $[\mathbf{x}_{jn}^T, 1]^T$ is an SV. Finding $\lambda_j(t+1)$ as in (16) requires solving a quadratic optimization problem similar to the one that a centralized SVM would solve, for example, via a gradient projection algorithm or an interior point method; see for example, Schölkopf and Smola (2002, Ch. 6). However, the number of variables involved in (16) per iteration per node is considerably smaller when compared to its centralized counterpart, namely N_j versus $\sum_{j=1}^{J} N_j$. Also, the optimal local slack variables ξ_j^* can be found via the KKT conditions for (13).

The ADMoM-based DSVM (MoM-DSVM) iterations (16)-(18) are summarized as Algorithm 1, and are illustrated in Figure 2. All nodes have available JC and η. Also, every node computes its local $N_j\times N_j$ matrix $\mathbf{Y}_j\mathbf{X}_j\mathbf{U}_j^{-1}\mathbf{X}_j^T\mathbf{Y}_j$, which remains unchanged throughout the entire algorithm. Every node then updates its local $(p+1)\times 1$ estimates $\mathbf{v}_j(t)$ and $\alpha_j(t)$; and the $N_j\times 1$ vector $\lambda_j(t)$.

At iteration $t+1$, node j computes vector $\mathbf{f}_j(t)$ locally to obtain its local $\lambda_j(t+1)$ via (16). Vector $\lambda_j(t+1)$ together with the local training set $\mathcal{S}_j$ are used at node j to compute $\mathbf{v}_j(t+1)$ via (17). Next, node j broadcasts its newly updated local estimates $\mathbf{v}_j(t+1)$ to all its one-hop neighbors $i \in \mathcal{B}_j$. Iteration $t+1$ resumes when every node updates its local $\alpha_j(t+1)$ vector via (18). Note that at any given iteration t of the algorithm, each node j can evaluate its own local discriminant function $g_j^{(t)}(\mathbf{x})$ for any vector $\mathbf{x} \in \mathcal{X}$ as

$$g_j^{(t)}(\mathbf{x}) = [\mathbf{x}^T, 1]\mathbf{v}_j(t) \tag{19}$$

which from Proposition 1 is guaranteed to converge to the same solution across all nodes as $t \to \infty$. Simulated tests in Section 5 will demonstrate that after a few iterations the classification performance of (19) outperforms that of the local discriminant function obtained based on the local training set alone. The effect of η on the convergence rate of MoM-DSVM will be tested numerically in Section 5.

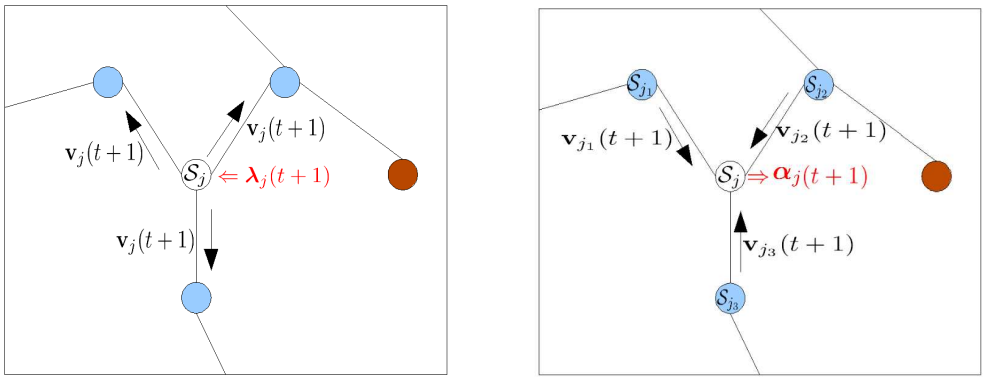


Figure 2: Visualization of iterations (16)-(18): (left) every node $j \in \mathcal{J}$ computes $\lambda_j(t+1)$ to obtain $\mathbf{v}_j(t+1)$, and then broadcasts $\mathbf{v}_j(t+1)$ to all neighbors $i \in \mathcal{B}_j$; (right) once every node $j \in \mathcal{J}$ has received $\mathbf{v}_i(t+1)$ from all $i \in \mathcal{B}_j$, it computes $\alpha_j(t+1)$.

Remark 1 *The messages exchanged among neighboring nodes in the MoM-DSVM algorithm correspond to local estimates $\mathbf{v}_j(t)$, which together with the local multiplier vectors $\alpha_j(t)$, convey sufficient information about the local training sets to achieve consensus globally. Per iteration and per node a message of* fixed size $(p+1)\times 1$ *is broadcasted (vectors α_j are not exchanged among nodes.) This is to be contrasted with incremental DSVM algorithms in, for example, Lu et al. (2008), Flouri et al. (2006) and Flouri et al. (2008), where the size of the messages exchanged between neighboring nodes depends on the number of SVs found at each incremental step. Although the SVs of each training set may be few, the overall number of SVs may remain large, thus consuming considerable power when transmitting SVs from one node to the next.*

Remark 2 *Real networks are prone to node failures, for example, sensors in a WSN may run out of battery during operation. Thanks to its fully decentralized mode of operation, the novel MoM-DSVM algorithm guarantees that the remaining nodes in the network will reach consensus as long as the*

Algorithm 1 MoM-DSVM

Require: Randomly initialize $\mathbf{v}_j(0)$, and $\alpha_j(0) = \mathbf{0}_{(p+1)\times 1}$ for every $j \in \mathcal{J}$

for $t = 0, 1, 2, \ldots$ **do**
  for all $j \in \mathcal{J}$ **do**
    Compute $\lambda_j(t+1)$ via (16).
    Compute $\mathbf{v}_j(t+1)$ via (17).
  end for
  for all $j \in \mathcal{J}$ **do**
    Broadcast $\mathbf{v}_j(t+1)$ to all neighbors $i \in \mathcal{B}_j$.
  end for
  for all $j \in \mathcal{J}$ **do**
    Compute $\alpha_j(t+1)$ via (18).
  end for
end for

node that fails, say $j_o \in \mathcal{J}$, does not correspond to a cut-vertex *of $\mathcal{G}$. In this case, the operational network graph $\mathcal{G}_o := \mathcal{G} - j_o$ remains connected, and thus surviving nodes can percolate information throughout $\mathcal{G}_o$. Of course, $\mathcal{S}_{j_o}$ will not participate in training the SVM. If j_o is a* cut-vertex *of $\mathcal{G}$, the algorithm will remain operational in each connected component of the resulting sub-graph $\mathcal{G}_o$, reaching consensus among nodes in each of the connected components.*

3.1 Online Distributed Support Vector Machine

In many distributed learning tasks data arrive sequentially, and possibly asynchronously. In addition, the processes to be learned may change with time. In such cases, training examples need to be added or removed from each local training set $\mathcal{S}_j$. Training sets of increasing of decreasing size can be expressed in terms of *time-varying* augmented data matrices $\mathbf{X}_j(t)$, and corresponding label matrices $\mathbf{Y}_j(t)$. An online version of DSVM is thus well motivated when a new training example $\mathbf{x}_{jn}(t)$ along with its label $y_{jn}(t)$ acquired at time t are incorporated into $\mathbf{X}_j(t)$ and $\mathbf{Y}_j(t)$, respectively. The corresponding modified iterations are given by (cf. (16)-(18))

$$\lambda_j(t+1) = \arg\max_{\lambda_j:\, \mathbf{0} \preceq \lambda_j \preceq JC\mathbf{1}_j(t+1)} -\frac{1}{2}\lambda_j^T \mathbf{Y}_j(t+1)\mathbf{X}_j(t+1)\mathbf{U}_j^{-1}\mathbf{X}_j(t+1)^T\mathbf{Y}_j(t+1)\lambda_j + \left(\mathbf{1}_j - \mathbf{Y}_j(t+1)\mathbf{X}_j(t+1)\mathbf{U}_j^{-1}\mathbf{f}_j(t)\right)^T \lambda_j, \quad (20)$$

$$\mathbf{v}_j(t+1) = \mathbf{U}_j^{-1}\left(\mathbf{X}_j(t+1)^T\mathbf{Y}_j(t+1)\lambda_j(t+1) - 2\alpha_j(t) + \eta\sum_{i\in\mathcal{B}_j}[\mathbf{v}_j(t) + \mathbf{v}_i(t)]\right), \quad (21)$$

$$\alpha_j(t+1) = \alpha_j(t) + \frac{\eta}{2}\sum_{i\in\mathcal{B}_j}[\mathbf{v}_j(t+1) - \mathbf{v}_i(t+1)]. \quad (22)$$

Note that the dimensionality of λ_j must vary to accommodate the variable number of $\mathcal{S}_j$ elements at every time instant t. The online MoM-DSVM classifier is summarized as Algorithm 2. For this algorithm to run, no conditions need to be imposed on how the sets $\mathcal{S}_j(t)$ increase or decrease. Their changes can be asynchronous and may comprise multiple training examples at once. In principle,

Algorithm 2 Online MoM-DSVM

Require: Randomly initialize $\mathbf{v}_j(0)$, and $\alpha_j(0) = \mathbf{0}_{(p+1)\times 1}$ for every $j \in \mathcal{J}$.

for $t = 0, 1, 2, \ldots$ **do**
  for all $j \in \mathcal{J}$ **do**
    Update $\mathbf{Y}_j(t+1)\mathbf{X}_j(t+1)\mathbf{U}_j^{-1}\mathbf{X}_j(t+1)^T\mathbf{Y}_j(t+1)$.
    Compute $\lambda_j(t+1)$ via (20).
    Compute $\mathbf{v}_j(t+1)$ via (21).
  end for
  for all $j \in \mathcal{J}$ **do**
    Broadcast $\mathbf{v}_j(t+1)$ to all neighbors $i \in \mathcal{B}_j$.
  end for
  for all $j \in \mathcal{J}$ **do**
    Compute $\alpha_j(t+1)$ via (22).
  end for
end for

the parameters η and C can also become time-dependent. The effect of these parameters will be discussed in Section 5.

Intuitively speaking, if the training sets remain invariant across a sufficient number of time instants, $\mathbf{v}_j(t)$ will closely track the optimal linear classifier. Rigorous convergence analysis of Algorithm 2 for any given rate of change of the training set goes beyond the scope of this work. Simulations will however demonstrate that the modified iterations in (20)-(22) are able to track changes in the training sets even when these occur at every time instant t.

Remark 3 *Compared to existing* centralized *online SVM alternatives in, for example, Cauwenberghs and Poggio (2000) and Fung and Mangasarian (2002), the online MoM-DSVM algorithm of this section allows seamless integration of both distributed and online processing. Nodes with training sets available at initialization and nodes that are acquiring their training sets online can be integrated to jointly find the maximum-margin linear classifier. Furthermore, whenever needed, the online MoM-DSVM can return a partially trained model constructed with examples available to the network at any given time. Likewise, elements of the training sets can be removed without having to restart the MoM-DSVM algorithm. This feature also allows adapting MoM-DSVM to jointly operate with algorithms that account for concept drift (Klinkenberg and Joachims, 2000). In the classification context, concept drift defines a change in the true classification boundaries between classes. In general, accounting for concept drift requires two main steps, which can be easily handled by the online MoM-DSVM: (i) acquisition of updated elements in the training set that better describe the current concept; and (ii) removal of outdated elements from the training set.*

4. Distributed Nonlinear Support Vector Machine

In Section 3, problem (1) was reformulated to allow all nodes to consent on $\mathbf{v}^*$. However, applying an identical reformulation to the nonlinear classification problem in (2) would require updates in

(16)-(18) to be carried in $\mathcal{H}$. As the dimensionality P of $\mathcal{H}$ increases, the local computation and communication complexities become increasingly prohibitive.

Our approach to mitigate this well-known "curse of dimensionality" is to enforce consensus of the local discriminants g_j^* on a subspace of reduced rank $L < P$. To this end, we project the consensus constraints corresponding to (2) and consider the optimization problem (cf. (6))

$$\begin{aligned} \min_{\{\mathbf{w}_j, b_j, \boldsymbol{\xi}_j\}} \quad & \frac{1}{2}\sum_{j=1}^{J}\|\mathbf{w}_j\|^2 + JC\sum_{j=1}^{J}\mathbf{1}_j^T\boldsymbol{\xi}_j \\ \text{s.t.} \quad & \mathbf{Y}_j(\boldsymbol{\Phi}(\mathbf{X}_j)\mathbf{w}_j + \mathbf{1}_j b_j) \succeq \mathbf{1}_j - \boldsymbol{\xi}_j \quad \forall j \in \mathcal{J} \\ & \boldsymbol{\xi}_j \succeq \mathbf{0}_j \quad \forall j \in \mathcal{J} \\ & \mathbf{G}\mathbf{w}_j = \mathbf{G}\mathbf{w}_i \quad \forall j \in \mathcal{J}, i \in \mathcal{B}_j \\ & b_j = b_i \quad \forall j \in \mathcal{J}, i \in \mathcal{B}_j \end{aligned} \tag{23}$$

where $\boldsymbol{\Phi}(\mathbf{X}_j) := [\phi(\mathbf{x}_{j1}), \ldots, \phi(\mathbf{x}_{jN_j})]^T$, and $\mathbf{G} := [\phi(\boldsymbol{\chi}_1), \ldots, \phi(\boldsymbol{\chi}_L)]^T$ is a fat $L \times P$ matrix common to all nodes with preselected vectors $\{\boldsymbol{\chi}_l\}_{l=1}^L$ specifying its rows. Each $\boldsymbol{\chi}_l \in \mathcal{X}$ corresponds to a $\phi(\boldsymbol{\chi}_l) \in \mathcal{H}$, which at the optimal solution $\{\mathbf{w}_j^*, b_j^*\}_{j=1}^J$ of (23), satisfies $\phi^T(\boldsymbol{\chi}_l)\mathbf{w}_1^* = \cdots = \phi^T(\boldsymbol{\chi}_l)\mathbf{w}_J^* = \phi^T(\boldsymbol{\chi}_l)\mathbf{w}^*$. The projected constraints $\{\mathbf{G}\mathbf{w}_j = \mathbf{G}\mathbf{w}_i\}$ along with $\{b_j = b_i\}$ force all nodes to agree on the value of the local discriminant functions $g_j^*(\boldsymbol{\chi}_l)$ at the vectors $\{\boldsymbol{\chi}_l\}_{l=1}^L$, but not necessarily for all $\mathbf{x} \in \mathcal{X}$. This is the price paid for reducing the computational complexity of (23) to an affordable level. Clearly, the choice of vectors $\{\boldsymbol{\chi}_l\}_{l=1}^L$, their number L, and the local training sets $\mathcal{S}_j$ determine how similar the local discriminant functions g_j^* are. If $\mathbf{G} = \mathbf{I}_P$, then (23) reduces to (6), and $g_1^*(\mathbf{x}) = \ldots = g_J^*(\mathbf{x}) = g^*(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$, but the high dimensionality challenge appears. At the end of this section, we will provide different design choices for $\{\boldsymbol{\chi}_l\}_{l=1}^L$, and test them via simulations in Section 5.

Because the cost in (23) is strictly convex w.r.t. $\mathbf{w}_j$, it guarantees that the set of optimal vectors $\{\mathbf{w}_j^*\}$ is unique even when $\mathbf{G}$ is a 'fat' matrix ($L < P$) and/or ill-conditioned (Bertsekas, 1999, Prop. 5.2.1). As in (2), having $\{\mathbf{w}_j^*\}$ known is of limited use, since the mapping ϕ may be unknown, or if known, evaluating vectors $\phi(\mathbf{x})$ may entail an excessive computational cost. Fortunately, the resulting discriminant function $g_j^*(\mathbf{x})$ admits a reduced-complexity solution because it can be expressed in terms of kernels, as shown by the following theorem.

Theorem 4 *For every positive semi-definite kernel $K(\cdot,\cdot)$, the discriminant functions $g_j^*(\mathbf{x}) = \phi^T(\mathbf{x})\mathbf{w}_j^* + b_j^*$ with $\{\mathbf{w}_j^*, b_j^*\}$ denoting the optimal solution of* (23)*, can be written as*

$$g_j^*(\mathbf{x}) = \sum_{n=1}^{N_j} a_{jn}^* K(\mathbf{x}, \mathbf{x}_{jn}) + \sum_{l=1}^{L} c_{jl}^* K(\mathbf{x}, \boldsymbol{\chi}_l) + b_j^*, \ \forall j \in \mathcal{J} \tag{24}$$

where $\{a_{jn}^*\}$ and $\{c_{jl}^*\}$ are real-valued scalar coefficients.

Proof See Appendix F. ■

The space of functions g_j described by (24) is fully determined by the span of the kernel function $K(\cdot,\cdot)$ centered at training vectors $\{\mathbf{x}_{jn}, n = 1, \ldots, N_j\}$ per node, and also at the vectors $\{\boldsymbol{\chi}_l\}_{l=1}^L$ which are common to all nodes. Thus, similarity of the discriminant functions g_j^* across nodes is

naturally constrained by the corresponding $\mathcal{S}_j$. Theorem 1 also reveals the effect of $\{\chi_l\}_{l=1}^L$ on the $\{g_j^*\}$. By introducing vectors $\{\chi_l\}_{l=1}^L$ common to all nodes, a subset of basis functions common to all local functional spaces is introduced a fortiori. Coefficients a_{jn}^* and c_{jl}^* are found so that all local discriminants g_j^* agree on their values at points $\{\chi_l\}_{l=1}^L$. Intuitively, at every node these coefficients summarize the global information available to the network.

Theorem 1 is an existence result whereby each nonlinear discriminant function g_j^* is expressible in terms of $\mathcal{S}_j$ and $\{\chi_l\}_{l=1}^L$. However, finding the coefficients a_{jn}^*, c_{jl}^* and b_j^* in a distributed fashion remains an issue. Next, it is shown that these coefficients can be obtained iteratively by applying the ADMoM solver to (23).

Similar to (7), introduce auxiliary variables $\{\omega_{ji}\}$ ($\{\zeta_{ji}\}$) to decouple the constraints $\mathbf{G}\mathbf{w}_j = \mathbf{G}\mathbf{w}_i$ ($b_j = b_i$) across nodes, and α_{jik} (β_{jik}) denote the corresponding Lagrange multipliers (cf. (8)). The surrogate augmented Lagrangian for problem (23) is then

$$\begin{aligned}\mathcal{L}(\{\mathbf{w}_j\},\{\xi_j\},\{\omega_{ji}\},\{\alpha_{jik}\},\{\zeta_{ji}\},\{\beta_{jik}\}) &= \frac{1}{2}\sum_{j=1}^J \|\mathbf{w}_j\|^2 + JC\sum_{j=1}^J \mathbf{1}_j^T\xi_j + \sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\alpha_{ji1}^T(\mathbf{G}\mathbf{w}_j-\omega_{ji}) \\ &+ \sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\alpha_{ji2}^T(\omega_{ji}-\mathbf{G}\mathbf{w}_i) + \sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\beta_{ji1}(b_j-\zeta_{ji}) + \sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\beta_{ji2}(\zeta_{ji}-b_i) \\ &+ \frac{\eta}{2}\sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\|\mathbf{G}\mathbf{w}_j-\omega_{ji}\|^2 + \frac{\eta}{2}\sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\|\omega_{ji}-\mathbf{G}\mathbf{w}_i\|^2 + \frac{\eta}{2}\sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\|b_j-\zeta_{ji}\|^2 + \frac{\eta}{2}\sum_{j=1}^J\sum_{i\in\mathcal{B}_j}\|\zeta_{ji}-b_i\|^2.\end{aligned}$$

Following the steps of Lemma 2, and with $\{\alpha_{jik}\}$ and $\{\beta_{jik}\}$ initialized at zero, the ADMoM iterations take the form

$$\{\mathbf{w}_j(t+1), b_j(t+1), \xi_j(t+1)\} = \arg\min_{\{\mathbf{w}_j,b_j,\xi_j\}\in\mathcal{W}} \mathcal{L}'(\{\mathbf{w}_j\},\{b_j\},\{\xi_j\},\{\alpha_j(t)\},\{\beta_j(t)\}), \tag{25}$$

$$\alpha_j(t+1) = \alpha_j(t) + \frac{\eta}{2}\sum_{i\in\mathcal{B}_j}\mathbf{G}[\mathbf{w}_j(t+1) - \mathbf{w}_i(t+1)], \tag{26}$$

$$\beta_j(t+1) = \beta_j(t) + \frac{\eta}{2}\sum_{i\in\mathcal{B}_j}[b_j(t+1) - b_i(t+1)] \tag{27}$$

where $\mathcal{L}'$ is defined similar to (8), $\alpha_j(t)$ as in Lemma 3, and $\beta_j(t) := \sum_{i\in\mathcal{B}_j}\beta_{ji1}(t)$. The ADMoM iterations (25)-(27) will not be explicitly solved since iterates $\mathbf{w}_j(t)$ lie in the high-dimensional space $\mathcal{H}$. Nevertheless, our objective is not to find $\mathbf{w}_j^*$, but rather the discriminant function $g_j^*(\mathbf{x})$. To this end, let $\Gamma := [\chi_1,\ldots,\chi_L]^T$, and define the kernel matrices with entries

$$[\mathbf{K}(\mathbf{X}_j,\mathbf{X}_j)]_{n,m} := K(\mathbf{x}_{jn},\mathbf{x}_{jm}), \tag{28}$$

$$[\mathbf{K}(\mathbf{X}_j,\Gamma)]_{n,l} := K(\mathbf{x}_{jn},\chi_l), \tag{29}$$

$$[\mathbf{K}(\Gamma,\Gamma)]_{l,l'} := K(\chi_l,\chi_{l'}). \tag{30}$$

From Theorem 1 it follows that each local update $g_j^{(t)}(\mathbf{x}) = \phi^T(\mathbf{x})\mathbf{w}_j(t) + b_j(t)$ admits per iteration t a solution expressed in terms of kernels. The latter is specified by the coefficients $\{a_{jn}(t)\}$, $\{c_{jl}(t)\}$ and $\{b_j(t)\}$ that can be obtained in closed form, as shown in the next proposition.

Proposition 2 *Let* $\lambda_j := [\lambda_{j1},\ldots,\lambda_{jN_j}]^T$ *denote the Lagrange multiplier corresponding to the constraint* $\mathbf{Y}_j(\Phi(\mathbf{X}_j)\mathbf{w}_j+\mathbf{1}_j b_j) \succeq \mathbf{1}_j-\xi_j$, *and* $\widetilde{\mathbf{w}}_j(t) := \mathbf{G}\mathbf{w}_j(t)$. *The local discriminant function* $g_j^{(t)}(\mathbf{x})$ *at iteration* t *is*

$$g_j^{(t)}(\mathbf{x}) = \sum_{n=1}^{N_j} a_{jn}(t)K(\mathbf{x},\mathbf{x}_{jn}) + \sum_{l=1}^{L} c_{jl}(t)K(\mathbf{x},\chi_l) + b_j(t) \tag{31}$$

where $\mathbf{a}_j(t) := [a_{j1}(t),\ldots,a_{jN_j}(t)]^T$, $\mathbf{c}_j(t) := [c_{j1}(t),\ldots,c_{jL}(t)]^T$, *and* $b_j(t)$ *are given by*

$$\mathbf{a}_j(t) := \mathbf{Y}_j\lambda_j(t), \tag{32}$$

$$\mathbf{c}_j(t) := 2\eta|\mathcal{B}_j|\widetilde{\mathbf{U}}_j^{-1}\left[\mathbf{K}(\Gamma,\Gamma)\mathbf{f}_j(t)-\mathbf{K}(\Gamma,\mathbf{X}_j)\mathbf{Y}_j\lambda_j(t)\right]-\widetilde{\mathbf{f}}_j(t), \tag{33}$$

$$b_j(t) := \frac{1}{2\eta|\mathcal{B}_j|}\left[\mathbf{1}_j^T\mathbf{Y}_j\lambda_j(t)-h_j(t)\right] \tag{34}$$

with $\lambda_j(t)$ *denoting the vector multiplier update available at iteration* t, $\widetilde{\mathbf{U}}_j := \mathbf{I}_L+2\eta|\mathcal{B}_j|\mathbf{K}(\Gamma,\Gamma)$, $\widetilde{\mathbf{f}}_j(t) := 2\alpha_j(t)-\eta\sum_{i\in\mathcal{B}_j}[\widetilde{\mathbf{w}}_j(t)+\widetilde{\mathbf{w}}_i(t)]$ *and* $h_j(t) := 2\beta_j(t)-\eta\sum_{i\in\mathcal{B}_j}[b_j(t)+b_i(t)]$.

Proof See Appendix G. ■

Proposition 2 asserts that in order to find $\mathbf{a}_j(t)$, $\mathbf{c}_j(t)$ and $b_j(t)$ in (32), (33) and (34), it suffices to obtain $\lambda_j(t)$, $\widetilde{\mathbf{w}}_j(t)$, $b_j(t)$, $\alpha_j(t)$, and $\beta_j(t)$. Note that finding the $L\times 1$ vector $\widetilde{\mathbf{w}}_j(t)$ from $\mathbf{w}_j(t)$ incurs complexity of order $O(L)$. The next proposition shows how to iteratively update $\lambda_j(t)$, $\widetilde{\mathbf{w}}_j(t)$, $b_j(t)$, $\alpha_j(t)$, and $\beta_j(t)$ in a distributed fashion.

Proposition 3 *The iterates* $\lambda_j(t)$, $\widetilde{\mathbf{w}}_j(t)$, $b_j(t)$, $\alpha_j(t)$ *and* $\beta_j(t)$ *can be obtained as*

$$\begin{aligned}\lambda_j(t+1) = \arg\max_{\lambda_j:\,\mathbf{0}_j\preceq\lambda_j\preceq JC\mathbf{1}_j} &-\frac{1}{2}\lambda_j^T\mathbf{Y}_j\left(\mathbf{K}(\mathbf{X}_j,\mathbf{X}_j)-\widetilde{\mathbf{K}}(\mathbf{X}_j,\mathbf{X}_j)+\frac{\mathbf{1}_j\mathbf{1}_j^T}{2\eta|\mathcal{B}_j|}\right)\mathbf{Y}_j\lambda_j+\mathbf{1}_j^T\lambda_j\\ &-\left(\widetilde{\mathbf{f}}_j^T(t)\left(\mathbf{K}(\Gamma,\mathbf{X}_j)-\widetilde{\mathbf{K}}(\Gamma,\mathbf{X}_j)\right)+h_j(t)\frac{\mathbf{1}^T}{2\eta|\mathcal{B}_j|}\right)\mathbf{Y}_j\lambda_j,\end{aligned} \tag{35}$$

$$\widetilde{\mathbf{w}}_j(t+1) = \left[\mathbf{K}(\Gamma,\mathbf{X}_j)-\widetilde{\mathbf{K}}(\Gamma,\mathbf{X}_j)\right]\mathbf{Y}_j\lambda_j(t+1)-\left[\mathbf{K}(\Gamma,\Gamma)-\widetilde{\mathbf{K}}(\Gamma,\Gamma)\right]\widetilde{\mathbf{f}}_j(t), \tag{36}$$

$$b_j(t+1) = \frac{1}{2\eta|\mathcal{B}_j|}\left[\mathbf{1}_j^T\mathbf{Y}_j\lambda_j(t+1)-h_j(t)\right], \tag{37}$$

$$\alpha_j(t+1) = \alpha_j(t)+\frac{\eta}{2}\sum_{i\in\mathcal{B}_j}[\widetilde{\mathbf{w}}_j(t+1)-\widetilde{\mathbf{w}}_i(t+1)], \tag{38}$$

$$\beta_j(t+1) = \beta_j(t)+\frac{\eta}{2}\sum_{i\in\mathcal{B}_j}[b_j(t+1)-b_i(t+1)] \tag{39}$$

where $\widetilde{\mathbf{K}}(\mathbf{Z},\mathbf{Z}') := 2\eta|\mathcal{B}_j|\mathbf{K}(\mathbf{Z},\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\mathbf{Z}')$. *With arbitrary initialization* $\lambda_j(0)$, $\widetilde{\mathbf{w}}_j(0)$, *and* $b_j(0)$*; and* $\alpha_j(0)=\mathbf{0}_{L\times 1}$ *and* $\beta_j(0)=0$, *the iterates* $\{a_{jn}(t)\}$, $\{c_{jl}(t)\}$ *and* $\{b_j(t)\}$ *in* (32), (33) *and* (34) *converge to* $\{a_{jn}^*\}$, $\{c_{jl}^*\}$ *and* $\{b_j^*\}$ *in* (24), *as* $t\to\infty$, $\forall\, j\in\mathcal{J}, n=1,\ldots,N_j$, *and* $l=1,\ldots,L$.

Proof See Appendix H. ■

Algorithm 3 MoM-NDSVM

Require: Randomly initialize $\widetilde{\mathbf{w}}_j(0)$ and $b_j(0)$; and $\alpha_j(0)=\mathbf{0}_{L\times 1}$ and $\beta_j(0)=0$ for every $j\in\mathcal{J}$.

for $t=0,1,2,\ldots$ **do**
  for all $j\in\mathcal{J}$ **do**
    Compute $\lambda_j(t+1)$ via (35).
    Compute $\widetilde{\mathbf{w}}_j(t+1)$ via (36).
    Compute $b_j(t+1)$ via (37).
  end for
  for all $j\in\mathcal{J}$ **do**
    Broadcast $\widetilde{\mathbf{w}}_j(t+1)$ and $b_j(t+1)$ to all neighbors $i\in\mathcal{B}_j$.
  end for
  for all $j\in\mathcal{J}$ **do**
    Compute $\alpha_j(t+1)$ via (38).
    Compute $\beta_j(t+1)$ via (39).
    Compute $\mathbf{a}_j(t)$, $\mathbf{c}_j(t)$ and $b_j(t)$ via (32), (33) and (34), respectively.
  end for
end for

The iterations comprising the ADMoM-based non-linear DSVM (MoM-NDSVM) are summarized as Algorithm 3. It is important to stress that Algorithm 3 starts by having all nodes agree on the common quantities Γ, JC, η, and $K(\cdot,\cdot)$. Also, each node computes its local kernel matrices as in (28)-(30), which remain unchanged throughout. Subsequently, Algorithm 3 runs in a manner analogous to Algorithm 1, with the difference that every node communicates an $(L+1)\times 1$ vector (instead of $(p+1)\times 1$) for its neighbors to receive $\widetilde{\mathbf{w}}_j(t)$ and $b_j(t)$.

4.1 On the Optimality of NDSVM and the Selection of Common Vectors

By construction, Algorithm 3 produces local discriminant functions whose predictions for $\{\chi_l\}_{l=1}^L$ are the same for all nodes in the network; that is, $g_1^*(\chi_l)=\ldots=g_J^*(\chi_l)=g^*(\chi_l)$ for $l=1,\ldots,L$, where $g^*(\chi_l)=\phi^T(\chi_l)\mathbf{w}^*+b^*$, and $\{\mathbf{w}^*,b^*\}$ are the optimal solution of the centralized problem (2). Viewing $\{\chi_l\}_{l=1}^L$ as a classification query, the proposed MoM-NDSVM algorithm can be implemented as follows. Having this query presented at any node j entailing a set of unlabeled vectors $\{\chi_l\}_{l=1}^L$, the novel scheme first percolates $\{\chi_l\}_{l=1}^L$ throughout the network.[3] Problem (23) is subsequently solved in a distributed fashion using Algorithm 3. Notice that in this procedure no database information is shared.

Although optimal in the sense of being convergent to its centralized counterpart, the algorithm just described needs to be run for every new classification query. Alternatively, one can envision procedures to find discriminant functions in a distributed fashion that classify new queries without having to re-run the distributed algorithm. The key is to pre-select a *fixed* set $\{\chi_l\}_{l=1}^L$ for which g^* in (5) is (approximately) equivalent to g_j^* in (24) for all $j\in\mathcal{J}$. From Theorem 1, we know that all local functions g_j^* share a common space spanned by the χ_l-induced kernels $\{K(\cdot,\chi_l)\}$. If the space $\mathcal{H}$ where g^* lies is finite dimensional, for example, when adopting linear or polynomial

3. Percolating $\{\chi\}_{l=1}^L$ in a distributed fashion through the network can be carried in a finite number of iterations at most equal to the diameter of the network.

kernels in (5), one can always find a finite-size set $\{\chi_l\}_{l=1}^L$ such that the space spanned by the set of kernels $\{K(\cdot,\chi_l)\}$ contains $\mathcal{H}$, and thus $g_1^*(\mathbf{x}) = \ldots = g_J^*(\mathbf{x}) = g^*(\mathbf{x})\ \forall \mathbf{x} \in \mathcal{X}$ (Predd et al., 2006). Indeed, when using linear kernels, the MoM-NDSVM developed here boils down to the MoM-DSVM developed in Section 3 for a suitable finite-size set $\{\chi_l\}_{l=1}^L$.

In general, however, the space spanned by $\{K(\cdot,\chi_l)\}$ may have lower dimensionality than $\mathcal{H}$; thus, local functions g_j^* do not coincide at every point. In this case, MoM-NDSVM finds local approximations to the centralized g^* which accommodate information available to all nodes. The degree of approximation depends on the choice of $\{\chi_l\}_{l=1}^L$. In what follows, we describe two alternatives to constructing such a set $\{\chi_l\}_{l=1}^L$.

- **Grid-based designs**. Consider every entry k of the training vectors $\{\mathbf{x}_{jn}\}$, and form the intervals $I_k := [x_k^{\min}, x_k^{\max}]$, $k = 1,\ldots,p$, where $x_k^{\min} := \min_{j\in\mathcal{J},\, n=1,\ldots,N_j}[\mathbf{x}_{jn}]_k$ and $x_k^{\max} := \max_{j\in\mathcal{J},\, n=1,\ldots,N_j}[\mathbf{x}_{jn}]_k$. Take for convenience $L = M^p$, and partition uniformly each I_k to obtain a set of M equidistant points $Q_k := \{q_{k1},\ldots,q_{kM}\}$. The set $\{\chi_l\}_{l=1}^L$ can be formed by taking all M^p possible vectors with entries drawn from the Cartesian product $Q_1 \times \ldots, \times Q_p$. One possible set we use for generating the $\{\chi_l\}_{l=1}^L$ vectors is obtained by selecting the k-th entry of the l-th vector as $[\chi_l]_k = q_{k,\left(\frac{l}{M^{k-1}} \bmod M\right)+1}$, where $l = 1,\ldots,M^p$ and $k = 1,\ldots,p$. In this case, MoM-NDSVM performs a global consensus step on the entry-wise maxima and minima of the training vectors $\{\mathbf{x}_{jn}\}$. Global consensus on the entry-wise maxima and minima can be computed exactly in a finite number of iterations equal to at most the diameter of the graph $\mathcal{G}$.

- **Random designs**. Once again, we consider every entry k of the training vectors $\{\mathbf{x}_{jn}\}$. MoM-NDSVM starts by performing a consensus step on the entry-wise maxima and minima of the local training vectors $\{\mathbf{x}_{jn}\}$. The set $\{\chi_l\}_{l=1}^L$ is formed by drawing elements χ_l randomly from a uniform p-dimensional distribution with extreme points per entry given by the extreme points $x_k^{\min}$ and $x_k^{\max}$, $k = 1,\ldots,p$. To agree on the set $\{\chi_l\}_{l=1}^L$, all nodes in the network are assumed to share a common seed used to initialize the random sampling algorithms.

As mentioned earlier, the number of points L affects how close local functions are to each other as well as to the centralized one. The choice of L also depends on the kernel used, prior knowledge of the discriminant function, and the available local training data N_j. Increasing L guarantees that local functions will be asymptotically close to each other regardless of N_j; however, the communication cost and computational complexity per node will increase according to L [cf. Algorithm 3]. On the other hand, a small L reduces the communication overhead at the price of increasing the disagreement among the g_j^*'s. This trade-off will be further explored in the ensuing section through simulated tests.

5. Numerical Simulations

In this section, we analyze the performance of both MoM-DSVM and MoM-NDSVM algorithms using different networks with synthetic and real-world training sets. Although we focus on the binary classification case, it is worth remembering that K-ary classification problems with $K > 2$ can be solved via binary classification schemes, for example, by using one versus all classifiers, or all versus all classifiers (Duda et al., 2002, Ch. 5).

5.1 Linear Classifiers

In this section, we present experiments on synthetic and real data to illustrate the performance of our distributed method for training linear SVMs.

5.1.1 TEST CASE 1: SYNTHETIC TRAINING SET

Consider a randomly generated network with $J = 30$ nodes. The network is connected with algebraic connectivity 0.0448 and average degree per node 3.267. Each node acquires labeled training examples from two different classes $\mathcal{C}_1$ and $\mathcal{C}_2$ with corresponding labels $y_1 = 1$ and $y_2 = -1$. Classes $\mathcal{C}_1$ and $\mathcal{C}_2$ are equiprobable and consist of random vectors drawn from a two-dimensional Gaussian distribution with common covariance matrix $\mathbf{\Sigma} = [1,\ 0;\ 0,\ 2]$, and mean vectors $m_1 = [-1,\ -1]^T$ and $m_2 = [1,\ 1]^T$, respectively. Each local training set $\mathcal{S}_j$ consists of $N_j = N = 10$ labeled examples and was generated by: (i) randomly choosing class $\mathcal{C}_k, k = 1,2$; and, (ii) randomly generating a labeled example $(\mathbf{x}_{jn}^T, y_{jn} = \mathcal{C}_k)$ with $\mathbf{x}_{jn} \sim \mathcal{N}(m_k, \mathbf{\Sigma})$. Thus, the global training set contains $JN = 300$ training examples. Likewise, a test set $\mathcal{S}_{\text{Test}} := \{(\widetilde{\mathbf{x}}_n, \widetilde{y}_n),\ n = 1,\ldots,N_T\}$ with $N_T = 600$ examples, drawn as in (i) and (ii), is used to evaluate the generalization performance of the classifiers. The Bayes optimal classifier for this 2-class problem is linear (Duda et al., 2002, Ch. 2), with risk $\mathbf{R}_{\text{Bayes}} = 0.1103$. The empirical risk of the centralized SVM in (1) is defined as

$$\mathbf{R}_{\text{emp}}^{\text{central}} := \frac{1}{N_T} \sum_{n=1}^{N_T} \frac{1}{2} |\widetilde{y}_n - \hat{y}_n|$$

where $\hat{y}_n$ is the predicted label for $\widetilde{\mathbf{x}}_n$. The average empirical risk of the MoM-DSVM algorithm as a function of the number of iterations is defined as

$$\mathbf{R}_{\text{emp}}(t) := \frac{1}{JN_T} \sum_{j=1}^{J} \sum_{n=1}^{N_T} \frac{1}{2} |\widetilde{y}_n - \hat{y}_{jn}(t)| \tag{40}$$

where $\hat{y}_{jn}(t)$ is the label prediction at iteration t and node j for $\widetilde{\mathbf{x}}_n$, $n = 1,\ldots,N_T$ using the SVM parameters in $\mathbf{v}_j(t)$. The average empirical risk of the local SVMs across nodes $\mathbf{R}_{\text{emp}}^{\text{local}}$ is defined as in (40) with $\hat{y}_{jn}$ found using only locally-trained SVMs.

Figure 3 (left) depicts the risk of the MoM-DSVM algorithm as a function of the number of iterations t for different values of JC. In this test, $\eta = 10$ and a total of 500 Monte Carlo runs are performed with randomly drawn local training and test sets per run. The centralized and local empirical risks with $C = 10$ are included for comparison. The average local prediction performance is also evaluated. Clearly, the risk of the MoM-DSVM algorithm reduces as the number of iterations increases, quickly outperforming local-based predictions and approaching that of the centralized benchmark. To further visualize this test case, Figure 3 (right) shows the global training set, along with the linear discriminant functions found by the centralized SVM and the MoM-DSVM at two different nodes after 400 iterations with $JC = 20$ and $\eta = 10$. Local SVM results for two different nodes are also included for comparison.

5.1.2 TEST CASE 2: MNIST TRAINING SET

Here, the MoM-DSVM is tested on the MNIST database of handwritten images (Lecun et al., 1998). The MNIST database contains images of digits 0 to 9. All images are of size 28 by 28 pixels. We consider the binary problem of classifying digit 2 versus digit 9 using a linear classifier. For this

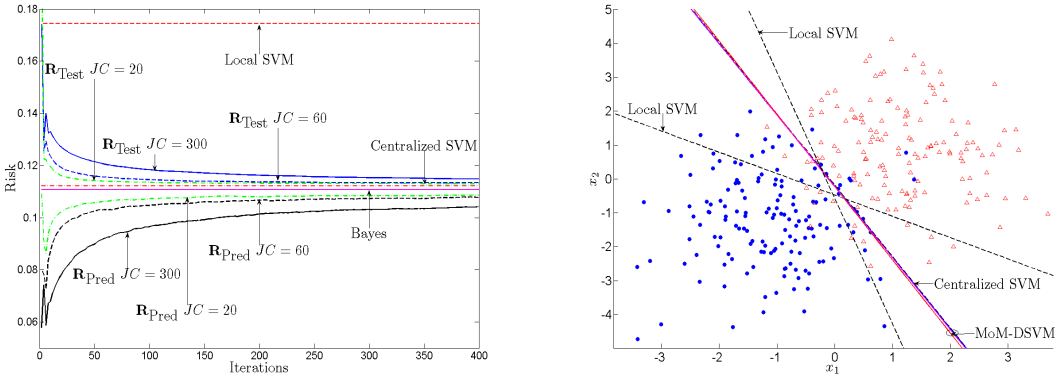


Figure 3: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) and prediction error ($\mathbf{R}_{\text{Pred}}$) of MoM-DSVM for a two-class problem using synthetic data and a network with $J = 30$ nodes. Centralized SVM performance and average local SVMs performance are also plot for comparison (left). Decision boundary comparison among MoM-DSVM, centralized SVM and local SVM results for synthetic data generated from two Gaussian classes (right).

experiment each image is vectorized to a 784 by 1 vector. In particular, we take 5,900 training samples per digit, and a test set of 1,000 samples per digit. Both training and test sets used are as given by the MNIST database, that is, there is no preprocessing of the data. For simulations, we consider two randomly generated networks with $J = 25$, algebraic connectivity 3.2425, and average degree per node 12.80; and $J = 50$ nodes, algebraic connectivity 1.3961, and average degree per node 15.92. The training set is equally partitioned across nodes, thus every node in the network with $J = 25$ has $N_j = 472$ training vectors, and every node in the network with $J = 50$ has $N_j = 236$ samples. The distribution of samples across nodes influences the training phase of MoM-DSVM. For example, if data per node are biased toward one particular class, then the training phase may require more iterations to percolate appropriate information across the network. In the simulations, we consider the two extreme cases: (i) training data are evenly distributed across nodes, that is, every node has the same number of examples from digit 2 and from digit 9; and, (ii) highly biased local data, that is, every node has data corresponding to a single digit; thus, a local binary classifier cannot be constructed.

The figures in this section correspond to one run of the MoM-DSVM for a network with noiseless communication links. Figure 4 shows the evolution of the test error for the network with 25 nodes and highly biased local data. Likewise, Figure 5 shows the evolution of the test error for the network with 50 nodes and highly biased local data. Different values for the penalties JC and η were used to illustrate their effect on both the classification performance and the convergence rate of MoM-DSVM. The parameter JC controls the final performance of the classifier; but for a finite number of iterations, η also influences the final performance of the classifier. Larger values of η may be desirable; however, if η is too large, the algorithm first focuses on reaching consensus across nodes disregarding the classification performance. Although, MoM-DSVM is guaranteed to converge for all η, a very large choice for η may hinder the convergence rate.

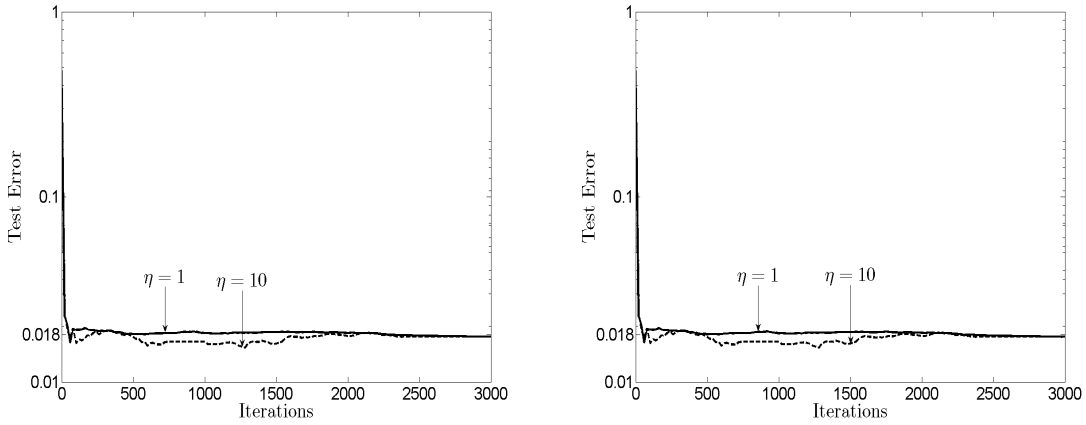


Figure 4: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) of MoM-DSVM, with penalty coefficients $JC = 1$ (left) and $JC = 5$ (right), for a two-class problem using digits 2 and 9 from the MNIST data set unevenly distributed across nodes, and a network with $J = 25$ nodes.

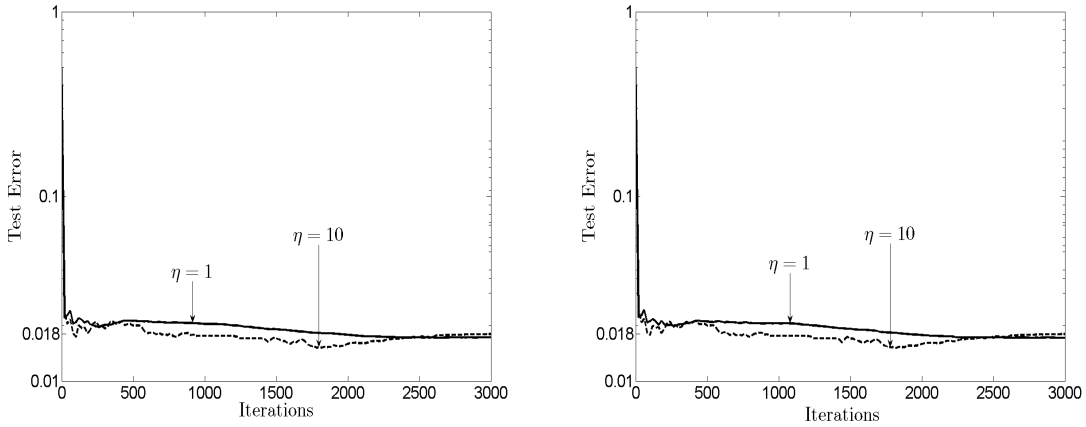


Figure 5: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) of MoM-DSVM, with penalty coefficients $JC = 1$ (left) and $JC = 5$ (right), for a two-class problem using digits 2 and 9 from the MNIST data set unevenly distributed across nodes, and a network with $J = 50$ nodes.

Next, the dispersion of the solutions after 3,000 iterations for different values of η is tested. For our experiment, dispersion refers to how similar are the local $\mathbf{v}_j(t)$ at every node. The mean-squared error (MSE) of the solution across nodes is defined as $\Delta(t) := \frac{1}{J}\sum_{j=1}^{J} ||\mathbf{v}_j(t) - \bar{\mathbf{v}}(t)||^2$ where $\bar{\mathbf{v}}(t) := \frac{1}{J}\sum_{j=1}^{J} \mathbf{v}_j(t)$. Table 1 shows $\Delta(t)$ at $t = 3{,}000$ for different values of η and JC. Note that larger values of η lead to smaller dispersion in the solution; however, as illustrated in Figure 5, they do not imply faster convergence rates.

	$\Delta(t)$ at $t = 3{,}000$			
	$J = 25$		$J = 50$	
η	$JC = 1$	$JC = 5$	$JC = 1$	$JC = 5$
1	3.1849×10^{-7}	3.1870×10^{-7}	2.3749×10^{-7}	2.3227×10^{-7}
5	1.5591×10^{-8}	1.4760×10^{-8}	2.6613×10^{-8}	2.6646×10^{-8}
10	2.9280×10^{-10}	2.9112×10^{-10}	3.6028×10^{-9}	3.6207×10^{-9}

Table 1: MSE $\Delta(t)$ with MNIST data set and biased local data for different values of η and JC.

Consider next data that are evenly distributed across nodes. The MNIST training set is partitioned across nodes ensuring that every node has an equal number of examples from digit 2 and digit 9. Figure 6 shows the evolution of the test error for the network with $J = 25$ nodes and Figure 7 shows the evolution of the test error for the network with $J = 50$ nodes for different values for the penalties JC and η. In this case, local classifiers achieve low test error after one iteration of the MoM-DSVM. In subsequent iterations MoM-DSVM forces all local classifiers to consent, but the test error does not decrease monotonically across iterations. This variation is small, ranging between 0.015 and 0.02, since all local classifiers already have low test error. Both Figures 6 and 7 show that between iterations 500 and $2{,}000$, the global test reaches a minimum value, then it increases and converges to a larger value. This non-monotonic behavior can be attributed to the fact that the MoM-DSVM iterates are not guaranteed to be monotonic. Moreover, before consensus is reached across all nodes the test error at any given node and iteration index does not necessarily need to be greater than the centralized one.

It is also worth noticing the resemblance of the curves in the left and right panels of Figures 4, 5, 6 and 7. Although the test error is nearly identical for $JC = 1$ and $JC = 5$, this does not imply that the $\mathbf{v}_j(t)$ are nearly identical across iterations. Furthermore, the insensitivity w.r.t. small changes in JC reveals that in order to affect the classifier performance, the parameter JC must vary in the order of J. Relating the distributed setting with its centralized counterpart, it follows that with, for example, $J = 25$ a change in JC from 1 to 5 in the distributed setup of (6), corresponds to a change in C from 0.04 to 0.20 for the centralized setting of (1). Such a small change in C explains why the classification performance of the equivalent centralized scenarios is nearly identical as reflected in the figures.

In both biased and evenly distributed data, after a few iterations, MoM-DSVM yields an average performance close to the optimal one. It is also interesting to note that in the biased data case, nodes alone cannot construct an approximate classifier since they do not have samples from both classes. If an incremental approach were used it would need at least one full cycle through the network to enable construction of local estimators per node.

Finally, the effect of network connectivity on the performance of MoM-DSVM is explored. In this experiment, we consider a network with $J = 25$ nodes, ring topology and biased data distribution as before. The performance of MoM-DSVM is illustrated by Figure 8. It is clear that in this case a larger η improves the convergence rate. Also, note that after a few iterations the average performance of the classifier across the network is close to the optimal. In practice, a small reduction of performance over the centralized classifier may be acceptable in which case MoM-DSVM can stop after a small number of iterations. Note that the communication cost of MoM-DSVM can be easily computed at any iteration in terms of the number of scalars transmitted across the network.

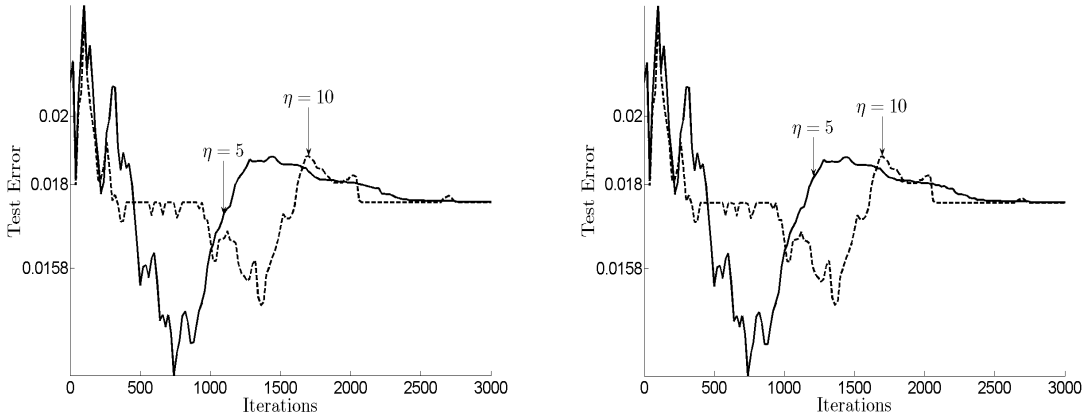


Figure 6: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) of MoM-DSVM, with penalty coefficients $JC = 1$ (left) and $JC = 5$ (right), for a two-class problem using digits 2 and 9 from the MNIST data set evenly distributed across nodes, and a network with $J = 25$ nodes.

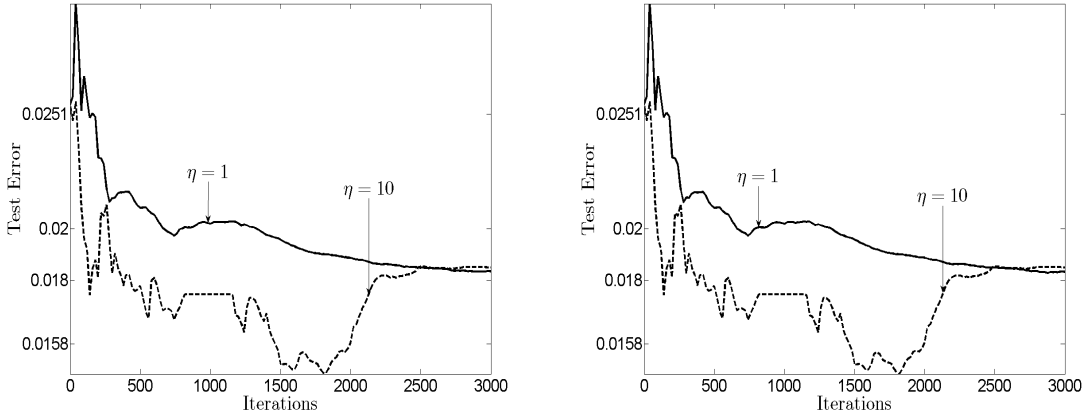


Figure 7: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) of MoM-DSVM, with penalty coefficients $JC = 1$ (left) and $JC = 5$ (right), for a two-class problem using digits 2 and 9 from the MNIST data set evenly distributed across nodes, and a network with $J = 50$ nodes.

For the MNIST data set, the total communication cost up to iteration t is $785Jt$ scalars (cf. Section 3).

5.1.3 TEST CASE 3: SEQUENTIAL OPERATION

Consider a network with $J = 10$ nodes, algebraic connectivity 0.3267, and average degree per node 2.80. Data from two classes arrive sequentially at each node in the following fashion: at $t = 0$ each node has available one labeled training example drawn from the class distributions $\mathcal{C}_1$ and $\mathcal{C}_2$

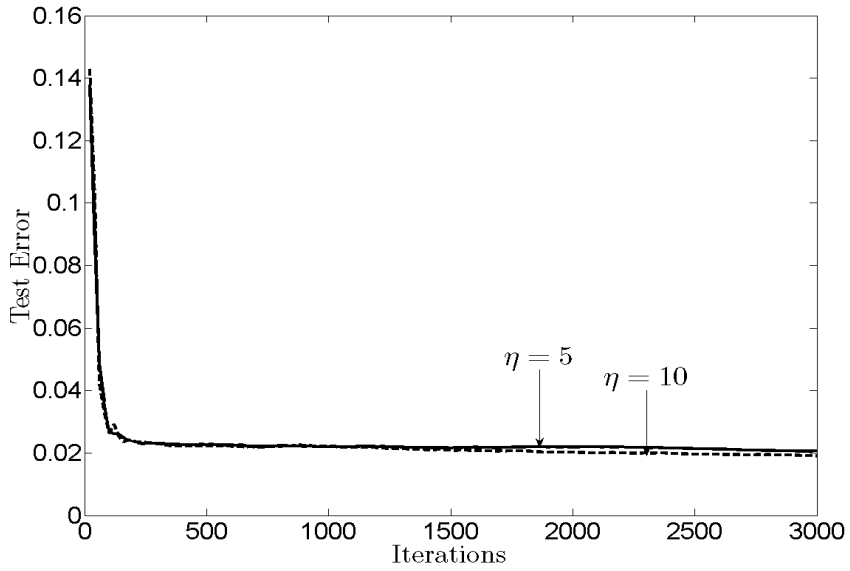


Figure 8: Evolution of the test error ($\mathbf{R}_{\text{Test}}$) of MoM-DSVM, with penalty coefficients $JC = 1$, for a two-class problem using digits 2 and 9 from the MNIST data set unevenly distributed across nodes, and a network with ring topology and $J = 25$ nodes.

described in Test Case 1. From $t = 0$ to $t = 19$, each node acquires a new labeled training example per iteration from this same distribution. From $t = 20$ to $t = 99$, no new training example is acquired. After iteration $t = 99$, the distribution from which training examples in class $\mathcal{C}_1$ were generated changes to a two-dimensional Gaussian distribution with covariance matrix $\Sigma_1 = [1,\ 0;\ 0,\ 2]$, and mean vector $m_1 = [-1,\ 5]^T$. From $t = 100$ to $t = 119$, each node acquires a new labeled training example per iteration using the new class-conditional distribution of $\mathcal{C}_1$, while the class-conditional distribution of C_2 remains unchanged. During these iterations, we remove the training examples from $\mathcal{C}_1$ that were generated during the interval $t = 0$ to $t = 19$, one per iteration. From $t = 120$ to $t = 299$ nodes do not acquire new labeled training examples. From iteration $t = 300$ to $t = 499$, we include 8 new training examples per node and per iteration drawn only from class $\mathcal{C}_1$ with the same class-conditional distribution as the one used at the beginning of the algorithm $t = 0$. Finally, at iteration $t = 500$ all labeled training samples drawn from $t = 300$ to $t = 499$ are removed at each node at once, returning to the global data set available prior to iteration $t = 300$. The algorithm continues without any further change in the training set until convergence.

Figure 10 illustrates the tracking capabilities of the online MoM-DSVM scheme for different values of η. A total of 100 Monte Carlo runs were performed. The figure of merit in this case is $\mathcal{V}(t) := \frac{1}{J}\sum_{j=1}^{J} \left\|\mathbf{v}_j(t) - \mathbf{v}_c(t)\right\|$, where $\mathbf{v}_c(t)$ contains the coefficients of the centralized SVM using the training set available at time t. The peaks in Figure 10 correspond to the changes described in our experiment. MoM-DSVM rapidly adapts the coefficients after the local training sets are modified. Clearly, the parameter η can be tuned to control the speed with which MoM-DSVM adapts. Notice that a large η may cause over-damping effects hindering the final performance of the algorithm. Figure 9 shows snapshots, for a single run of MoM-DSVM and $\eta = 30$, of the global training set

and local discriminant functions at different iterations. The solid training examples correspond to the current global SVs found by the online MoM-DSVM algorithm.

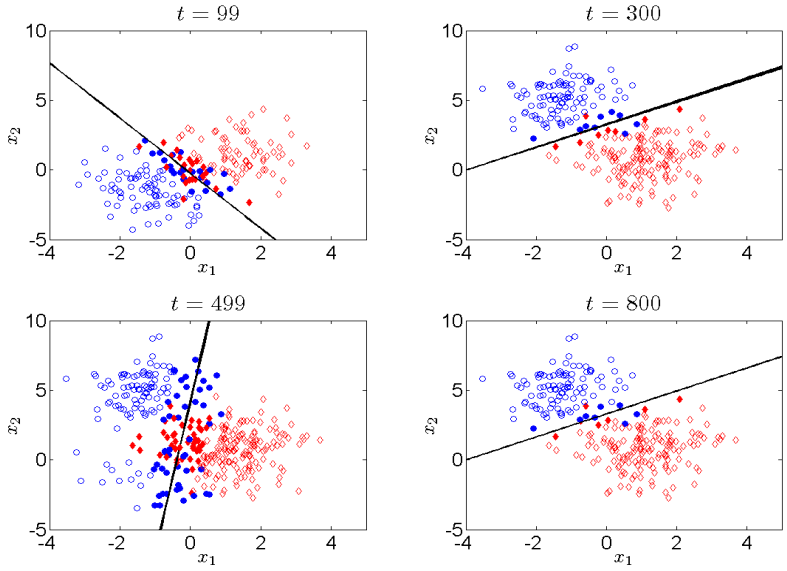


Figure 9: Snapshots of the global training data set and local linear discriminant $g_j^{(t)}(\mathbf{x})$ obtained with MoM-DSVM at all nodes for a synthetic training set evolving in time.

5.1.4 TEST CASE 4: COMMUNICATION COST COMPARISON

In this section, a comparison with the incremental SVM (ISVM) approach in Lu et al. (2008) is presented. The network with $J = 30$ nodes is considered again, where each node j has available a local training set with $N_j = N = 20$ with training vectors generated as in Test Case 1. A global test set with $N_T = 1,200$ was used, and 100 Monte Carlo runs were performed. The MoM-DSVM algorithm used $JC = 20$. The network topology is a ring; thus, ISVM entails no extra overhead due to inter-node communications. Nevertheless, in more general network topologies such overhead might dramatically increase the total communication cost incurred by ISVM. The communication cost is measured in terms of the number of scalars communicated per node. For MoM-DSVM, this cost is fixed per iteration and equal to $3J$ scalars; recall that per iteration every node broadcasts $\mathbf{v}_j(t)$ to its neighborhood (cf. Algorithm 1). The ISVM approach locally trains an SVM and passes its local SVs to the next node in the cycle; the algorithm continues traversing the network until no SVs are shared among neighboring nodes. Thus, the communication cost per iteration depends on the number of SVs found at each node, that is, $3 \times \{\#\text{ of SVs at node } j\}$. A contingency strategy to prevent SVs from being transmitted multiple times by the same node as well as to prevent repetition of training set elements at individual nodes is run in parallel with the ISVM algorithm.

Figure 11 depicts the cumulative communication cost for MoM-DSVM and ISVM as a function of their classification performance. In this particular case and with the most favorable network topology for an incremental approach, we observe that MoM-DSVM achieves a comparable risk to

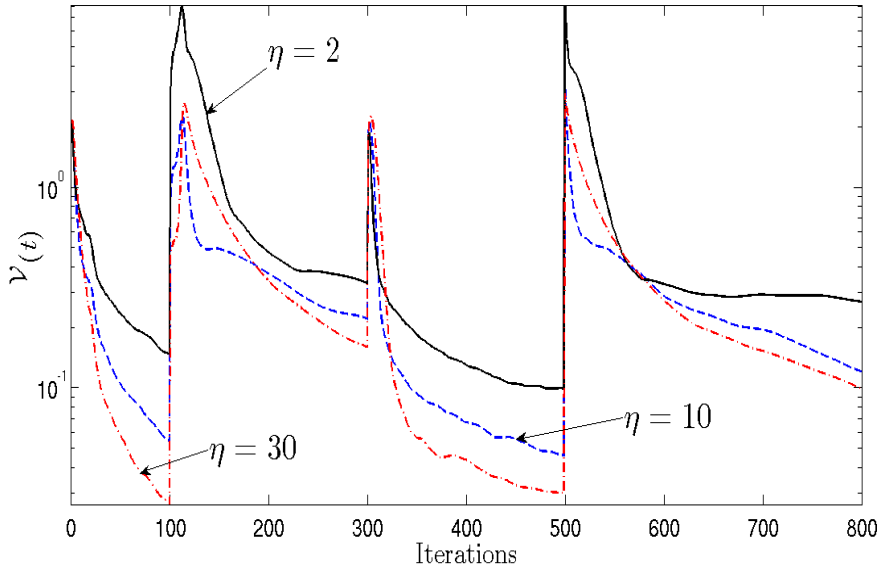


Figure 10: Average error $\mathcal{V}(t)$ of MoM-DSVM for a synthetic training set evolving in time evaluated for various values of η. The peaks correspond to iteration indexes where the local training sets were modified.

ISVM with a smaller number of transmitted scalars. Specifically, to achieve a risk of 0.1159, MoM-DSVM communicates on average 1,260 scalars whereas ISVM communicates on average 8,758 scalars. MoM-DSVM can largely reduce the amount of communications throughout the network, a gain that translates directly to lower power consumption, and thus, in the context of WSNs (see Example 1), longer battery life for individual nodes.

5.2 Nonlinear Classifier

In this section, we present experiments on synthetic and real data to illustrate the performance of our distributed method for training nonlinear SVMs.

5.2.1 TEST CASE 5: SYNTHETIC TRAINING SET

Consider the same network as in Test Case 1. Each node acquires labeled training examples from two different equiprobable classes $\mathcal{C}_1$ and $\mathcal{C}_2$. Class $\mathcal{C}_1$ contains now examples from a two-dimensional Gaussian distribution with covariance matrix $\Sigma_1 = [0.6, 0; 0, 0.4]$, and mean vector $m_1 = [0,0]^T$. Class $\mathcal{C}_2$ is a mixture of Gaussian distributions with mixing parameters $\pi_{21} = 0.3$ and $\pi_{22} = 0.7$; mean vectors $m_2 = [-1,-1]^T$ and $m_3 = [2,2]^T$; and, equal covariance matrix Σ. The optimal Bayes classifier here is clearly nonlinear.

We generate a matrix Γ with rows taken from a uniform two-dimensional grid of L points. The extreme values of the grid are chosen equal to the extreme points of the global training set. Local training sets are of size $N_j = 10\ \forall j \in \mathcal{J}$, and are generated from the distributions described in the previous paragraph. Each node uses its local training sets as well as the matrix Γ to build the local

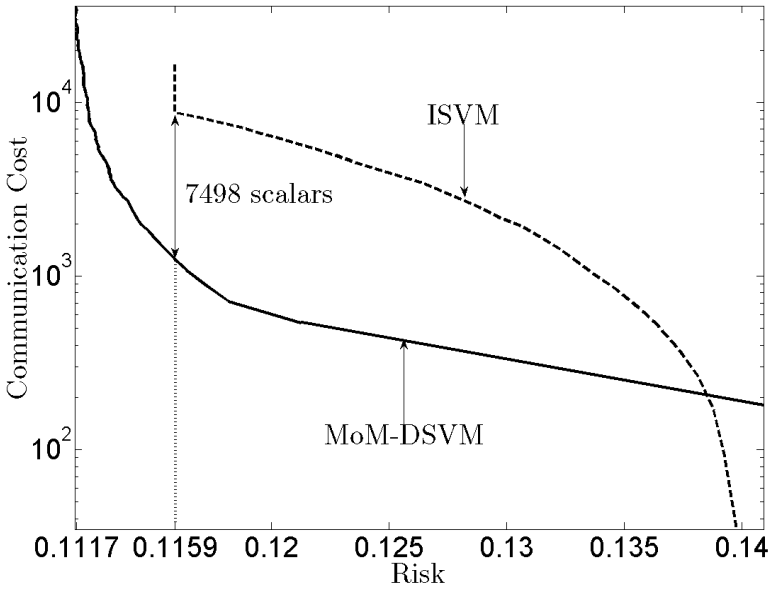


Figure 11: Communication cost, measured in terms of the number of scalars transmitted, of MoM-DSVM and ISVM for a network with ring topology and $J = 30$.

classifier, as described in (24). A Gaussian kernel with $\sigma^2 = 0.9$ and $\eta = 10$ was employed to construct a global nonlinear classifier. Figure 12 (left) shows the classification performance on the points $\{\chi_l\}_{l=1}^{L}$; that is, the classification performance when the testing set is given by $\{(\chi_l, y_l) : l = 1,\ldots,L\}$, where y_l indicates the class from which χ_l was originally drawn. For comparison, we have also included the Bayes risk, the centralized SVM empirical risk, and the local SVM risk. As expected, the classification performance of the distributed classifier approaches that of the centralized one.

Figure 12 (right) illustrates the performance of MoM-NDSVM on a randomly-generated test set of size $N_T = 600$ for various choices of L and JC. Matrix Γ was taken from a uniform two-dimensional grid of L points as before. A total of 500 Monte Carlo runs were performed. Clearly, the asymptotic performance of MoM-NDSVM rapidly outperforms the average performance of a locally-trained SVM and closely converges to the centralized SVM for larger values of L with all other parameters fixed. However, it is worth observing that the choice of the parameter JC also influences the performance. Large values for JC promote reduced number of prediction errors on the training set (possibly) leading to over-fitting. Various strategies, such as cross validation, can be implemented to select optimal values for both JC and σ^2 at the expense of training with MoM-NDSVM multiple times. To visualize the results, Figures 13 and 14 depicts the form of the discriminant function for several values of L at 6 different nodes in the network. Centralized and local discriminant functions are also included as benchmarks. Even though the nodes do not exactly agree on the final form of $g_j(\mathbf{x})$ at all points, their classification performance closely converges to the one obtained by the centralized SVM benchmark.

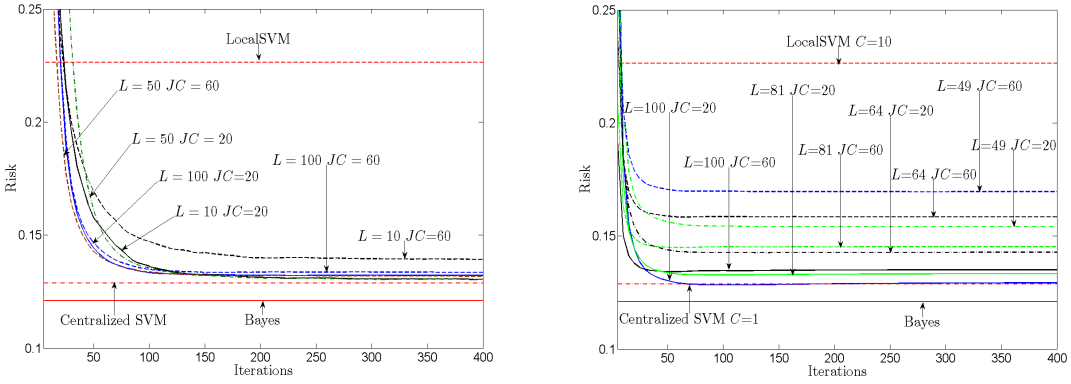


Figure 12: Evolution of prediction error $(\mathbf{R}_{\text{Pred}})$, where matrix Γ is considered a classification query of size L (left); and test error $(\mathbf{R}_{\text{Test}})$, where matrix Γ is constructed as a random grid with L points (right), for MoM-NDSVM applied to a two-class problem using synthetic data and a network with $J = 30$ nodes.

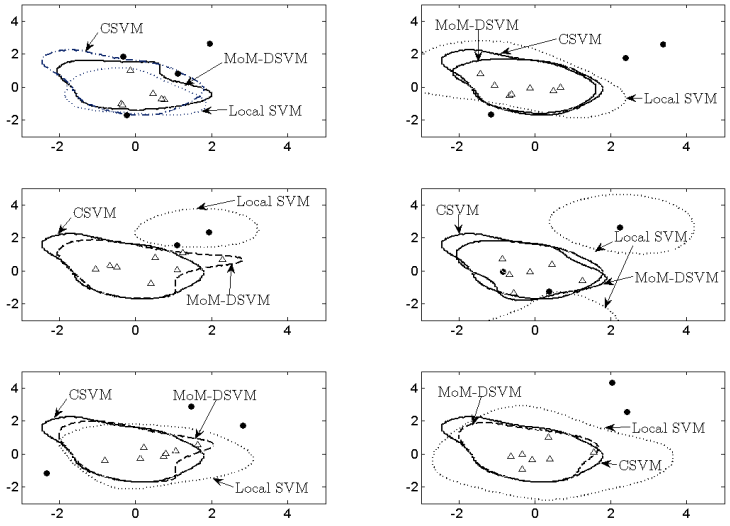


Figure 13: Comparison of the discriminant functions found by a centralized SVM, local SVMs, and the MoM-NDSVM algorithm at 6 different nodes of a network with $J = 30$ using synthetic data. A penalty term $JC = 20$ and a random grid with $L = 100$ were used.

5.2.2 TEST CASE 6: UCI TRAINING SETS

Four data sets from the UCI repository have been chosen to test our MoM-NDSVM algorithm: Iris, Wine, Puma Indians Diabetes, and Parkinsons (Asuncion and Newman, 2007). A brief description

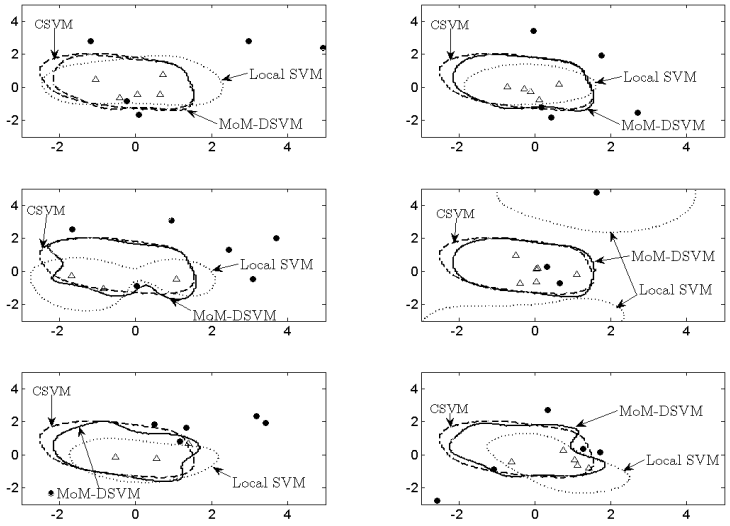


Figure 14: Comparison of the discriminant functions found by a centralized SVM, local SVMs, and the MoM-NDSVM algorithm at 6 different nodes of a network with $J = 30$ using synthetic data. A penalty term $JC = 60$ and a random grid with $L = 100$ were used.

Data set	Classes	Dim. Features	Size	Train. Set	Test Set
Iris	3	4	150	12	40
Wine	3	13	178	12	40
Diabetes	2	8	768	50	200
Parkinsons	2	23	197	12	40

Table 2: UCI data sets

of each of the data sets is shown in Table 2. Examples from the data sets are randomly split among $J = 5$ nodes in a fully-connected network. Focusing on the binary classification problem, only classes 2 and 3 from the Iris data set and classes 1 and 2 from the Wine data set are used. For simulation purposes, each local training set as well as the testing set have the same number of examples from each class.

Table 3 compares performance of the classifiers constructed via MoM-NDSVM with the average performance of the 5 local classifiers trained with local training sets only, and with the one of a centralized SVM trained with the training set available to the whole network. A total of 100 Monte Carlo runs were performed per data set, where both training and testing sets were drawn randomly per run. The MoM-NDSVM parameters JC and η were chosen via cross-validation for every training set as in Hastie et al. (2009, Ch. 7). Gaussian kernels as in the previous section were used. The local and centralized SVMs were trained using the Spider toolbox for MATLAB (Weston et al., 2006). To evaluate the local performance of the classifiers, each node trains a local SVM and its performance is compared with the one obtained via MoM-NDSVM. For each training set we

Data set	Local	Centralized	MoM-NDSVM $L = 150$	MoM-NDSVM $L = 300$	Class. Query
Iris	8.39%	4.43%	5.15%	5.26%	4.28%
Wine	15.71%	6.17%	7.37%	7.33%	6.60%
Diabetes	34.52%	24.40%	29.69%	28.92%	23.76%
Parkinsons 1	33.76%	18.45%	30.14%	31.13%	18.56%
Parkinsons 2	34.78%	18.86%	23.60%	24.05%	20.28%

Table 3: UCI data sets centralized versus local versus distributed performance comparison for $t = 1,000$. Parkinsons 2 is the normalized Parkinsons training set.

explore two cases: (i) local classifiers at each node; and (ii) Γ as a classification query. Figure 15 plots the training evolution of MoM-NDSVM for the Puma Indians Diabetes data set.

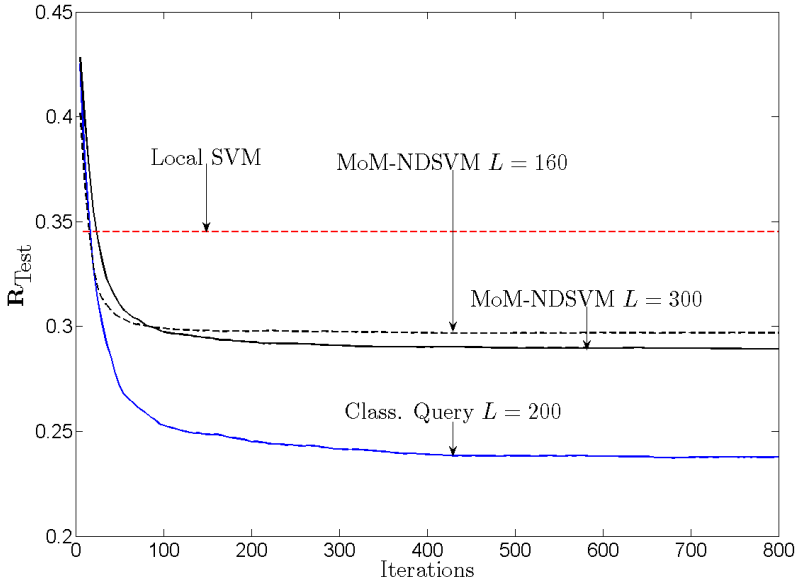


Figure 15: Evolution of test error ($\mathbf{R}_{\text{Test}}$) for Puma Indians Diabetes data set taken from the UCI repository, and a fully connected network with $J = 5$ nodes.

The performance of MoM-NDSVM for (i) depends heavily on the choice of Γ. To illustrate this point, the size of the local training sets per node has been chosen small when compared to the dimensionality of the feature space. Let $x_k^{\min}$ and $x_k^{\max}$ correspond to the smallest and largest values that the k-th feature can take. The row-elements of matrix Γ are chosen randomly and independently over the interval $[x_k^{\min}, x_k^{\max}]$ per component. Two different values of L were chosen to compare the performance of MoM-NDSVM. For small values of p, Γ can be chosen as a grid of M uniformly spaced points per dimension; therefore, $L = M^p$. The results summarized in Table 3 highlight

	$\eta = 10$		$\eta = 20$	
L	$p_0 = 49$	$p_0 = 81$	$p_0 = 49$	$p_0 = 81$
400	0.370%	1.246%	0.504%	1.448%
800	0.136%	0.242%	0.234%	0.654%

Table 4: Average MoM-NDSVM risk (Gaussian kernel) at iteration $t = 3{,}000$ for compressed MNIST data set with dimensionality p_0.

the fact that the classification performance at each node remains limited by the training examples available locally. However, in this extremely challenging case, collaboration among nodes improves the overall classification performance of the network.

Table 3 also illustrates how conditioning of the data together with the choice for the kernel function can impact the performance of MoM-NDSVM. In particular, its last two rows compare the classification performance achieved for the Parkinsons training set without normalization (Parkinsons 1) and with its features normalized to have maximum absolute size unity (Parkinsons 2). Although both centralized and local performance remain nearly unchanged, the MoM-NDSVM performance improves about 7% for both $L = 150$ and $L = 300$. An intuitive explanation follows from looking closer at the values of the features in the Parkinsons training set. Some features take values in the order of 10^2 while others take values in the order of 10^{-3}; thus, a Gaussian kernel that spans symmetrically along all directions is not the best kernel choice for this case. After normalization, a smaller number L of Gaussian kernels can be used to obtain a better representation of the decision surface. In conclusion, data across nodes must be preprocessed whenever possible to achieve a better trade off between classification performance and the computational complexity of MoM-NDSVM.

Note that the classification performance for case (ii) approaches the centralized SVM one. After a few iterations, the classification accuracy returned by the network surpasses that of locally trained SVMs. The speed of convergence might be hindered if L is chosen large. Fine tuning of η can achieve a desirable trade-off between speed of convergence and performance in terms of test error.

5.2.3 TEST CASE 7: MNIST TRAINING SET

Consider a network with $J = 25$, algebraic connectivity 3.2425, and average degree per node 12.8. Local training sets have been constructed based on the MNIST data set using digits 2 and 9 only. The nodes wish to train a nonlinear global classifier using a Gaussian kernel via MoM-NDSVM. Each node j has available a training set $\mathcal{S}_j$ with 472 examples from one class only, thus individual nodes cannot construct a classifier locally. The large size of the images in MNIST leads to an excessively large choice for $L = L_0 \gg 784$, hindering the convergence of MoM-NDSVM. Instead, each image has been compressed via principal component analysis (PCA) to vectors of dimensionality $p_0 < 784$. It was observed experimentally that after compression, the two classes become separable. Indeed, the centralized equivalent SVM yields test error zero. Figure 16 depicts the performance of MoM-NDSVM for various choices of η and p_0. Note that $\eta = 20$ leads to slower convergence of the average risk across the network. Table 4 summarizes the classification performance of MoM-NDSVM after 3,000 iterations. A larger value for L improves the average classification performance of the network. However, the number of iterations required for MoM-NDSVM to converge increases with L.

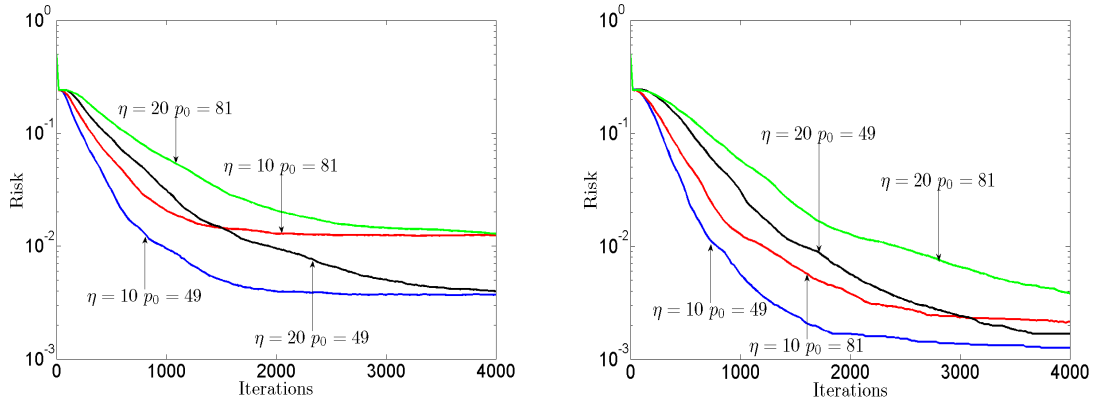


Figure 16: Average evolution of MoM-NDSVM risk (Gaussian kernel) for compressed MNIST data set for $L = 400$ (left), and $L = 800$ (right) in a network with $J = 25$ nodes. MNIST images have been compressed to dimensionality p_0 via PCA.

5.3 Noisy Inter-node Communications

This subsection presents robustness tests of the novel distributed classification scheme with noisy inter-node exchanges. Such noise is due to, for example, quantization error, additive Gaussian noise at the receiving ends, or, Laplacian noise intentionally added to transmitted samples in order to guarantee data privacy (Chaudhuri and Monteleoni, 2008; Dwork et al., 2006). Although focus is placed on MoM-DSVM, the results also carry over to MoM-NDSVM.

5.3.1 TEST CASE 8: MOM-DSVM WITH PERTURBED TRANSMISSIONS

In this setting, per iteration t, each node j purposely introduces a perturbation $\varepsilon_j(t)$ to the variable $\mathbf{v}_j(t)$ before transmission. Perturbed transmissions can be used to preserve data privacy (Dwork et al., 2006). Consider an eavesdropper accessing the noisy versions of $\mathbf{v}_j(t)$. The form and variance level Σ_j of the local perturbations $\varepsilon_j(t)$ can be adjusted per node to prevent the eavesdropper from learning $\mathcal{S}_j$. For instance, Dwork et al. (2006) suggests introducing zero-mean Laplacian random variable whose variance depends on the sensitivity of $\mathbf{v}_j(t)$ as a function of $\mathcal{S}_j$.

The MoM-DSVM iterations, with $JC = 5$ and $\eta = 10$, are modified by introducing local perturbations $\varepsilon_j(t)$ to $\mathbf{v}_j(t)$. Each $\varepsilon_j(t)$ is zero-mean Laplacian distributed and white across time and space, that is, $\mathbb{E}\{\varepsilon_j(t_1)\varepsilon_j^T(t_2)\} = 0$ if $t_1 \neq t_2$ and $\mathbb{E}\{\varepsilon_i(t)\varepsilon_j^T(t)\} = 0$ if $i \neq j\ \forall i, j \in \mathcal{J}$. The resulting MoM-DSVM iterations are

$$\begin{aligned}
\boldsymbol{\lambda}_j(t+1) &= \underset{\boldsymbol{\lambda}_j:\, \mathbf{0}_j \preceq \boldsymbol{\lambda}_j \preceq JC\mathbf{1}_j}{\arg\max} -\frac{1}{2}\boldsymbol{\lambda}_j^T \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{X}_j^T \mathbf{Y}_j \boldsymbol{\lambda}_j + \left(\mathbf{1}_j + \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{f}_j(t)\right)^T \boldsymbol{\lambda}_j, \\
\mathbf{v}_j(t+1) &= \mathbf{U}_j^{-1}\left[\mathbf{X}_j^T \mathbf{Y}_j \boldsymbol{\lambda}_j(t+1) - \mathbf{f}_j(t)\right], \\
\alpha_j(t+1) &= \alpha_j(t) + \frac{\eta}{2} \sum_{i \in \mathcal{B}_j} \left[(\mathbf{v}_j(t+1) + \varepsilon_j(t)) - (\mathbf{v}_i(t+1) + \varepsilon_i(t))\right]
\end{aligned}$$

where $\mathbf{U}_j = (1 + 2\eta|\mathcal{B}_j|)\mathbf{I}_{p+1} - \Pi_{p+1}$ and $\mathbf{f}_j(t) := 2\alpha_j(t) - \eta \sum_{i \in \mathcal{B}_j}[\mathbf{v}_j(t) + \varepsilon_j(t) + \mathbf{v}_i(t) + \varepsilon_i(t)]$ (cf. Proposition 1). In this case, MoM-DSVM operates in an analogous manner to Algorithm 1, differing

only in their broadcasting step. In the perturbed transmissions case, every node j broadcasts a perturbed vector $\mathbf{v}_j(t)+\varepsilon_j(t)$ (instead of $\mathbf{v}_j(t)$ alone) to its one-hop neighbors. Note that neighboring nodes $i \in \mathcal{B}_j$ only "see" the aggregate perturbed vector $\mathbf{v}_j(t)+\varepsilon_j(t)$ from node j.

Figure 17 illustrate the performance of MoM-DSVM after 100 Monte Carlo runs with perturbed transmissions for a network with $J = 8$ nodes, algebraic connectivity 0.4194, and average degree per node 2.5. Nodes collect observations from 2 classes $\mathcal{C}_1$ and $\mathcal{C}_2$, where $\mathcal{C}_1$ is $\mathcal{N}(m_1,\Sigma_1)$ with $m_1 = [0,\ 0]^T$, and $\Sigma_1 = [0.6,\ 0;\ 0,\ 0.4]$, and $\mathcal{C}_2$ is $\mathcal{N}(m_2,\Sigma_2)$ with $m_2 = [2,\ 2]^T$, and $\Sigma_2 = [1,\ 0;\ 0,\ 2]$. Each node collects an equal number of observations per class for a total of $N_j = N = 50$ observations. The noise $\varepsilon_j(t)$, inserted per transmission per node, has covariance matrix given by $\sigma^2\mathbf{I}_3$. The optimal classifier is determined by $\mathbf{v}^* = [-1.29,\ -0.76,\ 1.78]^T$, which is the one obtained by MoM-DSVM with $\sigma^2 = 0$. Interestingly, the average risk in the presence of perturbed transmissions remains close to the perturbation-free risk. Even for a large perturbation $\sigma^2 = 1$, the average risk hovers around 0.1075. Furthermore, the risk variance remains small. Indeed, it can be shown that the proposed scheme yields estimates $\mathbf{v}_j(t)$ with bounded variance.

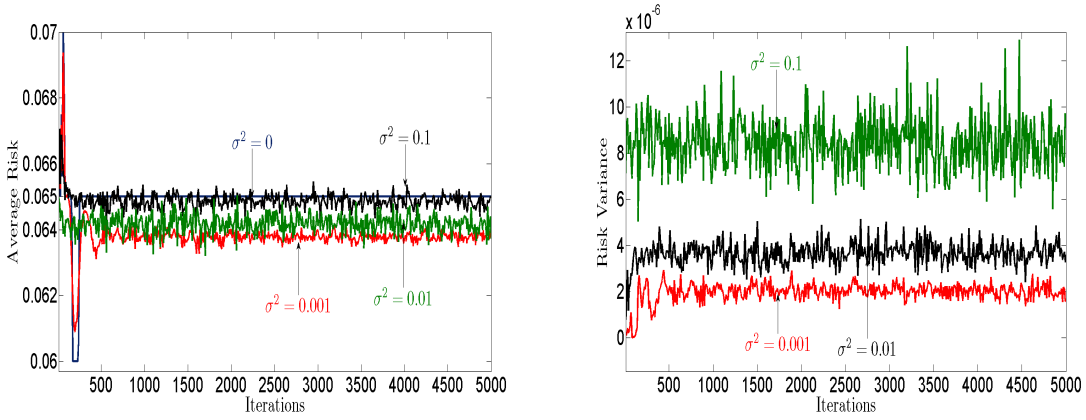


Figure 17: Average risk (left) and risk variance (right) for a network with $J = 8$, and a finite variance perturbation added to $\mathbf{v}_j(t)$ before it is broadcasted.

5.3.2 TEST CASE 9: NOISY COMMUNICATION LINKS

The MoM-DSVM is also robust to non-ideal inter-node links corrupted by additive noise due to, for example, quantization or additive Gaussian receiver noise. In this case, the noise is added at the receiver side. The MoM-DSVM must be modified to obtain a bounded variance on the estimates $\mathbf{v}_j(t)$, and the local Lagrange multipliers $\alpha_{ji}(t) := \alpha_{ji1}(t)$ must be exchanged among neighboring nodes; see Zhu et al. (2009) for similar approaches. Each communication link between node j and node $i \in \mathcal{B}_j$ introduces additive noise $\varepsilon^v_{ji}(t)$ ($\varepsilon^\alpha_{ji}(t)$) to the variable $\mathbf{v}_j(t)$ (α_{ji}). The perturbations $\{\varepsilon^v_{ji}(t)\}$ ($\{\varepsilon^\alpha_{ji}(t)\}$) are zero-mean random variables with covariance matrix Σ^v_{ji} (Σ^α_{ji}), white across

Algorithm 4 MoM-DSVM with noisy links

Require: Randomly initialize $\mathbf{v}_j(0)$, and $\alpha_{ji}(0) = \mathbf{0}_{(p+1)\times 1}\ \forall\ j \in \mathcal{J}\ \forall i \in \mathcal{B}_j$

for $t = 0, 1, 2, \ldots$ **do**
  for all $j \in \mathcal{J}$ **do**
    Compute $\lambda_j(t+1)$ via (41).
    Compute $\mathbf{v}_j(t+1)$ via (42).
  end for
  for all $j \in \mathcal{J}$ **do**
    Broadcast $\mathbf{v}_j(t+1)$ to all neighbors $i \in \mathcal{B}_j$.
  end for
  for all $j \in \mathcal{J},\ i \in \mathcal{B}_j$ **do**
    Compute $\alpha_{ji}(t+1)$ via (43).
  end for
  for all $j \in \mathcal{J},\ i \in \mathcal{B}_j$ **do**
    Transmit $\alpha_{ji}(t+1)$ to $i \in \mathcal{B}_j$.
  end for
end for

time and space. The modified MoM-DSVM iterations are

$$\lambda_j(t+1) = \underset{\lambda_j:\, \mathbf{0}_j \preceq \lambda_j \preceq JC\mathbf{1}_j}{\arg\max} -\frac{1}{2}\lambda_j^T \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{X}_j^T \mathbf{Y}_j \lambda_j + \left(\mathbf{1}_j + \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{f}_j(t)\right)^T \lambda_j, \tag{41}$$

$$\mathbf{v}_j(t+1) = \mathbf{U}_j^{-1}\left[\mathbf{X}_j^T \mathbf{Y}_j \lambda_j(t+1) - \mathbf{f}_j(t)\right], \tag{42}$$

$$\alpha_{ji}(t+1) = \alpha_{ji}(t) + \frac{\eta}{2}[\mathbf{v}_j(t+1) - (\mathbf{v}_i(t+1) + \varepsilon_{ij}^v(t))] \tag{43}$$

where $\mathbf{f}_j(t) := \sum_{i\in\mathcal{B}_j}\left\{\alpha_{ji}(t) - (\alpha_{ij}(t) + \varepsilon_{ij}^{\alpha}(t)) - \eta[\mathbf{v}_j(t) + (\mathbf{v}_i(t) + \varepsilon_{ij}^v(t))]\right\}$. The resulting MoM-DSVM algorithm with noisy links is summarized as Algorithm 4.

The left panels of Figures 18 and 19 depict the average performance after 100 Monte Carlo runs of MoM-DSVM for the same network of Test Case 8. As seen, the variance of the estimates $\mathbf{v}_j(t)$ yielded by the modified MoM-DSVM algorithm remains bounded.

Incremental approaches are hindered by noisy communication links because noise added to the SVs perturbs and accumulates in the local training sets. In ISVM, SVs are bound to percolate across the network, and even to come back to the node where they originated. Due to the noise, however, nodes cannot recognize noisy feature vectors already in $\mathcal{S}_j$. This is problematic since the size of local problems being solved per node increases linearly with the size of the training set, thus requiring a heuristic size-control scheme. The right panels of Figures 18 and 19 show the performance of an ISVM for different levels of noise variance. Noise is added to the SVs and noisy labels are rounded to 1 or -1. Different from MoM-DSVM, the performance of ISVM quickly deteriorates, even for low noise levels since the average risk approaches 0.5 after a few iterations, which amounts to pure guessing of the binary classifier.

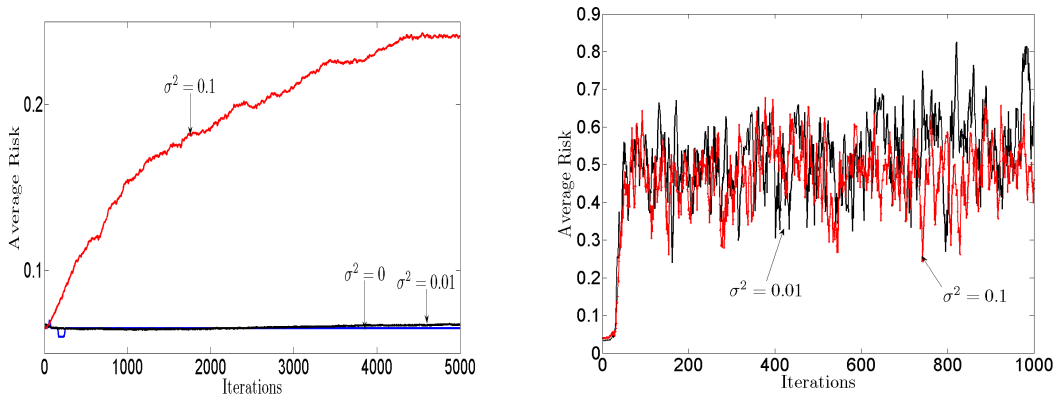


Figure 18: Average risk for a network with $J = 8$, and noisy communication links using a synthetic data set. MoM-DSVM (left) and incremental SVM approach (right).

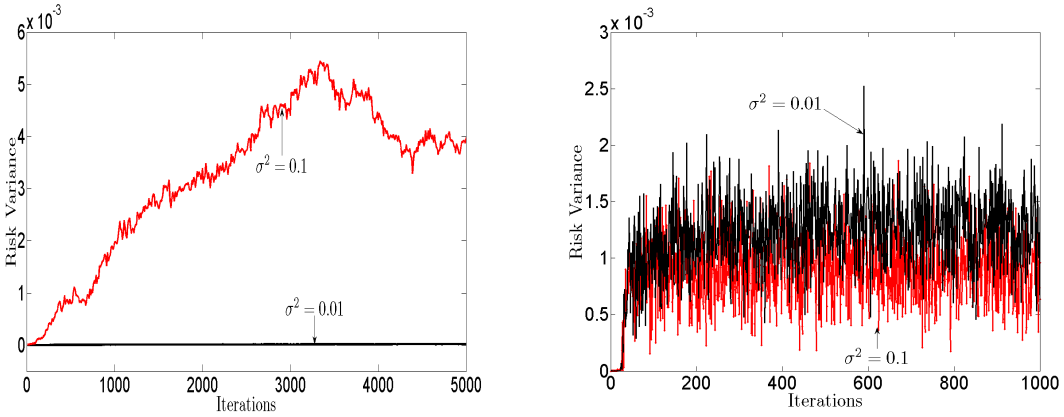


Figure 19: Risk variance for a network with $J = 8$ and noisy communication links corresponding to average risk in Figure 18. MoM-DSVM (left) and incremental SVM approach (right).

6. Conclusions

This work developed distributed SVM algorithms by reformulating the centralized SVM training problem into per-node separable sub-problems linked via consensus constraints, which can be solved using decentralized optimization tools. The novel algorithms are well suited for applications involving data that cannot be shared among collaborating nodes, which possibly operate under stringent resources, and may thus desire to reduce overhead of inter-node message exchanges.

Based on distributed training sets, the novel MoM-DSVM algorithm constructs a maximum-margin linear classifier iteratively. At every iteration, locally updated classifier vectors are exchanged among neighboring nodes. Convergence to the centralized linear SVM formulation is guaranteed. The approach lends itself naturally to online and asynchronous variants, which allow adaptation of the proposed DSVM to scenarios when elements of the local training sets become

available sequentially, or, when outdated elements need to be removed. Furthermore, the MoM-DSVM can be generalized to construct distributed nonlinear discriminant functions. The resulting iterative MoM-NDSVM algorithm is provably convergent to its centralized counterpart, and its complexity is kept at a manageable level by using the kernel trick. Local classifiers are limited by the span of their local training sets, and a set of basis common to all nodes.

Although not formally treated, the novel distributed classification algorithms can be readily extended to solve distributed support vector regression (DSVR) problems. The main characteristics of the present approach, such as its convexity, remain unchanged. Therefore, it is expected that linear and nonlinear estimators developed for MoM-DSVR, will enjoy convergence claims similar to those proved here for MoM-DSVM and MoM-NDSVM classifiers. To complement the distributed supervised classifiers introduced here, our current research deals with consensus-based distributed versions of the unsupervised k-means and expectation-maximization clustering algorithms.

Acknowledgments

This work was supported by NSF grants CCF 0830480 and CON 014658; and also through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. Part of this work appears in *Proc. of 9th ACM/IEEE Conference on Information Processing in Sensor Networks*, Stockholm, Sweden, April 16-19, 2010.

The authors also wish to thank Prof. Arindam Banerjee from the CS Department at the University of Minnesota for his feedback on this and related topics.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

Appendix A. The Alternating Direction Method of Multipliers

The ADMoM is a distributed optimization algorithm solving the following problem

$$\begin{aligned} \min_{\mathbf{v}} \quad & F_1(\mathbf{v}) + F_2(\mathbf{A}\mathbf{v}) \\ \text{s.t.} \quad & \mathbf{v} \in \mathcal{P}_1,\ \mathbf{A}\mathbf{v} \in \mathcal{P}_2 \end{aligned} \tag{44}$$

where $F_1 : \mathbb{R}^{p_1} \to \mathbb{R}$ and $F_2 : \mathbb{R}^{p_2} \to \mathbb{R}$ are convex functions, $\mathbf{A}$ is a $p_2 \times p_1$ matrix, while $\mathcal{P}_1 \subset \mathbb{R}^{p_1}$ and $\mathcal{P}_2 \subset \mathbb{R}^{p_2}$ denote non-empty polyhedral sets.

Upon introducing the auxiliary variable $\omega \in \mathbb{R}^{p_2}$, ADMoM solves the separable problem

$$\begin{aligned} \min_{\mathbf{v},\omega} \quad & F_1(\mathbf{v}) + F_2(\omega) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{v} = \omega \\ & \mathbf{v} \in \mathcal{P}_1,\ \omega \in \mathcal{P}_2 \end{aligned} \tag{45}$$

which is clearly equivalent to (44). With $\alpha \in \mathbb{R}^{p_2}$ denoting the Lagrange multiplier corresponding to the constraint $\mathbf{Av} = \omega$, the augmented Lagrangian corresponding to (45) is

$$\mathcal{L}(\mathbf{v}, \omega, \alpha) = F_1(\mathbf{v}) + F_2(\omega) + \alpha^T(\mathbf{Av} - \omega) + \frac{\eta}{2}\|\mathbf{Av} - \omega\|^2 \tag{46}$$

where the parameter $\eta > 0$ controls the impact of the constraint violation in (45). The ADMoM minimizes alternately $\mathcal{L}$ in (46) w.r.t. the primal variable $\mathbf{v}$, then w.r.t. the auxiliary variable ω, and after each cycle it uses these iterates to update the multiplier. Specifically, with t denoting iteration index, the ADMoM iterates at $t+1$ are given by

$$\mathbf{v}(t+1) = \arg\min_{\mathbf{v}\in\mathcal{P}_1} \mathcal{L}(\mathbf{v}, \omega(t), \alpha(t)), \tag{47}$$

$$\omega(t+1) = \arg\min_{\omega\in\mathcal{P}_2} \mathcal{L}(\mathbf{v}(t+1), \omega, \alpha(t)), \tag{48}$$

$$\alpha(t+1) = \alpha(t) + \eta(\mathbf{Av}(t+1) - \omega(t+1)). \tag{49}$$

Thanks to the auxiliary variable ω, each of the optimization problems (47) and (48) can be run separately, possibly by different processors. The following proposition states the main claim regarding convergence of the ADMoM iterates, and its proof can be found in Bertsekas and Tsitsiklis (1997, Ch. 3, Proposition 4.2).

Proposition 4 *Assume that the optimal solution set v^* of* (44) *is non-empty, and either $\mathcal{P}_1$ is bounded, or, $\mathbf{A}^T\mathbf{A}$ is nonsingular. Then, a sequence $\{\mathbf{v}(t), \omega(t), \alpha(t)\}$ generated by the iterations* (47)-(49) *is bounded, and every limit point of $\{\mathbf{v}(t)\}$ is an optimal solution of* (44). *Furthermore, $\{\alpha(t)\}$ converges to an optimal solution α^* of the dual problem [cf.* (45)*]*

$$\min_{\alpha\in\mathbb{R}^{p_2}} \quad H_1(\alpha) + H_2(\alpha)$$

where for all $\alpha \in \mathbb{R}^{p_2}$

$$H_1(\alpha) := \inf_{\mathbf{v}\in\mathcal{P}_1} [F_1(\mathbf{v}) + \alpha^T\mathbf{Av}],$$
$$H_2(\alpha) := \inf_{\omega\in\mathcal{P}_2} [F_2(\omega) - \alpha^T\omega].$$

Appendix B. Proof of Lemma 1

First, the equality constraints $\{\mathbf{w}_j = \mathbf{w}_i\}$ and $\{b_j = b_i\}$ will be shown equivalent to $\mathbf{w}_1 = \cdots = \mathbf{w}_J$ and $b_1 = \cdots = b_J$, respectively, for any feasible solution of (6). Consider any two nodes j_0 and j_k both in $\mathcal{J}$. Since the network is connected, there exists a path $\{j_0 j_1 \ldots j_{k-1} j_k\}$ of length at least one, which connects nodes j_0 and j_k. Because $j_{\ell+1} \in \mathcal{B}_\ell$ for $\ell = 0, 1, \ldots, k-1$, it is immediate that $\mathbf{w}_{j_\ell} = \mathbf{w}_{j_1} = \cdots = \mathbf{w}_{j_{k-1}} = \mathbf{w}_{j_k}$. Since $j_\ell, j_k \in \mathcal{J}$ are arbitrary, it follows readily that $\mathbf{w}_1 = \cdots = \mathbf{w}_J$. A similar argument leads to $b_1 = \cdots = b_J$.

As any feasible solution of (6) satisfies $\mathbf{w}_1 = \cdots = \mathbf{w}_J = \mathbf{w}$ and $b_1 = \ldots = b_J = b$, problem (6) becomes

$$\begin{aligned} \min_{\mathbf{w},b,\{\xi_j\}} \quad & J\left(\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j\right) \\ \text{s.t.} \quad & \mathbf{Y}_j(\mathbf{X}_j\mathbf{w} + b\mathbf{1}_j) \succeq \mathbf{1}_j - \xi_j \quad \forall j \in \mathcal{J} \\ & \xi_j \succeq \mathbf{0}_j \end{aligned} \tag{50}$$

which is equivalent to (1), since the constant J can be dropped from the cost function in (50).

Appendix C. Proof of Lemma 2

The objective here is to cast (7) in the form of (45), and thus show that iterations (9)-(12) correspond to the ADMoM iterations (47)-(49) in Appendix A. First, it will be shown that the set of consensus constraints in (7), namely $\{\mathbf{v}_j = \omega_{ji},\ \omega_{ji} = \mathbf{v}_i\ :\ \forall j \in \mathcal{J},\ \forall i \in \mathcal{B}_j\}$, can be written as the equality constraint $\mathbf{A}\mathbf{v} = \omega$ in (45). With this objective in mind, consider listing the constraints $\mathbf{v}_j = \omega_{ji}$ across nodes $j \in \mathcal{J}$ and neighbors $i \in \mathcal{B}_j$ as follows

$$\begin{aligned} \{\mathbf{v}_1 &= \omega_{1i}\}_{i\in\mathcal{B}_1} \\ &\vdots \\ \{\mathbf{v}_J &= \omega_{Ji}\}_{i\in\mathcal{B}_J}. \end{aligned} \tag{51}$$

Since for every $\mathbf{v}_j$ there are $|\mathcal{B}_j|$ constraints, the total number of equalities in (51) is $\sum_{j=1}^{J}|\mathcal{B}_j| = 2|\mathcal{E}|$, where $|\mathcal{E}|$ is the number of edges in the network. The factor 2 in $2|\mathcal{E}|$ is because for every edge $j \leftrightarrow i$ there are two constraints, namely $\mathbf{v}_j = \omega_{ji}$ and $\mathbf{v}_i = \omega_{ij}$.

The set of equalities in (51) can be written in matrix-vector form as

$$\begin{matrix} |\mathcal{B}_1| \text{ vectors} \\ |\mathcal{B}_J| \text{ vectors} \end{matrix} \left\{ \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_J \\ \vdots \\ \mathbf{v}_J \end{bmatrix} \right. = \underbrace{\begin{bmatrix} \underbrace{\begin{bmatrix} \mathbf{I}_{p+1} \\ \vdots \\ \mathbf{I}_{p+1} \end{bmatrix}}_{\mathbf{A}_1:=} & & \\ & \ddots & \\ & & \underbrace{\begin{bmatrix} \mathbf{I}_{p+1} \\ \vdots \\ \mathbf{I}_{p+1} \end{bmatrix}}_{\mathbf{A}_J:=} \end{bmatrix}}_{\mathbf{A}':=} \underbrace{\begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_J \end{bmatrix}}_{\mathbf{v}:=} = \underbrace{\begin{bmatrix} \{\omega_{1i}\}_{i\in\mathcal{B}_1} \\ \vdots \\ \{\omega_{Ji}\}_{i\in\mathcal{B}_J} \end{bmatrix}}_{\omega':=} \tag{52}$$

where $\mathbf{A}'\mathbf{v}$ replicates $\mathbf{v}$ in accordance with the left-hand-side (l.h.s.) of (51). Matrix $\mathbf{A}'$ in (52) is block-diagonal with block entries $\mathbf{A}_j := [\mathbf{I}_{p+1}, \ldots, \mathbf{I}_{p+1}]^T$ containing $|\mathcal{B}_j|$ identity matrices of size $(p+1)\times(p+1)$. Since $\mathbf{v}_j$ and ω_{ji} are $(p+1)\times 1$ vectors, $\mathbf{v}$ has size $(p+1)J \times 1$, and ω' has size $2(p+1)|\mathcal{E}| \times 1$.

Equation (52) shows that the constraints of the form $\mathbf{v}_j = \omega_{ji}$ can be compactly written as

$$\mathbf{A}'\mathbf{v} = \omega'. \tag{53}$$

Consider now the remaining constraints, which are of the form $\omega_{ji} = \mathbf{v}_i$, and can be listed explicitly as [cf. (51)]

$$\begin{aligned} \{\mathbf{v}_1 &= \omega_{j1}\}_{j\in\mathcal{B}_1} \\ &\vdots \\ \{\mathbf{v}_J &= \omega_{jJ}\}_{j\in\mathcal{B}_J}. \end{aligned} \tag{54}$$

Since the l.h.s. of (54) coincides with the l.h.s. of (51), the set of equations in (54) can be likewise written as [cf. (53)]

$$\mathbf{A}'\mathbf{v} = \omega'' \tag{55}$$

where now $\omega'' := [\{\omega_{j1}^T\}_{j\in\mathcal{B}_1},\ldots,\{\omega_{jJ}^T\}_{j\in\mathcal{B}_J}]^T$. Notice that ω'' is a permuted version of ω', since it can be obtained from ω' by replacing vector ω_{ji} in ω' with vector ω_{ij}. Hence, using a $2|\mathcal{E}|\times 2|\mathcal{E}|$ permutation matrix $\mathbf{E}$ and letting $\otimes$ denote the Kronecker product, ω'' can be related to ω' as

$$\omega'' = (\mathbf{E}\otimes\mathbf{I}_{p+1})\omega' \tag{56}$$

where $\mathbf{E} := [\{\mathbf{e}_{1i}\}_{i\in\mathcal{B}_1},\ldots,\{\mathbf{e}_{Ji}\}_{i\in\mathcal{B}_J}]$, and $\mathbf{e}_{ji}$ is a $2|\mathcal{E}|\times 1$ indicator vector given by

$$\mathbf{e}_{ji} := \left[\begin{array}{c} \mathbf{e}_1^{(ji)} \\ \vdots \\ \mathbf{e}_J^{(ji)} \end{array}\right]$$

with sub-blocks $\mathbf{e}_{i'}^{(ji)} := [\{\delta(j-j',i-i')\}_{j'\in\mathcal{B}_{i'}}]^T$, and $\delta(\cdot,\cdot)$ denoting Kronecker's delta function. Intuitively, $\mathbf{e}_{ji}$ identifies with a one the position where ω_{ji} in ω' is to be re-allocated in ω''.

Substituting (56) into (55) yields

$$\mathbf{A}'\mathbf{v} = (\mathbf{E}\otimes\mathbf{I}_{p+1})\omega'. \tag{57}$$

Concatenating (53) and (57) one arrives at

$$\mathbf{A}\mathbf{v} = \mathbf{E}'\omega' \tag{58}$$

where

$$\mathbf{A} := \left[\begin{array}{c} \mathbf{A}' \\ \mathbf{A}' \end{array}\right] \quad \text{and} \quad \mathbf{E}' := \left[\begin{array}{c} \mathbf{I}_{2(p+1)|\mathcal{E}|} \\ \mathbf{E}\otimes\mathbf{I}_{p+1} \end{array}\right]. \tag{59}$$

Using (58), problem (7) can be re-written as

$$\begin{array}{ll} \min\limits_{\mathbf{v},\omega,\{\xi_j\}} & \frac{1}{2}\sum\limits_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j + JC\sum\limits_{j=1}^{J}\mathbf{1}_j^T\xi_j \\ \text{s.t.} & \mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j \succeq \mathbf{1}_j - \xi_j \qquad \forall j\in\mathcal{J} \\ & \xi_j \succeq \mathbf{0}_j \qquad \forall j\in\mathcal{J} \\ & \mathbf{A}\mathbf{v} = \mathbf{E}'\omega'. \end{array} \tag{60}$$

It is known that the slack variables $\{\xi_j\}$ can be eliminated by introducing the hinge loss function $\ell(y,[\mathbf{x}^T,1]\mathbf{v}) := \max\{0,1-y[\mathbf{x}^T,1]\mathbf{v}\}$ (Schölkopf and Smola, 2002), which reduces (60) to its equivalent form

$$\begin{array}{ll} \min\limits_{\mathbf{v},\omega} & \frac{1}{2}\sum\limits_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j + JC\sum\limits_{j=1}^{J}\sum\limits_{n=1}^{N_j}\ell(y_{jn},[\mathbf{x}_{jn}^T,1]\mathbf{v}_j) \\ \text{s.t.} & \mathbf{A}\mathbf{v} = \mathbf{E}'\omega'. \end{array} \tag{61}$$

Comparing the latter with (45), it follows readily that (61), which is equivalent to (7), belongs to the ADMoM-solvable class since (61) is subsumed by (45) with the special choices

$$\begin{aligned} F_1(\mathbf{v}) &:= \frac{1}{2}\sum_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j + JC\sum_{j=1}^{J}\sum_{n=1}^{N_j}\ell(y_{jn},[\mathbf{x}_{jn}^T,1]\mathbf{v}_j), \\ F_2(\omega) &:= 0, \\ \mathcal{P}_1 &:= \mathbb{R}^{(p+1)J}, \\ \mathcal{P}_2 &:= \{\omega\in\mathbb{R}^{4(p+1)|\mathcal{E}|}\,|\,\omega=\mathbf{E}'\omega' \text{ for some } \omega'\in\mathbb{R}^{2(p+1)|\mathcal{E}|}\} \end{aligned} \tag{62}$$

where $\omega := \mathbf{E}'\omega'$ is now placed in the constraint set $\mathcal{P}_2$.

So far it has been proved that the problem in (7) can be cast as (45). The ADMoM iterations for (45) are (47)-(49), with corresponding iterates $\mathbf{v}(t)$, $\omega(t)$ and $\alpha(t)$. Given the constraint set $\mathcal{P}_2$ in (62), for every iterate $\omega(t)$ there exists a unique $\omega'(t)$ satisfying $\omega(t)=\mathbf{E}'\omega'(t)$, due to the fact that $\mathbf{E}'$ in (59) is full column rank. Hence, $\omega(t)$ can be replaced by $\mathbf{E}'\omega'(t)$ in iterations (47)-(49). Iteration (9) then follows by re-introducing the slack variables $\{\xi_j(t)\}$. Iteration (10) follows because now ω' is unconstrained. Finally, iterations (11) and (12) follow from (49) by splitting $\alpha(t)$ into appropriate sub-groups of vectors $\{\alpha_{ji1}(t)\}$ and $\{\alpha_{ji2}(t)\}$, respectively.

Appendix D. Proof of Lemma 3

The goal of this appendix is to show that iterations (9)-(12) reduce to (13)-(14). To start, notice that the cost in (10) is linear-quadratic w.r.t. ω_{ji}. Thus, setting the derivative of $\mathcal{L}$ w.r.t. ω_{ji} equal to zero, $\omega_{ji}(t+1)$ can be found in closed form as

$$\omega_{ji}(t+1)=\frac{1}{2\eta}(\alpha_{ji1}(t)-\alpha_{ji2}(t))+\frac{1}{2}(\mathbf{v}_j(t+1)+\mathbf{v}_i(t+1)). \tag{63}$$

Substituting (63) into (11) and (12), yields

$$\alpha_{ji1}(t+1) = \frac{1}{2}(\alpha_{ji1}(t)+\alpha_{ji2}(t))+\frac{\eta}{2}(\mathbf{v}_j(t+1)-\mathbf{v}_i(t+1)), \tag{64}$$

$$\alpha_{ji2}(t+1) = \frac{1}{2}(\alpha_{ji1}(t)+\alpha_{ji2}(t))+\frac{\eta}{2}(\mathbf{v}_j(t+1)-\mathbf{v}_i(t+1)). \tag{65}$$

Suppose now that $\alpha_{ji1}(t)$ and $\alpha_{ji2}(t)$ are initialized identically to zero at every node j; that is, $\alpha_{ji1}(0)=\alpha_{ji2}(0)=\mathbf{0}_{(p+1)\times 1}\ \forall j\in\mathcal{J}$ and $\forall i\in\mathcal{B}_j$. From (64) and (65), it follows easily that $\alpha_{ji1}(1)=\alpha_{ji2}(1)$. Similarly, if $\alpha_{ji1}(t-1)=\alpha_{ji2}(t-1)$, then by induction $\alpha_{ji1}(t)=\alpha_{ji2}(t)$. Thus, only one set of multipliers, say $\{\alpha_{ji1}\}$, needs to be stored and updated per node j.

Upon substituting $\omega_{ji}(t+1)=(1/2)(\mathbf{v}_j(t+1)+\mathbf{v}_i(t+1))$ into the objective function of (9) and using $\alpha_{ji1}(t)=\alpha_{ji2}(t)$, one obtains

$$\begin{aligned} \mathcal{L}'(\{\mathbf{v}_j\},\{\xi_j\},\{\mathbf{v}_j(t)\},\{\alpha_{ji1}(t)\}) &= \frac{1}{2}\sum_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j + JC\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j \\ &+\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ji1}^T(t)(\mathbf{v}_j-\mathbf{v}_i)+\frac{\eta}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{v}_j-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2+\frac{\eta}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{v}_i-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2. \end{aligned} \tag{66}$$

The first double sum in the right hand side (r.h.s.) of (66) can be rewritten as

$$\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ji1}^T(t)(\mathbf{v}_j-\mathbf{v}_i)=\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\mathbf{v}_j^T(\alpha_{ji1}(t)-\alpha_{ij1}(t))$$
$$=2\sum_{j=1}^{J}\mathbf{v}_j^T\sum_{i\in\mathcal{B}_j}\alpha_{ji1}(t) \tag{67}$$

where the first equality follows because $\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ji1}^T(t)\mathbf{v}_i=\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\alpha_{ij1}^T(t)\mathbf{v}_j$. Intuitively, the r.h.s. computes the sum by fixing a node j and adding the inner products of $\mathbf{v}_j$ with the incoming Lagrange multipliers $\alpha_{ij1}(t)$; while the left hand side performs the same sum by fixing a node j and adding the inner products of outgoing Lagrange multipliers $\alpha_{ji1}(t)$ and the corresponding $\mathbf{v}_i$ neighbors. The second equality on (67) holds because the all-zero initialization of the multipliers implies that $\alpha_{ji1}(t)=-\alpha_{ij1}(t)\ \forall t$ [cf. from (64)-(65)]. Likewise, the second and third double sums in the r.h.s. of (66) can be simplified to

$$\frac{\eta}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left[\left\|\mathbf{v}_j-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2+\left\|\mathbf{v}_i-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2\right]=\eta\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{v}_j-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2. \tag{68}$$

Lemma 3 follows after substituting (67) and (68) into (66), and defining $\alpha_j(t):=\sum_{i\in\mathcal{B}_j}\alpha_{ji1}(t)$.

Appendix E. Proof of Proposition 1

Letting $\lambda_j:=[\lambda_{j1},\ldots,\lambda_{jN_j}]^T$ and $\mu_j:=[\mu_{j1},\ldots,\mu_{jN_j}]^T$ denote Lagrange multipliers associated with the constraints $\mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j\succeq\mathbf{1}_j-\xi_j$ and $\xi_j\succeq\mathbf{0}_j$, respectively, the Lagrangian corresponding to (13) is given by

$$\begin{aligned}&\mathcal{L}''(\{\mathbf{v}_j\},\{\xi_j\},\{\lambda_j\},\{\mu_j\},\{\mathbf{v}_j(t)\},\{\alpha_j(t)\})\\&\quad=\frac{1}{2}\sum_{j=1}^{J}\mathbf{v}_j^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{v}_j+JC\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j-\sum_{j=1}^{J}\lambda_j^T(\mathbf{Y}_j\mathbf{X}_j\mathbf{v}_j-\mathbf{1}_j+\xi_j)\\&\quad-\sum_{j=1}^{J}\mu_j^T\xi_j+2\sum_{j=1}^{J}\alpha_j^T(t)\mathbf{v}_j+\eta\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{v}_j-\frac{1}{2}[\mathbf{v}_j(t)+\mathbf{v}_i(t)]\right\|^2.\end{aligned} \tag{69}$$

The KKT conditions yield per iteration the primal and dual variables in (69) as follows

$$\mathbf{v}_j(t+1) = \mathbf{U}_j^{-1}\left(\mathbf{X}_j^T\mathbf{Y}_j\lambda_j(t+1)-2\alpha_j(t)+\eta\sum_{i\in\mathcal{B}_j}(\mathbf{v}_j(t)+\mathbf{v}_i(t))\right), \tag{70}$$

$$\mathbf{0}_j = JC\mathbf{1}_j-\lambda_j-\mu_j \tag{71}$$

where $\lambda_j(t+1)$ is the optimal Lagrange multiplier after iteration $t+1$, and the inverse of $\mathbf{U}_j:=(1+2\eta|\mathcal{B}_j|)\mathbf{I}_{p+1}-\Pi_{p+1}$ always exists.

The KKT conditions also require $\lambda_j\succeq\mathbf{0}_j$ and $\mu_j\succeq\mathbf{0}_j$, which allows (71) to be replaced by $\mathbf{0}_j\preceq\lambda_j\preceq JC\mathbf{1}_j$. To carry out the iteration (70) at every node, the optimal values $\lambda_j(t+1)$ of the

Lagrange multipliers λ_j are found by solving the Lagrange dual problem associated with (69). The pertinent dual function is given by

$$\mathcal{L}_\lambda(\{\lambda_j\}) = \sum_{j=1}^{J} -\frac{1}{2}\lambda_j^T \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{X}_j^T \mathbf{Y}_j \lambda_j + \left(\mathbf{1}_j - \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{f}_j(t)\right)^T \lambda_j \tag{72}$$

where $\mathbf{f}_j(t) := 2\alpha_j(t) - \eta \sum_{i\in\mathcal{B}_j}[\mathbf{v}_j(t) + \mathbf{v}_i(t)]$. Note that the Lagrange multipliers $\{\mu_j\}$ are not present in $\mathcal{L}_\lambda$. From (72), the Lagrange dual problem can be decoupled if each node j has access to the $\mathbf{v}_i(t)$ estimates of its neighboring nodes. Thus, $\lambda_j(t+1)$ is given by

$$\lambda_j(t+1) = \arg \max_{\lambda_j:\, \mathbf{0}_j \preceq \lambda_j \preceq JC\mathbf{1}_j} -\frac{1}{2}\lambda_j^T \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{X}_j^T \mathbf{Y}_j \lambda_j + \left(\mathbf{1}_j - \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{f}_j(t)\right)^T \lambda_j \tag{73}$$

The dual variable update in (73) and the primal variable update in (70) coming from the KKT optimality, are precisely iterations (16) and (17) of Proposition 1, which together with (18) correspond to iterations (13) and (14). Lemma 3 shows that (13) and (14) are equivalent to (9)-(12). Lemma 2 establishes that (9)-(12) in turn correspond to the ADMoM iterations (47)-(49) of Appendix A. As stated in Proposition 4 in Appendix A, convergence of the ADMoM iterations (47)-(49) is guaranteed so long as: (i) $\mathcal{P}_1$ is bounded; or, (ii) $\mathbf{A}^T\mathbf{A}$ is nonsingular. Since for the problem at hand matrix $\mathbf{A}$ in (59) satisfies condition (ii), the iterates for (16)-(18) in Proposition 1 converge to the optimal solution of (7) for any $\eta > 0$.

Appendix F. Proof of Theorem 1

For simplicity, this theorem will be proved for purely linear discriminant functions $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. Consider the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ of functions $g(\mathbf{x})$ with corresponding positive semi-definite kernel $K : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$, defined as

$$\mathcal{H} := \overline{\left\{ g(\cdot) = \sum_{n=1}^{N} \gamma_n K(\cdot, \mathbf{x}_n) :\ N \in \mathbb{N},\ \gamma_1, \ldots, \gamma_N \in \mathbb{R},\ \mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathcal{X} \right\}}$$

with $\overline{\mathcal{A}}$ denotes the completion of the set $\mathcal{A}$.

The parameter optimization problem (23) can be written in terms of the Hinge loss function $\ell(y, g(\mathbf{x})) := \max\{0, 1 - yg(\mathbf{x})\}$, and the RKHS-induced norm $\|g\|_{\mathcal{H}}^2$ as a regularized optimization problem to obtain, (see, e.g., Schölkopf and Smola, 2002)

$$\begin{aligned} \min_{\{g_j\in\mathcal{H}\}} \quad & \frac{1}{2}\sum_{j=1}^{J} \left\|g_j\right\|_{\mathcal{H}}^2 + JC \sum_{j=1}^{J}\sum_{n=1}^{N_j} \ell(y_{jn}, g_j(\mathbf{x}_{jn})) \\ \text{s.t. } & g_j(\chi_l) = g_i(\chi_l) \quad \forall j \in \mathcal{J}, i \in \mathcal{B}_j, l = 1, \ldots, L. \end{aligned} \tag{74}$$

Given the optimal Lagrange multipliers ς_{jil}^* for the constraints $\{g_j(\chi_l) = g_i(\chi_l)\}$, the solution $\{g_j^*\}$ of (74) can be obtained from its Lagrangian as

$$\{g_j^*\} = \arg \min_{\{g_j\in\mathcal{H}\}} \frac{1}{2}\sum_{j=1}^{J} \left\|g_j\right\|_{\mathcal{H}}^2 + JC \sum_{j=1}^{J}\sum_{n=1}^{N_j} \ell(y_{jn}, g_j(\mathbf{x}_{jn})) + \sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\sum_{l=1}^{L} \varsigma_{jil}^*(g_j(\chi_l) - g_i(\chi_l)). \tag{75}$$

Arguing as in (67), the last term in (75) can be written as

$$\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\sum_{l=1}^{L}\varsigma_{jil}^{*}(g_j(\chi_l)-g_i(\chi_l))=\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\sum_{l=1}^{L}(\varsigma_{jil}^{*}-\varsigma_{ijl}^{*})g_j(\chi_l)$$

thus rendering the cost in (75) separable across j. Hence, each g_j^* can be obtained per node as

$$g_j^{*}=\arg\min_{g_j\in\mathcal{H}}V(\mathbf{x}_{j1},\ldots,\mathbf{x}_{jN_j},y_{j1},\ldots,y_{jN_j},\{\chi_l\},g_j)+\frac{1}{2}\left\|g_j\right\|_{\mathcal{H}}^{2} \tag{76}$$

where $V(\mathbf{x}_{j1},\ldots,\mathbf{x}_{jN_j},y_{j1},\ldots,y_{jN_j},\{\chi_l\},g_j):=JC\sum_{n=1}^{N_j}\ell(y_{jn},g_j(\mathbf{x}_{jn}))+\sum_{i\in\mathcal{B}_j}\sum_{l=1}^{L}(\varsigma_{jil}^{*}-\varsigma_{ijl}^{*})g_j(\chi_l)$. Applying the Representer Theorem to (76) as in Wahba (1990) and Schölkopf and Smola (2002) one readily arrives at

$$g_j^{*}(\mathbf{x})=\sum_{n=1}^{N_j}a_{jn}^{*}K(\mathbf{x},\mathbf{x}_{jn})+\sum_{l=1}^{L}c_{jl}^{*}K(\mathbf{x},\chi_l). \tag{77}$$

Appendix G. Proof of Proposition 2

Recall that $\mu_j:=[\mu_{j1},\ldots,\mu_{jN_j}]^T$ denotes the Lagrange multiplier associated with the constraint $\xi_j\succeq\mathbf{0}_j$ (cf. Appendix E). The Lagrangian corresponding to (25) is given by

$$\begin{aligned}
&\mathcal{L}''(\{\mathbf{w}_j\},\{b_j\},\{\xi_j\},\{\lambda_j\},\{\mu_j\},\{\mathbf{w}_j(t)\},\{b_j(t)\},\{\alpha_j(t)\},\{\beta_j(t)\})\\
&\quad=\frac{1}{2}\sum_{j=1}^{J}\|\mathbf{w}_j\|^2+JC\sum_{j=1}^{J}\mathbf{1}_j^T\xi_j-\sum_{j=1}^{J}\lambda_j^T(\mathbf{Y}_j\Phi(\mathbf{X}_j)\mathbf{w}_j-\mathbf{1}_jb_j+\xi_j)\\
&\quad-\sum_{j=1}^{J}\mu_j^T\xi_j+2\sum_{j=1}^{J}\alpha_j^T(t)\mathbf{G}\mathbf{w}_j+2\sum_{j=1}^{J}\beta_j(t)b_j\\
&\quad+\eta\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|\mathbf{G}\left[\mathbf{w}_j-\frac{1}{2}(\mathbf{w}_j(t)+\mathbf{w}_i(t))\right]\right\|^2+\eta\sum_{j=1}^{J}\sum_{i\in\mathcal{B}_j}\left\|b_j-\frac{1}{2}[b_j(t)+b_i(t)]\right\|^2.
\end{aligned}$$

From the KKT conditions for (25) it follows that

$$\mathbf{w}_j(t+1)=\widetilde{\mathbf{U}}_j^{-1}\left\{\Phi^T(\mathbf{X}_j)\mathbf{Y}_j\lambda_j(t+1)-\mathbf{G}^T\left[2\alpha_j(t)-\eta\sum_{i\in\mathcal{B}_j}\mathbf{G}(\mathbf{w}_j(t)+\mathbf{w}_i(t))\right]\right\}, \tag{78}$$

$$b_j(t+1)=\frac{1}{2\eta|\mathcal{B}_j|}\left[\mathbf{1}_j^T\mathbf{Y}_j\lambda_j(t+1)-2\beta_j(t)+\eta\sum_{i\in\mathcal{B}_j}(\beta_j(t)+\beta_i(t))\right] \tag{79}$$

where $\lambda_j(t+1)$ is the optimal Lagrange multiplier at iteration $t+1$, and $\widetilde{\mathbf{U}}_j:=\mathbf{I}_P+2\eta|\mathcal{B}_j|\mathbf{G}^T\mathbf{G}$. Using the Sherman-Morrison-Woodbury formula (Golub and Van Loan, 1996)

$$\widetilde{\mathbf{U}}_j^{-1}=\mathbf{I}_P-2\eta|\mathcal{B}_j|\mathbf{G}^T(\mathbf{I}_L+2\eta|\mathcal{B}_j|\mathbf{G}\mathbf{G}^T)^{-1}\mathbf{G}. \tag{80}$$

Substituting (80) into (78), left-multiplying by $\phi^T(\mathbf{x})$, and recalling that $\phi^T(\mathbf{x})\phi^T(\mathbf{x}')=K(\mathbf{x},\mathbf{x}')$, yields

$$\begin{aligned}
\phi^T(\mathbf{x})\mathbf{w}_j(t+1)=&\left(\mathbf{k}^T(\mathbf{x},\mathbf{X}_j)-2\eta|\mathcal{B}_j|\mathbf{k}^T(\mathbf{x},\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\mathbf{X}_j)\right)\mathbf{Y}_j\lambda_j(t+1)\\
&-\left(\mathbf{k}^T(\mathbf{x},\Gamma)-2\eta|\mathcal{B}_j|\mathbf{k}^T(\mathbf{x},\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\Gamma)\right)\widetilde{\mathbf{f}}_j(t+1)
\end{aligned} \tag{81}$$

where the entries of the kernel vector are $[\mathbf{k}(\mathbf{x},\mathbf{X}_j)]_n := K(\mathbf{x},\mathbf{x}_{jn})$ and $[\mathbf{k}(\mathbf{x},\Gamma)]_l := K(\mathbf{x},\chi_l)$.

Note that $g_j^{(t)}(\mathbf{x})$ in (31) follows from (77) and can be written as $g_j^{(t)}(\mathbf{x}) = \phi^T(\mathbf{x})\mathbf{w}_j(t) + b_j(t)$. Grouping terms in (81) that right-multiply $\mathbf{k}^T(\mathbf{x},\mathbf{X}_j)$ and those that right-multiply $\mathbf{k}^T(\mathbf{x},\Gamma)$, yields $\mathbf{a}_j(t)$ as in (32) and $\mathbf{c}_j(t)$ as in (33), respectively. Finally, $b_j(t)$ in (34) is given by (79).

Appendix H. Proof of Proposition 3

To obtain iteration (35), consider first the dual problem for (25), that is

$$\begin{aligned}\lambda_j(t+1) \;=\; \underset{\lambda_j:\,\mathbf{0}_j\preceq\lambda_j\preceq JC\mathbf{1}_j}{\arg\max} & -\frac{1}{2}\lambda_j^T\mathbf{Y}_j\left(\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j)+\frac{\mathbf{1}\mathbf{1}^T}{2\eta|\mathcal{B}_j|}\right)\mathbf{Y}_j\lambda_j \\ & +\left(\mathbf{1}_j-\mathbf{Y}_j\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T\widetilde{\mathbf{f}}_j(t)-\frac{h_j(t)\mathbf{1}^T}{2\eta|\mathcal{B}_j|}\right)^T\lambda_j \end{aligned} \tag{82}$$

where $\widetilde{\mathbf{U}}_j^{-1}$ is given by (80), and $\widetilde{\mathbf{f}}_j(t)$ as in Proposition 2. Using (80), the term $\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j)$ can be written in terms of inner products, and summarized via kernels as

$$\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j) = \mathbf{K}(\mathbf{X}_j,\mathbf{X}_j) - 2\eta|\mathcal{B}_j|\mathbf{K}(\mathbf{X}_j,\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\mathbf{X}_j). \tag{83}$$

Likewise, the term $\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T\widetilde{\mathbf{f}}_j(t)$ can be expressed as

$$\Phi(\mathbf{X}_j)\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T\widetilde{\mathbf{f}}_j(t) = \Big(\mathbf{K}(\mathbf{X}_j,\Gamma) - 2\eta|\mathcal{B}_j|\mathbf{K}(\mathbf{X}_j,\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\Gamma)\Big)\widetilde{\mathbf{f}}_j(t). \tag{84}$$

Plugging (83) and (84) into (82), yields (35).

To obtain iteration (36), left-multiply $\mathbf{w}_j(t+1)$ in (78) by $\mathbf{G}$ to arrive at

$$\widetilde{\mathbf{w}}_j(t+1) \;=\; \mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j)\mathbf{Y}_j\lambda_j - \mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T\left[2\alpha_j(t) - \eta\sum_{i\in\mathcal{B}_j}\left(\widetilde{\mathbf{w}}_j(t)+\widetilde{\mathbf{w}}_i(t)\right)\right]. \tag{85}$$

The terms $\mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j)$ and $\mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T$ can be respectively written as

$$\mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\Phi^T(\mathbf{X}_j) = \mathbf{K}(\Gamma,\Gamma) - 2\eta|\mathcal{B}_j|\mathbf{K}(\Gamma,\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\mathbf{X}_j) \tag{86}$$

and

$$\mathbf{G}\widetilde{\mathbf{U}}_j^{-1}\mathbf{G}^T = \mathbf{K}(\Gamma,\Gamma) - 2\eta|\mathcal{B}_j|\mathbf{K}(\Gamma,\Gamma)\widetilde{\mathbf{U}}_j^{-1}\mathbf{K}(\Gamma,\Gamma)\,. \tag{87}$$

Substituting (86) and (87) into (85), yields (36), and completes the proof of the proposition.

References

I. F. Akyildiz, D. Pompili, and T. Melodia. Underwater acoustic sensor networks: Research challenges. *Ad Hoc Networks (Elsevier)*, 3(3):257–279, Mar. 2005.

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd ed., 1999.

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, Mar. 2005.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *Proc. of Neural Info. Processing Systems Conf.*, pages 409–415, Denver, CO, USA, Nov. 2000.

E. Y. Chang, K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, and H. Cui. PSVM: Parallelizing support vector machines on distributed computers. In *21st Neural Info. Processing Systems Conf.*, Vancouver, Canada, Dec. 3-6, 2007.

K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*, 2008.

T. Do and F. Poulet. Classifying one billion data with a new distributed SVM algorithm. In *Proc. of International Conf. on Research, Innovation and Vision for the Future*, pages 59–66, Ho Chi Minh City, Vietnam, Feb. 12-16, 2006.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2nd edition, 2002.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proc. of 3rd Theory of Cryptography Conference*, pages 265–284, New York, NY, USA, Mar. 4-7, 2006.

I. El-Naqa, Y. Yang, M.N. Wernick, N.P. Galatsanos, and R.M. Nishikawa. A support vector machine approach for detection of microcalcifications. *IEEE Tran. on Medical Imaging*, 21(12): 1552–1563, Dec. 2002.

K. Flouri, B. Beferull-Lozano, and P. Tsakalides. Training a support-vector machine-based classifier in distributed sensor networks. In *Proc. of 14th European Signal Processing Conf.*, Florence, Italy, Sep. 4-8, 2006.

K. Flouri, B. Beferull-Lozano, and P. Tsakalides. Distributed consensus algorithms for SVM training in wireless sensor networks. In *Proc. of 16th European Signal Processing Conf.*, Laussane, Switzerland, Aug. 25-29, 2008.

G. Fung and O. L. Mangasarian. Incremental support vector machine classification. In *Proc. of the SIAM Intl. Conf. on Data Mining*, pages 247–260, Arlington, VA, USA, Apr. 11-13, 2002.

A. Ganapathiraju, J.E. Hamaker, and J. Picone. Applications of support vector machines to speech recognition. *IEEE Tran. on Signal Processing*, 52(8):2348–2355, Aug. 2004.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

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*, volume 17. MIT Press, 2005.

T. Hastie, R. Tishbiranie, and J. Friedman. *The Elements of Statistical Learning*. Spinger, 2nd edition, 2009.

R. Klinkenberg and T. Joachims. Detecting concept drift with support vector machines. In *Proc. of the Seventeenth International Conf. on Machine Learning*, pages 487–494, Stanford, CA, USA, Jun. 29 - Jul. 2, 2000.

K. Langendoen and N. Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Comput. Netw.*, 43(4):499–518, 2003.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, Nov. 1998.

F. F. Li. Estimation of intelligibility from received arbitrary speech signals with support vector machine. In *Proc. of International Conference on Machine Learning and Cybernetics*, volume 6, pages 3755–3760, Guangzhou, China, Aug. 2005.

Y. Liang, M.L. Reyes, and J.D. Lee. Real-time detection of driver cognitive distraction using support vector machines. *IEEE Tran. on Intelligent Transportation Systems*, 8(2):340–350, Jun. 2007.

Y. Lu, V. Roychowdhury, and L. Vandenberghe. Distributed parallel support vector machines in strongly connected networks. *IEEE Tran. on Neural Networks*, 19(7):1167–1178, Jul. 2008.

U. Markowska-Kaczmar and P. Kubacki. Support vector machines in handwritten digits classification. In *Proc. of 5th International Conference on Intelligent Systems Design and Applications*, pages 406–411, Wroclaw, Poland, Sep. 8-11, 2005.

A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J. J. Navarro-Abellan. Distributed support vector machines. *IEEE Tran. on Neural Networks*, 17(4):1091–1097, Jul. 2006.

C. H. Papadimitriou, *Computational Complexity*. Addison- Wesley, 1993.

J. B. Predd, S. R. Kulkarni, and H. V. Poor. Distributed kernel regression: An algorithm for training collaboratively. *IEEE Trans. on Information Theory*, 55(4):1856–1871, Apr. 2009.

B. Schölkopf and A. Smola. *Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1st edition, 1998.

G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conf. Series in Applied Mathematics*. SIAM, Philadelphia, PA, USA, 1990.

J. Weston, A. Elisseeff, G. BakIr, and F. Sinz. The spider machine learning toolbox, 2006. URL `http://www.kyb.tuebingen.mpg.de/bs/people/spider/`.

H. Zhu, G. B. Giannakis, and A. Cano. Distributed in-network channel decoding. *IEEE Trans. on Signal Processing*, 57(10):3970–3983, Oct. 2009.

Journal of Machine Learning Research 11 (2010) 1709-1731 Submitted 1/10; Published 5/10

Estimation of a Structural Vector Autoregression Model Using Non-Gaussianity

Aapo Hyvärinen AAPO.HYVARINEN@HELSINKI.FI
Department of Mathematics and Statistics
University of Helsinki
Helsinki, Finland

Kun Zhang KUN.ZHANG@TUEBINGEN.MPG.DE
Department of Computer Science and HIIT
University of Helsinki
Helsinki, Finland

Shohei Shimizu SSHIMIZU@AR.SANKEN.OSAKA-U.AC.JP
Institute of Scientific and Industrial Research
Osaka University
Osaka, Japan

Patrik O. Hoyer PATRIK.HOYER@HELSINKI.FI
Department of Computer Science and HIIT
University of Helsinki
Helsinki, Finland

Editor: Peter Dayan

Abstract

Analysis of causal effects between continuous-valued variables typically uses either autoregressive models or structural equation models with instantaneous effects. Estimation of Gaussian, linear structural equation models poses serious identifiability problems, which is why it was recently proposed to use non-Gaussian models. Here, we show how to combine the non-Gaussian instantaneous model with autoregressive models. This is effectively what is called a structural vector autoregression (SVAR) model, and thus our work contributes to the long-standing problem of how to estimate SVAR's. We show that such a non-Gaussian model is identifiable without prior knowledge of network structure. We propose computationally efficient methods for estimating the model, as well as methods to assess the significance of the causal influences. The model is successfully applied on financial and brain imaging data.

Keywords: structural vector autoregression, structural equation models, independent component analysis, non-Gaussianity, causality

1. Introduction

Analysis of causal influences or effects has become an important topic in statistics and machine learning, and has recently found applications in, for example, neuroinformatics (Roebroeck et al., 2005; Kim et al., 2007) and bioinformatics (Opgen-Rhein and Strimmer, 2007). While the deeper meaning of causality has been formalized in different ways (Pearl, 2000; Spirtes et al., 1993), we

∗. Also in Department of Computer Science and HIIT, University of Helsinki.

consider the problem here from a practical viewpoint, where coefficients in conventional statistical models are interpreted as causal influences.

For continuous-valued variables, such an analysis is typically performed in two different ways. First, if the time-resolution of the measurements is higher than the time-scale of causal influences, one can estimate a classic autoregressive (AR) model with time-lagged variables and interpret the autoregressive coefficients as causal effects. Second, if the measurements have a lower time resolution than the causal influences, or if the data has no temporal structure at all, one can use a model in which the influences are instantaneous, leading to Bayesian networks or structural equation models (SEM); see Bollen (1989).[1]

While estimation of autoregressive methods can be solved by classic regression methods, the case of instantaneous effects is much more difficult. Most methods suffer from lack of identifiability, because covariance information alone is not sufficient to uniquely characterize the model parameters. Prior knowledge of the structure (fixing some of the connections to zero) of the SEM is then necessary for most practical applications. However, a method was recently proposed which uses the non-Gaussian structure of the data to overcome the identifiability problem (Shimizu et al., 2006). If the disturbance variables (external influences) are non-Gaussian, no prior knowledge on the network structure is needed to estimate the linear SEM, except for the ubiquitous assumption of a directed acyclic graph (DAG) and the assumption of no latent variables. (The case of latent variables, that is, unobserved confounders, was later considered by Hoyer et al., 2008.)

Here, we consider the general case where causal influences can occur either instantaneously or with considerable time lags. Such models are one example of structural vector autoregressive (SVAR) models popular in econometric theory, in which numerous attempts have been made for its estimation, see, for example, Swanson and Granger (1997), Demiralp and Hoover (2003) and Moneta and Spirtes (2006). We propose to use non-Gaussianity to estimate the model. We show that this variant of the model is identifiable without other restrictions on the network structure than acyclicity and no latent variables. To our knowledge, no model proposed for this problem has been shown to be fully identifiable without prior knowledge of network structure. We further propose two computationally efficient methods for estimating the model based on the theory of independent component analysis or ICA (Hyvärinen et al., 2001).

The proposed non-Gaussian model not only allows estimation of both instantaneous and lagged effects; it also shows that taking instantaneous influences into account can change the values of the time-lagged coefficients quite drastically. Thus, we see that neglecting instantaneous influences can lead to misleading interpretations of causal effects. The framework further points towards generalizations of the well-known Granger causality measure (Granger, 1969).

The paper is structured as follows. We first define the model and discuss its relation to other models in Section 2. We motivate the key assumption of non-Gaussianity in Section 3. Next, we derive the likelihood and discuss some of its interpretations in Section 4. In Section 5 we propose two computationally efficient estimation methods and compare them with simulations. Assessement of the results using testing is considered in Section 6. Section 7 discusses some interesting phenomena concerning the interpretation of the estimated parameter values. Experiments on financial and neuroscientific data are made in Section 8. Some extensions of the model are discussed in Section 9, and Section 10 concludes the paper. Preliminary results were published in Hyvärinen et al. (2008) and Zhang and Hyvärinen (2009).

1. Here, we assume that the learning is unsupervised, that is, the inputs to the system are not known or used. If the inputs to the system are known, methods such as dynamic causal modelling can be used (Friston et al., 2003).

2. A Non-Gaussian Structural Vector Autoregressive Model

In this section, we define our new model.

2.1 Background and Notation

Let us denote the observed time series by $x_i(t), i = 1,\ldots,n, t = 1,\ldots,T$ where i is the index of the time series and t is the time index. All the time series (variables) are collected into a single column vector $\mathbf{x}(t)$. Without loss of generality, we can assume that the $x_i(t)$ have zero mean.

In autoregressive modelling, we would model the dynamics by a model of the form

$$\mathbf{x}(t) = \sum_{\tau=1}^{k} \mathbf{B}_\tau \mathbf{x}(t-\tau) + \mathbf{e}(t) \tag{1}$$

where k is the number of time-delays used, that is, the order of the autoregressive model, $\mathbf{B}_\tau, \tau = 1,\ldots,k$ are $n \times n$ matrices, and $\mathbf{e}(t)$ is the innovation process.

In structural equation models (SEM), or continuous-valued Bayesian networks, there is no time structure in the data, and the variables are simply modelled as functions of the other variables:

$$\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{e} \tag{2}$$

where the vector $\mathbf{e}$ is the vector of disturbances or external influences. The diagonal of $\mathbf{B}$ is defined to be zero. It is typically assumed that we have a sample of observations which are independent and identically distributed.

2.2 Definition of Our Model

In many applications, the influences between the $x_i(t)$ can be both instantaneous and lagged. Thus, we combine the two models in (1) and (2) into a single model. Denote by $\mathbf{B}_\tau$ the $n \times n$ matrix of the causal effects between the variables x_i with time lag $\tau, \tau = 0,\ldots,k$. For $\tau > 0$, the effects are ordinary autoregressive effects from the past to the present, while for $\tau = 0$, the effects are "instantaneous".

We define our model by a straightforward combination of (1) and (2) as

$$\mathbf{x}(t) = \sum_{\tau=0}^{k} \mathbf{B}_\tau \mathbf{x}(t-\tau) + \mathbf{e}(t) \tag{3}$$

where the $e_i(t)$ are random processes modelling the disturbances. We make the following assumptions on the $e_i(t)$:

1. The $e_i(t)$ are are mutually independent, both of each other and over time. This is a typical assumption in autoregressive models.

2. The $e_i(t)$ are *non-Gaussian*, which is an important assumption which distinguishes our model from classic models, whether autoregressive models, structural-equation models, or Bayesian networks.

3. The matrix modelling instantaneous effects, $\mathbf{B}_0$, corresponds to an *acyclic* graph, as is typical in causal analysis. However this assumption may not be strictly necessary as will be discussed

in Section 9. The acyclicity is equivalent to the existence of a permutation matrix $\mathbf{P}$, which corresponds to a "causal" ordering of the variables x_i, such that the matrix $\mathbf{PB_0P}^T$ is lower-triangular (i.e., entries above the diagonal are zero). Acyclicity also implies that the entries on the diagonal are zero, even before such a permutation.

2.3 Relation to Other Models

Next, we discuss the relationships of our model with other models.

2.3.1 LINEAR NON-GAUSSIAN ACYCLIC MODEL

Our model is a generalization of the linear non-Gaussian acyclic model (LiNGAM) proposed in Shimizu et al. (2006). If the order of the autoregressive part is zero, that is, $k = 0$, the model is nothing else than the LiNGAM model, modelling instantaneous effects only. As shown in Shimizu et al. (2006), the assumption of non-Gaussianity of the e_i enables estimation of the model. This is because the non-Gaussian structure of the data provides information not contained in the covariance matrix which is the only source of information in most methods.

Non-Gaussianity enables model estimation using independent component analysis, which solves the non-identifiability of factor analytic models using the assumption of non-Gaussianity of the factors (Comon, 1994; Hyvärinen et al., 2001). In fact, the estimation method in Shimizu et al. (2006) uses an ICA algorithm as an essential part. This is because we can transform (2) into the factor-analytic model of ICA:

$$\mathbf{x} = (\mathbf{I} - \mathbf{B})^{-1}\mathbf{e} \tag{4}$$

where $\mathbf{e}$ is now a vector of latent variables. Under the assumptions of the model, in particular the independence and non-Gaussianity of the disturbances e_i, the model can be essentially estimated (Comon, 1994). The acyclicity assumption also ensures that $\mathbf{I} - \mathbf{B}$ is invertible.

However, there is an important indeterminacy which ICA cannot solve: the order of the components. In a SEM, each disturbance corresponds to one of the observed variables. In contrast, ICA, like most factor-analytic models, gives the factors in no particular order. Thus, after ICA estimation (or as part of the ICA estimation) we have to establish the correspondence between the x_i and the e_i. It was proven by Shimizu et al. (2006) that the correspondence can in fact be established based on the acyclicity of $\mathbf{B}$. Basically, only one of the possible orderings of the rows of $(\mathbf{I} - \mathbf{B})$ is such that all the elements on the diagonal are non-zero, and can thus be scaled to equal one, which has to be the case by definition.

Thus, the LiNGAM model can be estimated by basically first performing ICA estimation and then finding the right ordering of the components based on acyclicity.

2.3.2 AUTOREGRESSIVE MODELS

On the other hand, if the matrix $\mathbf{B}_0$ has all zero entries, the model in Eq. (3) is a classic vector autoregressive model in which future observations are linearly predicted from preceding ones. If we knew in advance that $\mathbf{B}_0$ is zero, the model could thus be estimated by classic regression techniques since we do not have the same variables on the left and right-hand sides of Eq. (3). However, our model would still be different from classic autoregressive models because the disturbances $e_i(t)$ are non-Gaussian.

It is important to note here that an autoregressive model can serve two different goals: prediction and analysis of causality. Our goal here is the latter: We estimate the parameter matrices $\mathbf{B}_\tau$ in order to interpret them as causal effects between the variables. This goal is distinct from simply predicting future outcomes when passively observing the time series, as has been extensively discussed in the literature on causality (Pearl, 2000; Spirtes et al., 1993). Thus, we emphasize that our model is not intended to reduce prediction errors if we want to predict $x_i(t)$ using (passively) observed values of the past $\mathbf{x}(t-1), \mathbf{x}(t-2), \ldots$; for such prediction, an ordinary autoregressive model is likely to be just as good.

2.3.3 STRUCTURAL VECTOR AUTOREGRESSIVE MODELS

Combinations of SEM and vector autoregressive models have been proposed in the econometric literature, and called structural vector autoregressive models (SVAR). Although many of them are quite similar to our model in spirit (Swanson and Granger, 1997; Demiralp and Hoover, 2003; Moneta and Spirtes, 2006), we are not aware of any method in which non-Gaussianity would be an essential assumption. We will see below how the assumption of non-Gaussianity is essential for the identifiability of the model, which has been a serious problem in previous methods. In the next section, we thus consider the justification of this assumption.

3. Why Disturbances Could be Non-Gaussian

Non-Gaussianity is the new assumption in our model. In this section, we attempt to justify why, in many applications, we can consider the $e_i(t)$ to be non-Gaussian. The arguments are based on heteroscedasticity. We do not by any means claim that we are the first to develop these arguments; some of them are well-known and we merely re-iterate them here.

The principle of heteroscedasticity means that the variance of $e_i(t)$ depends on t: in some parts of the time series, it is larger, and in other parts it is smaller. The shape of the distribution conditional to the variance is the same always: often it is assumed to be Gaussian (normal).

We argue that heteroscedasticity is an important reason why, in many cases, the $e_i(t)$ should be strongly non-Gaussian. Even if the Central Limit Theorem is applicable in the sense that $e_i(t)$ is a sum of many different latent independent variables, the disturbances can be very non-Gaussian if, for some reason, the variance of the $e_i(t)$ is changing.

The connection between heteroscedasticity and non-Gaussianity can be developed in a few simple equations. Denote by $z(t)$ a standardized Gaussian random variable. Assume that a disturbance $e(t)$ (dropping the index i for simplicity) is a product of z and a random "variance" variable $v(t)$:

$$e(t) = z(t)v(t)$$

where $z(t)$ and $v(t)$ are independent by definition. We can, in fact, drop the time indices and just consider these time series as random variables. The distribution of v can be of different kinds, whereas the distribution of z is fixed to standardized Gaussian. In the simplest case, v takes only two different values, which means that the data points belong to just two different classes, and the density is then a finite mixture of two Gaussian distributions.

We can simply show the following well-known result: If z is Gaussian, e has always positive kurtosis,[2] regardless of the distribution of v (as long as v^2 has non-zero variance). This is because

$$\text{kurt}(e) = E\{e^4\} - 3(E\{e^2\})^2 = E\{v^4 z^4\} - 3(E\{v^2 z^2\})^2 = 3[E\{v^4\} - (E\{v^2\})^2] \tag{5}$$

which is always positive because it is the variance of v^2 multiplied by three (Beale and Mallows, 1959). It is easy to generalize this result to show that even if z is not Gaussian, the kurtosis is still positive if the variance of v^2 is large enough.

Heteroscedastity can be seen in some important application areas of causal modelling, in particular:

1. In econometrics, heteroscedastic models have a long tradition (Engle, 1995). For example, in financial markets the volatility of a price is often assumed to be changing over time, and volatility is nothing but the variance in some scaling.

2. In brain imaging, the power of rhytmic activity as measured by electroencephalography or magnetoencephalography is non-constant (Hari and Salmelin, 1997). The power is essentially the same as the variance.

We emphasize that the assumption of non-Gaussianity is fundamentally an empirical assumption. It is fulfilled in some application areas and not in others. It can be validated by examining the distributions of the estimates of the $e_i(t)$, which are simply obtained by solving for $\mathbf{e}(t)$ in (3) after estimation of the model. Those estimates are linear functions of the data, which implies that if the data were Gaussian, the $e_i(t)$ would necessarily be Gaussian. Thus, any non-Gaussianity in the estimates is valid evidence for the Gaussianity of the underlying $e_i(t)$. In addition to visual inspection, any formal tests for non-Gaussianity can be used, such as the Shapiro-Wilk test or the Kolmogorov-Smirnov test. (Independence of the $e_i(t)$ can be validated in the same way, although it seems to be more difficult to investigate by visualization or testing.)

However, in practice the question is not whether the disturbances are non-Gaussian but whether they are sufficiently far from Gaussian to enable sufficiently accurate estimation. In the theory of ICA, it has been shown that the asymptotic variance of the estimators is a function of the non-Gaussianity of the components: When their distribution approaches Gaussianity, the asymptotic variance goes to infinity (Cardoso and Laheld, 1996; Hyvärinen et al., 2001). Thus, instead of testing non-Gaussianity it may be much more useful to simply measure the accuracy of the estimates by bootstrapping and similar methods. If the disturbances are Gaussian (or very close to Gaussian), our estimation method is likely to fail completely. Some other assumptions are then needed to obtain identifiability of the model.

It should be also noted that the assumptions of non-Gaussianity and independence cannot be easily disentangled from the assumption of linearity. If there are non-linearities in the system, these may, for example, lead to non-Gaussian residuals even if the original disturbances were Gaussian.

4. Likelihood of the Model

To estimate our model, we start by formulating its likelihood.

2. We use here the definition of kurtosis given in Eq. (5), which is sometimes called excess kurtosis. Thus, kurtosis can be either positive or negative.

4.1 Likelihood of LiNGAM

First, we derive the likelihood of the LiNGAM model (Shimizu et al., 2006) which has not yet been given in the literature. The starting point is the likelihood of the ICA model which is well-known, see, for example, Pham and Garrat (1997) and Hyvärinen et al. (2001). Denote the ICA model by

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

for a square invertible matrix $\mathbf{A}$, and independent non-Gaussian latent variables s_i. Denote the observed sample by $\mathbf{X} = (\mathbf{x}(1), \ldots, \mathbf{x}(T))$ and $\mathbf{W} = \mathbf{A}^{-1}$. The log-likelihood is then usually given in the form

$$\log L_0(\mathbf{X}) = \sum_t \sum_i \log p_i(\mathbf{w}_i^T \mathbf{x}(t)) + T \log \det |\mathbf{W}|$$

where the p_i are the density functions of the independent components (here: disturbances). Since the densities of the disturbances are not specified, we have in general a semi-parametric problem. Different methods have been developed for approximating $\log p_i$, for example, Pham and Garrat (1997), Karvanen and Koivunen (2002) and Chen and Bickel (2006). Here, we have to take into account the fact that those methods usually assume that the independent components have been normalized to unit variance, which is not the case in LiNGAM. Thus, we prefer to modify the formula by normalizing the densities as follows:

$$\log L_1(\mathbf{X}) = \sum_t \sum_i \log \tilde{p}_i(\frac{\mathbf{w}_i^T \mathbf{x}(t)}{\sigma_i}) - T \sum_i \log \sigma_i + T \log \det |\mathbf{W}| \tag{6}$$

where the $\tilde{p}_i$ are the densities of the disturbances standardized to unit variance, and the σ_i^2 are their variances before standardization.

In fact, in practice it has been realized that often one can just fix the $\tilde{p}_i$ to a single function and still obtain a satisfactory estimator. In particular, if we know that the disturbances are all super-Gaussian (i.e., have positive kurtosis), fixing

$$\log \tilde{p}_i(s) = -\sqrt{2}|s| + \text{const.}$$

is enough to provide a consistent estimator under weak constraints (Cardoso and Laheld, 1996; Hyvärinen and Oja, 1998).

In LiNGAM, we have from (4) that in terms of the ICA model, $\mathbf{A}^{-1} = \mathbf{W} = \mathbf{I} - \mathbf{B}_0$ (we use the subindex 0 for $\mathbf{B}$ in LiNGAM to comply with the notation below). Now, we can simplify the likelihood because of the DAG structure. The DAG structure means that for the right permutation of its rows (corresponding to the causal ordering), $\mathbf{W}$ is lower-triangular. The determinant of a triangular matrix is equal to the product of its diagonal elements, and a permutation does not change the determinant, so the determinant of $\mathbf{W}$ is equal to the product of the diagonal elements when the variables are ordered in the causal order. But by definition of $\mathbf{W}$ in LiNGAM, those diagonal elements are all equal to one, so the last term in (6) is zero. So, the likelihood of the LiNGAM model is finally given by

$$\begin{aligned} \log L(\mathbf{X}) &= \sum_t \sum_i \log \tilde{p}_i \left(\frac{\mathbf{w}_i^T \mathbf{x}(t)}{\sigma_i} \right) - T \sum_i \log \sigma_i \\ &= \sum_t \sum_i \log \tilde{p}_i \left(\frac{x_i(t) - \mathbf{b}_{0,i}^T \mathbf{x}(t)}{\sigma_i} \right) - T \sum_i \log \sigma_i \end{aligned} \tag{7}$$

where the variances of the components can be estimated by taking the empirical variances as

$$\sigma_i^2 = \frac{1}{T}\sum_t (x_i(t) - \mathbf{b}_{0,i}^T \mathbf{x}(t))^2.$$

(Alternatively, the σ_i could be obtained by a separate maximization of the likelihood, but this would be more complicated computationally and conceptually.) Here, $\mathbf{W}$ is constrained to correspond to a DAG, with ones added in the diagonal.

4.2 Likelihood of Our Model

Now we can derive the likelihood of our model. First note that from (3) we can derive

$$\mathbf{x}(t) = \sum_{\tau=0}^{k} \mathbf{B}_\tau \mathbf{x}(t-\tau) + \mathbf{e}(t) \Leftrightarrow (\mathbf{I} - \mathbf{B}_0)[\mathbf{x}(t) - \sum_{\tau=1}^{k} (\mathbf{I} - \mathbf{B}_0)^{-1} \mathbf{B}_\tau \mathbf{x}(t-\tau)] = \mathbf{e}(t),$$

which gives

$$\mathbf{x}(t) - \sum_{\tau=1}^{k} (\mathbf{I} - \mathbf{B}_0)^{-1} \mathbf{B}_\tau \mathbf{x}(t-\tau) = \mathbf{B}_0[\mathbf{x}(t) - \sum_{\tau=1}^{k} (\mathbf{I} - \mathbf{B}_0)^{-1} \mathbf{B}_\tau \mathbf{x}(t-\tau)] + \mathbf{e}(t)$$

which shows that the our model in (3) is a LiNGAM model for the residuals $\mathbf{x}(t) - \sum_{\tau=1}^{k} (\mathbf{I} - \mathbf{B}_0)^{-1} \mathbf{B}_\tau \mathbf{x}(t-\tau)$. Denoting

$$\mathbf{M}_\tau = (\mathbf{I} - \mathbf{B}_0)^{-1} \mathbf{B}_\tau \text{ and } \mathbf{W} = \mathbf{I} - \mathbf{B}_0 \tag{8}$$

and replacing $\mathbf{x}(t)$ in (7) by the residuals, we have

$$\log L(\mathbf{X}) = \sum_i \sum_t \log \tilde{p}_i \left(\frac{\mathbf{w}_i^T [\mathbf{x}(t) - \sum_{\tau=1}^{k} \mathbf{M}_\tau \mathbf{x}(t-\tau)]}{\sigma_i} \right) - \log \sigma_i \tag{9}$$

with

$$\sigma_i^2 = \frac{1}{T}\sum_t \left(\mathbf{w}_i^T [\mathbf{x}(t) - \sum_{\tau=1}^{k} \mathbf{M}_\tau \mathbf{x}(t-\tau)] \right)^2.$$

4.3 Information-Theoretic Interpretation

An interesting point to note is that the likelihood is now a sum of the negative entropies of the residuals. The differential entropy of a random variable s can be written using the standardized version of s, denoted by $\tilde{s}$, as follows:

$$H(s) = -\int p_s(u) \log p_s(u) du = -\int p_{\tilde{s}}(u) \log p_{\tilde{s}}(u) du + \log \sigma_s$$

where σ_s^2 is the variance of s. Thus, we can interpret the terms in (9) as the (negative) entropies of the residuals. So, estimation is accomplished by minimizing the "prediction errors" or "uncertainties" in the DAG if the entropies are interpreted as the prediction errors when each variable is predicted by its parents. Note that for Gaussian variables, the entropies are very simple functions of the squared errors (variances), while for non-Gaussian variables, they are also functions of the non-Gaussianity (shape) of the distribution.

5. Practical Estimation Methods

Next we propose two practical methods for estimating the model, and further show how to include a sparseness penalty which may be very useful in practice.

5.1 A Two-Stage Method with Least-Squares Estimation

Optimization of the likelihood is difficult because it contains a complicated combinatorial optimization part due to the constraint that $\mathbf{B}_0$ is acyclic. A conceptually simple way of reinforcing this constraint would be to go through all possible orderings of the observed variables, and for each of them, maximize the likelihood as a function of the $\mathbf{B}_\tau$ so that $\mathbf{B}_0$ is constrained to be lower triangular. This is obviously computationally very expensive since the number of ordering is equal to $n!$ where n is the number of variables. Only for a very small n can this be computationally feasible. (Another difficulty is that the likelihood contains a semiparametric part because we do not specify a parametric model of the non-Gaussian distributions, but this problem has already been extensively treated in the theory of ICA, and has been found not to be very serious in practice, see Hyvärinen et al., 2001.)

To avoid the computational problems with likelihood, we propose a computationally simpler two-stage method for estimating our model. The method combines classic least-squares estimation of an autoregressive (AR) model with LiNGAM estimation.

5.1.1 DEFINITION

The basic idea is that the $\mathbf{M}_\tau$ in (8) can be consistently, and computationally efficiently, estimated by classic least-squares methods. Then, since the model is essentially a LiNGAM model for the residuals of the predictions by the $\mathbf{M}_\tau$, we simply use our previously developed estimation methods for LiNGAM to estimate the rest of the parameters. These methods (Shimizu et al., 2006) seem to tackle the combinatorial optimization problem in a satisfactory way. The ensuing method will be justified in more detail below; it is defined as follows:

1. Estimate a classic autoregressive model for the data

$$\mathbf{x}(t) = \sum_{\tau=1}^{k} \mathbf{M}_\tau \mathbf{x}(t-\tau) + \mathbf{n}(t) \tag{10}$$

   using any conventional implementation of a least-squares method. Note that here $\tau > 0$, so it is really a classic AR model. Denote the least-squares estimates of the autoregressive matrices by $\hat{\mathbf{M}}_\tau$.

2. Compute the residuals, that is, estimates of $\mathbf{n}(t)$

$$\hat{\mathbf{n}}(t) = \mathbf{x}(t) - \sum_{\tau=1}^{k} \hat{\mathbf{M}}_\tau \mathbf{x}(t-\tau).$$

3. Perform the LiNGAM analysis (Shimizu et al., 2006) on the residuals. This gives the estimate of the matrix $\mathbf{B}_0$ as the solution of the instantaneous causal model

$$\hat{\mathbf{n}}(t) = \mathbf{B}_0 \hat{\mathbf{n}}(t) + \mathbf{e}(t).$$

4. Finally, compute the estimates of the causal effect matrices $\mathbf{B}_\tau$ for $\tau > 0$ as

$$\hat{\mathbf{B}}_\tau = (\mathbf{I} - \hat{\mathbf{B}}_0)\hat{\mathbf{M}}_\tau \text{ for } \tau > 0. \tag{11}$$

5.1.2 CONSISTENCY PROOF

We now prove that this provides a consistent estimator of $\mathbf{B}_\tau$. First, the basic model definition in (3) can be manipulated to yield

$$(\mathbf{I} - \mathbf{B}_0)\mathbf{x}(t) = \sum_{\tau=1}^{k} \mathbf{B}_\tau \mathbf{x}(t-\tau) + \mathbf{e}(t)$$

and thus

$$\mathbf{x}(t) = \sum_{\tau=1}^{k} (\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{B}_\tau \mathbf{x}(t-\tau) + (\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{e}(t). \tag{12}$$

Now, a well-known result is that least-squares estimation of an AR model as in (10) is consistent even if the innovation vector $\mathbf{n}(t)$ does not have independent or even uncorrelated elements (for fixed t), and even if it is heteroscedastic and non-Gaussian. Thus, comparing (12) with (10), in the limit we can equate the autoregressive matrices, which gives $(\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{B}_\tau = \mathbf{M}_\tau$ for $\tau \geq 1$, and thus (11) is justified. (In fact, we anticipated (11) in the notation used in the likelihood in (9).)

Second, comparison of (12) with (10) shows that the residuals $\hat{\mathbf{n}}(t)$ are, asymptotically, of the form $(\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{e}(t)$. This means

$$\hat{\mathbf{n}}(t) = (\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{e}(t) \Leftrightarrow (\mathbf{I} - \mathbf{B}_0)\hat{\mathbf{n}}(t) = \mathbf{e}(t) \Leftrightarrow \hat{\mathbf{n}}(t) = \mathbf{B}_0\hat{\mathbf{n}}(t) + \mathbf{e}(t)$$

which is the LiNGAM model for $\hat{\mathbf{n}}(t)$. This shows that $\mathbf{B}_0$ is obtained as the LiNGAM analysis of the residuals, and the consistency of our estimator of $\mathbf{B}_0$ follows from the consistency of LiNGAM estimation (Shimizu et al., 2006). Thus, our method is consistent for all the $\mathbf{B}_\tau$. This obviously proves, by construction, the identifiability of the model as well.

5.1.3 INTERPRETATION RELATED TO ICA OF RESIDUALS

An interesting viewpoint of the two-stage estimation method is analysis of the dependencies of the innovations after estimating a classic AR model. Suppose we just estimate an AR model as in (1), and interpret the coefficients as causal effects. Such an interpretation more or less presupposes that the innovations $e_i(t)$ are independent of each other, because otherwise there is some structure in the model which has not been modelled by the AR model. If the innovations are not independent, the causal interpretation may not be justified. Thus, it seems necessary to further analyze the dependencies in the innovations in cases where they are strongly dependent.

Analysis of the dependency structure in the residuals (which are, by definition, estimates of innovations) is precisely what leads to the two-stage estimation method. As a first approach, one could consider application of something like principal component analysis or independent component analysis on the residuals. The problem with such an approach is that the interpretation of the obtained results in the framework of causal analysis would be quite difficult. Our solution is to fit a causal model like LiNGAM to the residuals, which leads to a straightforward causal interpretation of the analysis of residuals which is logically consistent with the AR model.

5.2 Method Based on Multichannel Blind Deconvolution

While the two-stage method proposed above is computationally very efficient, it is far from being statistically optimal. The estimation of the autoregressive part takes in no way non-Gaussianity into account and is thus likely to be suboptimal. However, it is useful because it provides a good initial guess for any further iterative method.

Thus, to improve the results of the two-stage method, we further propose an estimation method based on the similarity of our model with convolutive versions of ICA which are also called multichannel blind deconvolution (MBD). Estimation of the model Eq. (3) is, in fact, closely related to the multichannel blind deconvolution problem with causal finite impulse response (FIR) filters (Cichocki and Amari, 2002; Hyvärinen et al., 2001). MBD, as a direct extension of ICA, assumes that the observed signals are convolutive mixtures of some spatially and independently and identically distributed (i.i.d.) sources.

Using MBD methods is justified here due to the possibility or transforming an autoregressive model into a moving-average model: In Eq. (3), the observed variables $x_i(t)$ can be considered as convolutive mixtures of the disturbances $e_i(t)$. Thus, we can find estimates of $\mathbf{B}_\tau$, as well as $e_i(t)$, in Eq. (3), by using MBD methods to estimate the filter matrices $\mathbf{W}_\tau$

$$\hat{\mathbf{e}}(t) = \sum_{\tau=0}^{k} \mathbf{W}_\tau \mathbf{x}(t-\tau). \tag{13}$$

Comparing (13) with (3), we can see that the $\mathbf{B}_\tau$ can then be recovered from the estimated $\mathbf{W}_\tau$; details are given below.

The basic statistical principle to estimate the MBD model is that the disturbances $e_i(t)$ should be mutually independent for different i and different t. Under the assumption that at most one of the sources is Gaussian, by making the estimated sources spatially and temporally independent, MBD can recover the mixing system (here corresponding to $e_i(t)$ and $\mathbf{B}_\tau$) up to some scaling, permutation, and time shift indeterminacies (Liu and Luo, 1998). This implies that our SVAR model is identifiable by MBD if at most one of the disturbances e_i is Gaussian.

There exist several well-developed algorithms for MBD. For example, one may adopt the one based on natural gradient (Cichocki and Amari, 2002). By extending the LiNGAM analysis procedure (Shimizu et al., 2006), we can find the estimate of $\mathbf{B}_\tau$ in the following three steps, based on the MBD estimates of $\mathbf{W}_\tau$.

1. Find the permutation of rows of $\mathbf{W}_0$ which yields a matrix $\widetilde{\mathbf{W}}_0$ with only significantly nonzero entries on the main diagonal. The permutation can be found using similar methods (e.g., the Hungarian algorithm) as in LiNGAM (Shimizu et al., 2006). Note that here we also need to apply the same permutations to rows of $\mathbf{W}_\tau$ ($\tau > 0$) to produce $\widetilde{\mathbf{W}}_\tau$.

2. Divide each row of $\widetilde{\mathbf{W}}_0$ and $\widetilde{\mathbf{W}}_\tau$ ($\tau > 0$) by the corresponding diagonal entry in $\widetilde{\mathbf{W}}_0$. This gives $\widetilde{\mathbf{W}}'_0$ and $\widetilde{\mathbf{W}}'_\tau$, respectively. The final estimates of $\mathbf{B}_0$ and $\mathbf{B}_\tau$ ($\tau > 0$) can then be computed as $\widehat{\mathbf{B}}_0 = \mathbf{I} - \widetilde{\mathbf{W}}'_0$ and $\widehat{\mathbf{B}}_\tau = -\widetilde{\mathbf{W}}'_\tau$, respectively.

3. To obtain the causal order in the instantaneous effects, find the permutation matrix $\mathbf{P}$ (applied equally to both rows and columns) of $\widehat{\mathbf{B}}_0$ which makes $\widetilde{\mathbf{B}}_0 = \mathbf{P}\widehat{\mathbf{B}}_0\mathbf{P}^T$ as close as possible to strictly lower triangular.

5.3 Sparsification of the Causal Connections

For the purposes of interpretability and generalizability, it is often useful to sparsify the causal connections, that is, to set insignificant entries of $\hat{\mathbf{B}}_\tau$ to zero. Analogously to the development of ICA with sparse connections (Zhang et al., 2009), we can incorporate an adaptive L_1 penalty into the likelihood of the MBD method to achieve fast model selection which performs such sparsification. We use a penalty-based approach because traditional model selection based on information criteria involves a combinatorial optimization problem whose complexity increases exponentially in the dimensionality of the parameter space. In the MBD problem, this is often not computationally feasible.

Thus, to make $\mathbf{W}_\tau$ in Eq. (13) as sparse as possible, we maximize the penalized likelihood defined as

$$pl(\{\mathbf{W}_\tau\}) = \log L(\{\mathbf{W}_\tau\}) - \lambda \sum_{i,j,\tau} |w_{i,j,\tau}|/|\hat{w}_{i,j,\tau}|, \tag{14}$$

where $L(\{\mathbf{W}_\tau\})$ is the likelihood, $w_{i,j,\tau}$ the (i,j)th entry of $\mathbf{W}_\tau$, and $\hat{w}_{i,j,\tau}$ a consistent estimate of $w_{i,j,\tau}$, such as the maximum likelihood estimate. The parameter λ is the general weight of the penalty.

The idea here is that we first compute an initial estimate of the $w_{i,j,\tau}$ by a conventional method (such as maximum likelihood) and then use those estimates to compute a parameter-wise weighting in the L_1 penalty. The end result is that those $w_{i,j,\tau}$ for which the initial estimates $\hat{w}_{i,j,\tau}$ were small are heavily penalized, and likely to be zero in the final estimate obtained by maximization of pl.

This penalized likelihood is a special case of adaptive Lasso and therefore has the same consistency in variable selection (Zou, 2006). In fact, it can also be used for selecting the order k of the autoregressive model. In particular, to achieve model selection similar to the Bayesian Information Criterion (BIC), one can set $\lambda = \log T$, where T is the sample size (Zhang et al., 2009).

It may be also useful to penalize groups of parameters. In particular, to see if the historical values of $x_i(t)$ causes $x_j(t)$ $(i \neq j)$, one needs to examine the combined effect of the group of parameters $[\hat{\mathbf{B}}_\tau]_{i,j}, \tau = 1, ..., p$, and therefore it makes sense to apply penalization on the parameter group. Combining the above approach with group Lasso (Bach, 2008) leads to the following penalized likelihood:[3]

$$pl(\{\mathbf{W}_\tau\}) = \log L(\{\mathbf{W}_\tau\}) - \lambda \sum_{i,j,\tau} |w_{i,j,0}|/|\hat{w}_{i,j,0}| - k\lambda \sum_{i,j} \Big(\sum_{\tau=1}^{k} w_{i,j,\tau}^2 \Big)^{1/2} \Big/ \Big(\sum_{\tau=1}^{k} \hat{w}_{i,j,\tau}^2 \Big)^{1/2},$$

where the last term has the coefficient k because the parameter group $w_{i,j,\tau}, \tau = 1, ..., k$ has k parameters.

5.4 Simulations

To investigate the performance of the proposed estimation methods, we conducted a series of simulations. We set the number of lags $k = 1$ and the dimensionality $n = 5$. We randomly constructed the strictly lower-triangular matrix $\mathbf{B}_0$ and matrix $\mathbf{B}_1$. To make the causal effects sparse, we set about 60% of the entries in the matrix $\mathbf{B}_1$ and the lower-triangular part of $\mathbf{B}_0$ to zero, while the magnitude of the others is uniformly distributed between 0.05 and 0.5 and the sign is random. Super-Gaussian

3. Here we treat the instantaneous effects separately. If one would like to see if the total influence from $x_i(t-\tau), \tau = 0, 1, ..., p$ to $x_j(t)$ is significant, all parameters $w_{i,j,\tau}, \tau = 0, 1, ..., p$ should be treated as a group.

disturbances $e_i(t)$ were generated by passing standardized i.i.d. Gaussian variables through a power nonlinearity with exponent between 1.5 and 2.0 while keeping the original sign. The observations $\mathbf{x}(t)$ were then generated according to the model in Eq. (3). Various sample sizes (T = 100, 300, and 1000) were tested. We compared the performance of the two-stage method (Section 5.1), the method by MBD (Section 5.2) and the MBD-based method with the sparsity penalty (Section 5.3). In the last method, we set the penalization parameter in Eq. (14) as $\lambda = \log T$ to make its results consistent with those obtained by BIC. The densities of the independent components were adaptively estimated using the method in Pham and Garrat (1997). In each case, we repeated the experiments for 5 replications.

Fig. 1 shows the scatter plots of the estimated parameters (including the strictly lower triangular part of $\mathbf{B_0}$ and all entries of $\mathbf{B}_1$) versus the true ones. Different subplots correspond to different sample sizes or different methods. The mean square error (MSE) of the estimated parameters is also given in each subplot. One can see that as the sample sizes increases, all methods give better results. For each sample size, the method based on MBD is always better than the two-stage method, showing that the estimate by the MBD-based method is more efficient. Furthermore, due to the prior knowledge that many parameters are zero, the MBD-based method with the sparsity penalty performed best.

6. Assessment of the Significance of Causality

In practice, we also need to assess the significance of the estimated causal relations. While the sparsification method in Section 5.3 is related to this goal, here we propose a more principled approach for testing the significance of the causal influences.

For the instantaneous effects $x_i(t) \to x_j(t)$ $(i \neq j)$, the significance of causality is obtained by assessing if the entries of $\hat{\mathbf{B}}_0$ are statistically significantly different from zero. For the lagged effects $x_i(t-\tau) \to x_j(t)$ $(i \neq j, \tau > 0)$, however, one is often not interested in the significance of any single coefficient in $\hat{\mathbf{B}}_\tau$: More frequently one aims to find out if the total effect from $x_i(t-\tau)$ to $x_j(t)$ is significant.

We propose two simple statistics. One is a measure of instantaneous variance contributed by $x_i(t)$ to $x_j(t)$: $S_0(i \leftarrow j) = [\mathbf{B}_0]_{ij}^2 \cdot \mathrm{var}(x_i(t))/\mathrm{var}(x_j(t))$. If all time series have the same variance, it is simplified to $S_0(i \leftarrow j) = [\mathbf{B}_0]_{ij}^2$. The other measures how strong the total lagged causal influence from $x_i(t)$ to $x_j(t)$ is; it is a measure of contributed variance from $x_i(t-\tau), \tau > 0$ to $x_j(t)$: $S_{lag}(i \leftarrow j) = \mathrm{var}(\sum_{\tau>0}[\mathbf{B}_\tau]_{ij} x_j(t-\tau))/\mathrm{var}(x_j(t))$. If all series $x_i(t)$ have the same variance and are approximately temporally uncorrelated, the above statistic can be approximated by $\sum_{\tau>0}[\mathbf{B}_\tau]_{ij}^2$. (Note that these quantities are not exactly proportions of variance explained because the explaining variables are not necessarily uncorrelated.)

The asymptotic distributions of these statistics under the null hypothesis (with no causal effects) are very difficult to derive, and they may also behave poorly in the finite sample case. Therefore, like in Diks and DeGoede (2001) and Theiler et al. (1992), we use bootstrapping with surrogate data to find the empirical distributions of each statistic under the null hypothesis. To generate the surrogate data under the null hypothesis, in each bootstrapping replication we "scramble" the original series $x_i(t)$, that is, each time series is randomly permuted in temporal order. We then calculate $\hat{S}^*$, the estimate of the statistic S (which may be $S_0(i \leftarrow j)$ or $S_{lag}(i \leftarrow j)$) for the surrogate data. Next, the α-level bootstrapping critical value $c_{t\alpha}^*$ is found as the α-th quantile of the bootstrapping distribution

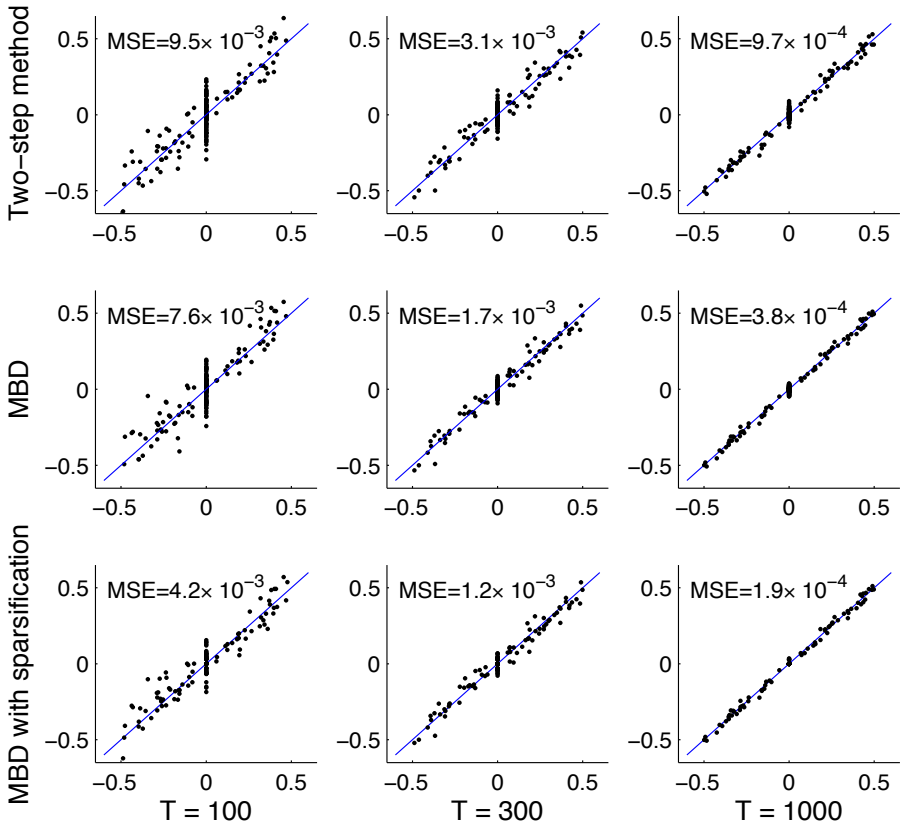


Figure 1: Scatter plots of the estimated coefficients (y axis) versus the true ones (x axis) for different sample sizes and different methods.

of $\hat{S}^*$. Finally, we reject the null hypothesis if $\hat{S} > c^*_{t\alpha}$, where $\hat{S}$ is the estimate of S for the original data.

7. Remarks on the Interpretation of the Parameters

In this section, we discuss how the autoregressive parameters are changed by taking into account the instantaneous effects, and how our model can be interpreted in the framework of Granger causality.

7.1 Interaction Between Instantaneous and Lagged Effects

Equation (11) shows the interesting fact already mentioned in the Introduction: Consistent estimates of the $\mathbf{B}_\tau$ are not obtained by a simple AR model fit even for $\tau > 0$. Taking instantaneous effects into account changes the estimation procedure for all the autoregressive matrices, if we want consistent estimators as we usually do. Of course, this is only the case if there are instantaneous effects, that is, $\mathbf{B}_0 \neq 0$; otherwise, the estimates are not changed.

While this phenomenon is, in principle, well-known in econometric literature (Swanson and Granger, 1997; Demiralp and Hoover, 2003; Moneta and Spirtes, 2006), Eq. (11) is seldom applied because estimation methods for $\mathbf{B}_0$ have not been well developed. To our knowledge, no estimation method for $\mathbf{B}_0$ has been proposed which is consistent for the whole matrix without strong prior assumptions on $\mathbf{B}_0$.

Next we present some theoretical examples of how the instantaneous and lagged effects interact based on the formula in (11).

7.1.1 EXAMPLE 1: AN INSTANTANEOUS EFFECT MAY SEEM TO BE LAGGED

Consider first the case where the instantaneous and lagged matrices are as follows:

$$\mathbf{B}_0 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad \mathbf{B}_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.9 \end{pmatrix}.$$

That is, there is an instantaneous effect $x_2 \to x_1$, and no lagged effects (other than the purely autoregressive $x_i(t-1) \to x_i(t)$). Now, if an AR(1) model is estimated for data coming from this model, without taking the instantaneous effects into account, we get the autoregressive matrix

$$\mathbf{M}_1 = (\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{B}_1 = \begin{pmatrix} 0.9 & 0.9 \\ 0 & 0.9 \end{pmatrix}.$$

Thus, the effect $x_2 \to x_1$ seems to be lagged although it is, actually, instantaneous.

7.1.2 EXAMPLE 2: SPURIOUS EFFECTS APPEAR

Consider three variables with the instantaneous effects $x_1 \to x_2$ and $x_2 \to x_3$, and no lagged effects other than $x_i(t-1) \to x_i(t)$, as given by

$$\mathbf{B}_0 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \qquad \mathbf{B}_1 = \begin{pmatrix} 0.9 & 0 & 0 \\ 0 & 0.9 & 0 \\ 0 & 0 & 0.9 \end{pmatrix}.$$

If we estimate an AR(1) model for the data coming from this model, we obtain

$$\mathbf{M}_1 = (\mathbf{I} - \mathbf{B}_0)^{-1}\mathbf{B}_1 = \begin{pmatrix} 0.9 & 0 & 0 \\ 0.9 & 0.9 & 0 \\ 0.9 & 0.9 & 0.9 \end{pmatrix}.$$

This means that the estimation of the simple autoregressive model leads to the inference of a direct lagged effect $x_1 \to x_3$, although no such direct effect exists in the model generating the data, for any time lag.

A more reassuring result is the following: if the data follows the same causal ordering for all time lags, that ordering is not contradicted by the neglect of instantaneous effect. A rigorous definition of this property is the following.

Theorem 1 *Assume that there is an ordering $i(j), j = 1 \ldots n$ of the variables such that no effect goes backward, that is,*

$$\mathbf{B}_\tau(i(j-\delta), i(j)) = 0 \text{ for } \delta > 0, \tau \geq 0, 1 \leq j \leq n. \tag{15}$$

(In the purely instantaneous case, existence of such an ordering is equivalent to acyclicity of the effects as noted in Section 2.2.) Then, the same property applies to the $\mathbf{M}_\tau, \tau \geq 1$ *as well. Conversely, if there is an ordering such that (15) applies to* $\mathbf{M}_\tau, \tau \geq 1$ *and* $\mathbf{B}_0$*, then it applies to* $\mathbf{B}_\tau, \tau \geq 1$ *as well.*

Proof: When the variables are ordered in this way (assuming such an order exists), all the matrices $\mathbf{B}_\tau$ are lower-triangular. The same applies to $\mathbf{I} - \mathbf{B}_0$. Now, the product of two lower-triangular matrices is lower-triangular; in particular the $\mathbf{M}_\tau$ are also lower-triangular according to (11), which proves the first part of the theorem. The converse part follows from solving for $\mathbf{B}_\tau$ in (11) and the fact that the inverse of a lower-triangular matrix is lower-triangular.

What this theorem means is that if the variables really follow a single "causal ordering" for all time lags, that ordering is preserved even if instantaneous effects are neglected and a classic AR model is estimated for the data. Thus, there is some limit to how (11) can change the causal interpretation of the results.

7.2 Generalizations of Granger Causality

The classic interpretation of causality in instantaneous SEMs would be that x_i causes x_j if the (j,i)-th coefficient in $\mathbf{B}_0$ is non-zero. On the other hand, in the time series context, the concept of Granger causality (Granger, 1969) formalizes causality as the ability to reduce prediction error. A simple operational definition of Granger causality can be based on the autoregressive coefficients $\mathbf{M}_\tau$: If at least one of the coefficients from $x_i(t-\tau), \tau \geq 1$ to $x_j(t)$ is (significantly) non-zero, then x_i Granger-causes x_j. This is because then the variable x_i reduces the prediction error in x_j in the mean-square sense if it is included in the set of predictors, which is the very definition of Granger causality.

In light of the results in this paper, we can generalize the concept of Granger causality in two ways. First we can combine the two aspects of instantaneous and lagged effects. In fact, such a concept of instantaneous causality was already alluded to by Granger (1969), but presumably due to lack of proper estimation methods, that paper as well as most future developments considered mainly non-instantaneous causality. The second generalization is to measure prediction error by the information-theoretic definition of Section 4.3, essentially using entropy instead of mean squared error. These two generalization are independent of each other in the sense that we can use any one of them, omitting the other.

Both of these extensions are implicit in estimation of our model. Thus, we define that *a variable* x_i *causes* x_j *if at least one of the coefficients* $\mathbf{B}_\tau(j,i)$*, giving the effect from* $x_i(t-\tau)$ *to* $x_j(t)$*, is (significantly) non-zero for* $\tau \geq 0$. The condition for τ is different from Granger causality since the value $\tau = 0$, corresponding to instantaneous effects, is included. Moreover, since estimation of the instantaneous effects changes the estimates of the lagged ones, the lagged effects used in our definition are different from those usually used with Granger causality. Using entropy instead of mean-squared error is implicit in this definition because non-Gaussianity is used in the estimation of the model. In general, entropy minimization is closely related to ICA estimation (Hyvärinen, 1999) as well as the estimation of the present model as was discussed in Section 4.3. Notice that we assume here, as in the general theory of Granger causality, that there are no unobserved confounders.

8. Real Data Experiments

We applied our model together with the estimation and testing method on both financial data and magnetoencephalography (MEG) data. In the former application, we used the sparsity penalty to select significant effects, while in the latter one, bootstrapping was used.

8.1 Application in Finance

First, we use the model in Eq. (3) to find the causal relations among several world stock indices. The chosen indices are Dow Jones Industrial Average (DJI) in USA, Nikkei 225 (N225) in Japan, Hang Seng Index (HSI) in Hong Kong, and the Shanghai Stock Exchange Composite Index (SSEC) in China. We used the daily dividend/split adjusted closing prices from 4th Dec 2001 to 11th Jul 2006, obtained from the Yahoo finance database. For the few days when the price is not available, we use simple linear interpolation to estimate the price. Denoting the closing price of the ith index on day t by $P_i(t)$, the corresponding return is calculated by $x_i(t) = \frac{P_i(t)-P_i(t-1)}{P_i(t-1)}$. The data for analysis are $\mathbf{x}(t) = [x_1(t), ..., x_4(t)]^T$, with $T = 1200$ observations.

We applied the MBD-based method with the sparsity penalty to $\mathbf{x}(t)$. The kurtoses of the estimated disturbances $\hat{e}_i$ are 3.9, 8.6, 4.1, and 7.6, respectively, implying that the disturbances are non-Gaussian. We found that more than half of the coefficients in the estimated $\mathbf{W}_0$ and $\mathbf{W}_1$ are exactly zero due to sparsity penalty. $\widehat{\mathbf{B}}_0$ and $\widehat{\mathbf{B}}_1$ were constructed based on $\mathbf{W}_0$ and $\mathbf{W}_1$, using the procedure given in Section 5.2. It was found that $\widehat{\mathbf{B}}_0$ can be permuted to a strictly lower-triangular matrix, meaning that the instantaneous effects follow a linear acyclic causal model. Finally, based on $\widehat{\mathbf{B}}_0$ and $\widehat{\mathbf{B}}_1$, one can plot the causal diagram, which is shown in Fig. 2.

Fig. 2 reveals some interesting findings. First, DJI_{t-1} has significant impacts on N225_t and HSI_t, which is a well-known fact in the stock market. Second, the causal relations $\text{DJI}_{t-1} \to \text{N225}_t \to \text{DJI}_t$ and $\text{DJI}_{t-1} \to \text{HSI}_t \to \text{DJI}_t$ are consistent with the time difference between Asia and USA. That is, the causal effects from N225_t and HSI_t to DJI_t, although seeming to be instantaneous, may actually be mainly caused by the time difference. Third, unlike SSEC, HSI is very sensitive to others; it is even strongly influenced by N225, another Asian index. Fourth, it may be surprising that there is a significant negative effect from DJI_{t-1} to DJI_t; however, it is not necessary for DJI_t to have significant negative autocorrelations, due to the positive effect from DJI_{t-1} to DJI_t going through N225_t and HSI_t.

8.2 Application on MEG Data

Second, we applied the proposed model on the magnitudes of brain sources obtained from magnetoencephalographic (MEG) signals to analyze their causal relationships. The raw recordings consisted of the 204 gradiometer channels measured by the Vectorview helmet-shaped neuromagnetometer (Neuromag Ltd., Helsinki, Finland) in a magnetically shielded room at the Brain Research Unit of the Low Temperature Laboratory of the Aalto University School of Science and Technology. They were obtained from a healthy volunteer and lasted about 12 minutes. The data was downsampled to 75 Hz.

To begin with, we separated sources underlying the recorded MEG data using a recently proposed blind source separation method, Fourier-ICA (Hyvärinen et al., 2010). We manually selected 17 sources which are expected to correspond to brain activity, rejecting clear artifacts based on the Fourier spectra and topographic distributions of the sources.

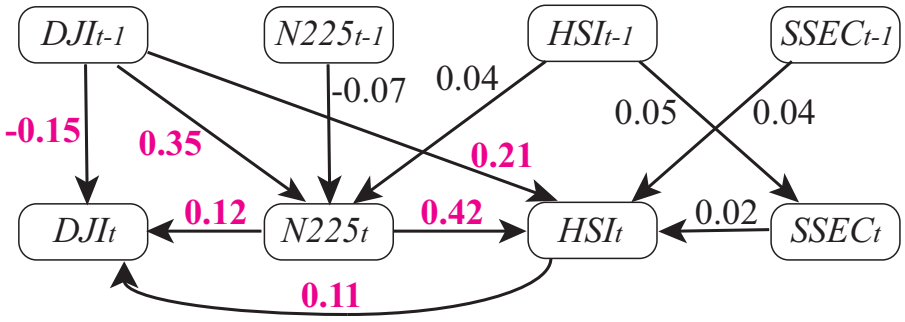


Figure 2: Results of application of our model to daily returns of the stock indices DJI, N225, HSI, and SSEC, with $k = 1$ lag. Large coefficients (greater than 0.1) are shown in bold and red.

Next, we fitted an ordinary vector autoregressive model with 10 lags on the estimated sources, finding the corresponding innovation series which we denote by $y_i(t), i = 1, ..., 17$. Our goal was to analyze if there are some influences between the magnitudes of these innovations. We prefer to analyze the innovations because the innovations are approximately white both temporally and spatially, and thus we can analyze the magnitudes with no contamination by linear (auto)correlations of the source signals. The autoregressive model order 10 was chosen because it was the smallest order that gave approximately white innovations.

We then fitted the SVAR model on the logarithmically transformed magnitudes $x_i(t) = \log(0.2 + |y_i(t)|), i = 1, ..., 17$. We determined the order k of our SVAR model by minimizing the AIC criterion (Akaike, 1973), which is the negative log-likelihood of the MBD model plus a term measuring the complexity of the model. The log-likelihood involves the densities of the MBD outputs $\hat{e}_i(t)$, which were modelled by a mixture of three Gaussians. From the candidate orders between 0 and 20, we found that $k = 2$ gave the minimum AIC.

After finding the estimate of the coefficients $\hat{\mathbf{B}}_\tau, \tau = 0, 1, 2$ with the MBD-based approach, one can easily calculate the estimates of the statistics $S_0(i \leftarrow j)$ and $S_{lag}(i \leftarrow j)$. The bootstrapping approach given in Section 6 was used to evaluate if these estimated statistics are significant. Here we need to test multiple hypotheses simultaneously; to reduce the type I error, we adopted the Bonferroni correction (Shaffer, 1995) for multiple testing correction. We used the significance level 5%. For both the instantaneous and lagged effects, one needs to perform $17 \times 16 = 272$ tests; therefore, the significance level for each individual test is then $0.05/272 \approx 2 \times 10^{-4}$. We used 10^4 replications for the bootstrapping.

For illustration, we give the empirical distribution of the statistics $S_0(7 \leftarrow 14)$ and $S_{lag}(7 \leftarrow 14)$, as well as their estimated values for the original series $x_i(t)$, in Fig. 3. Clearly $\hat{S}_0(7 \leftarrow 14)$ is significant, while $\hat{S}_{lag}(7 \leftarrow 14)$ is not.

Fig. 4 shows the resulting diagram of causal analysis with instantaneous effects between the magnitudes of the selected MEG sources, with the influences significant at 5% level (corrected for multiple testing). What we see is that the connections tend to be strong between sources which are close to each other. For example, the occipitoparietal sources such as #1, #2, #3, #8, and #11 have strong interconnections. Some perirolandic sources such as #5, #7, #10, and #14 are also interconnected. Sources #4 and #16 seems to mediate between these two groups.

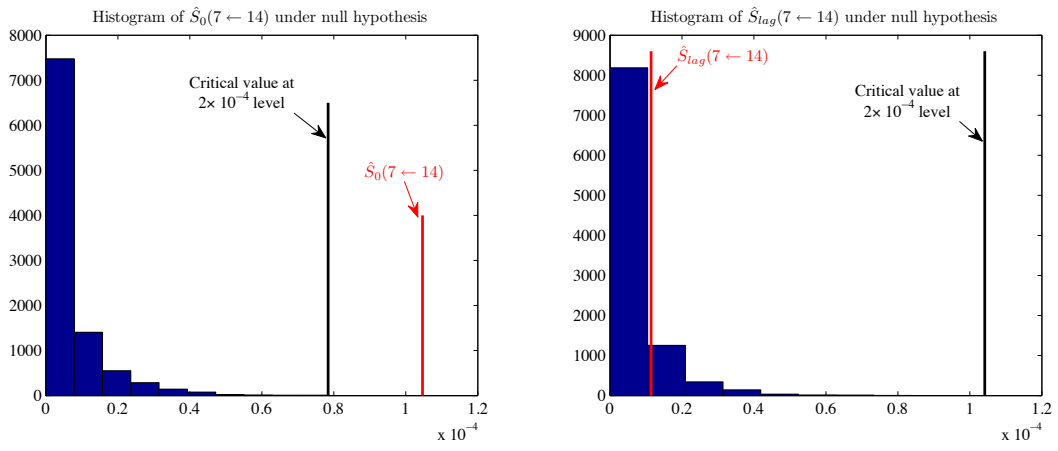


Figure 3: Illustration of the empirical distribution of the statistics under the null hypothesis obtained by bootstrapping. (a) For the statistic $S_0(7 \leftarrow 14)$. (b) For $S_{lag}(7 \leftarrow 14)$.

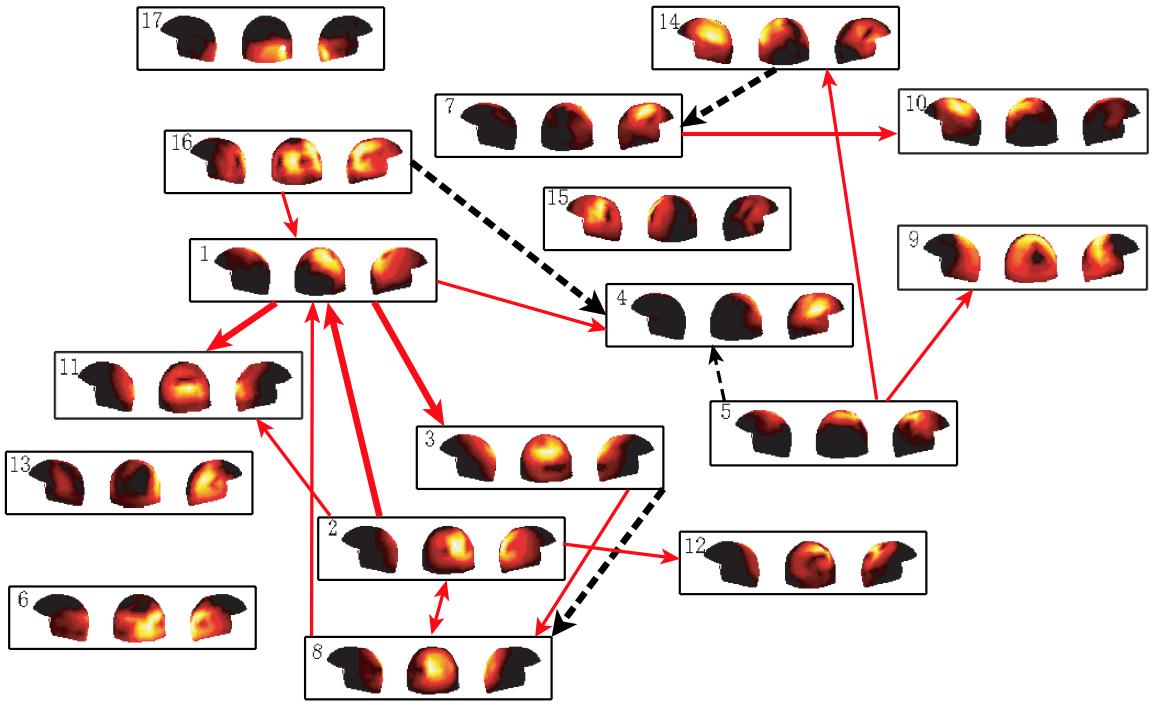


Figure 4: Results of application of our model on the log-magnitudes of the MEG sources (significant at 5% level, corrected for multiple testing). Black dashed line: instantaneous effect. Red solid line: lagged effect. The thickness of the lines indicates the strength of the influences.

9. Extensions of the Model

We have here assumed that $\mathbf{B}_0$ is acyclic, as is typical in causal analysis. However, this assumption is only made because we do not know very well how to estimate a linear non-Gaussian Bayesian network (or SEM) in the cyclic case. If we have a method which can estimate cyclic models, we do not need the assumption of acyclicity in our combined model either; see Lacerda et al. (2008) for one proposal. We could just use such a new method in our two-stage method instead of LiNGAM, and nothing else would be changed. However, development of methods for estimating cyclic models is orthogonal to the main contribution of our paper in the sense that we can use any such new method to estimate the instantaneous model in our framework.

In formulating the likelihood we had to assume that the $\mathbf{e}(t)$ are independent and identically distributed for different time points. However, in our two-stage estimation method, no such assumption was needed to guarantee consistency. In particular, the $\mathbf{e}(t)$ can be heteroscedastic, as long as $\mathbf{e}(t)$ and $\mathbf{e}(t')$ are uncorrelated for $t \neq t'$. In such a case, it might also be advantageous to change the LiNGAM estimation method so that the ICA part is replaced by methods estimating (4) explicitly based on temporal heteroscedasticity (Matsuoka et al., 1995; Hyvärinen, 2001; Pham and Cardoso, 2001); this is quite straightforward and necessitates no further changes in the method.

An interesting class of methods which is related to ours has been recently proposed by Gómez-Herrero et al. (2008). The idea is to combine blind source separation with a linear autoregressive model of the latent sources. The estimation of such a model can be accomplished by methods which are quite similar to our estimation methods, see also Haufe et al. (2009). However, the interpretation of the model is very different since, first, Gómez-Herrero et al. (2008) separate linear sources and analyze their (causal) connections whereas we analyze connections between the observed variables, and second, we estimate instantaneous causal influences whereas Gómez-Herrero et al. (2008) only estimate lagged ones.

10. Conclusion

We showed how non-Gaussianity enables estimation of a causal discovery model in which the linear effects can be either instantaneous or time-lagged. Like in the purely instantaneous case (Shimizu et al., 2006), non-Gaussianity makes the model identifiable without explicit prior assumptions on existence or non-existence of given causal effects. The theoretical developments are closely related to independent component analysis. The classic assumption of acyclicity was made, although it may not be necessary. From the practical viewpoint, an important implication is that considering instantaneous effects changes the coefficient of the time-lagged effects as well. We proposed methods for computationally efficient estimation of the model, as well as for sparsification and testing of the model coefficients.

Acknowledgments

We are grateful to Pavan Ramkumar, Lauri Parkkonen, and Riitta Hari for the MEG data. This work was partly funded by the Academy of Finland Centre-of-Excellence in Algorithmic Data Analysis.

References

H. Akaike. Information theory and an extension of the maximum likelihood principle. *Proc. 2nd International Symposium on Information Theory*, pages 267–281, 1973.

F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

E. M. L. Beale and C. L. Mallows. Scale mixing of symmetric distributions with zero means. *The Annals of Mathematical Statistics*, 30(4):1145–1151, 1959.

K. A. Bollen. *Structural Equations with Latent Variables*. John Wiley & Sons, 1989.

J.-F. Cardoso and B. Hvam Laheld. Equivariant adaptive source separation. *IEEE Trans. on Signal Processing*, 44(12):3017–3030, 1996.

A. Chen and P. J. Bickel. Efficient independent component analysis. *The Annals of Statistics*, 34 (6):2824–2855, 2006.

A. Cichocki and S.-I. Amari. *Adaptive Blind Signal and Image Processing: Learning Algorithms*. Wiley, 2002.

P. Comon. Independent component analysis—a new concept? *Signal Processing*, 36:287–314, 1994.

S. Demiralp and K. D. Hoover. Searching for the causal structure of a vector autoregression. *Oxford Bulletin of Economics and Statistics*, 65 (supplement):745–767, 2003.

C. Diks and J. DeGoede. A general nonparametric bootstrap test for granger causality. In H. Broer and B. Krauskopf nd G. Vegter, editors, *Global Analysis of Dynamical Systems*, pages 391–403 (Chapter 16). Taylor & Francis, London, 2001.

R. F. Engle, editor. *ARCH: Selected Readings*. Oxford University Press, 1995.

K. J. Friston, L. Harrison, and W. Penny. Dynamic causal modelling. *NeuroImage*, 19(4):1273–1302, 2003.

G. Gómez-Herrero, M. Atienza, K. Egiazarian, and J.L. Cantero. Measuring directional coupling between EEG sources. *NeuroImage*, 43:497–508, 2008.

C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37:424–438, 1969.

R. Hari and R. Salmelin. Human cortical oscillations: a neuromagnetic view through the skull. *Trends in Neuroscience*, 20:44–49, 1997.

S. Haufe, R. Tomioka, G. Nolte, K.-R. Müller, and M. Kawanabe. Modeling sparse connectivity between underlying brain sources for EEG/MEG. 2009. Arxiv preprint.

P. O. Hoyer, S. Shimizu, A. J. Kerminen, and M. Palviainen. Estimation of causal effects using linear non-Gaussian causal models with hidden variables. *International Journal of Approximate Reasoning*, 49:362–378, 2008.

A. Hyvärinen. Blind source separation by nonstationarity of variance: A cumulant-based approach. *IEEE Transactions on Neural Networks*, 12(6):1471–1474, 2001.

A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

A. Hyvärinen and E. Oja. Independent component analysis by general nonlinear Hebbian-like learning rules. *Signal Processing*, 64(3):301–313, 1998.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley Interscience, 2001.

A. Hyvärinen, S. Shimizu, and P. O. Hoyer. Causal modelling combining instantaneous and lagged effects: an identifiable model based on non-Gaussianity. In *Proc. Int. Conf. on Machine Learning (ICML2008)*, pages 424–431, Helsinki, Finland, 2008.

A. Hyvärinen, P. Ramkumar, L. Parkkonen, and R. Hari. Independent component analysis of short-time Fourier transforms for spontaneous EEG/MEG analysis. *NeuroImage*, 49(1):257–271, 2010.

J. Karvanen and V. Koivunen. Blind separation methods based on pearson system and its extensions. *Signal Processing*, 82(4):663–573, 2002.

J. Kim, W. Zhu, L. Chang, P. M. Bentler, and T. Ernst. Unified structural equation modeling approach for the analysis of multisubject, multivariate functional MRI data. *Human Brain Mapping*, 28:85–93, 2007.

G. Lacerda, P. Spirtes, J. Ramsey, and P. O. Hoyer. Discovering cyclic causal models by independent components analysis. In *Proc. 24th Conf. on Uncertainty in Artificial Intelligence (UAI2008)*, Helsinki, Finland, 2008.

R. W. Liu and H. Luo. Direct blind separation of independent non-Gaussian signals with dynamic channels. In *Proc. Fifth IEEE Workshop on Cellular Neural Networks and their Applications*, pages 34–38, London, England, April 1998.

K. Matsuoka, M. Ohya, and M. Kawamoto. A neural net for blind separation of nonstationary signals. *Neural Networks*, 8(3):411–419, 1995.

A. Moneta and P. Spirtes. Graphical models for the identification of causal structures in multivariate time series models. In *Proc. Joint Conference on Information Sciences*, Kaohsiung, Taiwan, 2006.

R. Opgen-Rhein and K. Strimmer. From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, 1(37), 2007.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

D.-T. Pham and J.-F. Cardoso. Blind separation of instantaneous mixtures of non stationary sources. *IEEE Trans. Signal Processing*, 49(9):1837–1848, 2001.

D.-T. Pham and P. Garrat. Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. *IEEE Trans. on Signal Processing*, 45(7):1712–1725, 1997.

A. Roebroeck, E. Formisano, and R. Goebel. Mapping directed influence over the brain using Granger causality and fMRI. *NeuroImage*, 25(1):230–242, 2005.

J. P. Shaffer. Multiple hypothesis testing. *Annual Review of Psychology*, 46:561–584, 1995.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *J. of Machine Learning Research*, 7:2003–2030, 2006.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer-Verlag, 1993.

N. R. Swanson and C. W. J. Granger. Impulse response functions based on a causal approach to residual orthogonalization in vector autoregression. *J. of the Americal Statistical Association*, 92: 357–367, 1997.

J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. Farmer. Testing for nonlinearity in time series: the method of surrogate data. *Physica D*, 58:77–94, 1992.

K. Zhang and A. Hyvärinen. Causality discovery with additive disturbances: An information-theoretical perspective. In *Proc. European Conference on Machine Learning (ECML2009)*, pages 570–585, 2009.

K. Zhang, H. Peng, L. Chan, and A. Hyvärinen. ICA with sparse connections: Revisited. In *Proc. Int. Conference on Independent Component Analysis and Blind Signal Separation (ICA2009)*, pages 195–202, Paraty, Brazil, 2009.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1417–1429, 2006.

Journal of Machine Learning Research 11 (2010) 1733-1736 Submitted 12/09; Revised 4/10; Published 5/10

FastInf: An Efficient Approximate Inference Library

Ariel Jaimovich ARIELJ@CS.HUJI.AC.IL
Ofer Meshi MESHI@CS.HUJI.AC.IL
School of Computer Science and Engineering
Hebrew University of Jerusalem
Jerusalem, Israel 91904

Ian McGraw IMCGRAW@MIT.EDU
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139 USA

Gal Elidan GALEL@HUJI.AC.IL
Department of Statistics
Hebrew University of Jerusalem
Jerusalem, Israel 91905

Editor: Cheng Soon Ong

Abstract

The FastInf C++ library is designed to perform memory and time efficient approximate inference in large-scale discrete undirected graphical models. The focus of the library is propagation based approximate inference methods, ranging from the basic loopy belief propagation algorithm to propagation based on convex free energies. Various message scheduling schemes that improve on the standard synchronous or asynchronous approaches are included. Also implemented are a clique tree based exact inference, Gibbs sampling, and the mean field algorithm. In addition to inference, FastInf provides parameter estimation capabilities as well as representation and learning of shared parameters. It offers a rich interface that facilitates extension of the basic classes to other inference and learning methods.

Keywords: graphical models, Markov random field, loopy belief propagation, approximate inference

1. Introduction

Probabilistic graphical models (Pearl, 1988) are a framework for representing a complex joint distribution over a set of n random variables $\mathcal{X} = \{X_1 \dots X_n\}$. A qualitative graph encodes probabilistic independencies between the variables and implies a decomposition of the joint distribution into a product of local terms:

$$P(\mathcal{X}) = \frac{1}{Z} \prod_i \psi_i(C_i),$$

where C_i are subsets of $\mathcal{X}$ defined by the cliques of the graph structure and $\psi_i(C_i)$ are the quantitative parameters (potential functions) that define the distribution. Computing marginal probabilities and likelihood in graphical models are critical tasks needed both for making predictions and to facilitate learning. Obtaining exact answers to these inference queries is often infeasible even for relatively

modest problems. Thus, there is a growing need for inference methods that are both efficient and can provide reasonable approximate computations. Despite few theoretical guarantees, the Loopy Belief Propagation (LBP, Pearl, 1988) algorithm has gained significant popularity in the last two decades due to impressive empirical success, and is now being used in a wide range of applications ranging from transmission decoding to image segmentation (Murphy and Weiss, 1999; McEliece et al., 1998; Shental et al., 2003). Recently there has been an explosion in practical and theoretical interest in propagation based inference methods, and a range of improvements to the convergence behavior and approximation quality of the basic algorithms have been suggested (Wainwright et al., 2003; Wiegerinck and Heskes, 2003; Elidan et al., 2006; Meshi et al., 2009).

We present the *FastInf* library for efficient approximate inference in large scale discrete probabilistic graphical models. While the library's focus is propagation based inference techniques, implementations of other popular inference algorithms such as mean field (Jordan et al., 1998) and Gibbs sampling are also included. To facilitate inference for a wide range of models, FastInf's representation is flexible allowing the encoding of standard Markov random fields as well as template-based probabilistic relational models (Friedman et al., 1999; Getoor et al., 2001), through the use of shared parameters. In addition, FastInf also supports learning capabilities by providing parameter estimation based on the Maximum-Likelihood (ML) principle, with standard regularization. Missing data is handled via the Expectation Maximization (EM) algorithm (Dempster et al., 1977).

FastInf has been used successfully in a number of challenging applications, ranging from inference in protein-protein networks with tens of thousands of variables and small cycles (Jaimovich et al., 2005), through protein design (Fromer and Yanover, 2008) to object localization in cluttered images (Elidan et al., 2006).

2. Features

The FastInf library was designed while focusing on generality and flexibility. Accordingly, a rich interface enables implementation of a wide range of probabilistic graphical models to which all inference and learning methods can be applied. A basic general-purpose propagation algorithm is at the base of all propagation variants and allows straightforward extensions.

A model is defined via a graph interface that requires the specification of a set of cliques $C_1 \ldots C_k$, and a corresponding set of tables that quantify the parametrization $\psi_i(C_i)$ for each joint assignment of the variables in the clique C_i. This general setting can be used to perform inference both for the directed Bayesian network representation and the undirected Markov one.

2.1 Inference Methods

FastInf includes implementations of the following inference methods:

- Exact inference by the Junction-Tree algorithm (Lauritzen and Spiegelhalter, 1988)
- Loopy Belief Propagation (Pearl, 1988)
- Generalized Belief Propagation (Yedidia et al., 2005)
- Tree Re-weighted Belief Propagation (Wainwright et al., 2005)
- Propagation based on convexification of the Bethe free energy (Meshi et al., 2009).
- Mean field (Jordan et al., 1998)
- Gibbs sampling (Geman and Geman, 1984)

By default, all methods are used with standard asynchronous message scheduling. We also implemented two alternative scheduling approaches that can lead to better convergence properties (Wainwright et al., 2002; Elidan et al., 2006). All methods can be applied to both sum and max product propagation schemes, with or without damping of messages.

2.2 Relational Representation

In many domains, a specific local interaction pattern can recur many times. To represent such domains, it is useful to allow multiple cliques to share the same parametrization. In this case a set of template table parametrizations $\psi_1, \ldots, \psi_T$ are used to parametrize all cliques using

$$P(\mathcal{X}) = \frac{1}{Z} \prod_t \prod_{i \in I(t)} \psi_t(C_i),$$

where $I(t)$ is the set of cliques that are mapped to the t'th potential. This template based representation allows the definition of large-scale models using a relatively small number of parameters.

2.3 Parameter Estimation

FastInf can also be used for learning the parameters of the model from evidence. This is done by using gradient-based methods with the Maximum-Likelihood (ML) objective. The library also handles partial evidence by applying the EM algorithm (Dempster et al., 1977). Moreover, FastInf supports L_1 and L_2 regularization that is added as a penalty term to the ML objective.

3. Documentation

For detailed instructions on how to install and use the library, examples for usage and documentation on the main classes of the library visit FastInf home page at: `http://compbio.cs.huji.ac.il/FastInf`.

Acknowledgments

We would like to acknowledge Menachem Fromer, Haidong Wang, John Duchi and Varun Ganapathi for evaluating the library and contributing implementations of various functions. This library was initiated in Nir Friedman's lab at the Hebrew University and developed in cooperation with Daphne Koller's lab at Stanford. AJ was supported by the Eshkol fellowship from the Israeli Ministry of Science. IM and GE were supported by the DARPA transfer learning program under contract FA8750-05-2-0249

References

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*, B 39:1–39, 1977.

G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *UAI* 2006.

G. Elidan, G. Heitz, and D. Koller. Learning object shape: From drawings to images. In *CVPR* 2006.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI* 1999.

M. Fromer and C. Yanover. A computational framework to empower probabilistic protein design. In *Bioinformatics*, pages 214–222, 2008.

S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 721–741, 1984.

L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure. In *ICML* 2001.

A. Jaimovich, G. Elidan, H. Margalit, and N. Friedman. Towards an integrated protein-protein interaction network. In *RECOMB*, 2005.

M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul. An introduction to variational approximations methods for graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer, Dordrecht, Netherlands, 1998.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, 1988.

R. McEliece, D. McKay, and J. Cheng. Turbo decoding as an instance of pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communication*, 16:140–152, 1998.

O. Meshi, A. Jaimovich, A. Globerzon, and N. Friedman. Convexifying the bethe free energy. In *UAI* 2009.

K. Murphy and Y. Weiss. Loopy belief propagation for approximate inference: An empirical study. In *UAI* 1999.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Francisco, California, 1988.

N. Shental, A. Zommet, T. Hertz, and Y. Weiss. Learning and inferring image segmentations with the GBP typical cut algorithm. In *ICCV* 2003.

M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree-based reparameterization for approximate estimation on loopy graphs. In *NIPS* 2002.

M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Exact map estimates by (hyper)tree agreement. In *NIPS* 2002.

M. J. Wainwright, T.S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.

W. Wiegerinck and T. Heskes. Fractional belief propagation. In *NIPS* 2002.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282–2312, 2005.

Journal of Machine Learning Research 11 (2010) 1737-1769 Submitted 6/09; Revised 3/10; Published 5/10

Evolving Static Representations for Task Transfer

Phillip Verbancsics VERB@EECS.UCF.EDU
Kenneth O. Stanley KSTANLEY@EECS.UCF.EDU
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816, USA

Editor: Michael Littman

Keywords: transfer learning, task transfer, evolutionary computation, neuroevolution, indirect encoding

Abstract

An important goal for machine learning is to transfer knowledge between tasks. For example, learning to play RoboCup Keepaway should contribute to learning the full game of RoboCup soccer. Previous approaches to transfer in Keepaway have focused on transforming the original representation to fit the new task. In contrast, this paper explores the idea that transfer is most effective if the representation is designed to be the *same* even across different tasks. To demonstrate this point, a *bird's eye view* (BEV) representation is introduced that can represent different tasks on the same two-dimensional map. For example, both the 3 vs. 2 and 4 vs. 3 Keepaway tasks can be represented on the same BEV. Yet the problem is that a raw two-dimensional map is high-dimensional and unstructured. This paper shows how this problem is addressed naturally by an idea from evolutionary computation called *indirect encoding*, which compresses the representation by exploiting its geometry. The result is that the BEV learns a Keepaway policy that transfers *without further learning* or manipulation. It also facilitates transferring knowledge learned in a different domain, Knight Joust, into Keepaway. Finally, the indirect encoding of the BEV means that its geometry can be changed without altering the solution. Thus static representations facilitate several kinds of transfer.

1. Introduction

Representation is a critical factor in the ability of any algorithm to learn autonomously (Clark, 1989). For example, a soccer player might represent the world through raw vision, distances and angles to other objects, or qualitative features such as close and far. Different such representations provide different perspectives to the learning algorithm. While one might be appropriate for learning physical control, another might better suit strategic planning. This paper focuses in particular on the effect of representation on *task transfer*, that is, bootstrapping knowledge gained learning one task to facilitate learning another, related task (Caruana, 1997; Talvitie and Singh, 2007; Taylor et al., 2007a). It turns out that representation not only affects the performance of such transfer, but also the elegance of its implementation. For example, transferring an artificial neural network (ANN) that takes as inputs parameters associated with objects (e.g., location, size, etc.) to a task with more such objects may require transforming the network by adding inputs for parameters associated with each new object (Taylor et al., 2007a). Yet such transformation can disrupt previous learning, thereby requiring the transformed network to undergo additional training to regain even its former

capabilities within the new scenario. As an alternative, this paper argues that an ideal representation would require no such transformations (i.e., it would remain static) when transferring to a new task.

The idea that input (i.e., state) representation might remain static during transfer is plausible because the raw inputs to biological organisms, for example, vision, remain the same even when new tasks are confronted. For example, when a child graduates from playing Keepaway to full-blown soccer, the number of photoreceptors in the eye do not change. The main idea in this paper is that such static representation, when possible, facilitates transfer by ensuring that the semantics of the representation are preserved even when the task changes.

To demonstrate the critical role of static representation in transfer, a novel state representation is introduced called a *bird's eye view* (BEV), which is a two-dimensional depiction of objects on the ground from above. Conceptually, the BEV is a metaphor for an internal representation of the state of the world from above. The BEV places objects into the context of the world geometry, allowing geometric relationships to be more easily learned. Another advantage is that its input dimensionality (i.e., number of inputs) is constant no matter how many objects are on the field. That way, even if the task is transferred to a version with more objects, the representation remains the same (i.e., static), significantly simplifying task transfer.

However, the challenge for the BEV is that representing a high-resolution two-dimensional field requires many input dimensions (i.e., many parameters), similarly to an eye. An outgrowth of evolutionary computation designed to address such high-dimensional problems is *indirect encoding*, which compresses the representation of the solution by reusing information. The particular indirect encoding in this paper, called a *compositional pattern producing network* (CPPN; Stanley 2007), represents artificial neural network (ANN) mappings between high-dimensional spaces by exploiting *regularities* in their geometry, which is well-suited to the BEV. An evolutionary algorithm called Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT; Gauci and Stanley 2008; Stanley et al. 2009; Gauci and Stanley 2010) that is designed to evolve CPPNs is therefore able to learn effectively from the BEV.

The HyperNEAT BEV is tested in the common RoboCup Keepaway soccer reinforcement-learning (RL) benchmark (Stone et al., 2005). Keepaway is important because it can potentially serve as a stepping stone to full-blown soccer in the future, which is a major current goal in machine learning (Kalyanakrishnan et al., 2007; Kitano et al., 1997; Kok et al., 2005; Kyrylov et al., 2005; Mackworth, 2009; Stolzenburg et al., 2006). One interesting result with the BEV is the longest holding time in the 3 vs. 2 variant of the task yet recorded. However, more importantly, unlike any method so far, HyperNEAT can transfer from 3 vs. 2 to 4 vs. 3 Keepaway with no change in representation and no further learning, demonstrating the critical role static representation plays in learning and transfer. Furthermore, these transferred policies can then be further trained on the new task without the need to alter the representation. Additional types of transfer within Keepaway are investigated wherein the *representation* of the policy (i.e., the CPPN, or indirect encoding) remains static while the BEV itself is changed by increasing resolution and by accommodating different field sizes. Finally, cross-domain transfer is demonstrated by training on a distinctly different domain, Knight Joust (Taylor et al., 2007a), which is a simple predator-prey type domain, and then transferring to 3 vs. 2 Keepaway.

The main result is that transfer through a static representation is consistently more robust and often provides immediate benefits even without any further training. While static representations are inherently high-dimensional because they must encompass many tasks, indirect encodings like HyperNEAT's CPPNs show that high-dimensionality need not be prohibitive. Thus, while machine

learning often focuses on the learning *algorithm*, the hope is that this paper provokes a fruitful conversation on the role of *representation* in transfer and learning in general.

It is finally important to acknowledge that the extent to which maintaining a static representation is realistic depends upon the learning method, state information, and differences between domains. Thus static representation is an ideal that when met can provide an advantage, as shown in this paper.

The next section describes the importance of representation in learning, prior research in transfer learning, and the methods that underlie the BEV representation. Section 3 explains how the BEV is configured, how information is represented in the BEV, and how HyperNEAT trains the BEV. In Section 4, the experiments that investigate the performance of the BEV in learning and transfer are described. Finally, Section 5 presents the results of the BEV, followed by a discussion in Section 6.

2. Background

This section examines the critical role of representation in RL and then explains the geometry-based methods that underlie the static representation investigated in this paper, and their relation to task transfer.

2.1 Representation in Learning

A convenient model for problems in RL is the *Markov decision process* (MDP). In the MDP, the learner knows its environment through a state observation $s \in S$, which is characterized by a set of state variables $s = \langle p_1, p_2, \cdots, p_n \rangle$, in which each p_i denotes a particular state parameter. By taking an action $a \in A$, the agent transitions to a new state in S through the transition function $T : S \times A \mapsto S$. The reward function $R : S \mapsto \mathbb{R}$ determines the instantaneous reward associated with reaching each state. Finally, the action that the agent takes from its current state is selected by the policy function $\pi : S \mapsto A$ (Puterman, 1994). For example, in the Keepaway soccer domain, the state space S for the keeper with the ball can be defined as the set of distances and angles to each other player. The set of actions A can be defined as a set of passes to teammates and holding the ball (Metzen et al., 2007; Stone et al., 2001; Stone and Sutton, 2001; Stone et al., 2005). A simple policy π would be to pass to the most open teammate when takers are close and hold the ball otherwise.

While the MDP framework provides a solid foundation for developing learning algorithms, it does not suggest how to select a state and action *representation* appropriate for both the domain and the learning algorithm. One popular approach to state representation, for example, in the RoboCup Keepaway soccer domain, is to express the state as distances and angles to the other players relative to the agent with the ball (Metzen et al., 2007; Stone et al., 2001, 2005; Taylor et al., 2006). However, this common representation is not the only one possible, which is important because representation critically influences what is learned (Gauci and Stanley, 2008; Diuk et al., 2008; Tadepalli et al., 2004; Tesauro, 1992).

To see the powerful effect of representation on learning, consider the common representation in 3 vs. 2 Keepaway of 13 state parameters that are value-attributes for distance and angle relationships among the players and the field. In contrast, 4 vs. 3 Keepaway, a similar task, requires an increased number of state parameters to represent the distances and angles for the additional players. These additional parameters mean that the same representation cannot be applied to both tasks, thereby complicating the transfer of knowledge between tasks. For example, the same concept must

be learned repeatedly when the same decisions are made separately for multiple objects, such as whether to pass to teammates who are out of bounds.

Relational RL addresses problems such as scaling and repetitious concepts by generalizing the representation of information for learning algorithms to a relational form (Deroski et al., 2001). For example, in the RoboCup Keepaway domain, instead of real-valued state parameters, general *relations* can be defined. An example for deciding to whom to pass in 3 vs. 2 Keepaway is:

$$Pass(Teammate) : -Threatened(PlayerWithBall), Open(Teammate).$$

The relational form provides a more expressive representation that can be combined with reinforcement-learning methods (Tadepalli et al., 2004). By focusing on the logic of relationships, instead of on individual parameter values, these relationships can be applied to any number of objects. Furthermore, once a relationship is learned for one set of objects, it is learned for all similar sets of objects.

One of relational RL's goals is to provide an easier representation for transferring knowledge. This transfer could be across objects in the domain or across different tasks. However, the design and definition of these relationships are dependent upon the human designer, requiring expert domain knowledge. Learning is dependent upon the a priori defined relations. Continuous and noisy domains present additional challenges to designing appropriate relations (Morales, 2003). Nevertheless, relational RL highlights the importance of representation to learning.

However, while state representation is important, it is not the only type of representation that affects learning. Also significant is the representation of the learner itself, which impacts which types of relationships can be learned and how easily they are found. For example, research in temporal difference learning can employ look-up tables or more compact representations (Sutton, 1988, 1996; Tesauro, 1992), which work by encoding regularities. An important difference between these representations is that the look-up table contains enough parameters to store associated actions with every state, while compact representations must encode the solution with significantly fewer parameters than states. To guarantee convergence with a *look-up table*, every state must be visited an infinite number of times (Sutton and Barto, 1998) while *compact* representations need only discover underlying regularities in the problem (Sutton, 1996; Tesauro, 1992).

Another important factor in representation is the geometry of the domain (e.g., which position is adjacent to which and in what direction). Geometry plays a critical role in learning. For example, if a checkers board is scrambled while the relationships among locations that have been moved remain the same, the game would become more difficult to learn. This effect has been investigated in checkers, wherein learning based on board geometry was demonstrated to enhance performance versus learning while blind to geometry (Gauci and Stanley, 2008, 2010). Ideally, the solution should be a function of the domain geometry, enabling the learner to take advantage of geometric regularities. This paper focuses further on the critical role of representing geometry, particularly in task transfer, which is described next.

2.2 Task Transfer

Task transfer means applying knowledge learned in one task to a new, related task (Caruana, 1997; Talvitie and Singh, 2007; Taylor et al., 2007a). It allows learning to be recycled instead of starting anew, thereby avoiding wasted computation. Additionally, a task may be so complex that it requires initial training on a simpler version to reduce learning time and increase performance (Caruana,

1997; Schmidhuber and Informatik, 1994; Tadepalli, 2008; Thrun and Mitchell, 1994). Thus the capability to transfer is becoming increasingly important as the tasks studied in RL increase in complexity. However, transfer learning faces several challenges: First, transfer is only effective among compatible tasks and the particular knowledge that can transfer from one task to another must be identified. Second, a method must be derived to actually implement the transfer of knowledge. Finally, cases in which transfer hinders performance, or *negative transfer*, must be avoided (Pan and Yang, 2008). There are several types of transfer learning problems and a variety of methods that exploit their characteristics. These methods include translating the knowledge learned in one task to another task (Ramon et al., 2007; Taylor et al., 2007a), choosing the best policy for the current task from a set of previously learned policies (Talvitie and Singh, 2007), extracting advice from previously learned tasks (Torrey et al., 2008a,b), and learning multiple tasks at the same time (Collobert and Weston, 2008). This section reviews several such approaches.

An intuitive approach to transfer learning is to transform the representation of knowledge learned in one task to a suitable form for a new task and then continue learning from that point. A successful method that takes this approach is *transfer via inter-task mapping for policy search methods* (TVITM-PS; Taylor et al. 2007a). TVITM-PS is such a leading method for transforming the policy learned in the source task into a policy usable in the target task. In TVITM-PS, a transfer functional ρ is defined to transform the policy π for a source task into the policy for a target task, such that $\rho(\pi_{source}) = \pi_{target}$. This functional is often hand-coded based on domain knowledge, though learning it is possible. When there are novel state variables or actions, an *incomplete mapping* is defined from the source to the target. TVITM-PS can be adapted to multiple representations. For example, in an ANN, input or output nodes whose connections are not defined in the mapping (i.e., it is incomplete) are made fully connected to the existing hidden nodes with random weights. This incomplete mapping implies that further training is needed to optimize the policies with respect to the new state variables and actions. However, it makes it possible to begin in the target domain from a better starting point than from scratch. TVITM-PS is a milestone in task transfer because it introduces a formal approach to moving from one domain to another that defines how ambiguous variables in the target domain should be treated. The performance of TVITM-PS is compared to results in this paper.

Another method of transfer, which is one that is explored in this paper, is to recycle the exact *same* policy from a source task in a later target task. The idea is that the policy can then continue to improve in the target task. An existing approach to recycling past policies is to maintain a *set* of policies and select among them. *Alternating trusting Exploration and suspicious exploitation* (AtEase; Talvitie and Singh 2007) is such a transfer method; it aims to recognize when tasks are related and when to exploit knowledge gained from previous tasks. It exploits knowledge from previous tasks by judging when to invoke the previously gained knowledge and from which policies. To facilitate this process, a set of policies previously developed by learning source tasks are first evaluated. This evaluation estimates the performance of these previously learned source policies on the new target task. Second, the strategies are ranked by their expected performance on the target task and the source policy with the best estimated performance is chosen. Finally, the chosen best policy is set as the current policy for the target task. It remains as the policy for the target task until the policy's actual performance on the task falls below expectation (i.e., the estimated performance from the evaluation of source policies is greater than the current performance) or reaches a maximum number of iterations (allowing other policies to be explored). If the expert policy falls below expectation, the next best policy is selected and is set as the current expert policy. This method

allows an accurate estimate of which policy from previously learned tasks is appropriate for the current task. In contrast to this approach, this paper focuses on how to effectively leverage knowledge gained in a *single* source policy to continue learning in the target domain. Thus the approach in this paper can potentially combine with a multi-policy approach such as AtEase.

An important consideration in transfer is whether a human can understand the knowledge being transferred among tasks. An alternative method to recycling previously learned policies directly is to take advice from learned policies to augment decision making. This advice may take the form of geometric knowledge, causal relationships, predictions, or any other type of information, allowing researchers to more easily interpret the transferred knowledge. *Rule extraction* is one such method that takes knowledge learned from a source domain and translates it into advice that aids a policy in a target domain (Torrey et al., 2008b). The advice is generated as a conditional, if-then statement. Torrey et al. (2008b) describe two methods for generating advice. One method is to compose rules by decomposing the policy learned on a source task. For example, Q-values can be examined directly and rules can be generated based on which actions are preferred. An alternative method for generating advice is to analyze the *behavior* (instead of the policy) of an agent to generate rules. Consider observing agents playing a game of Keepaway soccer. Through observation, it may be apparent that a learned policy always passes the ball if opponents approach within one meter, which may then be transformed into a rule to transfer to another task. These sets of rules have the advantage of being understandable to humans, allowing researchers to know what knowledge is being transferred and how it is contributing.

Interestingly, transfer learning does not always require a designated source and target task. Instead, knowledge may transfer among several tasks that are simultaneously being learned. By encoding the knowledge for multiple tasks within the same policy, the knowledge gained from each individual task may combine with and complement the knowledge from other tasks. For example, Collobert and Weston (2008) demonstrate transfer learning through multi-task training for natural language processing (NLP) with deep neural networks. There are many tasks related to NLP, including part-of-speech tagging, chunking, named entity recognition, semantic role labeling, language modeling, and relating words syntactically. The idea is that learning about one such task may contribute to learning the others. By training the policies simultaneously for all these capabilities, knowledge can be continually passed back and forth among all these tasks. In particular, Collobert and Weston (2008) show that this method improves generalization and achieves competitive results on the task of relating words with similar meaning.

This paper adds to our understanding of task transfer by focusing on the role of *representation*. The next section reviews the NEAT method, upon which this representation-centric approach is built.

2.3 NeuroEvolution of Augmenting Topologies (NEAT)

NEAT (Stanley and Miikkulainen, 2002, 2004) is a popular policy search method that evolves ANNs. The main idea in this paper focuses on an extension of NEAT called HyperNEAT. Nevertheless, the basic principles of NEAT still supply the foundation of the approach. Traditionally, ANNs evolved by NEAT control agents that select actions based on their sensor inputs. It is proven in a variety of challenging control and decision-making tasks (Aaltonen et al., 2009; Cardamone et al., 2009; Stanley and Miikkulainen, 2002, 2004; Stanley et al., 2005; Taylor et al., 2006; Whiteson, 2005; Whiteson and Whiteson, 2007). This section briefly reviews NEAT.

NEAT is an evolutionary algorithm that starts with a population of small, simple ANNs that increase their complexity over generations by adding new nodes and connections through mutation. That way, the topology of the network does not need to be known a priori and NEAT finds a suitable level of complexity for the task. NEAT is unlike many previous methods that evolved neural networks, that is, *neuroevolution* methods, which historically evolved either fixed-topology networks (Gomez and Miikkulainen, 1999; Saravanan and Fogel, 1995), or arbitrary random-topology networks (Angeline et al., 1993; Gruau et al., 1996; Yao, 1999). Unlike these approaches, NEAT begins evolution with a population of small, simple networks and increases the complexity of the network topology into *diverse species* over generations, leading to increasingly sophisticated behavior. A similar process of gradually adding new genes has been confirmed in natural evolution (Martin, 1999; Watson et al., 1987) and shown to improve adaptation in a few prior evolutionary (Altenberg, 1994) and neuroevolutionary (Harvey, 1993) approaches. However, a key feature that distinguishes NEAT from prior work in growing ANNs is its unique approach to maintaining a healthy diversity of increasingly complex structures simultaneously, as this section reviews. Complete descriptions of the NEAT method, including experiments confirming the contributions of its components, are available in Stanley and Miikkulainen (2002, 2004) and Stanley et al. (2005).

The NEAT method is based on three key ideas. First, to allow network structures to increase in complexity over generations, a method is needed to keep track of which gene is which. Otherwise, it is not clear in later generations which individual is compatible with which in a population of diverse structures, or how their genes should be combined to produce offspring. NEAT solves this problem by assigning a unique *historical marking* to every new piece of network structure that appears through a structural mutation. The historical marking is a number assigned to each gene corresponding to its order of appearance over the course of evolution. The numbers are inherited during crossover unchanged, and allow NEAT to perform crossover among diverse topologies without the need for expensive topological analysis.

Second, NEAT divides the population into species so that individuals compete primarily within their own niches instead of with the population at large. Because adding new structure is often initially disadvantageous, this separation means that unique topological innovations are protected and therefore have the opportunity to optimize their structure without direct competition from other niches in the population. The historical markings help NEAT determine to which species different individuals belong.

Third, many approaches that evolve network topologies and weights begin evolution with a population of random topologies (Gruau et al., 1996; Yao, 1999). In contrast, NEAT begins with a uniform population of simple networks with no hidden nodes, differing only in their initial random weights. Because of speciation, novel topologies gradually accumulate over evolution, thereby allowing diverse and complex phenotype topologies to be represented. No limit is placed on the size to which topologies can grow. New structures are introduced incrementally as structural mutations occur, and only those structures survive that are found to be beneficial through fitness evaluations. In effect, then, NEAT searches for a compact, appropriate topology by incrementally adding complexity to existing structure.

The important concept for the approach in this paper is that NEAT is a policy search method that discovers the right topology and weights of a network to maximize performance on a task. The next section reviews the extension of NEAT called HyperNEAT that allows it to exploit geometry through representation.

2.4 CPPNs and HyperNEAT

The primary reason that NEAT is chosen as the main vehicle to study alternate representations is that it is easily extended to become an *indirect encoding*, which means a *compressed* description of the solution network. Such compression makes the policy search practical even if the state space is high-dimensional. One effective approach to indirect encoding is to compute the network structure as a function of the domain's geometry. This section describes such an extension of NEAT, called Hypercube-based NEAT (HyperNEAT; Gauci and Stanley 2008; Stanley et al. 2009; Gauci and Stanley 2010), which enables the novel state representation in this paper from a bird's eye view. The effectiveness of the geometry-based learning in HyperNEAT has been demonstrated in multiple domains, such as checkers (Gauci and Stanley, 2008, 2010), multi-agent predator prey (D'Ambrosio and Stanley, 2008; D'Ambroiso and Stanley, 2010), visual discrimination (Stanley et al., 2009), and quadruped locomotion (Clune et al., 2009). For a full HyperNEAT description, see Stanley et al. (2009) and Gauci and Stanley (2010).

The main idea in HyperNEAT is that it is possible to learn geometric relationships in the domain through an indirect encoding that describes how the *connectivity* of the ANN can be *generated* as a function of the domain geometry. Unlike a *direct* representation, wherein every dimension in the policy space (i.e., each connection in the ANN) is described individually, an indirect representation can describe a pattern of parameters in the policy space without explicitly enumerating every such parameter. That is, information is reused in such an encoding, which is a major focus in the field of *generative and developmental systems* from which HyperNEAT originates (Bentley and Kumar, 1999; Hornby and Pollack, 2002; Lindenmayer, 1968; Turing, 1952). Such information reuse is what allows indirect encodings to search a compressed space. That is, HyperNEAT discovers the *regularities* in the domain geometry and learns a policy based on them.

The indirect encoding in HyperNEAT is called a *compositional pattern producing network* (CPPN; Stanley 2007), which encodes the *connectivity pattern* of an ANN (Gauci and Stanley, 2007, 2008; Stanley et al., 2009; Gauci and Stanley, 2010). The idea behind CPPNs is that a geometric pattern can be encoded by a *composition of functions* that are chosen to represent several common regularities. For example, because the Gaussian function is symmetric, when it is composed with any other function, the result is a symmetric pattern. The internal structure of a CPPN is a weighted network, similar to an ANN, that denotes which functions are composed and in what order. The appeal of this encoding is that it can represent a pattern of connectivity, with regularities such as symmetry, repetition, and repetition with variation, through a network of simple functions (i.e., the CPPN), which means that, instead of evolving ANNs directly, NEAT can evolve *CPPNs* that generate ANN connectivity patterns (Figure 1). Furthermore, the indirect encoding represents the connectivity of the ANN regardless of its size, which allows ANNs of arbitrary dimensionality to be represented.

Formally, CPPNs are *functions* of geometry (i.e., locations in space) that output connectivity patterns whose nodes are situated in n dimensions, where n is the number of dimensions in a Cartesian space. For each connection between two nodes in that space, the CPPN inputs their *coordinates* and outputs their connection weight. That way, NEAT can evolve CPPNs that represent ANNs with symmetries and regularities that are computed *directly* from the geometry of the state space. Consider a CPPN that takes four inputs labeled x_1, y_1, x_2, and y_2; this point in four-dimensional space can *also* denote the connection between the two-dimensional points (x_1, y_1) and (x_2, y_2). The output of the CPPN for that input thereby represents the weight of that connection (Figure 1). By querying

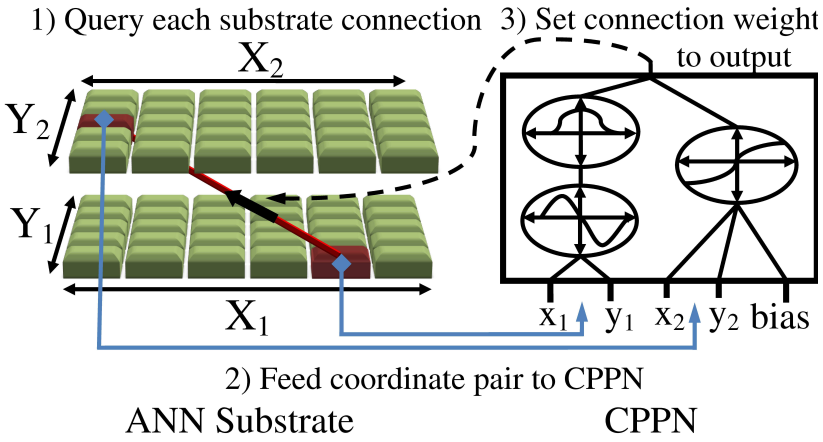
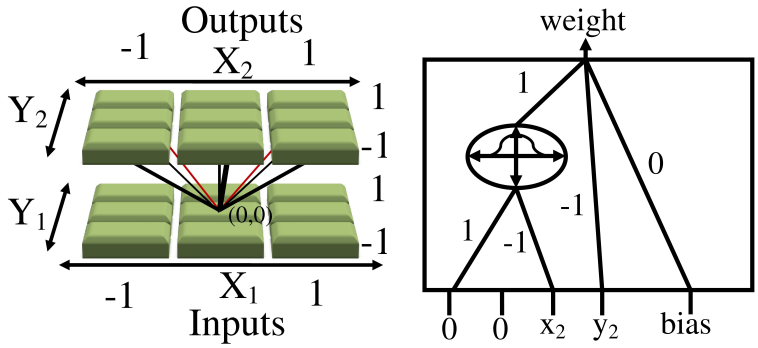


Figure 1: A CPPN Describes Connectivity. A grid of nodes, called the ANN *substrate*, is assigned coordinates. (1) Every connection between layers in the substrate is queried by the CPPN to determine its weight; the line connecting layers in the substrate represents a sample such connection. (2) For each such query, the CPPN inputs the coordinates of the two endpoints, which are highlighted on the input and output layers of the substrate. (3) The weight between them is output by the CPPN. Thus, CPPNs, whose internal topology and connection weights are evolved by HyperNEAT, can generate regular patterns of connections.

every pair of points in the space, the CPPN can produce an ANN, wherein each queried point is the position of a neuron. While CPPNs are themselves networks, the distinction in terminology between CPPN and ANN is important for explicative purposes because in HyperNEAT, CPPNs *encode* ANNs. Because the connection weights are produced as a function of their endpoints, the final structure is produced with *knowledge* of the domain geometry, which is literally depicted geometrically within the constellation of nodes. In other words, parameters p_i of the state vector s actually exist at *coordinates* in space, giving it a geometry.

To help explain how CPPNs can compactly encode regular connectivity patterns, Figure 2 shows how a very simple CPPN encodes a symmetric network. In effect, the CPPN paints a pattern within a four-dimensional hypercube that is interpreted as an isomorphic connectivity pattern. The example in Figure 2 illustrates the natural connection between the function embodied by the CPPN and the geometry of the resultant network.

Connectivity patterns produced by a CPPN in this way are called *substrates* so that they can be verbally distinguished from the CPPN, whose internal topology is independent of the substrate. The experimenter defines both the location and role (i.e., hidden, input, or output) of each node in the substrate. As a rule of thumb, nodes are placed on the substrate to reflect the geometry of the domain (i.e., the state), which makes the setup straightforward (Gauci and Stanley, 2007, 2008; Clune et al., 2009; Stanley et al., 2009; Gauci and Stanley, 2010). This way, the connectivity of the substrate becomes a direct function of the domain geometry, which means that knowledge about the problem can be injected into the search and HyperNEAT can exploit the regularities (e.g., adjacency,



(a) ANN Substrate (b) CPPN

Figure 2: Example CPPN Describing Connections from a Single Node. An example CPPN (b) with five inputs $(x_1, y_1, x_2, y_2, bias)$ and one output (*weight*) contains a single Gaussian node and five connections. The function produced is symmetric about x_1 and x_2 (because of the Gaussian) and linear with respect to y_2 (which directly connects to the CPPN output). For the given fixed input node coordinate $(x_1 = 0, y_1 = 0)$, the CPPN in effect produces the function Gaussian$(-x_2) - y_2$. This pattern of weights from input node $(0,0)$ is shown on the substrate (a). Weight magnitudes are indicated by thickness and black lines indicate positive values. Note that the pattern produces a set of weights that are symmetric about the x-axis and linearly decreasing as the values of y_2 increases. In this way, the function embodied by the CPPN encodes a geometric pattern of weights in space. HyperNEAT evolves the topologies and weights of such CPPNs.

or symmetry, which the CPPN sees) of a problem that are invisible to traditional encodings. For example, one way that geometric knowledge can be imparted is by including a hidden node in the CPPN that computes Gaussian$(x_2 - x_1)$, which imparts the concept of locality on the x-axis, an idea employed in the implementation in this paper. The HyperNEAT algorithm is outlined in algorithm 1.

In summary, instead of evolving the ANN directly, HyperNEAT, through the NEAT method, evolves the internal topology and weights of the CPPN that *encodes* it, which is significantly more compact. The next section explains how this encoding makes it possible to learn from a bird's eye view.

3. Approach: Bird's Eye View

A major challenge for the state representation in RL tasks is that specific state variables are often tied to agents or individual objects, which makes it difficult to add more such objects without expanding the state space (Taylor et al., 2007a). To address this problem, this section proposes a static representation, the bird's eye view (BEV) perspective, which enables scaling to higher complexity states without the need to alter the representation. The BEV is explained first, followed by its implementation, which is based on the HyperNEAT approach. Because it is relatively simple, the BEV is chosen in this paper to exemplify the advantage of static representation in task transfer.

```
   Input: Substrate Configuration
   Output: Solution CPPN
 1 Initialize population of minimal CPPNs with random weights;
 2 while Stopping criteria is not met do
 3     foreach CPPN in the population do
 4         foreach Possible connection in the substrate do
 5             Query the CPPN for weight w of connection;
 6             if Abs(w) >Threshold then
 7                 Create connection with a weight scaled proportionally to w (Figure 1);
 8             end
 9         end
10         Run the substrate as an ANN in the task domain to ascertain fitness;
11     end
12     Reproduce CPPNs according to the NEAT method to produce the next generation;
13 end
14 Output the Champion CPPN.
```

Algorithm 1: Basic HyperNEAT Algorithm

3.1 Bird's Eye View

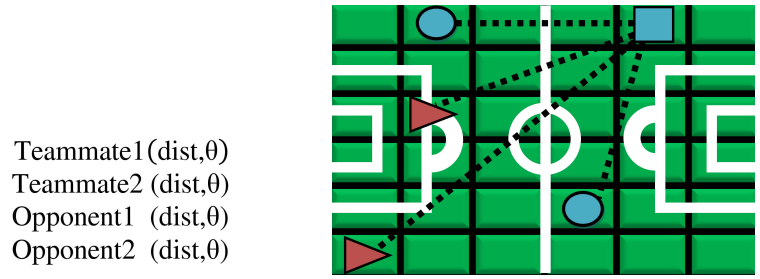
Humans often visualize data from a BEV. Examples include maps for navigation, construction blue prints, and sports play books. Key to these representations is that they remain the same (i.e., they are *static*) no matter how many objects are represented on them. For example, a city map does not change size or format when new buildings are constructed or new roads are created. Additionally, the physical geometry of such representations allow agents to understand spatial relationships among objects in the environment by placing them in the context of physical space. The BEV also implicitly represents its borders by excluding space outside them from its field of view. As suggested in Kuipers' Spatial Semantic Hierarchy (SSH), such *metrical* representation of the geometry of large-scale space is a critical component of human spatial reasoning (Kuipers, 2000).

A distinctive feature of the proposed representation is that not only is the agent state represented from a BEV, but it *also* requests *actions* within the same BEV perspective. For example, to request a pass the agent can indicate its target by simply highlighting it on a two-dimensional output array. That way, instead of making decisions blind to the geometry of physical space, it can be taken into account.

Egocentric data (Figure 3a) can be mapped to an equivalent BEV by translating from local (relative) coordinates to global coordinates established by static points of reference (i.e., fiducials). The global coordinates mark the location of objects in the BEV (Figure 3b). This translation allows mapping any number of objects into the static representation of the BEV.

Importantly, the continuous coordinate system must be discretized so that each variable in the state representation corresponds to a single discrete location. This discretization allows the two-dimensional field to be represented with a finite set of parameters. The values of these parameters denote objects in their respective regions.

Note that while the division of the field in Figure 3b appears reminiscent of *tile coding* (Sutton, 1996), that appearance is superficial because (1) a tile coding of the state variables in Figure 3a



(a) Egocentric view (b) BEV

Figure 3: Alternative Representations of a Soccer Field. Several parameters (a) represent the agent's relationship with other agents on a soccer field (taken from a standard RoboCup representation; Cheny et al. 2003). Each distance and angle pair represents a specific relationship of the agent to another agent. The BEV (b) represents the same relationships as paths in the geometric space. A square depicts the agent, circles depicts its teammates, and triangles its opponents. The overhead perspective also makes it possible to represent any number of agents without changing the representation.

would still be egocentric whereas the BEV is not, and (2) tile coding breaks the state representation into chunks that can be optimized separately whereas the HyperNEAT CPPN derives the connectivity of the policy network directly from the geometric relationships among the squares in Figure 3b, as explained next.

3.2 HyperNEAT: Learning from the BEV

Geometric patterns often exhibit spatial regularities. Examples include repetition and symmetry. Furthermore, important geometric relationships such as locality and topological connectedness often critically influence informed spatial decision-making. The challenge for machine learning is that learning is often blind to the geometry of the problem, making it difficult to exploit such relationships (Gauci and Stanley, 2008, 2010). To understand the impact of learning from the true geometry of the domain, consider a two-dimensional field converted to a traditional vector of parameters, which removes the geometry (Figure 4). For example, consider a set of input values to an ANN such as in to Figure 3a. Though each *dist* and θ pair is critically related in such a traditional representation, an ANN has no inherent knowledge or explicit access to this relationship. In contrast, HyperNEAT *sees* the task geometry, thereby exploiting geometric regularities and relationships, such as locality, which the BEV naturally makes explicit.

For HyperNEAT to exploit patterns in a two-dimensional BEV (e.g., in soccer), the geometry of the input layer of the substrate is made two-dimensional, as in Figure 5. That way, CPPNs can compute the connectivity of the substrate as a function of that geometry. The x and y coordinates of each input unit (i.e., each p_i) are in the range $[-1,1]$. Furthermore, the output layer of the substrate matches the dimensions of the BEV so that the CPPN can exploit the geometric relationship between the input space and output space as well (Figure 5). That the *outputs* are themselves a discretized two-dimensional plane is another significant difference from tile coding. Each coordinate in this substrate represents a discretized region of the overhead view of physical space. A four-dimensional CPPN with inputs x_1, y_1, x_2, and y_2 determines the weights between coordinates in the two-dimensional input layer and the two-dimensional output layer, creating a pattern of con-

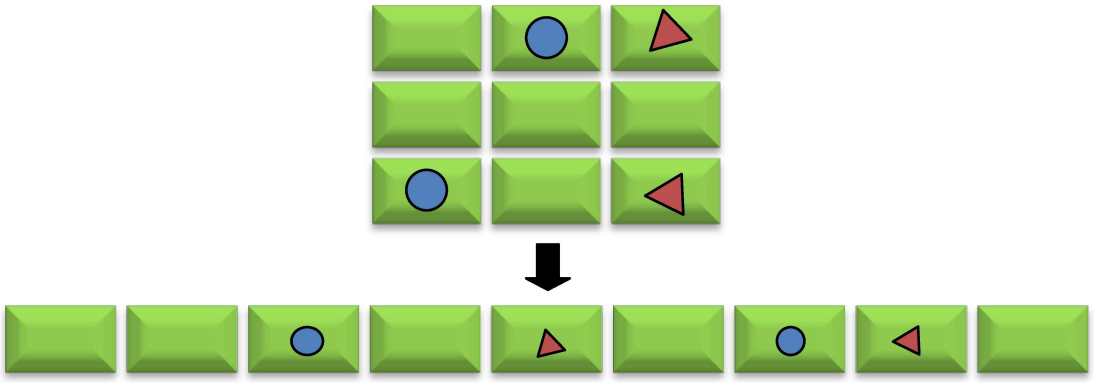


Figure 4: The Importance of True Geometry. A two-dimensional field transformed into a vector of parameters without any geometry forfeits knowledge of the geometry of the domain.

nections between regions in the physical space. To represent world state, objects and agents are literally "drawn" onto the input substrate, which is a static size, like marking a map. The generated network then can make decisions based on the relationships of such features in physical space and thereby learn the significance of certain kinds of geometric relationships among objects that are not identified a priori by the designer.

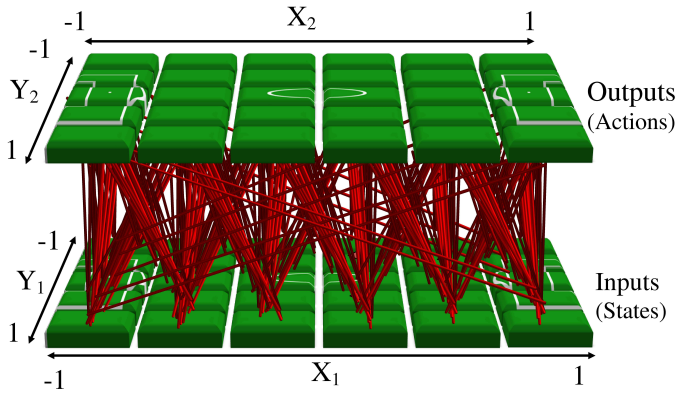


Figure 5: BEV Implemented in the Substrate. Each dimension ranges between $[-1, 1]$ and the input and output planes of the substrate are equivalently constructed to take advantage of geometric regularities between states and actions. Because CPPNs are an indirect encoding, the high dimensionality of the weights does not affect performance. (The CPPN is the search space.)

In this way, the BEV makes it possible to add new features (e.g., a new player) to the state space *without* the need to add new inputs. Instead, they can now simply be drawn onto the existing representation with no additional apparatus. That way, task transfer to different numbers of players is made simple through the static representation.

Interestingly, although the BEV is naturally held static its size or resolution can be changed *without* retraining. A unique feature of CPPNs (which encode the BEV connectivity) is that the same CPPN can query substrates of arbitrary size or resolution. It is important to note that even when size or resolution are changed, the *CPPN* itself remains the same. Thus the BEV can extend its representation to different field sizes or to different levels of detail (i.e., resolutions), as shown in Figure 6. In this way, the CPPN allows not only transfer to different numbers of players, but to different field sizes and resolutions, all without the need for retraining.

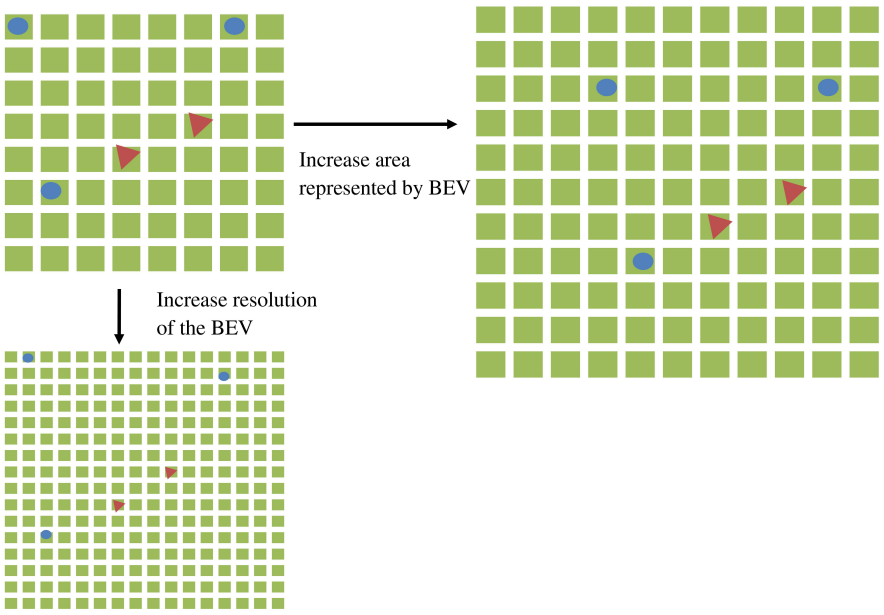


Figure 6: Changing the BEV. Two kinds of alterations are depicted in this figure. First, the BEV can be altered by increasing the *area* of the substrate while maintaining the size of each discrete cell by extrapolating new connection weights associated with previously unseen cells. Second, resolution is increased by increasing the number of cells and shrinking the area represented by each discrete cell. The CPPN automatically interpolates connection weights for the new locations. Thus, the BEV allows new forms of transfer to differing field sizes or levels of precision.

It is important to understand that the dimensionality of the search space in HyperNEAT is not the same as the dimensionality of the substrate because the search space is the *CPPN*, which is a compact encoding of the pattern of connections in the substrate. For example, if the substrate resolution is 20×20 then the number of possible connections in the substrate is $400 \times 400 = 160,000$. However, a CPPN that encodes this connectivity can itself contain orders of magnitude fewer connections. This fact also explains why resolution can increase without retraining. For example, if resolution increases to 40×40 (2,560,000 possible connections), there are new connections that connect locations that previously did not exist at 20×20. However, the same CPPN can simply query the (x_1, y_1, x_2, y_2) coordinate of the new connections, thereby interpolating the weights of the

new connections automatically. Although the number of connections in this example increases from 160,000 to 2,560,000, the dimensionality of the CPPN does not change at all.

The next section introduces the experiments that demonstrate the benefits of this geometric approach.

4. Experimental Setup

The experiments in this paper are designed to investigate the role of representation in task transfer. Of course, some representations are better suited to transfer in a given domain than others. Further, the ability to transfer between tasks is dependent on the similarity of the tasks. However, this paper focuses on the idea that a particularly effective representation for transfer is one that *does not need to change* from one task to the next. Because the representation is consistent, it has the potential to exhibit improved performance in the target domain immediately after transfer, without further learning. The advantage of a consistent representation is that the semantic relationships learned previously are preserved and then can be built upon. Because the BEV is the same irrespective of the number of players on either side, it satisfies this requirement and allows the hypothesis that consistent representation leads to immediate improvement in the target domain to be tested. This section explains the domains, the methods compared, and the experimental configurations.

4.1 RoboCup Keepaway Domain

RoboCup simulated soccer Keepaway (Stone et al., 2001) is well-suited to such an investigation because it is a popular RL performance benchmark and can be scaled to different numbers of agents to create new versions of the same task. All experiments are run on the Keepaway 0.6 player benchmark (Stone et al., 2006) and the RoboCup Simulator Soccer Server v. 12.1.1 (Cheny et al., 2003). RoboCup Keepaway is a popular benchmark (Metzen et al., 2007; Stone et al., 2005; Taylor et al., 2007a; Whiteson et al., 2005) in part because it includes a large state space, partially observable state, and noisy sensors and actuators. It is also a stepping stone to full-blown RoboCup Soccer, one of the hottest tasks in machine learning (Kalyanakrishnan et al., 2007; Kitano et al., 1997; Kok et al., 2005; Kyrylov et al., 2005; Mackworth, 2009; Stolzenburg et al., 2006). In Keepaway, *keepers* try to maintain possession of the ball within a fixed region and *takers* attempt to take it away. The number of agents and size of the field can be varied to make the task more or less difficult: The smaller the field and the more players in the game, the harder it becomes.

4.2 Keepaway Benchmark

Each learning method in this paper is initially compared in the standard benchmark setup (Stone et al., 2005) of the three keepers versus two takers task on a 20m$\times$20m field. In this setup, agents' sensors are noisy and their actions are nondeterministic. Takers follow static policies, wherein the first two takers go towards the ball and additional takers attempt to block open keepers. The learner only controls the keeper who possesses the ball; its choices are to hold the ball or pass to a specific teammate. The keepers' reward is the length of time they hold ball. In the 3 vs. 2 task, 13 variables represent the agent's state (Stone et al., 2005). These include each player's distance to the center of the field, the distance from the keeper with the ball to each other player, the distance from each other keeper to the closest taker, and the minimum angle between the other keepers and the takers (Figure 7). The three possible actions are holding the ball or passing to one of the other two keepers.

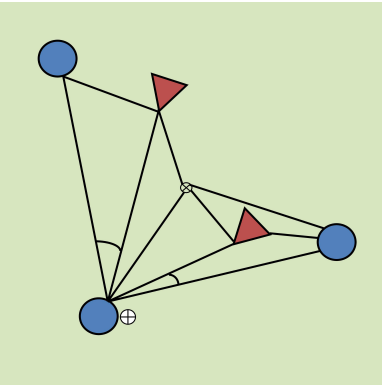


Figure 7: Visualization of Traditional State Variables in 3 vs. 2 Keepaway. The 13 state parameters that represent the state in the 3 vs. 2 Keepaway task are depicted in this figure. The three keepers are represented by the circles and the takers represented by the triangles. The state parameters include the distances from each player to the center of the field (marked by the circle with the $\times$), the distances from the keeper with the ball (denoted by the circle with the +) to each other player, the distance from each other keeper to the taker nearest them, and the angles along the passing lanes.

To investigate the ability of a static representation, that is, the HyperNEAT BEV, to learn this task, it is compared to both static policies (Stone et al., 2006) and the learning algorithms Sarsa (Rummery and Niranjan, 1994), NEAT (Stanley and Miikkulainen, 2004), and EANT (Metzen et al., 2007). Unlike the BEV, the traditional representation (with 13 state variables) through which these methods learn in 3 vs. 2 Keepaway must be changed for different versions of the task, such as 4 vs. 3 Keepaway. The static benchmarks are Always-Hold, Random, and a Hand-Coded policy, which holds the ball if no takers are within 10m (Stone and Sutton, 2001). These static benchmarks provide a baseline to validate that the BEV learns a non-trivial policy in the initial task.

State action reward state action (Sarsa; Rummery and Niranjan 1994) is an on-policy temporal difference RL method that learns the action-value function $Q(s,a)$. The quintuple (s,a,r,s',a') defines the update function for $Q(s,a)$ by determining for a current state (s) and action (a) what the reward (r) and the expected reward for the predicted next state (s') and action (a') will be. The update equation is:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma Q(s',a')),$$

where α is the learning rate and γ is the discount factor for the future reward. The values in $Q(s,a)$ determine which action is taken in a given state by selecting the maximal value. Each keeper separately learns which action to take in a given state to maximize the reward it receives (Taylor et al., 2006).

Regular NEAT (Stanley and Miikkulainen, 2002) evolves ANNs to maximize a fitness function. The ANN receives the 13 state inputs (like Sarsa) to define the state of the system and produce three outputs to select an action. The fitness in RoboCup Keepaway is the average length of time that keepers can hold the ball over a number of trials (Taylor et al., 2006). EANT (Metzen et al., 2007) is an additional neuroevolution algorithm based on NEAT that learned Keepaway. Though similar to NEAT, it distinguishes itself by more explicitly controlling the ratio of exploration to exploitation

during the evolutionary process. These methods were chosen for comparison because they have been tested in the same Keepaway configuration.

As described in Section 3, the HyperNEAT BEV transforms the traditional state representation to explicitly capture the geometry. The standard substrate is a two-dimensional 20×20 input layer connected to a 20×20 output layer. Thus both the state and action spaces have 400 dimensions each ($p_1 \ldots p_{400}$ and $a_1 \ldots a_{400}$). As with Sarsa in Stone and Sutton (2001), this policy representation does not include a hidden layer. However, the CPPN that encodes its weights *does* evolve internal nodes. Each node in a substrate layer represents a 1m^2 discrete chunk of Keepaway field. Each keeper's position is marked on the input layer with a positive value of 1.0 in its containing node and takers are similarly denoted by -1.0. Paths are literally drawn from the keeper with the ball to the other players (as in Figure 8).

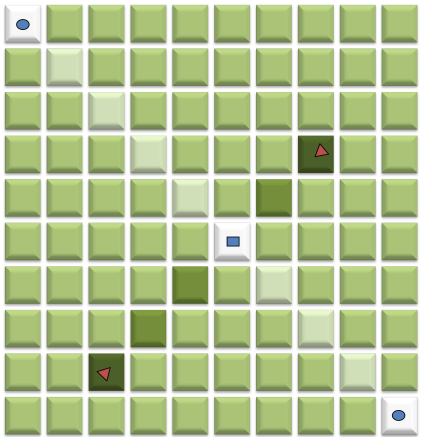


Figure 8: Visualizing the BEV Input Layer in 3 vs. 2 Keepaway. The input layer of the BEV is marked with the positions of keepers, takers and paths. The keeper with the ball is the small square, other keepers are circles, and the takers are triangles. Positive input values are denoted by lighter shades (for keepers and paths to keepers) and negative input values are denoted by darker shades (for takers and paths to takers). The middle shade represents an input of 0.0, the lightest shade is $+1.0$, and the darkest shade is -1.0. The BEV represents the distances and angles to other players in a geometric configuration, allowing geometric relationships to be exploited by HyperNEAT. Paths implicitly represent which keeper possesses the ball by converging on that keeper. (Note that the actual standard input layer in the experiments is 20×20.)

Positive values of 0.3 depict paths to other keepers and values of -0.3 depict paths to takers. These input values for agents and paths are experimentally determined and robust to minor variation. Actions are selected from among the output nodes (top layer of Figure 5) that correspond to where the keepers are located: If the highest output is the node where the keeper with the ball is located, it holds the ball. Otherwise, it passes to the teammate with the highest output at its node. This method of action selection thus corresponds exactly to the three actions available to Sarsa, NEAT, and EANT. A key property of this representation is that it is independent of the number of players on either side, unlike the representation in the traditional approaches.

The population size in HyperNEAT is 100. Available CPPN activation functions are absolute value, bipolar sigmoid, Gaussian, linear, sine, and step. Activation is signed, resulting in a node out-

put range of $[-1,1]$. By convention, a connection is not expressed if the magnitude the corresponding CPPN output is below a minimal threshold of 0.2 (Gauci and Stanley, 2007). The probability of adding a node to the CPPN is 0.05 and the probability of adding a connection is 0.18. The disjoint and excess node coefficients were both 1.0 and the weight difference coefficient was 1.0. The initial compatibility threshold was 20.0. These parameters were found to be robust to moderate variation in preliminary experimentation.

HyperNEAT evolves the CPPN that encodes the connectivity between the ANN layers in the substrate (up to 160,000 connections with a 20×20 resolution). Fitness is assigned according to the generated network's ball possession time averaged over at least 30 trials, with additional trials up to 100 assigned to those above the mean, following Taylor et al. (2006). Additionally, the CPPNs in the initial population are given the geometric concept of locality (Section 2.4).

4.3 Keepaway Transfer

Task transfer, the focus of this work, is first evaluated by training a HyperNEAT BEV on the 3 vs. 2 task on a 25m×25m field (instead of the standard 20m×20m) and then testing the trained BEVs on the 4 vs. 3 version of the task on the same field *without any further training*. The larger field is needed to accommodate the larger version of the task (Taylor et al., 2007b). To switch from 3 vs. 2 to 4 vs. 3, the additional players and paths are simply drawn on the input layer as usual, with no transformation of the representation or further training. The resulting performance on 4 vs. 3 is compared to TVITM-PS (Taylor et al. 2007b; described in Section 2.2), which is the leading transfer method for this task. TVITM-PS results are from policies represented by an ANN trained by NEAT (Taylor et al., 2007b). Unlike the HyperNEAT BEV, TVITM-PS requires further training after transfer because ρ expands the ANN by adding new state variables.

Additionally, two alternative forms of transfer are evaluated in Keepaway. The first is transfer to increasing field sizes, which is evaluated by first training individuals on a small (15m×15m) field size and then testing trained individuals on the trained and larger field sizes (each of 15m×15m, 20m×20m, and 25m×25m). To adjust for field size changes, the size of the HyperNEAT BEV substrate is changed to match the different field sizes (i.e., if the field size is 15m×15m, the substrate is 15×15; if it is 25m×25m, the substrate is 25×25). In this way, the relative meaning of each discrete input unit is held constant (e.g., $\frac{15\text{m}\times 15\text{m}}{15\times 15} = 1\text{m}^2$ per input and $\frac{25\text{m}\times 25\text{m}}{25\times 25} = 1\text{m}^2$ per input). The indirect encoding of the BEV extrapolates the trained knowledge from one field size to the other field sizes.

Second, transfer to substrates of different resolutions is evaluated by training individuals on a single field size, then doubling the resolution in each dimension of the substrate (i.e., an individual trained on a 20m×20m field with a 20×20 substrate is reevaluated on a substrate changed to 40×40). This increase in resolution results in a smaller section of the field being represented by each input (e.g., $\frac{20\text{m}\times 20\text{m}}{20\times 20} = 1\text{m}^2$ per input and $\frac{20\text{m}\times 20m}{40\times 40} = \frac{1}{4}\text{m}^2$ per input). The higher resolution BEV is then tested on the same field size to evaluate the ability to transfer knowledge between substrate resolutions. The new connections in the BEV are interpolated by the indirect encoding. In principle, this ability to raise resolution could allow computational cost to be reduced by training on a lower resolution and later raising resolution to increase the precision of the BEV.

4.4 Knight Joust

Knight Joust is a predator-prey variant domain wherein the player (prey) starts on one side of the field and the opponent (predator) starts on the opposite side (Taylor et al., 2007b). The player must then travel to the opposite side of the field while evading the opponent. The name *Knight Joust* reflects that the player is allowed three potential moves: move forward, knight jump left, and knight jump right, where a *knight jump* is two steps in the direction left or right and then forward (as in chess). The opponent follows a stochastic policy that attempts to intercept the player. The traditional state representation consists of the distance to the opponent, the angle between the opponent and the left side, and the angle between the opponent and the right side (Figure 9).

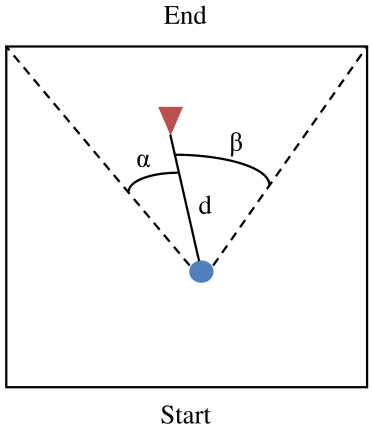


Figure 9: Knight Joust World. In Knight Joust, the player (circle) begins on the side marked *Start* and must reach the side marked *End*, while evading the opponent (triangle). The player is given the state information of the distance to the opponent, d, the angle between the opponent and the left side, α, and the angle between the opponent and the right side, β. This state information can similarly be drawn on the substrate of the BEV by marking the position of the player, opponent, the path between them, and the paths to the corners.

While Knight Joust is significantly different from Keepaway, a feature of both is that at each step the agent must make the decision that best avoids the opponent. However, Knight Joust is simpler, eliminating such complexity as multiple agents, noise, and kicking a ball, making it more tractable. The simplification makes it ideal for cross-domain transfer; because training is quicker and easier than in Keepaway, knowledge is more quickly bootstrapped. In Taylor et al. (2007a), cross-domain transfer from Knight Joust to Keepaway was shown to enhance learning. Additionally, the HyperNEAT BEV can represent the state information in Figure 9 by drawing the state information onto the inputs.

In particular, the player and opponent are indicated by $+1.0$ and -1.0 respectively. The path to the opponent is shown by values of -0.3 and the paths to the goal-side corners are marked with $+0.3$. Actions are selected from among the output nodes representing the position in front of the player (move forward), the left corner (knight jump left), and the right corner (knight jump right). This representation of state is similar to Keepaway, but the semantics are different: The player in Knight Joust is selecting a direction of movement instead of a passing position and the paths to the corners indicate the direction of the goal rather than teammate positions.

The evaluation of cross-domain transfer is completed by first training for 20 generations in the Knight Joust domain. Fitness is assigned to the individuals in Knight Joust by awarding 1 point for only moving forward and a bonus of 20 points for reaching the end. Next, the champions of these runs seed the runs for 3 vs. 2 Keepaway. Finally, Keepaway training is run for ten generations. The runs seeded with individuals trained in Knight Joust can then be compared to Keepaway runs without such transfer. This experiment is interesting because it can help to show that static transfer is beneficial with the BEV even in cases where the input semantics of the two tasks have slightly different meaning.

5. Results

This section describes the results of training the BEV on the Keepaway benchmark, the transfer performance among variations of the Keepaway task, and finally the performance of the BEV in cross-domain transfer from Knight Joust to Keepaway. Videos of evolved Keepaway behaviors are available at http://eplex.cs.ucf.edu/hyperneat-keepaway.html.

5.1 RoboCup Keepaway Performance Evaluation

In the RoboCup Keepaway benchmark, performance is measured by the number of seconds that the keepers maintain possession (Stone and Sutton, 2001; Stone et al., 2006; Taylor et al., 2007b). After training, the champion of each epoch is tested over 1,000 trials. Performance results are averaged over five runs with each consisting of 50 generations of evolution. This number of generations was selected because the corresponding *simulated* time spent in RoboCup during training equals simulated time (800-1,000 hours) for previous approaches (Taylor et al., 2006; Metzen et al., 2007). The test on the 3 vs. 2 benchmark is intended to validate that the BEV learns competitively with other leading methods.

In 3 vs. 2 Keepaway on the 20m$\times$20m field, the best keepers from each of the five runs controlled by a BEV substrate trained by HyperNEAT maintain possession of the ball on average for 15.4 seconds ($sd = 1.31$), which significantly outperforms ($p < 0.05$) all static benchmarks (Table 1). Furthermore, assuming similar variance, this performance significantly exceeds ($p < 0.05$) the current best reported average results (Stone et al., 2001, 2005; Taylor et al., 2006) on this task for both temporal difference learning (12.5 seconds) and NEAT (14.0 seconds), and matches EANT (14.9 seconds; Table 1). The important implication of this result is that the HyperNEAT BEV is at least competitive with the top learning algorithms on this task.

5.2 Keepaway Transfer Results

In transfer learning, the main focus of this work, the BEV is evaluated by testing individuals trained for 20 generations *only* on the 3 vs. 2 task on a 25m$\times$25m field. Learned policies are then tested on *both* the 3 vs. 2 and 4 vs. 3 tasks for 1,000 trials each without any further training. Note that this evaluation of transfer differs from Taylor et al. (2007b), in which teams trained on the smaller task are *further trained* on the larger task after the transfer because new parameters are added. In contrast, transfer within the BEV requires no changes or transformations. Performance is averaged over five runs, following Taylor et al. (2006). Figure 10 shows the average test performance on *both* 3 vs. 2 (trained) and 4 vs. 3 (untrained; immediately after transfer) of each generation champion.

METHOD	AVERAGE HOLD TIME
HYPERNEAT BEV	15.4S
EANT	14.9S
NEAT	14.0S
SARSA	12.5S
HAND-TUNED BENCHMARK	8.3S
ALWAYS HOLD BENCHMARK	7.5S
RANDOM BENCHMARK	3.4S

Table 1: Average Best Performance by Method. The HyperNEAT BEV holds the ball longer than previously reported best results for neuroevolution and temporal difference learning methods. Results are shown for Evolutionary Acquisition of Neural Topologies (EANT) from Metzen et al. (2007), NeuroEvolution of Augmenting Topologies (NEAT) from Taylor et al. (2006), and State action reward state action (Sarsa) from Stone and Sutton (2001).

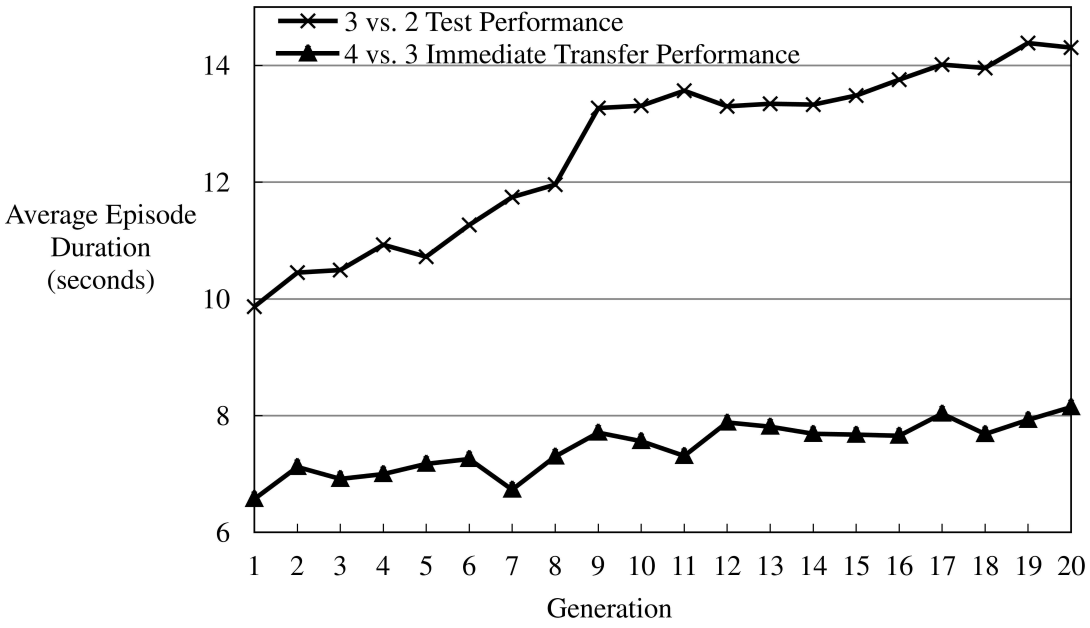


Figure 10: Transfer Learning From 3 vs. 2 to 4 vs. 3 Keepaway on a 25m×25m Field. As the performance (averaged over five runs) of the champion on the 3 vs. 2 task improves, the transfer performance on the 4 vs. 3 task also consequently improves from 6.6 seconds to 8.1 seconds without *ever* training for it. The improvement is positively correlated ($r = 0.87$).

Testing performance on the 3 vs. 2 task improves to 14.3 seconds on average over each run. At the same time, the test performance of these same individuals on the 4 vs. 3 task, which was not trained, improves from 6.6 seconds to 8.1 seconds on average. In contrast, the previous best approach to transfer learning in this domain required executing a transfer function and additional

training for between 50 and 200 hours (depending on the chosen transfer function) *beyond* the initial bootstrap training in 3 vs. 2 to achieve a comparable 8.0 second episode duration (Taylor et al., 2007b). Thus, because the BEV is static, transfer is instantaneous and requires no special adjustments to the representation to achieve the same result as many hours of *further* training with the TVITM-PS transfer method.

Although the BEV improves in 4 vs. 3 Keepaway even when only trained in 3 vs. 2, it is still informative to investigate the effect of further training in the 4 vs. 3 task. For this purpose, individuals are trained on the 3 vs. 2 task for 20 generations and then further trained on the 4 vs. 3 task for 30 generations. The performance of these policies is contrasted with keepers trained on 4 vs. 3 from scratch for 50 generations. Performance is averaged over five runs and generation champions are evaluated over 1,000 episodes. Figure 11 shows the average test performance of the generation champions. The individuals trained solely on 4 vs. 3 improve from 6.2 seconds to 8.0 seconds. Interestingly, this performance is equivalent to policies trained only in the 3 vs. 2 task and transferred to 4 vs. 3. However, individuals trained on 3 vs. 2 for the first 20 generations increase their test performance on 4 vs. 3 to 9.1 seconds over the last 30 generations. The final difference between further training after transfer and training from scratch is significant ($p < 0.05$).

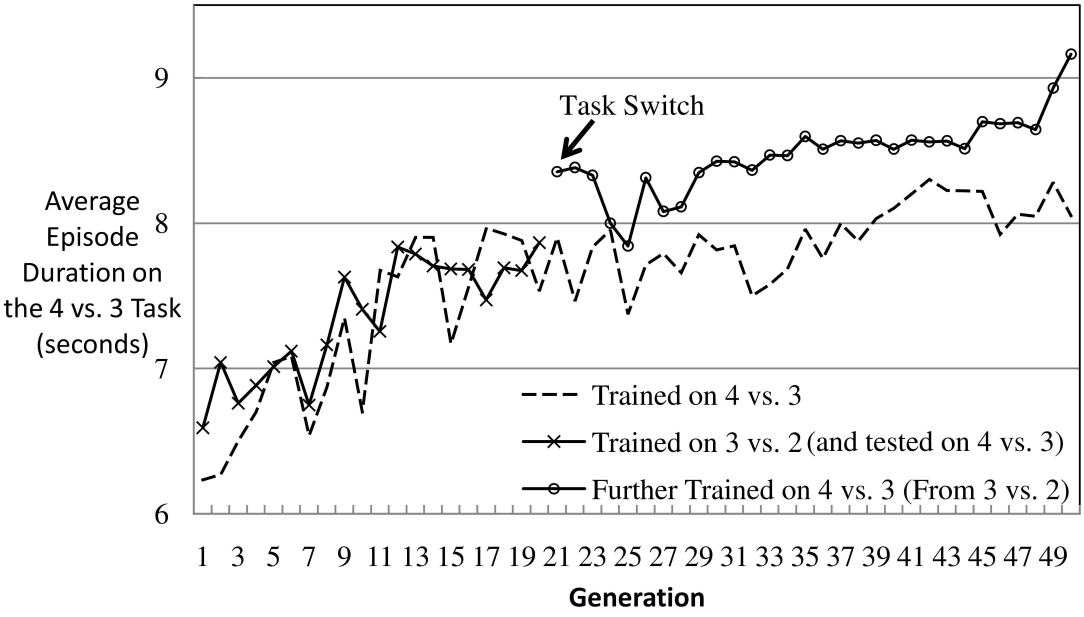


Figure 11: Further Training After Transfer From 3 vs. 2 to 4 vs. 3 Keepaway on a 25m×25m Field. Performance of individuals trained on 3 vs. 2 then transferred to 4 vs. 3 and further trained are contrasted with individuals solely trained on 4 vs. 3. All depicted results are performance on the 4 vs. 3 task. Prior training on 3 vs. 2 and transfer to the 4 vs. 3 enhances keeper performance by beginning in a more optimal area of the search space.

An important factor in the superior performance of the learner that was transferred is the behavior of the third taker in the 4 vs. 3 task, which seeks to block the most open player. This behavior differs from 3 vs. 2, in which the two takers attempt to take the ball only by always heading towards

it. When training on 4 vs. 3 without previously learning on 3 vs. 2, the third taker's behavior may inhibit performance by preventing important knowledge from being learned. For example, in 3 vs. 2 an important concept is to pass to the most open player. However, in 4 vs. 3 the most open player is *not* always the best choice because of the behavior of the third taker; therefore policies that learn the concept of passing to the most open player, which is still an important skill, are not discovered.

A thorough evaluation of transfer recognizes that there is more than one way to alter a task. Thus transfer learning is also evaluated by testing the best policy trained in 3 vs. 2 on varied field sizes. Stone et al. (2001) previously investigated this kind of transfer on their best Sarsa solution in an easier version of the Keepaway task that does not include noise by testing a single high-performing individual that was trained on a fixed field size (15m×15m) not only on the trained field size, but also on the other two field sizes. The best policy trained by the HyperNEAT BEV (which, unlike Sarsa, was subject to noise) on the 15m×15m field size was also tested in this way (Figure 12).

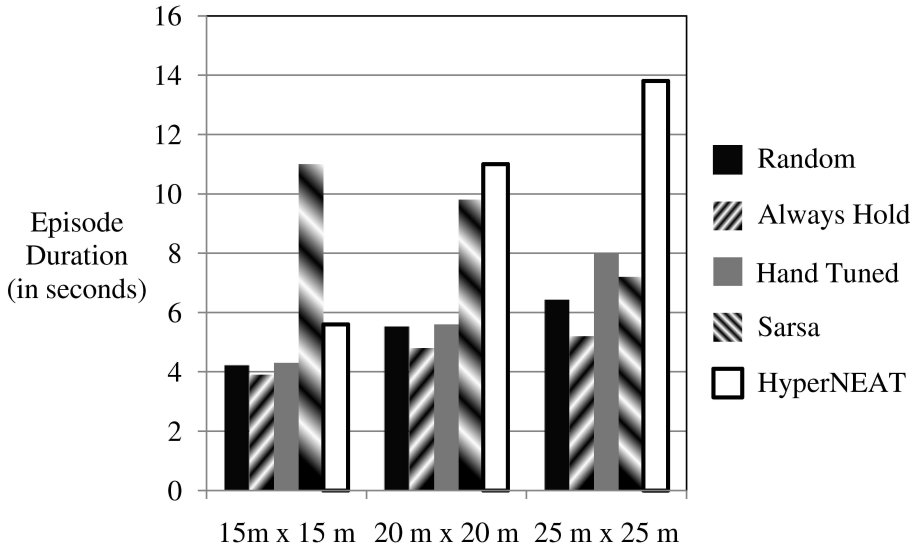


Figure 12: 3 vs. 2 Transfer Performance To Larger Field Sizes. Transfer to larger field sizes is evaluated by testing an individual trained on a single field size (15m×15m) on two larger field sizes (20m×20m and 25m×25m) as well. The BEV is scaled by matching the substrate size to the field size, thus maintaining the same field area represented by each discrete unit on the substrate. Depicted results from Stone and Sutton (2001) show that as a policy trained by Sarsa is transferred to larger field sizes, it *decreases* in performance. However, the task is *easier* as field size increases, as shown by the performance of hand-designed policies (Random, Always Hold, and Hand-Tuned) that *increase* in performance as field size increases. In contrast, the BEV learns a policy that outperforms the hand-designed policies and transfers to the larger field sizes, *significantly improving* performance.

The results are interesting because they show that the representation can cause performance to vary in unexpected ways. For example, even though larger field sizes are easier, Stone and Sutton (2001) report that the performance of the best keepers trained by Sarsa *declines* when they are

transferred to larger fields. However, even hard-coded policies, such as Random, Always Hold, or Hand Tuned, increase in performance as field size increases, demonstrating the decreased difficulty of the task. Also, in contrast to Sarsa, when transferred to larger fields, the keepers trained with the HyperNEAT BEV improve performance (as would be expected) from 5.6 seconds to 11.0 seconds and 13.8 seconds, respectively, and outperform the hand-designed policies (Figure 12). These improvements make sense because the task should become easier when there is more room on the field.

The BEV's advantage is that the geometric relationships encoded in the CPPN can be extrapolated as the field size increases, thereby extending the knowledge from the smaller field size to the newer areas of the larger field. For Sarsa, such extrapolation is not possible because as field size increases, the new areas represent previously unseen distances for which Sarsa was not trained. Sarsa has no means to extrapolate geometric knowledge from the distances it has seen because, unlike the CPPN, the knowledge learned is not a *function* of the domain geometry (i.e., the geometric relationships on a two-dimensional soccer field). Instead, Sarsa learns a function of the examples presented, which do not explicitly describe the geometry of the domain.

Another important lesson from changing the field size is that BEV performance requires a minimal resolution. When the field size is 15m×15m, the BEV performance appears to underperform compared to Sarsa. In part, this difference is because Sarsa was tested originally without noise (Stone et al., 2001). A later experiment with Sarsa trained on the 15m×15m with noise (Stone et al., 2005) shows that its performance is similar to the BEV. However, another factor is simply that when the field size is 15m×15m, the BEV resolution is *also* at 15×15, which may be too low to capture the detail necessary to succeed in the task. Confirming this hypothesis, if the BEV is trained at 30×30 resolution on a 15m×15m field, its performance rises significantly, to 7.1 seconds compared to 7.4s for Sarsa when it *is* trained with noise on 15m×15m (Stone et al., 2005). This result raises the interesting question of whether resolution can be *raised* above the training resolution without negative impact, as the next experiment addresses.

The final result in Keepaway is that the knowledge learned through the indirect encoding, that is, the CPPN, is not negatively impacted by later increasing resolution from that at which the BEV was trained. The substrate resolution of the champion individuals from five runs from training on three field sizes (15m×15m, 20m×20m, and 25m×25m) are doubled in each dimension and then tested again on the same field size. For example, a 20×20 BEV becomes 40×40, which means that each input represents one quarter as much of the space as before. This BEV quadruples the number of inputs and outputs while increasing the number of connections by a factor of 16 (from 160,000 to 2,560,000 connections). Table 2 shows that no matter the field size, even massively increasing the resolution does not degrade performance and can even lead to a *free* performance increase.

For the 15m×15m, 20m×20m, and 25m×25m field sizes, doubling the size of each dimension on average changes performance from 4.6 seconds to 5.3 seconds, 15.4 seconds to 15.9 seconds, and 16.8 seconds to 16.9 seconds respectively. In one instance, on the 20m×20m field, performance improved instantly from 16.6 seconds to 18.9 seconds. The advantage of this capability is that the BEV resolution can be selectively increased while maintaining the same performance, which makes possible further training with a higher resolution BEV.

	PERFORMANCE	
TRAINING FIELD SIZE	TRAINED RESOLUTION	INCREASED RESOLUTION
15M×15M	4.6S	5.3S
20M×20M	15.4S	15.9S
25M×25M	16.8S	16.9S

Table 2: Average Performance of the Best Individuals at Different Resolutions. The regularities learned by the indirect encoding are not dependent on the particular substrate resolution and may be extrapolated to higher resolutions. Increasing the number of connections in the substrate by a factor of 16 (by doubling the size of each dimension) does not degrade performance; in fact, it even improves it significantly in some cases.

5.3 Knight Joust Transfer Results

Cross-domain transfer is evaluated from the non-Keepaway task of Knight Joust on a 20×20 grid to 3 vs. 2 Keepaway on a 20m×20m field. Evolution is run for 20 generations on the Knight Joust task and then the champions seed the beginning generations of 3 vs. 2 Keepaway. Further training is then performed over ten additional generations of evolution. Performance in Keepaway of the champion players from Knight Joust is on average 0.3 seconds above the performance of initial random individuals. After one generation of evolution, the best individuals from transfer exceed the raw performance by 0.6 seconds. Finally, after ten further generations, the best individuals with transfer hold the ball for 1.1 seconds longer than without transfer (Figure 13).

The differences are significant ($p < 0.05$). Thus even preliminary learning in a significantly different domain proved beneficial to the BEV. In contrast, previous transfer results from Knight Joust to Keepaway from Taylor and Stone (2007) demonstrated an initial performance advantage, but after training for five simulator hours (which is less than the duration of ten generations) there was no performance difference between learning with transfer and without it.

Overall, the results establish that the BEV is highly effective in transfer in Keepaway. The next section discusses the deeper implications of these results.

6. Discussion and Future Work

Methods that alter representation remain important tools in task transfer for domains in which the representation must change with the task. However, the BEV shows that a carefully chosen representation with the right encoding can sometimes eliminate the need to change the representation, even across different domains.

The deeper lesson is the critical role of representation in transfer and the consequent need for algorithms that can learn from relatively high-dimensional static representations of task geometry. Indeed, the human eye contains *millions* of photoreceptors, which provide the same set of inputs to every visual task tackled by humans. No new photoreceptor is added for a new task. In effect, visual input to the human eye is a static representation (i.e., it does not alter when changing tasks) of state to the human brain. While it is true that the information from the eye is interpreted by the visual cortex, the set of inputs to the cortex, which are the photoreceptors of the eye, remains the same. In this paper, the BEV contains no hidden layers. However, by adding hidden layers it is possible to

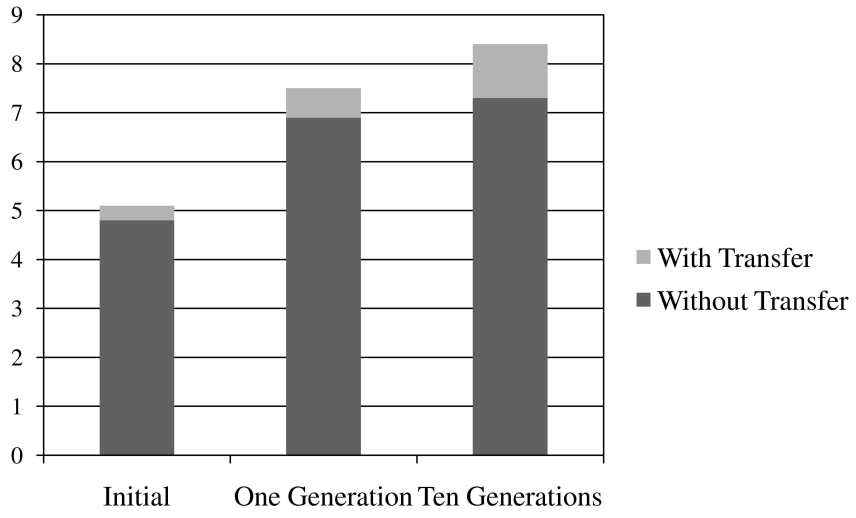


Figure 13: Transfer Results from Knight Joust to Keepaway. Direct transfer and further training performance averaged over 30 runs is shown. The performance of raw champions from Knight Joust on Keepaway outperforms initial random individual by 0.3 seconds. After one generation, this advantage from transfer increases to 0.6 seconds and at 10 generations the advantage is 1.1 seconds. Thus performance on Keepaway, both instantaneous and with further training, benefits from transfer from the Knight Joust domain with significance $p < 0.05$.

add the intervening interpretation of the input state analogously to how the visual cortex interprets data from the eye. HyperNEAT substrates with hidden layers have been shown to work in the past in domains without transfer (D'Ambrosio and Stanley, 2008; Clune et al., 2009; Gauci and Stanley, 2010). Thus the prospects are good for expanding the scope of static transfer. Nevertheless, of course the human eye represents an ideal, and not all possible domains are amenable to keeping the representation static. Yet for those that are, the investigation in this paper shows that it can provide an advantage.

The ability of a representation to remain static is dependent upon the particular differences between the tasks. Tasks that are semantically similar should be able to represent state information similarly in the BEV, requiring no changes to the representation. Tasks that are significantly different, either through state information or actions, may not allow the representation to remain the same. However, even then there are potential ways to allow the BEV to retain what was learned in the previous task and build upon it while being altered. For example, one option is to add new input layers or output layers. These new layers can denote new information or actions associated with new objects in the environment while the previously-trained input and output layers retain the prior knowledge. Geometry thus remains an advantage because state information that is connected with the same location (e.g., all the state data for a single agent) would be located at the same coordinate on separate layers. In contrast, an ANN without geometry would have no means to discern which original inputs are associated with which new inputs and would thus instead have to learn such relationships.

The role of representation in transfer is relevant to all approaches to learning because transfer is always an option for extending the scope of learning. Thus encoding research, such as in generative and developmental systems (Bentley and Kumar, 1999; Hornby and Pollack, 2002; Lindenmayer, 1968; Stanley, 2007; Bentley and Kumar, 1999; Turing, 1952), and representation research, such as in relational reinforcement learning (Deroski et al., 2001; Morales, 2003; Tadepalli et al., 2004), is important to machine learning in general. Static representations mean that instead of training a new policy, or retraining a previous one, the same policy can be transferred without change. Additionally, the static nature of the representation allows the same policy to train on multiple tasks simultaneously. For example, a soccer player does not practice by playing *only* soccer games. Players improve through multiple drills and continually practice in-between games to refine skills.

The encoding of the solution also impacts the kinds of policies that are found. For example, in this paper the policy is encoded by a CPPN that is expressed as a function of the task geometry, which enables the solution to exploit regularities in the geometry and extrapolate to previously unseen areas of the geometry. It should also be possible to simplify the search for a policy that is a function of the geometry in other learning approaches as well. The challenge is that gradient information (i.e., error) cannot directly pass through the indirection between the ANN and its generating CPPN. A method that solves this problem would open up the power of indirect encoding to all of RL.

6.1 Prospects for Full RoboCup Soccer

An exciting implication of this work is that the power of static transfer and indirect encoding can potentially bootstrap learning the complete game of soccer. After all, the key elements of soccer are present in Keepaway as well. In fact, the results in this paper demonstrate that a static representation can competitively learn to hold the ball in Keepaway and that this skill transfers immediately through the BEV to variations of that task. The static BEV state representation enables the learned policy to transfer to variations of the task in which the number of players is changed (e.g., 3 vs. 2 to 4 vs. 3). Furthermore, indirectly encoding the policy enables the same policy to be applied to variations of the task in which the geometry has been changed (e.g., moving from 20m$\times$20m to 25m$\times$25m field size) HyperNEAT has also been proven effective in a wide variety of tasks (D'Ambrosio and Stanley, 2008; Clune et al., 2009; Stanley et al., 2009; Gauci and Stanley, 2010).

Interestingly, the Keepaway domain was designed as a stepping stone to scaling machine learning methods to the full RoboCup soccer domain (Stone and Sutton, 2001). The same principles that enable the BEV to transfer among variations of the Keepaway domain *also* can potentially enable the BEV to scale to full Keepaway soccer. For example, because the representation remains static no matter how many players are on the field, training can begin with a small number of players, such as 3 vs. 3 soccer, and iteratively add more players, eventually scaling up to the full 11 vs. 11 soccer game. Furthermore, varying the substrate configuration while the solution encoding remains static makes it possible to train skills relevant to RoboCup on subsets of the full field, for example, half-field offense/defense. In this way, varying the number of players and varying the field size are *both* required to transfer from the RoboCup Keepaway domain to full RoboCup soccer. Thus this study suggests a novel path to learning full-fledged soccer.

A distinctive feature of the BEV representation is that actions are *also* selected in the BEV, that is, the outputs are in the same geometry as the field. In the RoboCup Keepaway domain, actions are constrained to holding the ball and directly passing to a teammate. However, there are many

other actions that are possible, such as clearing the ball, kicking the ball out of bounds, dribbling, and passing to a location close to a teammate. Furthermore, the BEV can potentially control players without the ball. By requesting actions in the BEV geometry, actions can be selected based on positions instead of objects.

For example, the keeper with the ball can potentially select *any* position on the BEV to which to kick the ball. That way, the BEV is not constrained in its actions. The player with the ball can then choose from passes to teammates, passes to positions near teammates, or dribble by kicking the ball to a nearby position and then pursuing the ball. Players without the ball can be controlled by interpreting the outputs of the BEV as the desired location towards which that player should move. Thus an interesting property of the BEV is that the state space can transfer, by accommodating new players or field sizes, and the action space can *also* transfer in the same way. Ultimately, the promise of such transfer is tied to the idea of static representation, whose potential was highlighted in this paper.

7. Conclusion

This paper introduced the BEV representation, which simplifies task transfer by making the state representation static. That way, no matter how many objects are in the domain, the size of the state representation remains the same. In contrast, in traditional representations, changing the number of players (e.g., in the RoboCup Keepaway task) forces changes in the representation by adding dimensions to the state space. In addition to results competitive with leading methods on the Keepaway benchmark, the BEV, which is enabled by an indirect encoding, achieved transfer learning from 3 vs. 2 to 4 vs. 3 Keepaway *without* further training. Improvement after further training then demonstrated that the knowledge gained from the transfer does indeed facilitate further learning the more difficult task. Transfer also proved successful not only among variations on the number of players, but also among different field sizes and substrate resolutions. Finally, cross-domain transfer was demonstrated, from Knight Joust to Keepaway. The cross-domain transfer improved not only immediate performance, but also enhanced further learning. All these results highlight the critical role that representation plays in learning and transfer. By altering the representation, transfer learning is simplified. Yet high-dimensional static representations require indirect encodings that take advantage of their expressive power, such as in HyperNEAT. The hope is that advanced representations in conjunction with indirect encoding can later contribute to scaling learning techniques to more challenging tasks, such as the complete RoboCup soccer domain.

Acknowledgments

This research is supported in part by a Science, Mathematics, and Research for Transformation (SMART) fellowship from the American Society of Engineering Education (ASEE) and the Naval Postgraduate School.

References

Timo Aaltonen et al. Measurement of the top quark mass with dilepton events selected using neuroevolution at CDF. *Physical Review Letters*, 2009.

Lee Altenberg. Evolving better representations through selective genome growth. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 182–187, Piscataway, NJ, 1994. IEEE Press.

Peter J. Angeline, Gregory M. Saunders, and Jordan B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5:54–65, 1993.

Petet J. Bentley and Sanjeev Kumar. The ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, pages 35–43, San Francisco, 1999. Kuafmann.

Luigi Cardamone, Daniele Loiacono, and Pier Luca Lanzi. On-line neuroevolution applied to the open racing car simulator. In *Proceedings of the 2009 IEEE Congress on Evolutionary Computation (IEEE CEC 2009)*, Piscataway, NJ, USA, 2009. IEEE Press.

Rich Caruana. Multitask learning. In *Machine Learning*, pages 41–75, 1997.

Mao Cheny, Klaus Dorer, Ehsan Foroughi, Fredrik Heintz, ZhanXiang Huangy, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Jan Murray, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, Yi Wangy, and Xiang Yin. *Robocup Soccer Server: User's Manual*. The Robocup Federation, 4.00 edition, February 2003.

Peter Clark. *Machine and Human Learning*. London: Kogan Page, 1989.

Jeff Clune, Benjamin E. Beckmann, Charles Ofria, and Robert T. Pennock. Evolving coordinated quadruped gaits with the hyperneat generative encoding. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC-2009) Special Section on Evolutionary Robotics*, Piscataway, NJ, USA, 2009. IEEE Press.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, New York, NY, 2008. ACM Press.

David D'Ambroiso and Kenneth O. Stanley. Evolving policy geometry for scalable multiagent learning. In *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010)*, page 8, New York, NY, USA, 2010. ACM Press.

David B. D'Ambrosio and Kenneth O. Stanley. Generative encoding for multiagent learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, New York, NY, 2008. ACM Press.

Carlos Diuk, Andre Cohen, and Michael L. Littman. An object-oriented representation for efficient reinforcement learning. In *ICML '08: Proceedings of the 25th International Conference on Machine learning*, pages 240–247, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: http://doi.acm.org/10.1145/1390156.1390187.

Sao Deroski, Luc De Raedt, and Kurt Driessens. Relational reinforcement learning. *Machine Learning*, 43(1-2):7–52, April-May 2001.

Jason Gauci and Kenneth O. Stanley. Generating large-scale neural networks through discovering geometric regularities. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 8, New York, NY, 2007. GECCO-2007, ACM.

Jason Gauci and Kenneth O. Stanley. A case study on the critical role of geometric regularity in machine learning. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-2008)*, Menlo Park, CA, 2008. AAAI Press.

Jason Gauci and Kenneth O. Stanley. Autonomous evolution of topographic regularities in artificial neural networks. *Neural Computation*, page 38, 2010.

Faustino Gomez and Risto Miikkulainen. Solving non-Markovian control tasks with neuroevolution. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1356–1361, San Francisco, 1999. Kaufmann.

Frederic Gruau, Darrell Whitley, and Larry Pyeatt. A comparison between cellular encoding and direct encoding for genetic neural networks. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 81–89, Cambridge, MA, 1996. MIT Press.

Inman Harvey. *The Artificial Evolution of Adaptive Behavior*. PhD thesis, School of Cognitive and Computing Sciences, University of Sussex, Sussex, 1993.

Gregory S. Hornby and Jordan B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3), 2002.

Shivaram Kalyanakrishnan, Yaxin Liu, and Peter Stone. *RoboCup 2006: Robot Soccer World Cup X*, volume 4434 of *Lecture Notes in Computer Science*, chapter Half Field Offense in RoboCup Soccer: A Multiagent Reinforcement Learning Case Study. Springer Berlin / Heidelberg, 2007.

Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, Eiichi Osawa, and Hitoshi Matsubara. Robocup: A challenge problem for AI. *AI Magazine*, 18(1):73–87, Spring 1997.

Jelle R. Kok, Matthijs T. J. Spaan, and Nikos Vlassis. Non-communicative multi-robot coordination in dynamic environments. *Robotics and Autonomous Systems*, 50(2-3):99–114, February 2005.

Benjamin Kuipers. The spatial semantic heirarchy. *Artifical Intelligence*, 119:191–233, 2000.

Vadym Kyrylov, Martin Greber, and David Bergman. Multi-criteria optimization of ball passing in simulated soccer. *Journal of Multi-Criteria Decision Analysis*, 13:103–113, 2005.

Aristid Lindenmayer. Mathematical models for cellular interaction in development parts I and II. *Journal of Theoretical Biology*, 18:280–299 and 300–315, 1968.

Alan Mackworth. Agents, bodies, constraints, dynamics, and evolution. *AI Magazine*, 30(1):7–28, Spring 2009.

Andrew P. Martin. Increasing genomic complexity by gene duplication and the origin of vertebrates. *The American Naturalist*, 154(2):111–128, 1999.

Jan FH. Metzen, Mark Edgington, Yohannes Kassahun, and Frank Kirchner. Performance evaluation of EANT in the robocup keepaway benchmark. In *ICMLA '07: Proceedings of the Sixth International Conference on Machine Learning and Applications*, pages 342–347, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3069-9. doi: http://dx.doi.org/10.1109/ICMLA.2007.80. URL `http://dx.doi.org/10.1109/ICMLA.2007.80`.

Eduardo Morales. Scaling up reinforcement learning with a relational representation. In *Proceedings of the Workshop on Adaptability in Multi-agent Systems (AORC-2003)*, pages 15–26, Sydney, Austrailia, January 2003.

Sinno Pan and Qiang Yang. A survey on transfer learning. Technical Report HKUST-CS08-08, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, November 2008.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, April 1994.

Jan Ramon, Kurt Driessens, and Tom Croonenborghs. Transfer learning in reinforcement learning problems through partial policy recycling. In *Proceedings of the 18th European Conference on Machine Learning*, pages 699–707, Berlin, Germany, 2007. Springer-Verlag.

Gavin A. Rummery and Mahesan Niranjan. On-line Q-learning using connectionist systems. CUED/F-INFENG/TR 166, Cambridge University Engineering Department, 1994.

Natarajan Saravanan and David B. Fogel. Evolving neural control systems. *IEEE Expert*, pages 23–27, June 1995.

Jurgen Schmidhuber and Fakultat Fur Informatik. On learning how to learn learning strategies. Technical report, Fakultat fur Informatik, Technische Universitat Munchen. Revised, 1994.

Kenneth O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines Special Issue on Developmental Systems*, 8(2):131–162, 2007.

Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127, 2002.

Kenneth O. Stanley and Risto Miikkulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

Kenneth O. Stanley, Bobby D. Bryant, and Risto Miikkulainen. Real-time neuroevolution in the NERO video game. *IEEE Transactions on Evolutionary Computation Special Issue on Evolutionary Computation and Games*, 9(6):653–668, 2005.

Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life*, 15(2), 2009.

Frieder Stolzenburg, Jan Murray, and Karsten Sturm. Multiagent matching algorithms with and without coach. *Journal of Decision Systems*, 15(2-3):215–240, 2006. Special issue on Decision Support Systems. Guest editors: Fatima C. C. Dargam and Pascale Zarate.

Peter Stone and Richard S. Sutton. Scaling reinforcement learning to robocup soccer. In *The Eighteenth International Conference on Machine Learning*, pages 537–544, New York, NY, June 2001. ICML 2001, ACM.

Peter Stone, Richard S. Sutton, and Satinder Singh. Reinforcement learning in 3 vs. 2 keepaway. In Peter Stone, T. Balch, and G. Kraetszchmar, editors, *Robocup-2000: Robot soccer world cup IV*, pages 249–258. Springer Verlag, Berlin, 2001.

Peter Stone, Richard S. Sutton, and Gregory Kuhlmann. Reinforcement learning for RoboCup-soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

Peter Stone, Gregory Kuhlmann, Matthew E. Taylor, and Yaxin Liu. Keepaway soccer: From machine learning testbed to benchmark. In *RoboCup-2005: Robot Soccer World Cup IX*, pages 93–105. Springer Verlag, 2006.

Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Richard S. Sutton. Learning to predict by the methods of temporal differences. In *Machine Learning*, pages 9–44, 1988.

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.

Prasad Tadepalli. Learning to solve problems from exercises. *Computational Intelligence*, 4(24): 257–291, 2008.

Prasad Tadepalli, Robert Givan, and Kurt Driessens. Relational reinforcement learning: An overview. In *International Conference on Machine Learning Workshop on Relational Reinforcement Learning*, New York, NY, 2004. ACM Press.

Erik Talvitie and Satinder Singh. An experts algorithm for transfer learning. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 1065–1070, 2007.

Matthew E. Taylor and Peter Stone. Cross-domain transfer for reinforcement learning. In *Proceedings of the 24th International Conference on Machine learning*, pages 879–886, New York, NY, USA, 2007. ACM.

Matthew E. Taylor, Shimon Whiteson, and Peter Stone. Comparing evolutionary and temporal difference methods in a reinforcement learning domain. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006)*, pages 1321–1328, New York, NY, July 2006. ACM Press.

Matthew E. Taylor, Peter Stone, and Yaxin Liu. Transfer learning vis inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 1(8):2125–2167, September 2007a.

Matthew E. Taylor, Shimone Whiteson, and Peter Stone. Transfer via intertask mappings in policy search reinforcement learningn. In *The Autonomous Agents and Multi-Agent Systems Conference*, New York, NY, May 2007b. AAMAS-2007, ACM Press.

Gerald Tesauro. Practical issues in temproal difference learning. *Machine Learning*, 8(3-4):257–277, May 1992.

Sebastian Thrun and Tom M. Mitchell. Learning one more thing. Technical report, Carnegie Mellon University, 1994.

Lisa Torrey, Jude W. Shavlik, Trevor Walker, and Richard Maclin. Rule extraction for transfer learning. In *Rule Extraction from Support Vector Machines*, pages 67–82. Springer-Verlag, Berlin, Germany, 2008a.

Lisa Torrey, TrevorWalker, Richard Maclin, and Jude Shavlik. Advice taking and transfer learning: Naturally inspired extensions to reinforcement learning. In *AAAI Fall Symposium on Naturally Inspired AI*, Washington, DC, 2008b. AAAI Press.

Alan Turing. The Chemical Basis of Morphogenesis. *Royal Society of London Philosophical Transactions Series B*, 237:37–72, August 1952.

James D. Watson, Nancy H. Hopkins, Jeffrey W. Roberts, Joan A. Steitz, and Alan M. Weiner. *Molecular Biology of the Gene Fourth Edition*. The Benjamin Cummings Publishing Company, Inc., Menlo Park, CA, 1987.

Shimon Whiteson. Improving reinforcement learning function approximators via neuroevolution. In *AAMAS '05: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1386–1386, New York, NY, USA, 2005. ACM. ISBN 1-59593-093-0. doi: http://doi.acm.org/10.1145/1082473.1082794.

Shimon Whiteson and Daniel Whiteson. Stochastic optimization for collision selection in high energy physics. In *IAAI 2007: Proceedings of the Nineteenth Annual Innovative Applications of Artificial Intelligence Conference*, Vancouver, British Columbia, Canada, July 2007. AAAI Press.

Shimon Whiteson, Nate Kohl, Risto Miikkulainen, and Peter Stone. Evolving soccer keepaway players through task decomposition. *Mach. Learn.*, 59(1-2):5–30, 2005.

Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

Journal of Machine Learning Research 11 (2010) 1771-1798 Submitted 5/09; Revised 1/10; Published 5/10

Bayesian Learning in Sparse Graphical Factor Models via Variational Mean-Field Annealing

Ryo Yoshida YOSHIDAR@ISM.AC.JP
Department of Statistical Modeling
Institute of Statistical Mathematics
Tachikawa, Tokyo 190-8562, Japan

Mike West MW@STAT.DUKE.EDU
Department of Statistical Science
Duke University
Durham, NC 27708-0251, USA

Editor: Michael Jordan

Abstract

We describe a class of sparse latent factor models, called graphical factor models (GFMs), and relevant sparse learning algorithms for posterior mode estimation. Linear, Gaussian GFMs have *sparse, orthogonal* factor loadings matrices, that, in addition to sparsity of the implied covariance matrices, also induce conditional independence structures via zeros in the implied precision matrices. We describe the models and their use for robust estimation of sparse latent factor structure and data/signal reconstruction. We develop computational algorithms for model exploration and posterior mode search, addressing the hard combinatorial optimization involved in the search over a huge space of potential sparse configurations. A mean-field variational technique coupled with annealing is developed to successively generate "artificial" posterior distributions that, at the limiting temperature in the annealing schedule, define required posterior modes in the GFM parameter space. Several detailed empirical studies and comparisons to related approaches are discussed, including analyses of handwritten digit image and cancer gene expression data.

Keywords: annealing, graphical factor models, variational mean-field method, MAP estimation, sparse factor analysis, gene expression profiling

1. Introduction

Bayesian sparse modelling in multivariate analysis is of increasing interest in applications as diverse as life science, economics and information science, and is driving a need for effective computational methods for learning model structure, that is, sparse configurations. Parallel developments of sparse latent factor models (e.g., West, 2003; Griffiths and Ghahramani, 2006; Lucas et al., 2006; Wang et al., 2007; Archambeau and Bach, 2009; Carvalho et al., 2008; Guan and Dy, 2009; Rai and Daumé, 2009) and inherently sparsely structured graphical models (e.g., Jordan, 1999, 2004; Dobra et al., 2004; Jones et al., 2005; Carvalho and West, 2007) have explored Bayesian computations using a range of stochastic and deterministic search methods. With a view to scaling to higher dimensions and identification of regions of interest in model structure space, efficient and effective computation remains a challenge. We describe a previously undeveloped class of sparse graphical factor models (GFMs)—a subclass of linear, Gaussian latent factor models with sparse factor loadings that also induce sparse conditional independencies. In this context, we develop a compu-

tational technique for posterior mode evaluation using a hybrid of variational mean-field method (Attias, 1999; Wainwright and Jordan, 2008) and annealing-based optimization.

As a previously unexplored class of sparse (linear, Gaussian) factor models, the intrinsic graphical structure of the GFM arises from use of an orthogonal factor loadings matrix and appropriate scaling of its columns, together with the usual diagonal covariance matrix for latent factors (with no loss of generality). We show that this generally induces zero elements in the precision matrix of the GFM, as well as the covariance matrix. Particularly, the zero entries in the covariance matrix have corresponding zeros in the precision matrix. We also show that covariance matrices of fitted values (i.e., "data reconstructions") from such a model have the same sparse structure, and demonstrate aspects of robustness of the model in inferring variable-latent factor relationships in the presence of outliers. These properties are not shared in general by sparse factor models that lack the graphical structure on variables, nor of course by non-sparse approaches. These intrinsic properties of the GFM, along with relationships with earlier studies on sparse factor analyses, are discussed in Section 2.

Our *variational mean-field annealing algorithm (VMA2)* addresses the combinatorial optimization involved in aiming to compute approximate posterior modes for GFM parameters in the context of the huge space of zero/non-zero potential patterns in factor loadings. Using a prescribed schedule of decreasing temperatures, VMA2 successively generates tempered "artificial" posteriors that, at the limiting zero temperature, yield posterior modes for both GFM parameters and the 0/1 loadings indicators. Defined via an artificial, dynamic regularization on the posterior entropy of configured sparse structures, VMA2 is developed in Section 3.

Section 4 provides additional algorithmic details, including prior modelling for evaluating degree of sparseness, and a stochastic variant of VMA2 for higher-dimensional problems is described in Section 5. Performance and comparisons on artificial data appear in Section 6. Section 7 summarizes extensive, detailed empirical comparisons with related approaches in analyses of hand-written digit images and cancer gene expression data. Section 8 concludes with brief additional comments. A range of detailed supplementary materials, extended discussion on the gene expression studies and `R` code, is accessible from `http://daweb.ism.ac.jp/~yoshidar/anneals/`.

2. Sparse Graphical Factor Models

We describe the GFM with some intrinsic graphical properties, followed by connections to previously developed classes of sparse latent factor analyses.

2.1 GFM Form

Observed sample vectors $x_i \in \mathbb{R}^p$ in p dimensional feature space are each linearly related to independent, unobserved Gaussian latent factor vectors $\lambda_i \in \mathbb{R}^k$ with additional Gaussian noise. We are interested in sparse variable-factor relationships so that the bipartite mapping $\lambda \to x$ is sparse, with the underlying $p \times k$ matrix of coefficients—the *factor loadings matrix*—having a number of zero elements; the $p \times k$ binary matrix Z defines this *configured sparsity pattern*. We use a sparse, orthogonal loading matrix and diagonal covariance matrices for both latent factors and residuals; the model is mathematically identified in the usual sense in factor analysis (Anderson, 2003).

With Z as the $p \times k$ binary matrix with elements z_{gj} such that variable g is related to factor j if and only if $z_{gj} = 1$, the GFM is

$$x_i = \Psi^{1/2}\Phi_Z\lambda_i + \nu_i \quad \text{with } \lambda_i \sim \mathcal{N}(\lambda_i|0,\Delta) \text{ and } \nu_i \sim \mathcal{N}(\nu_i|0,\Psi)$$

where: *(a)* the factor loading matrix $\Psi^{1/2}\Phi_Z$ has $\Phi_Z \equiv \Phi \circ Z$ with $\circ$ representing element-wise product; *(b)* Φ_Z is orthogonal, that is, $\Phi_Z'\Phi_Z = I_k$; *(c)* the factors have diagonal covariance matrix $\Delta = \text{diag}(\delta_1,\ldots,\delta_k)$; and *(d)* the idiosyncratic Gaussian noise (or residual) ν_i is independent of λ_i and has covariance matrix $\Psi = \text{diag}(\psi_1,\ldots,\psi_p)$. The implied covariance matrix of the sampling model, Σ, and the corresponding *precision matrix*, Σ^{-1}, are

$$\Sigma = \Psi^{1/2}\{I + \Phi_Z\Delta\Phi_Z'\}\Psi^{1/2} \quad \text{and} \quad \Sigma^{-1} = \Psi^{-1/2}\{I - \Phi_Z T\Phi_Z'\}\Psi^{-1/2} \tag{1}$$

where $T = \text{diag}(\tau_1,\ldots,\tau_k)$ with $\tau_j = \delta_j/(1+\delta_j)$ $(j = 1:k)$. In general, sparse loading matrices induce some zero elements in the covariance matrix whether or not they are orthogonal, but *not* in the implied precision matrix. In the GFM here, however, a sparse factor model also induces off-diagonal zeros in Σ^{-1}. Zeros in the precision matrix defines a conditional independence or graphical model, hence the GFM terminology. In (1), the pattern of sparsity (location of zero entries) in the covariance and precision matrices are the same. The set of variables associated with one specific factor forms a clique in the induced graphical model, with sets of variables that have non-zero loadings on any two factors lying in the separating subgraph between the corresponding cliques. Hence, we have a natural and appealing framework in which sparse factor models and graphical models are reconciled and consistent.

2.2 Some Model Attributes

In general, a non-orthogonal factor model with the sparse loading matrix W—a sparse extension of probabilistic PCA (Bishop, 1999, 2006)—has the form

$$x_i = W\lambda_i + \nu_i \text{ with } \lambda_i \sim N(0,I) \text{ and } \nu_i \sim N(0,\Psi).$$

The GFM arises when a singular value decomposition is applied to the scaled-factor loading matrix $\Psi^{-1/2}W = \Phi_Z\Delta^{1/2}R$ with a $k \times k$ orthogonal matrix R being removed. This non-orthogonal model defines a Bayes optimal reconstruction of the data via the fitted values (or extracted signal)

$$\hat{x}(x_i) := W\mathbb{E}[\lambda_i|x_i] = WW'(WW'+\Psi)^{-1}x_i.$$

Then, asymptotically,

$$\frac{1}{n}\sum_{i=1}^{n}\hat{x}(x_i)\hat{x}(x_i)' \xrightarrow{p} \mathbb{C}ov[\hat{x}(x_i)] = WW'(WW'+\Psi)^{-1}WW'$$

and this is generally a non-sparse matrix (no zero entries) even though W is sparse. This is an inconsistency in the sense that data reconstructions should be expected to share the dominant patterns of covariance sparsity evident in the original covariance matrix $\mathbb{C}ov[x_i] = WW'+\Psi$. In the GFM, however, $\mathbb{C}ov[\hat{x}(x_i)] = \Psi^{1/2}\Phi_Z G\Phi_Z'\Psi^{1/2}$ where G is diagonal with entries $\delta_j^2/(1+\delta_j)$. In such cases, $\mathbb{C}ov[\hat{x}(x_i)]$ is sparse and shares the same 0 elements as $\mathbb{C}ov[x_i]$.

Another feature of the GFM is related to a robust property acquired by the implied graphical structure. Consider an example of 4 variables $x_i' = (x_{i1},x_{i2},x_{i3},x_{i4})$ and 2 factors $\lambda_i' = (\lambda_{i1},\lambda_{i2})$

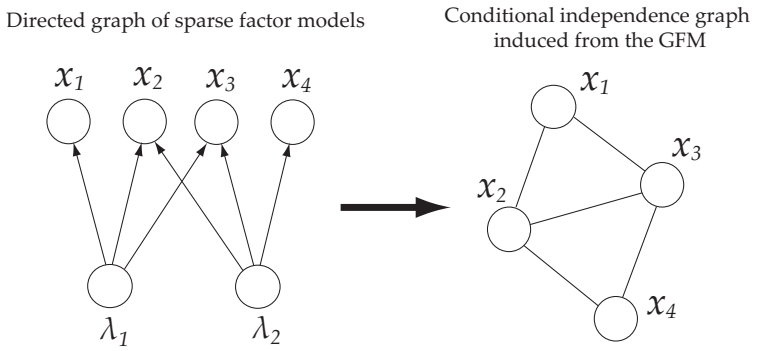


Figure 1: Graphical model structure of an example GFM.

with two cliques in the conditional independence graph; $\{x_{i1}, x_{i2}, x_{i3}\} \leftarrow \lambda_{i1}$ and $\{x_{i2}, x_{i3}, x_{i4}\} \leftarrow \lambda_{i2}$ (see Figure 1). The graph defines the decomposition of the joint density $p(x_{i1}, x_{i2}, x_{i3}, x_{i4}) = p(x_{i1}|x_{i2}, x_{i3})p(x_{i2}, x_{i3}|x_{i4})p(x_{i4})$ or $p(x_{i1}, x_{i2}, x_{i3}, x_{i4}) = p(x_{i4}|x_{i2}, x_{i3})p(x_{i2}, x_{i3}|x_{i1})p(x_{i1})$. This implies that presence of one or more outliers in the isolated feature variable, that is, x_{i1} or x_{i4}, associated with a single factor clique, has no effect on the variables, x_{i4} or x_{i1}, once the intermediate variables x_{i2} and x_{i3} are given. Then, the parameters involved in $p(x_{i1})$ or $p(x_{i4})$, for instance, the loading components and the noise variances corresponding to the isolated variable, can be estimated independently of the impact of outliers in x_{i4} or x_{i1}. The numerical experiment shown in Section 7.1 highlights this robustness property in terms of data compression/restoration tasks, with comparison to other sparse factor models.

2.3 Likelihood, Priors and Posterior

Denote by Θ the full set of parameters $\Theta = \{\Phi, \Delta, \Psi\}$. Our computations aim to explore model structures Z and corresponding posterior modes of parameters Θ under the posterior $p(Z, \Theta|X)$ using specified priors and based on the n observations forming the columns of the $p \times n$ data matrix X.

2.3.1 LIKELIHOOD FUNCTION

The likelihood function is

$$p(X|Z,\Theta) \propto |\Psi|^{-n/2}|I-T|^{n/2}\text{etr}(-S\Psi^{-1}/2 + \Psi^{-1/2}S\Psi^{-1/2}\Phi_Z T\Phi_Z'/2) \tag{2}$$

where $\text{etr}(A) = \exp(\text{trace}(A))$ for any square matrix A, and S is the sample sum-of-square matrix $S = XX'$ with elements s_{gh}. In (2), the factor loadings appear only in the last term and form the important statistic

$$\text{trace}(\Psi^{-1/2}S\Psi^{-1/2}\Phi_Z T\Phi_Z') = \sum_{j=1}^{k} \tau_j \phi_{z_j}' \Psi^{-1/2}S\Psi^{-1/2}\phi_{z_j}$$

where ϕ_{z_j} is column j of Φ_Z, or $\phi_{z_j} = \phi_j \circ z_j$ where ϕ_j is column j of Φ and z_j is column j of Z.

2.3.2 PRIORS ON Θ AND Z

Priors over non-zero factor loadings may reflect substantive *a priori* knowledge if available, and will then be inherently context specific. For examples here, however, we use uniform priors $p(\Theta|Z)$ for exposition. Note that, on the critical factor loadings elements Φ, this involves a uniform on the hypersphere defined by the orthogonality constraint that is then simply conditioned (by setting implied elements of Φ to zero) as we move across candidate models Z.

Concerning the sparse structure Z, we adopt independent priors on the binary variates z_{gj} with $\text{logit}(\Pr(z_{gj}=1|\zeta_{gj})) = -\zeta_{gj}/2$ where $\text{logit}(p) = \log(p/(1-p))$ and the parameters ζ_{gj} are assigned hyperpriors and included in the overall parameter set in later. Beta priors are obvious alternatives to this; the logit leads to a minor algorithmic simplification, but otherwise the choice is arbitrary. Using beta priors can be expected to lead to modest differences, if any of practical relevance, in many cases, and users are free to explore variants. The critical point is that including Bayesian inference on these $p \times k$ sparsity-determining quantities leads to "self-organization" as their posterior distributions concentrate on larger or smaller values. Examples in Section 6 highlight this.

2.4 MAP Estimation for (Θ, Z) in GFMs

Conditional on the $p \times k$ matrix of sparsity control hyperparameters ζ whose elements are the ζ_{gj}, it follows that posterior modes (Z, Θ) maximize

$$\begin{aligned} 2\log p(Z,\Theta|X,\zeta) &= 2\log p(\Theta|Z) - \sum_{g=1}^{p}\sum_{j=1}^{k} z_{gj}\zeta_{gj} - \sum_{g=1}^{p}(n\log\psi_g + s_{gg}\psi_g^{-1}) \\ &\quad + \sum_{j=1}^{k}(n\log(1-\tau_j) + \tau_j \phi_{zj}'\Psi^{-1/2}S\Psi^{-1/2}\phi_{zj}). \end{aligned} \tag{3}$$

The first two terms in (3) arise from the specified priors for Θ and Z, respectively. The quadratic form in the last term is $\phi_{zj}'\Psi^{-1/2}S\Psi^{-1/2}\phi_{zj} = \phi_j' S(z_j,\Psi)\phi_j$ for each j, where the key $p \times p$ matrices $S(z_j,\Psi)$ have elements $(S(z_j,\Psi))_{gh}$ given by

$$(S(z_j,\Psi))_{gh} = z_{gj}z_{hj}s_{gh}(\psi_g\psi_h)^{-1/2}, \qquad \text{for } g,h = 1:p. \tag{4}$$

The (relative) signal-to-noise ratios $\tau_j = \delta_j/(1+\delta_j)$ control the roles played by the last term in (3).

Optimizing (3) over Θ and Z involves many discrete variables and the consequent combinatorial computational challenge. Greedy hill-climbing approaches will get stuck at improper local solutions, often and quickly. The VMA2 method in Section 3 addresses this.

2.5 Links to Previous Sparse Factor Modelling and Learning

In the MAP estimation defined by (3), there are evident connections with traditional sparse principal component analyses (sparse PCA; Jolliffe et al., 2003, Zou et al., 2006 and d'Aspremont et al., 2007). If $\Psi = I$ and $\Delta = I$, the latter likelihood component in (3) is the pooled-variance of projections, that is, $\sum_{j=1}^{k}\phi_j' S(z_j,I)\phi_j$, constructed by the k sparse loading vectors. This is the central statistic optimized in many sparse PCAs. Differences among existing sparse PCAs arise in the way they regulate degrees of sparseness and whether or not orthogonality is imposed on the loading vectors.

The direct sparse PCA of d'Aspremont et al. (2007) imposes an upper-bound $d > 0$ on the cardinality of z_j (the number of non-zero elements), with a resulting semidefinite programming of computational complexity $O(p^4\sqrt{log(p)})$. The applicability of that approach is therefore limited to problems with p rather small. Such cardinality constraints can be regarded as suggestive of structure for the prior distribution on ζ in our model.

The SCoTLASS algorithm of Jolliffe et al. (2003) uses ℓ_1-regularization on loading vectors, later extended to SPCA using elastic nets by Zou et al. (2006). Recently, Mairal et al. (2009) presented a ℓ_1-based dictionary learning for sparse coding in which the method aims to explore sparsity on factor-sample mapping rather than that on factor-variable relations. Setting Laplace-like prior distributions on scale loadings is a counterpart of ℓ_1-based penalization (Jolliffe et al., 2003; Zou et al., 2006). However, our model-based perspective aims for a more probabilistic analysis, with advantages in probabilistic assessment of appropriate dimension of the latent factor space as well as flexibility in the determination of effective degrees of sparseness via the additional parameters ζ. Other than the preceding studies, ℓ_1-regularizations have widely been employed to make sparse latent factor analyses. Archambeau and Bach (2009) developed a general class of sparse latent factor analyses involving sparse probabilistic PCA (Guan and Dy, 2009) and a sparse variant of probabilistic canonical correlation analysis. A key idea of Archambeau and Bach (2009) is to place the automatic relevance determination (ARD) prior of Mackay (1995) on each loading component, and to apply a variational mean-field learning method.

Key advances in Bayesian sparse factor analysis build on non-parametric Bayesian modelling in Griffiths and Ghahramani (2006) and Rai and Daumé (2009), and developments in Carvalho et al. (2008) stemming from the original sparse Bayesian models in West (2003). Carvalho et al develop MCMC and stochastic search methods for posterior exploration. MCMC posterior sampling can be effective but is hugely challenged as the dimensions of data and factor variables increase. Our focus here is MAP evaluation with a view to scaling to increasingly large dimensions, and we leave open the opportunities for future work on MCMC methods in GFMs.

Most importantly, as remarked in Section 2.2, the GFM differs from some of the forgoing models in the conditional independence graphical structures induced. This characteristic contributes to preserving sparse structure in the data compression/reconstruction process and also to the outlier robustness issue. We leave further comparative discussion to Section 7.1, where we evaluate some of the foregoing methods relative to the sparse GFM analysis in an image processing study.

3. Variational Mean-Field Annealing for MAP Search

Finding MAP estimates of the augmented posterior distribution (3) involves many discrete variables z_{gj}. Then, commonly applied search methods such as greedy hill-climbing algorithm often get stuck in improper local solutions. Here, we present a general framework of VMA2 enabling us to escape local mode traps by exploiting annealing.

3.1 Basic Principle

Relative to (3), consider the class of extended objective functions

$$\mathcal{G}_T(\Theta,\omega) = \sum_{Z\in\mathcal{Z}} \omega(Z)\log p(X,Z,\Theta|\zeta) - T\sum_{Z\in\mathcal{Z}} \omega(Z)\log\omega(Z) \tag{5}$$

where $\omega(Z)$—the *sparsity configuration probability*—represents *any distribution* over $Z \in \mathscr{Z}$ that may depend on (X, Θ, ζ), and where $T \geq 0$. This modifies the original criterion (3) by taking the expectation of $p(X, Z, \Theta|\zeta)$ with respect to $\omega(Z)$—the expected complete data log-likelihood in the context of EM algorithm—and by the inclusion of Shannon's entropy of $\omega(Z)$ with the *temperature* multiplier T.

Now, view (5) as a criterion to maximize over (Θ, ω) jointly for any given T. The following is a key result:

Proposition 1 *For any given parameters* Θ *and temperature* T*, (5) is maximized with respect to* ω *at*

$$\omega_T(Z) \propto p(Z|X, \Theta, \zeta)^{1/T}. \tag{6}$$

Proof See the Appendix. ■

For any given Θ, a large T leads to $\omega_T(Z)$ being rather diffuse over sparse configurations Z so that iterative optimization—alternating between Θ and ω—will tend to move more easily and freely around the high-dimensional space Z. This suggests annealing beginning with the temperature T large and successively reducing towards zero. We note that:

- As $T \to 0$, $\omega_T(Z)$ converges to a distribution degenerate at the conditional mode $\hat{Z}(\Theta, \zeta)$ of $p(Z|X, \Theta, \zeta)$, so that
- joint maximization of $\mathcal{G}_T(\Theta, \omega)$ would approach the *global maximum of the exact posterior* $p(\Theta, Z|X, \zeta)$ as $T \to 0$.

The notion of the annealing operation is to realize a gradual move of successively-generated solutions for Θ and $\omega_T(Z)$, and to escape local mode traps by exploiting annealing. Note that, for any given tempered posterior (6), the expectation in the first term of (5) is virtually impossible to be taken due to the combinatorial explosion. In what follows, we introduce VMA2 as a mean-field technique coupled with the annealing-based optimization to overcome this central computational difficulty.

3.2 VMA2 based on Factorized, Tempered Posteriors

To define and implement a specific algorithm, we constrain the otherwise arbitrary *"artificial configuration probabilities"* ω, and do so using a construction that induces analytic tractability. We specify the simplest, factorized form

$$\omega(Z) = \prod_{g=1}^{p} \prod_{j=1}^{k} \omega(z_{gj}) := \prod_{g=1}^{p} \prod_{j=1}^{k} \omega_{gj}^{z_{gj}} (1 - \omega_{gj})^{1 - z_{gj}}$$

in the same way as conventional Variational Bayes (VB) procedures do. In this GFM context, the resulting optimization is eased using this independence relaxation as it gives rise to tractability in computing the conditional expectation in the first term of (5).

If $T = 1$, and given the factorized ω, the objective function $\mathcal{G}_1$ exactly agrees with the *free energy*, which bounds the posterior marginal as

$$\log \sum_{Z \in \mathscr{Z}} p(X, \Theta, Z|\zeta) \geq \mathcal{G}_1(\Theta, \omega).$$

The lower-bound $\mathcal{G}_1$ is the criterion that the conventional VB methods aim to maximize (Wainwright and Jordan, 2008). This indicates that any solutions corresponding to the VB inference can be obtained by stopping the cooling schedule at $T = 1$ in our method. Similar ideas have, of course, been applied in deterministic annealing EM and annealed VB algorithms (e.g., Ueda and Nakano, 1998). These methods exploit annealing schemes to escape from local traps during coordinate-basis updates in aiming to define variational approximations of posteriors.

Even with this relaxation, maximization over $\omega(Z)$ cannot be done for all elements of Z simultaneously and so is approached sequentially—sequencing through each ω_{gj} in turn while conditioning the others. For any given T this yields the optimizing value given by

$$\omega_{gj}(T) \propto \exp\Big\{\frac{1}{T}\sum_{\mathscr{Z}_{C\setminus\{g,j\}}}\prod_{h\neq g}\prod_{l\neq j}\omega(z_{hl})\log p(z_{gj}=1|X,Z_{C\setminus\{g,j\}},\Theta,\zeta)\Big\} \tag{7}$$

where $\mathcal{C}$ denotes the collection of all indices (g, j) for the p features and k factor variables, $\mathcal{C}\setminus\{g, j\}$ is the set of the paired indices (h,l) such that $(h,l) \neq (g, j)$, and $\mathscr{Z}_{C\setminus\{g,j\}}$ stands for the set of z_{hl}s other than z_{gj}.

Starting with $\omega_{gj} \simeq 1/2$ at an initial large value of T, (7) gradually concentrates to the point mass as T decays to zero slowly:

$$\hat{z}_{gj} := \lim_{T\downarrow 0}\omega_{gj}(T) = \begin{cases} 1, & \text{if } \displaystyle\sum_{\mathscr{Z}_{C\setminus\{g,j\}}}\prod_{h\neq g}\prod_{l\neq j}\omega(z_{hl})\log\frac{p(z_{gj}=1,X,Z_{C\setminus\{g,j\}},\Theta,\zeta)}{p(z_{gj}=0,X,Z_{C\setminus\{g,j\}},\Theta,\zeta)} > 0, \\ 0, & \text{otherwise.} \end{cases}$$

It remains true that, at the limiting zero temperature, the global maximum of $\mathcal{G}_T(\Theta,\omega)$ is the set of $p\times k$ point masses at the global posterior mode of $p(\Theta,Z|X,\zeta)$. This is seen trivially as follows: *(i)* As $T\to 0$, and with the non-factorized ω in (5), we have limiting value

$$\sup_Z \log p(X,\Theta,Z|\zeta) = \sup_\omega \mathcal{G}_0(\Theta,\omega) \tag{8}$$

with the point mass $\omega(Z) = \delta_{\hat{Z}}(Z)$ at the location of the global maximum $(\hat{Z})_{gj} = \hat{z}_{gj}$. Further, *(ii)* any point mass $\delta_{\hat{Z}}(Z)$ is representable by a fully factorized $p\times k$ point masses as $\delta_{\hat{Z}}(Z) = \prod_{g,j}\delta_{\hat{z}_{gj}}(z_{gj})$.

It is stressed that the coordinate-basis updates (7) cannot, of course, guarantee convergence to the *global* optimum even with prescribed annealing. Nevertheless, VMA2 represents a substantial advance in its ability to move more freely and escape local mode traps. We also note the generality of the idea, beyond factor models and also potentially using penalty functions other than entropy.

4. Sparse Learning in Graphical Factor Models

We first provide a specific form of VMA2 for the GFM, and then address the issue of evaluating relevant degrees of sparseness.

4.1 MAP Algorithm

Computations alternate between conditional maximization steps for ω and Θ while reducing the temperature T. At each step, the value of the objective function (5) is kept to refine until convergence where the temperature reaches to zero. Specifically:

1: Set a cooling schedule $\mathcal{T} = \{T_1, \ldots, T_d\}$ of length d where $T_d = 0$;
2: Set ζ;
3: Initialize Θ;
4: Initialize $\omega(Z)$;
5: $i \leftarrow 0$;
6: `while` ($\{$the loop is not converged $\} \wedge \{i \leq d\}$)
7: $i \leftarrow i + 1$;
8: Compute configuration probabilities $\omega_{gj}(T_i)$;
9: Optimize with respect to each column ϕ_j $(j = 1 : k)$ of Φ in turn under full-conditioning;
10: Optimize with respect to Δ under full-conditioning;
11: Optimize with respect to Ψ under full-conditioning;
12: Optimize with respect to ζ under full-conditioning;
13: `end while`

We now summarize key components in the iterative, annealed computation defined above.

4.2 Sparse Configuration Probabilities

First consider maximization with respect to each sparse configuration probability ω_{gj} conditional on all others. We note that the first term in (5) involves the expectation over Z with respect to the probabilities ω, denoted by $\mathbb{E}_\omega[\cdot]$. Accordingly, for the key terms $S(z_j, \Psi)$ we have

$$\mathbb{E}_\omega[S(z_j, \Psi)] = \Omega_j \circ (\Psi^{-1/2} S \Psi^{-1/2}) \text{ with } (\Omega_j)_{gh} = \begin{cases} \omega_{gj}, & \text{if } g = h, \\ \omega_{gj}\omega_{hj}, & \text{otherwise.} \end{cases} \tag{9}$$

Introduce the notation $\Psi^{-1/2} S \Psi^{-1/2} = (s_1(\Psi), \ldots, s_p(\Psi))$ to represent the p columns of the scaled-sample sum-of-square matrix here, and define the $p-$vector

$$\tilde{\omega}_{gj} = (\omega_{1j}, \ldots, \omega_{g-1,j}, 1, \omega_{g+1,j}, \ldots, \omega_{pj})'.$$

Then, the partial derivative of (5) with respect to ω_{gj} conditional on Θ and the other configuration probabilities leads to

$$\text{logit}(\omega_{gj}(T)) = H_{gj}(\zeta_{gj})/T \quad \text{where} \quad H_{gj}(\zeta_{gj}) := \tau_j \phi_{gj} (\phi_j \circ \tilde{\omega}_{gj})' s_g(\Psi) - \zeta_{gj}.$$

This directly yields the conditional maximizer for ω_{gj} in terms of the tempered negative energy $H_{gj}(\zeta_{gj})/T$. As the temperature T is reduced towards zero, the resulting estimate tends towards 0 or 1 according to the sign of $H_{gj}(\zeta_{gj})$.

4.3 Conditional Optimization over Φ

The terms in (5) that involve Φ are simply the expectation of the quadratic forms in the last term of (3), with the term for each column ϕ_j involving the key matrices $S(z_j, \Psi)$ defined in (4), for each $j = 1 : k$. At each step through the overall optimization algorithm in Section 4.1, we sequence through these columns of the loadings matrix in turn conditioning on the previously optimized

values of all other columns. In the context of the overall iterative MAP algorithm, this yields global optimization over Φ as $T \to 0$.

Conditional optimization then reduces to the following: for each $j = 1:k$, sequence through each column ϕ_j in turn and at each step

$$\begin{aligned} &\underset{\phi_j}{\text{maximize}} && \phi_j' \mathbb{E}_\omega[S(z_j, \Psi)]\phi_j \\ &\text{subject to} && \phi_j'\phi_j = 1 \quad \text{and} \quad \phi_m'\phi_j = 0 \text{ for } m \neq j, m = 1:k. \end{aligned} \tag{10}$$

The optimization conditions on the most recently updated values of all other columns $m \neq j$ at each step, and is performed as one sweep as the line 9 in the algorithm of Section 4.1. Column order can be chosen randomly or systematically each time while still maintaining convergence. In this step, we stress that the original orthogonality condition is modified to $\Phi_Z'\Phi_Z = I \to \Phi^T\Phi = I$ in (10). It remains the case that iteratively refined estimates obtained from (10) satisfy the original condition at the limiting zero temperature, yielding sparsity for $\mathbb{E}_\omega[S(z_j, \Psi)]$, as detailed in the mathematical derivations in supplementary material.

The specific computations required for the conditional optimization in (10) are as follows (with supporting details in the Appendix). Note that the central matrices $\mathbb{E}_\omega[S(z_j, \Psi)]$ required here are trivially available from Equation (9).

1: Compute the $p \times (k-1)$ matrix $\Phi_{(-j)} = \{\phi_m\}_{m \neq j}$ by simply deleting column j from Φ;

2: Compute the $p \times p$ projection matrix $N_j = I_p - \Phi_{(-j)}\Phi_{(-j)}'$;

3: Compute the eigenvector φ_j corresponding to the most dominant eigenvalue of $N_j\mathbb{E}_\omega[S(z_j, \Psi)]N_j$;

4: Compute the required optimal vector $\phi_j = N_j\varphi_j / \|N_j\varphi_j\|$.

This procedure solves (10) by optimizing over an eigenvector already constrained by the orthogonality conditions. Here N_j spans the null space of the current $k-1$ columns of $\Phi_{(-j)}$, so $N_j\mathbb{E}_\omega[S(z_j, \Psi)]N_j$ defines the projection of $\mathbb{E}_\omega[S(z_j, \Psi)]$ onto the orthogonal space and eigenvectors φ_j lie in the null space. It remains to ensure that the computed value ϕ_j is of unit length, which involves the normalization in the final step in part 4. Selecting the eigenvector with maximum eigenvalue ensures the conditional maximization in (10).

4.4 Conditional Optimization over Δ

The variances δ_j of the latent factors appear in Equations (3) and (5) in the sum over $j = 1:k$ of terms

$$-n\log(1+\delta_j) + \delta_j(1+\delta_j)^{-1}\phi_j'\mathbb{E}_\omega[S(z_j, \Psi)]\phi_j.$$

This is unimodal in δ_j with maximizing value

$$\hat{\delta}_j = \max\{0, n^{-1}\phi_j'\mathbb{E}_\omega[S(z_j, \Psi)]\phi_j - 1\}, \tag{11}$$

and so the update at the line 10 of the MAP algorithm of Section 4.1 computes these values in parallel for each factor $j = 1:k$. Note that this may generate zero values, indicating the removal of the corresponding factors from the model, and so inducing an intrinsic ability to prune the number

of factors as being redundant in a model specified initially with a larger, encompassing value of k. The configured sparse structure drives this pruning; any specific factor j that is inherently very sparse generates a smaller value of the projected "variance explained" $\phi_j' \mathbb{E}_\omega[S(z_j, \Psi)]\phi_j$, and so can lead to $\hat{\delta}_j = 0$ as a result.

4.5 Conditional Optimization over Ψ

The diagonal noise covariance matrix Ψ appears in the objective function of Equation (5) in terms that can be re-expressed as

$$-n \log|\Psi| - \text{trace}(S\Psi^{-1}) + \sum_{j=1}^{k} \tau_j \text{trace}(\phi_j \phi_j' \Psi^{-1/2} (\Omega_j \circ S) \Psi^{-1/2})$$

where $\tau_j = \delta_j/(1+\delta_j)$ for each j. Differentiating this with respect to $\Psi^{-1/2}$ yields the gradient equation:

$$n\text{diag}^{-1}(\Psi^{1/2}) - \text{diag}^{-1}(S\Psi^{-1/2}) + \sum_{j=1}^{k} \tau_j \text{diag}^{-1}(\phi_j \phi_j' \Psi^{-1/2} (\Omega_j \circ S)) = 0,$$

where $\text{diag}^{-1}(A)$ denotes the vector of the diagonal elements in A. Iterative solution of this non-linear equation in Ψ can be performed via the reduced implicit equation

$$\text{diag}^{-1}(\Psi) = n^{-1}\text{diag}^{-1}(\{I_p - \sum_{j=1}^{k} \tau_j (\phi_j \phi_j') \circ (\Psi^{-1/2} \Omega_j \Psi^{1/2})\} S).$$

4.6 Degrees of Sparseness

The prior over the logistic hyperparameters $\zeta = \{\zeta_{gj}\}$ defining the Bernoulli probabilities for the z_{gj} is important in encouraging relevant degrees of sparseness. Extending the model via an hierarchical prior for these parameters enables adaptation to data in evaluating relevant degrees of sparseness. One first class of priors is used here, taking the ζ_{gj} to be conditionally independent and drawn from the prior with positive part Gaussian distribution $N_+(\zeta_{gj}|\mu,\sigma)$ for some specified mean and variance (μ,σ). The annealing search can now be extended to include ζ, simply embedding conditional optimization of (5) under this prior within each step of the iterative search. The conditional independence structure of the model easily yields unique solutions for each of the ζ_{gj} in parallel as values satisfying

$$\omega_{gj} = \frac{\exp(-\zeta_{gj}/2)}{1+\exp(-\zeta_{gj}/2)} - \frac{\zeta_{gj}-\mu}{2\sigma}. \tag{12}$$

Solutions to (12) are trivially, iteratively computed. Evidently, as ω_{gj} approaches 0 or 1, the solution for ζ_{gj} is shifted to the corresponding boundary. ζ_{gj} as a function of ω_{gj} for several values of (μ,σ).

As mentioned earlier, the choice of this logit/truncated normal prior is a subjective preference and could be replaced by others, such as beta priors. Again, we expect that this would typically lead to modest differences, if any of practical relevance, in many cases.

5. A Stochastic Search Variant for Large p

In problems with larger numbers of variables, the computations quickly become challenging, especially in view of the repeated eigen-decompositions required for updating factor loading matrix. In our examples and experiments, analysis with dimensions $p \sim 500$ would be feasible using our own R code (`vma2gfm()` available from the supplementary web site), but computation time then rapidly increases with increasing p. More efficient low level coding will speed this, but nevertheless it is of interest to explore additional opportunities for increasing the efficiency of the MAP search.

To reduce the computational time, we explore a stochastic variant of the original deterministic VMA2 that uses realized Z matrices from current, conditional configuration probabilities $\omega_{gj}(T)$ at each stage of the search process. The realized binary matrix $Z = [z_1, \ldots, z_k]$ replaces the full matrix $\mathbb{E}_\omega[S(z_j, \Psi)]$ with a sparse alternative $S(z_j, \Psi)$. In larger, very sparse problems, this will enable us to greatly reduce the computing time as each eigen-decomposition can be computed based only on the components related to non-zero z_{gj} values. This leads to a stochastic annealing search with all other steps unchanged. We also have the additional benefit of the introduced randomness aiding in potentially moving away from the stuck in suboptimal solutions. It should be stressed that this is an optional complement to the deterministic algorithm and one that may be used for an initial period of time prior to enable swifter initial iterations from arbitrary initial values, prior to switching to the deterministic annealing once in the region of a posterior mode.

The modified search procedure over ϕ_j in Equation (10) is:

1. Draw a set of binary values $\hat{z}_{gj}, g = 1, \ldots, p$, according to the current configuration probabilities $\omega_{gj}(T)$;
2. Define the set of *active variables* by $\mathcal{A}_j = \{g | g \in 1:p, \hat{z}_{gj} = 1\}$; denote by $\phi_{j,\{\mathcal{A}_j\}}$ the sub-vector of ϕ_j for only the active variables, and $S_{\{\mathcal{A}_j\}}(z_j, \Psi)$ the submatrix of $S(z_j, \Psi)$ whose rows and columns correspond to only the active variables;
3. Solve the reduced optimization conditional on the $\mathcal{A}_j$, via:

$$\begin{aligned} \underset{\phi_{j,\{\mathcal{A}_j\}}}{\text{maximize}} \quad & \phi'_{j,\{\mathcal{A}_j\}} S_{\{\mathcal{A}_j\}}(\hat{z}_j, \Psi) \phi_{j,\{\mathcal{A}_j\}} \\ \text{subject to} \quad & \|\phi_{j,\{\mathcal{A}_j\}}\|^2 = 1 \ \text{ and } \ \phi'_{m,\{\mathcal{A}_j\}} \phi_{j,\{\mathcal{A}_j\}} = 0 \text{ for } m \neq j. \end{aligned}$$

4. Update the full $p-$vector ϕ_j with elements $\phi_{j,\{\mathcal{A}_j\}}$ for the active variables and all other elements zero.

For example, in a problem with $p = 5000$ but sparseness of the order of 5%, the $\mathcal{A}_j$ will involve a few hundred active variables, and eigenvalue decomposition will then be performed on matrices of that order rather than 5000×5000. We note also that this strategy requires a modification to the update operation for the configuration probabilities: the ω_{gj} will be updated at any one step only for the current indices $g \in \mathcal{A}_j$, keeping the remaining z_{gj} at values previously obtained.

6. Experimental Results on Synthetic Data

Performance and comparisons on artificial data are shown to highlight some learning properties of the GFM.

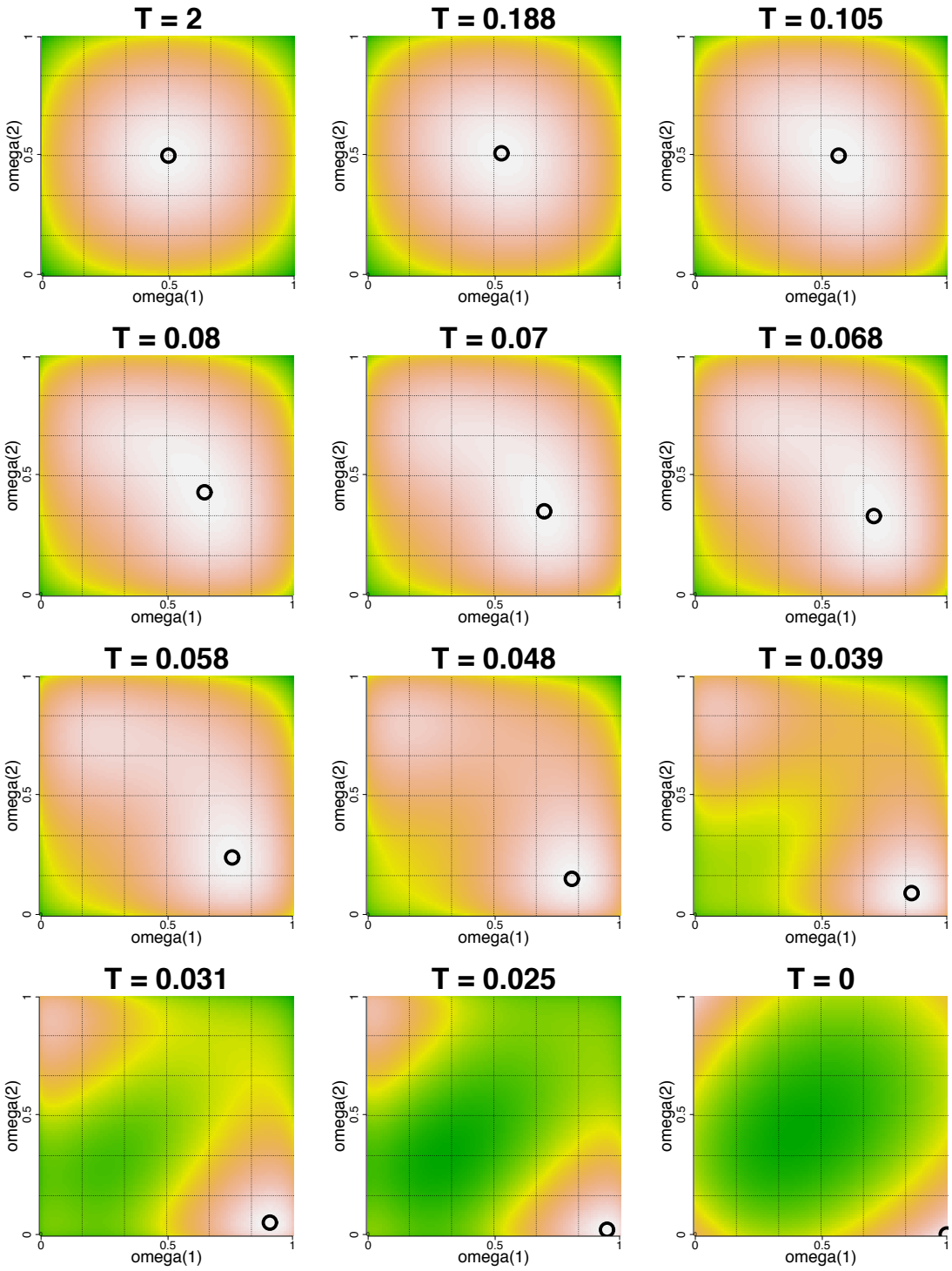


Figure 2: Display of evolving $\mathcal{G}_T(\hat{\Theta}, \omega)$ in the annealing process (from $T = 2$ to $T = 0$) with contour plots. The black circle in each panel indicates the maximum point, and that corresponding to $T = 0$ in the panel on the bottom-right corner indicates the optimal sparse structure.

6.1 Visual Tracking of Annealing Process with a Toy Problem

The first experiment shows how the VMA2 method can solve the combinatorial optimization. Consider 3 variables and 1 factor, so that $x_i = (\phi_1 \cdot z_1)\lambda_{1i} + \nu_i$ where all parameters except ϕ_1 are fixed as $\Psi = I$ and $\Delta = I$. The likelihood function in (2) is then $p(X|Z,\Theta) \propto \exp(\phi'_{z_1} S \phi_{z_1}/2)$. Assume that true edge on $z_{31} = 1$, indicating $x_{i3} \leftarrow \lambda_{i1}$, is known, but $z_{11} = 1$ and $z_{21} = 0$ are treated as unknown. Then, with the prior for z_{11} and z_{21} as $\text{logit}(\Pr(z_{11} = 1)) = \text{logit}(\Pr(z_{21} = 1)) = -1.5$, we explored values for ϕ_1 and ω_{g1}, $g = 1,2$, based on an artificial data set drawn from the GFM, so as to refine $\mathcal{G}_T(\Theta,\omega)$ under the factorized $\omega(Z) = \omega(z_{11})\omega(z_{21})$.

We can map the surface $\mathcal{G}_T(\Theta,\omega)$ over $(\omega_{11},\omega_{21})$ when Θ is set at the optimized value $\hat{\Theta}$ for each $(\omega_{11},\omega_{21})$. Figure 2 on the bottom-right corner displays a contour plot of $\mathcal{G}_0(\hat{\Theta},\omega)$. The maximum point lies in one of the four corners corresponding to $\omega_{g1} \in \{0,1\}$ and the global MAP estimate has $\omega_{11} = 1$ and $\omega_{21} = 0$.

Figure 2 also shows a tracking result of the VMA2 search process starting from $T = 2$ and stopping at $T = 0$. The change in $\mathcal{G}_T(\hat{\Theta},\omega)$ and the corresponding maximizing values of $(\omega_{11},\omega_{21})$ can be monitored through the contour plots at selected temperatures. Starting from the initial values, $\omega_{11} \approx 0.5$ and $\omega_{21} \approx 0.5$, at the highest temperature, the successively-generated maximum points gradually come closer to the global optimum ($\omega_{11} = 1$ and $\omega_{21} = 0$) as the annealing process proceeds. At higher temperatures, $\mathcal{G}_T(\hat{\Theta},\omega)$ is unimodal. In the overall search, the tempered criterion begins to become bimodal after the trajectory moves into regions close to the global maximum.

This simple illustrative example highlights the key to success in the search: moving the trajectory of solutions closer to the global maximum in earlier phases of the cooling schedule, before the tempered criterion function exhibits substantial multimodality. Looking ahead, we may be able to raise the power of the annealing search by, for example, using dynamic control of the cooling schedule or more general penalty functions for ω.

6.2 Snapshot of Algorithm with 30 Variables and 4 Factors

In what follows, we will show some simulation studies to provide insights and comparisons. The data sets have $n = 100$ data points drawn from the GFM with $p = 30$ and $k_{\text{true}} = 4$, and with $\Psi = 0.05I$ and $\Delta = \text{diag}(1.5, 1.2, 1.0, 0.8)$. The z_{gj} were independently generated with $\Pr(z_{gj} = 1) = 0.3$, yielding roughly 70% sparsity; then, non-zero elements of Φ where generated as independent standard normal variates, following which Φ_Z was constrained to orthogonality.

To explore sensitivity to the chosen temperature schedule for annealing, experiments were run using three settings:

- (Log-inverse decay) $T_i = 3/\log_2(i+1)$ for $i = 1,\ldots,6999$, and $T_{7000} = 0$
- (Linear decay) $T_i = 3 - 6 \times 10^3 \times (i-1)$ for $i = 1,\ldots,1999$, and $T_{2000} = 0$
- (Power decay) $T_i = 3 \times 0.99^{-(i-1)}$ for $i = 1,\ldots,1999$, and $T_{2000} = 0$

For each, we evaluated the resulting MAP search in terms of comparison with the true model and computational efficiency, in each case using a model with redundant factor dimension $k = 8$.

6.3 Annealing with Fixed Hyper-parameters

First analyses fixed $\zeta_{gj} = c$ and was run repeatedly across some grid points of $c \in [0,5]$. Figure 3 summarizes the evaluation of the receiver operating characteristics (ROC) for the three cooling

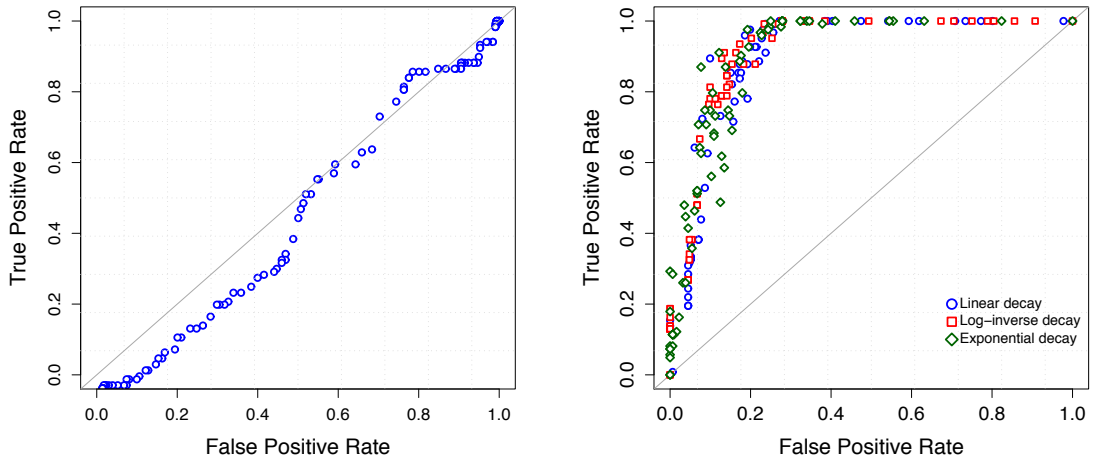


Figure 3: ROC for threshold PCA assuming a known, true $k = 4$ factors (left), compared to VMA2 estimation of the GFM under the three cooling schedules and with $k = 8$ (right). TPR (vertical) and FPR (horizontal) were calculated according to TP/P and FP/N where P and N denote the numbers of non-zero and zero elements in true loadings, TP and FP are the numbers of true positives and false positives, respectively.

schedules. The true positive (TPR) and false positive rates (FPR) were computed based on the correspondences between estimated and true values of the z_{gj}. For comparison, we used standard PCA, extracting the dominant 4 eigenvectors and setting entries below a threshold (in absolute value) to zero; sliding the threshold towards zero gives a range of truncated loadings vectors in the PCA that define the ROC curve for this approach. The resulting ROC curve, shown in the left panel, is very near to the 45° line, comparing very poorly with the annealed GFM; for the latter, each ROC curve indicates rather accurate identification of the sparse structure and the curves differ in small ways only as a function of cooling schedule. The choice of cooling schedule can, however, have a more marked influence on results if initialized at temperatures that are too low.

6.4 Inference on Degrees of Sparseness

A second analysis uses the sparsity prior $p(\zeta_{gj}) = \mathcal{N}_+(\zeta_{gj}|\mu,\sigma)$ with $\mu = 3$ and $\sigma = 6$, and adopts the log-inverse cooling schedule. As shown in the right panel of Figure 4, the analysis realized a reasonable control of FNR (15.4%) and FPR (0%), inducing a slightly less sparse solution than the true structure. The GFM analysis automatically prunes the redundant factors, identifying the true model dimension. Figure 5 displays a snapshot of evolving configuration probabilities ω_{gj} and hyper-parameters ζ_{gj} during the annealing schedule, demonstrating convergence over 2000 steps. At around $T_i \simeq 0.45$, all the configuration probabilities corresponding to the redundant four factors reached to zero.

We further evaluated sensitivity to the choice of cooling schedules; in addition to the previous three cooling schedules, we compared with:

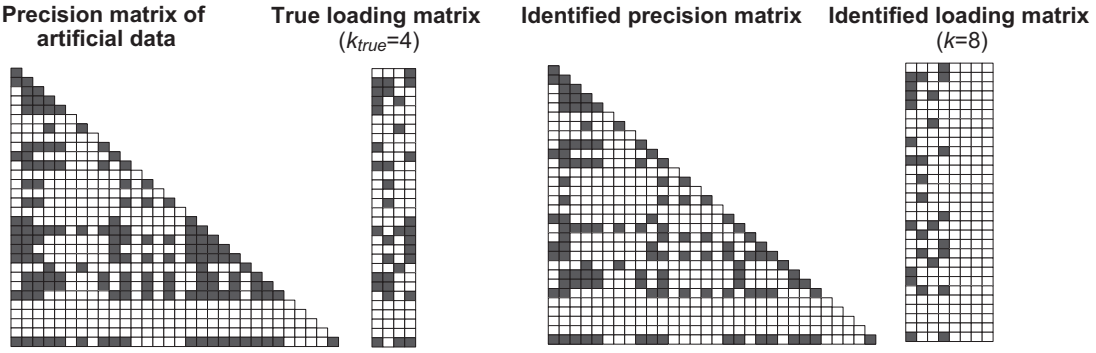


Figure 4: Result of the VMA2 estimation using the log-inverse rate cooling in analysis of synthetic data. (Left) Precision and factor loadings matrix used for generating the synthetic data ($p = 30$, $k_{\text{true}} = 4$). Non-zero elements are colored black. (Right) Estimated precision and factor loadings matrix ($k = 8$); note that the MAP estimate sets the last four loading vectors to zero and so identifies the true number of factors automatically.

- (Log-inverse decay) $T_i = 0.7/\log_2(i+1)$ for $i = 1,\ldots,6999$ and, $T_{7000} = 0$
- (Linear decay) $T_i = 0.7 - 6 \times 10^3 \times (i-1)$ for $i = 1,\ldots,1999$ and, $T_{2000} = 0$
- (Power decay) $T_i = 0.7 \times 0.99^{-(i-1)}$ for $i = 1,\ldots,1999$ and, $T_{2000} = 0$

The initial temperatures are reduced from 3 to 0.7. Figure 6 shows the variations of TPR and FPR in the use of the six cooling schedules, evaluated in 20 analyses with replicated synthetic models and data sets. The left and center panels indicate significant dominance of the annealing starting from the higher initial temperatures. Performance in identifying model structure seriously degrades when using a temperature schedule that starts too low, and the sensitivity to schedule is very limited when beginning with reasonably high initial temperatures.

The right panel in Figure 6 shows TPR and FPR for the sparse PCA (SPCA) proposed by Zou et al. (2006), evaluated on the same 20 data sets using the `R` code `spca()` available at CRAN (`http://cran.r-project.org/`). With `spca()`, we can specify the number of nonzero elements (cardinality) in each column of the factor loading matrix. We executed `spca()` after the assignment of the true cardinality as well as the known factor dimension $k_{\text{true}} = 4$. The figure indicates a better performance of GFM annealed with high initial temperature than the sparse PCA, and this is particularly notable in that the GFM analysis uses $k = 8$ and involves no *a priori* knowledge on the degree of sparseness. It is important to see that the conducted comparison is biased since the data were drawn from the GFM with the orthogonal loading matrix where SPCA does not make orthogonality assumptions. In Section 7.1, we provide deeper comparisons among several existing sparse factor analyses based on image processing in hand-written digits recognition.

6.5 Computing Time Questions

Figure 7 shows the CPU times required for the execution of the GFM analyses as above, repeated with increasing dimension $p \in \{100, 200, 300, 500, 700, 1000\}$. The data sets were again generated from GFMs with 4 factors and roughly 70% sparseness. We then performed the VMA2 using a

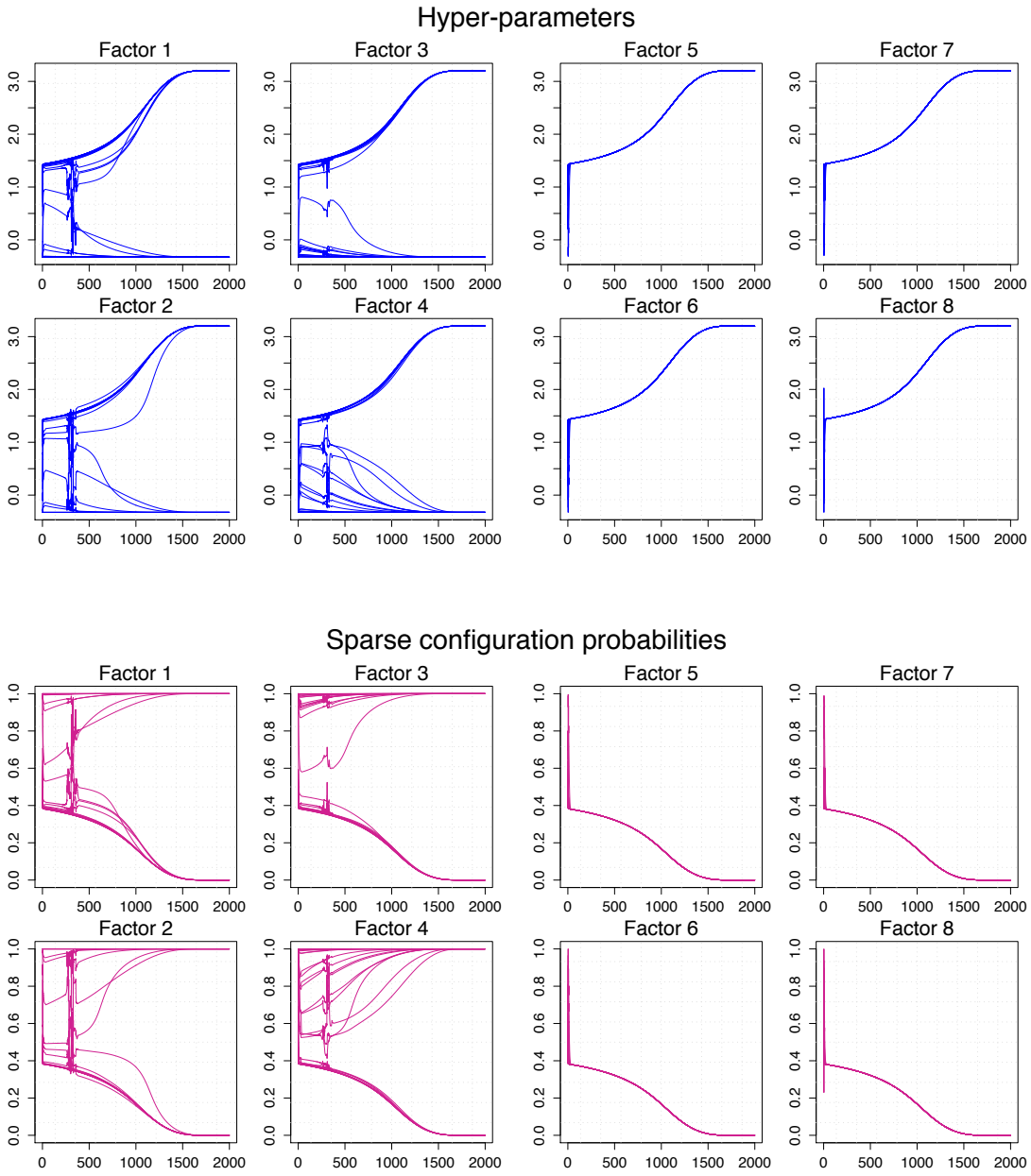


Figure 5: Convergence trajectories of the ζ_{gj} (upper) and ω_{gj} (lower) in analysis of synthetic data over 2000 steps of annealed MAP estimation.

linear decay cooling of length 2000, and using both deterministic and stochastic annealing in a model with $k = 8$. The deterministic algorithm was not used for $p \geq 500$ due to substantial increase in CPU times; this was eased via use of the stochastic search algorithm.

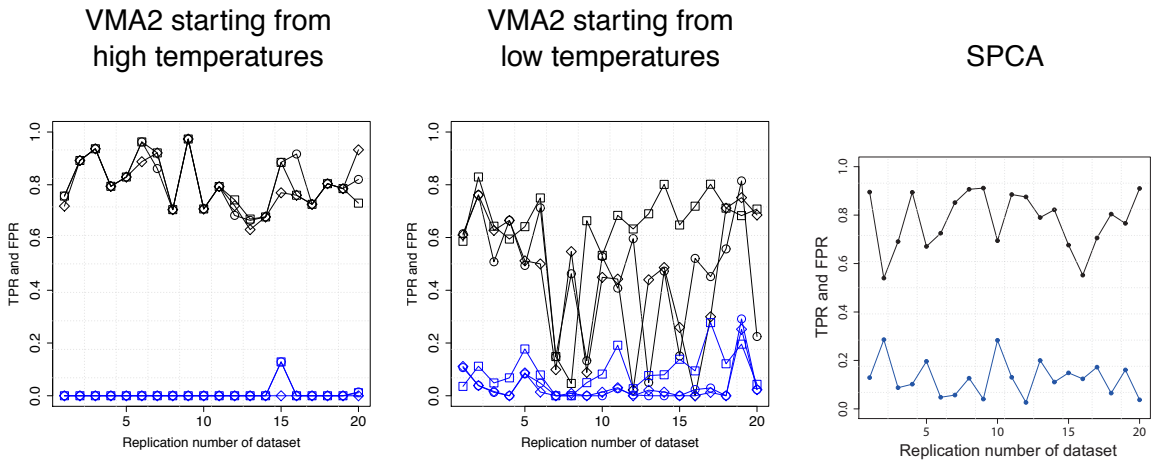


Figure 6: Performance tests on 20 synthetic data sets for different cooling schedules and comparison between the GFM and a sparse PCA (SPCA). For each panel, TPR (black) and FPR (blue) are plotted (vertical axis) against the 20 replicate simulations of artificial data. The results of annealing with the higher and lower initial temperatures are shown in the left and center panels respectively where the rates of cooling with log-inverse, linear and power decays are denoted by box, diamond and circle, respectively. The right panel shows the results of SPCA.

7. Real Data Applications

Experimental results on image analyses of hand-written digits (Section 7.1) and breast cancer gene expression data (Section 7.2) are shown to demonstrate practical relevance of the GFMs in analyses of high dimensional data.

7.1 Application: Hand-written Digit Recognition

We evaluate GFM in pattern recognition analyses of hand-written digit images, and make comparisons to three existing methods; (i) SPCA (Zou et al., 2006), (ii) sparse probabilistic PCA with ARD prior (Archambeau and Bach, 2009), and (iii) MCMC-driven sparse factor analysis (West, 2003; Carvalho et al., 2008). These three methods are all based on models with non-orthogonal sparse loading matrices. The training data set was made from 16×16 gray-scale images of 100 digits (i.e., $n = 100$, $p = 256$) of '3' that were randomly drawn from the postal zip code data available at `http://www-stat.stanford.edu/~tibs/ElemStatLearn/` (Hastie et al., 2001). To evaluate robustness of the four approaches, we added artificial outliers to 15 pixels (features) for about 5% of the original 100 images. Some of the contaminated images are shown in the top-left panels of Figure 8.

For the non-probabilistic method, that is, (i) SPCA, we performed data reconstruction in the standard manner; $x(x_i) = WW'x_i$ with W the matrix of sparse, non-orthogonal loading vectors. In applications of (ii) and (iii) that are inherently driven by probabilistic models, data reconstruction was made via the posterior mean $x(x_i) = W\mathbb{E}[\lambda_i|x_i]$ using obtained sparse loading matrix W. For all

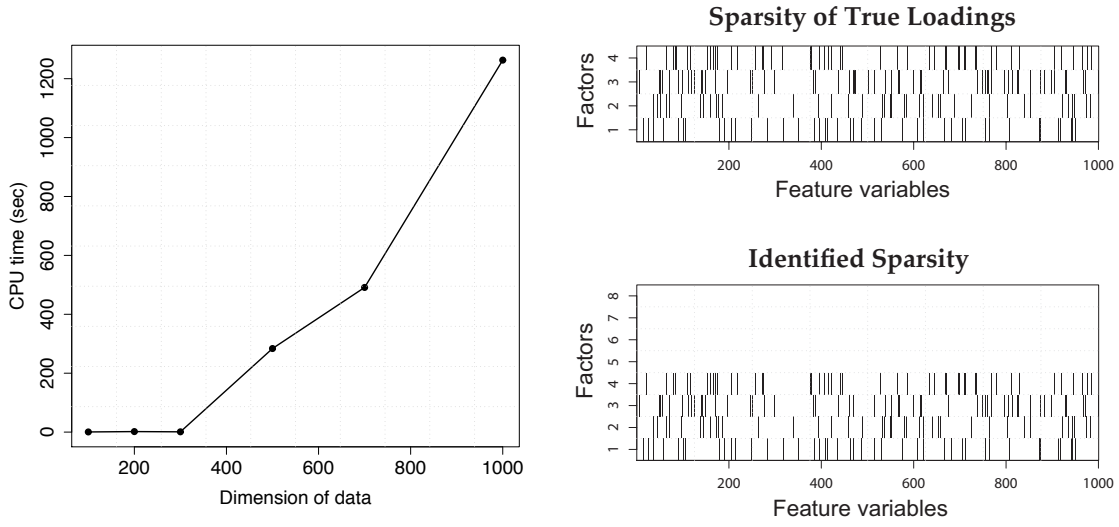


Figure 7: (Left) CPU times (in seconds; Intel(R) Core(TM)2 Duo processor, 2.60Ghz) versus model dimension p for the stochastic VMA2. For the deterministic VMA2, we terminated the tests with the data larger than $p = 300$. Execution times for the deterministic algorithm were approximately 468, 812 and 1100 sec for $p = 100$, 200 and 300. (Right) Identified sparse loadings matrix, displayed as transpose, for the case of $p = 1000$ where the MAP estimation achieved FPR $= 12.0\%$ and FNR $= 18.4\%$.

the methods, setting factor dimensions to $k = 10$, we explored sparse estimates so that the degrees of sparseness become approximately 30% (see Figure 8). For SPCA, we use the same number of non-zero elements in each loading vector as in the estimated GFM. The GFM was estimated using VMA2 with a fixed value for ζ and a linear cooling schedule of length 2000.

A set of 100 test data samples was created from the 100 samples above by adding outliers drawn from a uniform distribution to randomly-chosen pixels with probability 0.2. Performance of the four approaches to data compressions/reconstruction were assessed via mean square error (MSE) between $x(x_i)$s and the true, original test images without the outliers. The right four panels in Figure 8 show some digit images reconstructed by each method with the corresponding original/contaminated test data. The reconstruction errors for the training and test instances are also summarized in the figure. For the results on (ii) and (iii)—the non-orthogonal probabilistic analyses—the reconstructed digits were vaguely-outlined. Such poor reconstructions arise partly from effects of the outliers spread from pixel to pixel along the complete graph defined by non-sparse precision matrix. This empirical result indicates the vulnerability issue of non-restricted sparse factor models in presence of outliers. In the reconstructions of the test instances, the GFM could capture characteristics of original digits with the highest accuracy among the methods. SPCA attained the second highest accuracy in terms of MSE. These observations highlight the substantial merit of using sparse linear mapping in data reconstructions. The GFM and SPCA limit the propagations of outliers within some factor cliques, as most pixel images in the other isolated, non-adjacent factor cliques could be restored clearly.

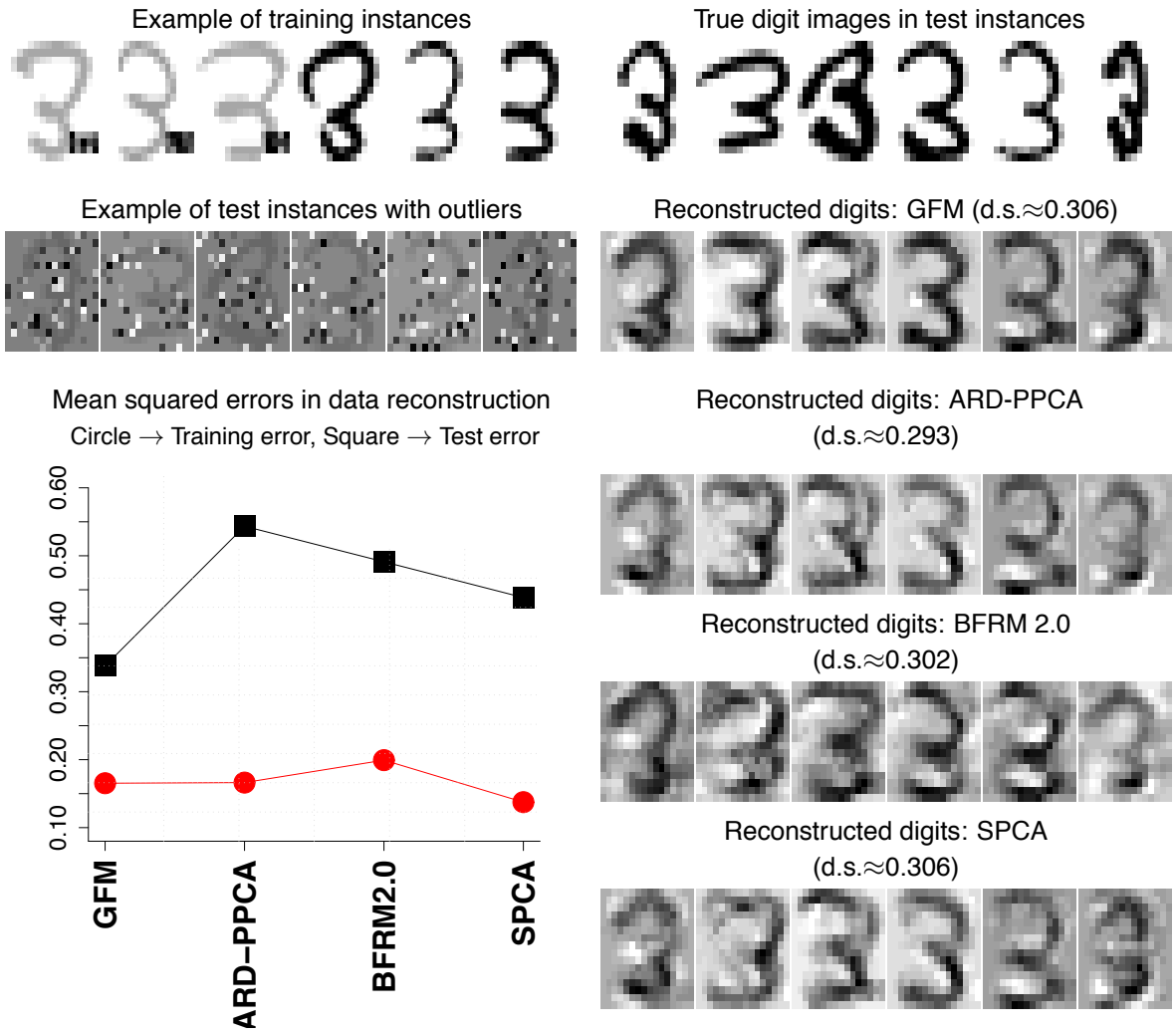


Figure 8: Comparison between GFM and the three alternative methods ((i)-(iii)) in the data reconstruction of outlier hand-written digit images. For the implementation of the sparse probabilistic PCA with ARD prior (ARD-PPCA), we prepared our own `R` function which is available at Supplementary web site. In the application of the MCMC-based sparse factor analysis, we used BFRM 2.0 distributed at http://www.stat.duke.edu/research/software/west/bfrm/. In the four panels on the bottom-right, $d.s.$ denotes the degree of sparseness.

7.2 Application: Breast Cancer Gene Expression Study

Latent factor models are being more used in microarray-based gene expression studies in both basic biological and clinical studies, such as cancer genomics. An example in breast cancer gene expression study here further illustrates the practical relevance of GFM structure and adds to comparisons with other approaches. In addition to the summary details below, a much extended discussion of

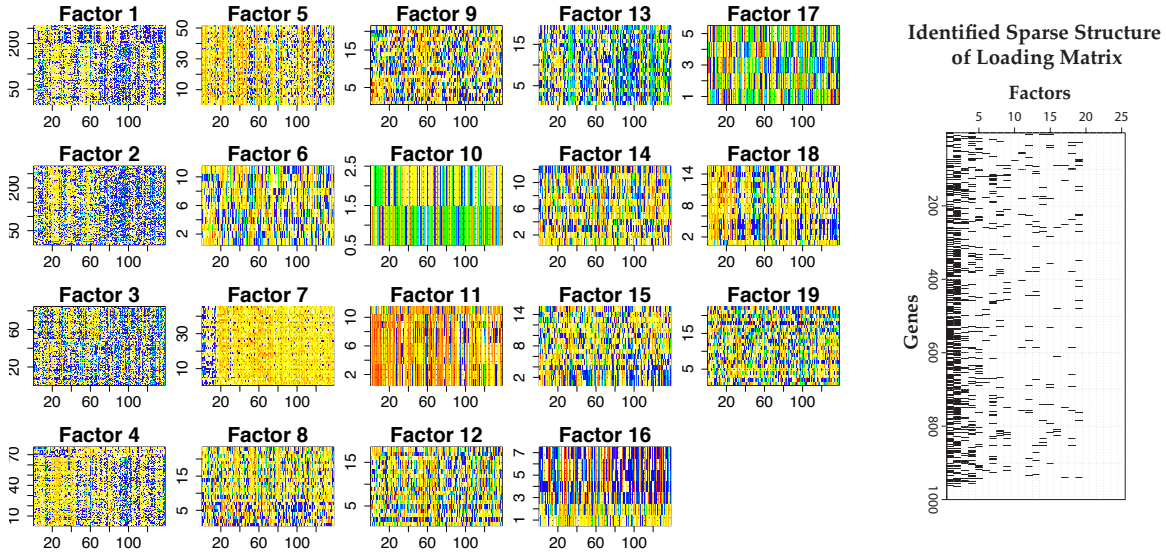


Figure 9: Identified factor probes (left) and sparse structure (right; binary matrix). In each image of the left panel, expression signatures of the probes associated with each factor are depicted across 138 samples (ordered along horizontal axis).

both statistical and biological aspects is available in supporting material at the first author's web site (see link below).

Among the goals of most such studies are identification of multiple factors that may represent underlying activity of biological pathways and provide opportunities for improved predictive models with estimated latent factors for physiological and clinical outcomes (e.g., Carvalho et al., 2008; Chang et al., 2009; Hirose et al., 2008; Lucas et al., 2006, 2009; West, 2003; Yoshida et al., 2004, 2006). Here we discuss an example application of our sparse GFM in analysis of data from previously published breast cancer studies (Cheng et al., 2006; Huang et al., 2003; Pittman et al., 2004; West et al., 2001).

The GFM approach was applied to a sample of gene expression microarray data collected in the CODeX breast cancer genomics study (Carvalho et al., 2008; Pittman et al., 2004; West et al., 2001) at the Sun-Yat Sen Cancer Center, Taipei, during 2001-2004. In addition to summary expression indices from Affymetrix Human Genome U95 arrays, the data set includes immunohistochemistry (IHC) test for key hormonal receptor proteins in clinical prognostics; ERBB2 (Her2) and estrogen (ER). The IHC measures are discrete: ER negative (ER=0), ER positive with low/high-level expression (ER=1 and ER=2), Her2 negative (Her2=0), and Her2 positive with low/high-level (Her2=1 and Her=2). We performed analysis of $p = 996$ genes with the expression levels that, on a log2 (fold change) scale, exceed a median level of 7 and a range of at least 3-fold changes across the tumors. The data set, including the expression data and the IHC hormonal measures, are available on-line as supplementary material.

The annealed estimation of GFM was run with $k = 25$, $\mu = 7$ and $\sigma = 10$. The cooling schedule was prescribed by a linearly-decreasing sequence of 2000 temperatures under which the decay rate and initial temperature were set to 0.006 and 3, respectively. The applied GFM identified 19 factors,

pruning from the model maximum $k = 25$. Heatmaps of gene expression for genes identified in each of the factors appear in Figure 9 with the identified sparse pattern of the loadings matrix.

Evaluation and Annotation of Inferred Factors: To investigate potential biological connections of the factors, we evaluated enrichment of the functional annotations shared by genes in each factor through the Gene Ontology (GO). This exploration revealed associations between some factors and GO biological processes; the complete and detailed results, including tables of the GO enrichment analyses for each factor and detailed biological descriptions, are available from the web site of supporting information.

Factors Related to ER: Figure 10 displays boxplots of fitted values of the factor scores for each sample, plotted across all 19 factors and stratified by levels of each of the clinical ER and Her2 (0/1/2) categories. For each sample i, the posterior mean of the factor vector, namely $\hat{\lambda}_i = (I_k + \Delta)^{-1}\Delta\Phi_Z'\Psi^{-1/2}x_i$, is evaluated at the estimated model, providing the fitted values displayed. We note strong association of ER status to factors 8 (GO: hormone metabolic process), 9 (GO: glucose metabolic process, negative regulation of MAPK activity), 12 (GO: C21-steroid hormone metabolic process), 14 (GO: apoptotic program, positive regulation of caspase activity), 18 (GO: M phase of meiotic cell cycle) and 19 (GO: regulation of Rab protein signal transduction). These clear relationships of ER status to multiple factors with intersecting but also distinct biological pathway annotations is consistent with the known complexity of the broader ER network, as estrogen receptor-induced signaling impacts multiple cellular growth and developmentally related downstream target genes and strongly defines expression factors linked to breast cancer progression.

Her2 Status and Oncogenomic Recombination Hotspot on 17q12: Figure 10 indicates factor 16 as strongly associated with Her2 status $(0,1)$ versus 2. Factor 16 significantly loads on only 7 genes that include STARD3, GRB7 and two probe sets on the locus of ERBB2 (which encodes Her2). This is consistent with earlier gene expression studies that have consistently identified a single expression pattern related to Her2 and a very small number of additional genes, and that have found the "low Her2 positives" level(1) to be generally comparable to negatives. Interestingly, we note that STARD3, GRB7 and ERBB2 are all located on the same chromosomal locus 17q12, which is known as PPP1R1B-STARD3-TCAP-PNMT-PERLD1-ERBB2-MGC14832-GRB7 locus. This locus has been reported in many studies (e.g., Katoh and Katoh, 2004) as an oncogenomic recombination hotspot which is amplified frequently in breast tumor cells, and the purely exploratory (i.e., unsupervised) GFM analysis clearly identifies the "Her2 factor" as strongly reflective of increased expression of genes in this hotspot, consistent with the amplification inducing Her2 positivity.

Comparison to Non-sparse Analysis: Finally, we show a comparison to non-sparse traditional PCA. Supplementary Fig.1 and 2 show the estimated factors (principal components) corresponding to the most dominant 19 eigenvalues, stratified by the levels of ER and Her2. The PCA failed to capture the existing factor relevant to Her2-specific phenotypes in the analysed data. Note that the foregoing sparse analysis identified the Her2-relevant factor only through the 7 non-zero loadings. Indeed, our post-analysis has found that the data set contains very few genes exhibiting significant fold change across the Her2 phenotypes. The non-sparse analysis would capture many irrelevant features through too redundant non-zero loadings. The failure of PCA signifies the importance of sparse modelling in handling high-dimensional data having inherently sparse structure.

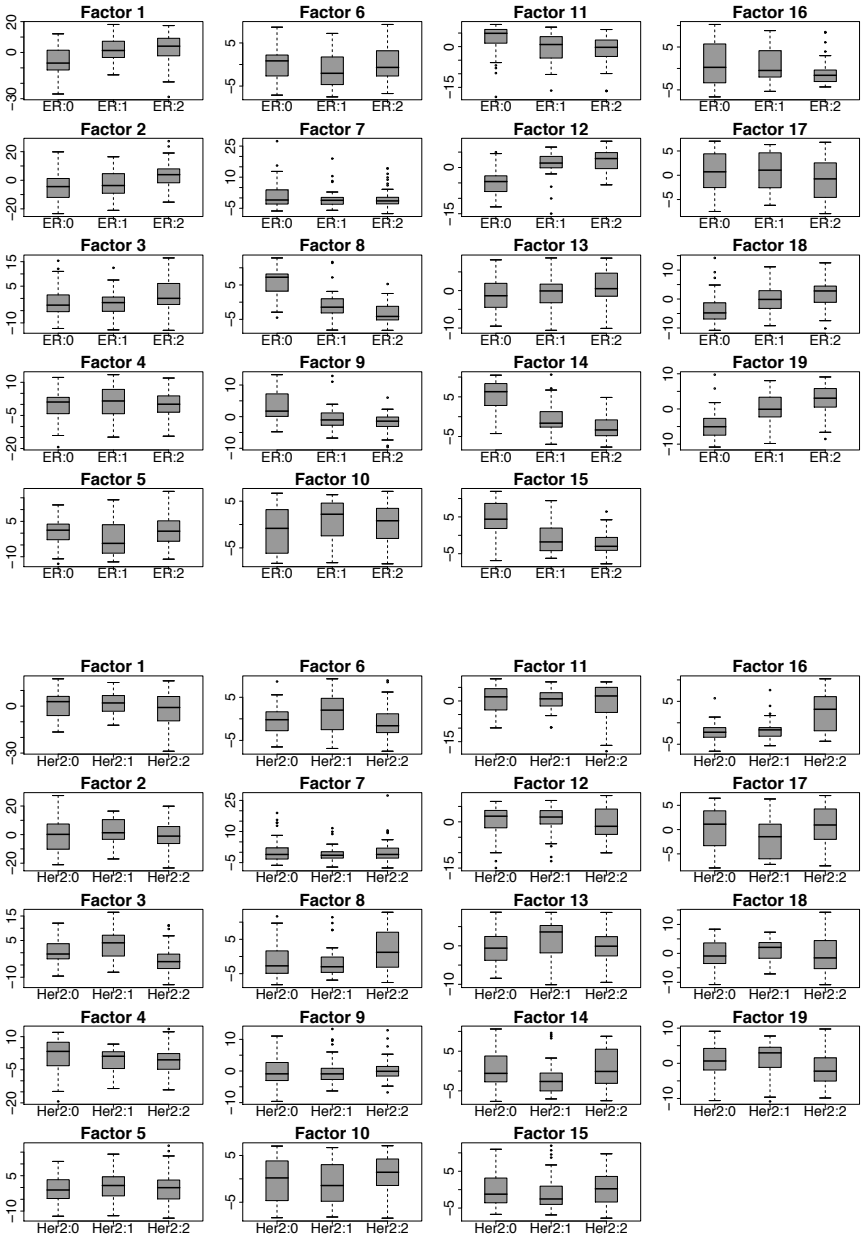


Figure 10: Boxplots of fitted values of breast tumor-specific factor scores, stratified by protein IHC determinations of clinical ER status (upper) and Her2 status (lower) in their 0/1/2 categories.

8. Additional Comments

The novel graphical property of GFMs provides a nice reconciliation of sparse covariance models with sparse precision models—sparse latent factor analysis and graphical models, respectively. Some of the practical benefits of this arise from the ability of GFM to define data reconstructions

exhibiting the same patterns of covariances as the model/data predict, and the potential to induce robustness to outliers relative to non-graphical factor models, whether sparse or not. Some theoretical questions remain about precise conditions under which the sparsity patterns of covariance and precision matrices are guaranteed to agree in general sparse Gaussian factor models other than the GFM form. Additionally, extensions to integrate non-parametric Bayesian model components for factors, following Carvalho et al. (2008), are of clear future interest.

The ability of the VMA2 to aid in the identification of model structure in sparse GFM, and to provide an additional computational strategy and tools to address the inherently challenging combinatorial optimization problem, has been demonstrated in our examples. Scaling to higher dimensional models is enabled by relaxation of the direct deterministic optimization viewpoint, with stochastic search components that promote greater exploration of model space and can speed up search substantially. Nevertheless, moving to higher dimensions will require new, creative computational implementations, such as using distributed computing, that will themselves require novel methodological concepts.

The annealed search methodology evidently will apply in other contexts beyond factor models. At one level, sparse factor models are an instance of problems of variable selection in multivariate regression, in which the regression predictors (feature variables) are themselves unknown (i.e., are the factors). The annealed entropy approach is therefore in principle applicable to problems involving regression model search and uncertainly in general classes of linear or nonlinear multivariate regression with potentially many predictor variables. Beyond this, the same can be said about potential uses in other areas of graphical modelling involving structural inference of directed or undirected graphical models, and also in multivariate time series problems where some of the sparse structure may relate to relationships among variables over time.

We also remark on generalization of the basic form of VMA2 here that might use penalty functions other than the Shannon's entropy used here. The central idea of the VMA2 is the design of a temperature-controlled iterative optimization that converges to the joint posterior distribution of model parameters and sparse structure indicators. The entropy formulation used in our GFM context was inspired by the form of the posterior itself, but similar algorithms—with the same convergent property—could be designed using other forms. This, along with computational efficiency questions and applications in models beyond the sparse GFM framework, and also potential extensions to consider heavy-tailed or Bayesian nonparametric distributions for latent factors and/or residuals (e.g., Carvalho et al., 2008), are open areas for future research.

Acknowledgments

The authors are grateful to the Action Editor and three anonymous referees for their detailed and most constructive comments on the original version of this paper. Elements of the research reported here were developed while Ryo Yoshida was visiting SAMSI and Duke University during 2008-09. Aspects of the research of Mike West were partially supported by grants from the U.S. National Science Foundation (DMS-0342172) and National Institutes of Health (NCI U54-CA-112952). The research of Ryo Yoshida was partially supported by the Japan Science and Technology Agency (JST) under the Core Research for Evolutional Science and Technology (CREST) program. Any opinions, findings and conclusions or recommendations expressed in this work are those of the authors and do not necessarily reflect the views of the NSF or NIH.

Appendix A.

We present a proof of Proposition 1 and a derivation of optimization over Φ.

A.1 Proof of Proposition 1

Replace the objective function of (5) by multiplying by inverse temperature $1/T$:

$$\frac{1}{T}\mathcal{G}_T(\Theta,\omega) = \sum_{Z\in\mathscr{Z}} \omega(Z)\log p(X,Z,\Theta|\zeta)^{1/T} - \sum_{Z\in\mathscr{Z}} \omega(Z)\log\omega(Z).$$

An upper-bound of this modified criterion is derived as follows:

$$\begin{aligned}
\frac{1}{T}\mathcal{G}_T(\Theta,\omega) &= \sum_{Z\in\mathscr{Z}} \omega(Z)\log\frac{p(Z|X,\Theta,\zeta)^{1/T}p(X,\Theta|\zeta)^{1/T}}{\omega(Z)} \\
&= \sum_{Z\in\mathscr{Z}} \omega(Z)\log\frac{p(Z|X,\Theta,\zeta)^{1/T}}{\omega(Z)\sum_{Z'\in\mathscr{Z}} p(Z'|X,\Theta,\zeta)^{1/T}} + K_0 \\
&\leq K_0.
\end{aligned}$$

In the second equality, the terms irrelevant to $\omega(Z)$ are included in $K_0 = \log p(X,\Theta|\zeta)^{1/T} + \log\sum_{Z'\in\mathscr{Z}} p(Z'|X,\Theta,\zeta)^{1/T}$. The first term in the second line is the negative of the Kullback-Leibler divergence between $\omega(Z)$ and the normalized tempered posterior distribution. The lower-bound of the Kullback-Leibler divergence is attained if and only if

$$\omega(Z) = \frac{p(Z|X,\Theta,\zeta)^{1/T}}{\sum_{Z'\in\mathscr{Z}} p(Z'|X,\Theta,\zeta)^{1/T}},$$

as required.

A.2 Derivation: Optimization over Φ

Let ρ_j, $j\in\{1,\ldots,k\}$ be the Lagrange multipliers to ensure the restrictions in (10). We now write down the Lagrange function:

$$\phi_j'\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j - \rho_j(\|\phi_j\|^2-1) - \sum_{m\neq j}\rho_m\phi_m'\phi_j. \tag{13}$$

Differentiation of (13) with respect to ϕ_j yields

$$\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j - \rho_j\phi_j - \sum_{m\neq j}\rho_m\phi_m = 0. \tag{14}$$

In order to solve this equation, the first step to be addressed is to find the closed form solution for the vector of the Lagrange multipliers, $\rho_{(-j)} = \{\rho_m\}_{m\neq j}\in\mathbb{R}^{k-1}$. Multiplying (14) by each ϕ_m', $m\neq j$, from the left, we have the $k-1$ equations as follows:

$$\phi_m'\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j - \sum_{m\neq j}\rho_m\phi_m'\phi_j = 0 \quad \text{for } m \text{ s.t. } m\neq j.$$

This yields the matrix representation

$$\Phi'_{(-j)}\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j - \Phi'_{(-j)}\Phi_{(-j)}\rho_{(-j)} = 0,$$

which in turn leads to the solution for $\rho_{(-j)}$ as

$$\rho_{(-j)} = (\Phi'_{(-j)}\Phi_{(-j)})^{-1}\Phi'_{(-j)}\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j.$$

Substituting this into the original Equation (14) yields the eigenvalue equation

$$N_j\mathbb{E}_\omega[S(z_j,\Psi)]\phi_j - \rho_j\phi_j = 0 \quad \text{with } N_j = I - \Phi_{(-j)}\Phi'_{(-j)}. \tag{15}$$

Now consider the alternative, symmetrized eigenvalue equation

$$N_j\mathbb{E}_\omega[S(z_j,\Psi)]N_j\varphi_j - \rho_j\varphi_j = 0. \tag{16}$$

Since N_j is idempotent, left-multiplication of (16) by N_j yields

$$N_j\mathbb{E}_\omega[S(z_j,\Psi)]N_j\varphi_j - \rho_j N_j\varphi_j = 0.$$

which is equivalent to the required Equation (15) when $\phi_j = N_j\varphi_j$.

References

T.W. Anderson. *An Introduction to Multivariate Statistical Analysis, 3rd ed*. Wiley-Interscience; New Jersey, 2003.

C. Archambeau and F. Bach. Sparse probabilistic projections. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 73–80, Cambridge, MA, 2009. MIT Press.

H. Attias. Inferring parameters and structure of latent variable models by variational Bayes. *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI99)*, pages 21–30, 1999.

C.M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society (Series B)*, 61:611–622, 1999.

C.M. Bishop. *Pattern Recognition and Machine Learning, 1st ed*. Springer: Singapore, 2006.

C.M. Carvalho and M. West. Dynamic matrix-variate graphical models. *Bayesian Analysis*, 2: 69–98, 2007.

C.M. Carvalho, J.E. Lucas, Q. Wang, J.T. Chang, J.R. Nevins, and M. West. High-dimensional sparse factor modelling: Applications in gene expression genomics. *Journal of the American Statistical Association*, 103:1438–1456, 2008.

J. Chang, C.M. Carvalho, S. Mori, A. Bild, Q. Wang, M. West, and J.R. Nevins. Decomposing cellular signaling pathways into functional units: A genomic strategy. *Molecular Cell*, 34:104–114, 2009.

S.H. Cheng, M. West C.F.Horng, E. Huang, J. Pittman, H. Dressman M.H. Tsou, C.M. Chen, S.Y. Tsail, J.J. Jian, J.R. Nevins M.C. Liu, and A.T. Huang. Genomic prediction of loco-regional recurrence following mastectomy in breast cancer. *Journal of Clinical Oncology*, 24:4594–4602, 2006.

A. d'Aspremont, L. El Ghaoui, M.I. Jordan, and G.R.G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

A. Dobra, B. Jones, C. Hans, J.R. Nevins, and M. West. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90:196–212, 2004.

T.L. Griffiths and Z Ghahramani. Infinite latent feature models and the indian buffet process. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 475–482, Cambridge, MA, 2006. MIT Press.

Y. Guan and J Dy. Sparse probabilistic principal component analysis. *Proceedings of AISTATS 2009*, 5:185–192, 2009.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Newv York: Springer, 2001.

O. Hirose, R. Yoshida, S. Imoto, R. Yamaguchi, T. Higuchi, D. S. Charnock-Jones, C. Print, and S. Miyano. Statistical inference of transcriptional module-based gene networks from time course gene expression profiles by using state space models. *Bioinformatics*, 24(7):932–942, 2008.

E. Huang, S Chen, H. K. Dressman, J. Pittman, M. H. Tsou, C. F Horng, A. Bild, E. S. Iversen, M. Liao, C. M. Chen, M. West, J. R. Nevins, and A. T. Huang. Gene expression predictors of breast cancer outcomes. *The Lancet*, 361:1590–1596, 2003.

I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the lasso. *Journal of Computational and Graphical Statistics*, 112(3):531–547, 2003.

B. Jones, A. Dobra, C.M. Carvalho, C. Hans, C. Carter, and M. West. Experiments in stochastic computation for high-dimensional graphical models. *Statistical Science*, 20:388–400, 2005.

M.I. Jordan, editor. *Learning in Graphical Models*. Cambridge MA: MIT Press, 1999.

M.I. Jordan. Graphical models. *Statistical Science*, 19:140–155, 2004.

M. Katoh and M. Katoh. Evolutionary recombination hotspot around GSDML-GSDM locus is closely linked to the oncogenomic recombination hotspot around the PPP1R1B-ERBB2-GRB7 amplicon. *International Journal of Oncology*, 24:757–63, 2004.

J.E. Lucas, C.M. Carvalho, Q. Wang, A.H. Bild, J.R. Nevins, and M. West. Sparse statistical modelling in gene expression genomics. In P. Müller, K.A. Do, and M. Vannucci, editors, *Bayesian Inference for Gene Expression and Proteomics*, pages 155–176. Cambridge University Press, 2006.

J.E. Lucas, C.M. Carvalho, J-T.A. Chi, and M. West. Cross-study projections of genomic biomarkers: An evaluation in cancer genomics. *PLoS ONE*, 4(2):e4523, 2009.

D.J.C. Mackay. Probable networks and plausible predictions - a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6:469–505, 1995.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML2009)*, pages 689–696, New York, NY, USA, 2009. ACM.

J. Pittman, E. Huang, H. Dressman, C. F. Horng, S. H. Cheng, M. H. Tsou, C. M. Chen, A. Bild, E. S. Iversen, A. T. Huang, J. R. Nevins, and M. West. Integrated modeling of clinical and gene expression information for personalized prediction of disease outcomes. *Proceedings of the National Academy of Sciences*, 101:8431–8436, 2004.

P. Rai and H. Daumé. The infinite hierarchical factor regression model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1321–1328, Cambridge, MA, 2009. MIT Press.

N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11:271–282, 1998.

M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

Q. Wang, C.M. Carvalho, J.E. Lucas, and M. West. BFRM: Bayesian factor regression modelling. *Bulletin of the International Society for Bayesian Analysis*, 14(2):4–5, 2007.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. In J.M. Bernardo, M.J. Bayarri, J.O. Berger, A.P. Dawid, D. Heckerman, A.F.M. Smith, and M. West, editors, *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, H. Zuzan R. Spang, J.R. Marks, and J.R. Nevins. Predicting the clinical status of human breast cancer utilizing gene expression profiles. *Proceedings of the National Academy of Sciences*, 98:11462–11467, 2001.

R. Yoshida, T. Higuchi, and S. Imoto. A mixed factors model for dimension reduction and extraction of a group structure in gene expression data. *Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference*, pages 161–172, 2004.

R. Yoshida, T. Higuchi, S. Imoto, and S. Miyano. Arraycluster: an analytic tool for clustering, data visualization and module finder on gene expression profiles. *Bioinformatics*, 22(12):1538–1539, 2006.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.

Journal of Machine Learning Research 11 (2010) 1799-1802 Submitted 10/09; Revised 3/10; Published 6/10

The SHOGUN Machine Learning Toolbox

Sören Sonnenburg[*] SOEREN.SONNENBURG@TU-BERLIN.DE
Berlin Institute of Technology
Franklinstr. 28/29, 10587 Berlin, Germany

Gunnar Rätsch GUNNAR.RAETSCH@TUEBINGEN.MPG.DE
Sebastian Henschel SHOGUN@KODEAFFE.DE
Christian Widmer CWIDMER@TUEBINGEN.MPG.DE
Jonas Behr JONAS.BEHR@TUEBINGEN.MPG.DE
Alexander Zien[†] ALEXANDER.ZIEN@TUEBINGEN.MPG.DE
Fabio de Bona FABIO.DE.BONA@TUEBINGEN.MPG.DE
Friedrich Miescher Laboratory, Max Planck Society
Spemannstr. 39, 72076 Tübingen, Germany

Alexander Binder ALEXANDER.BINDER@TU-BERLIN.DE
Christian Gehl[‡] CHRISTIAN.GEHL@TRIFENSE.DE
Berlin Institute of Technology
Franklinstr. 28/29, 10587 Berlin, Germany

Vojtěch Franc XFRANCV@CMP.FELK.CVUT.CZ
Center for Machine Perception, Czech Technical University
Technicka 2, 166 27 Praha 6, Czech Republic

Editor: Cheng Soon Ong

Abstract

We have developed a machine learning toolbox, called SHOGUN, which is designed for unified large-scale learning for a broad range of feature types and learning settings. It offers a considerable number of machine learning models such as support vector machines, hidden Markov models, multiple kernel learning, linear discriminant analysis, and more. Most of the specific algorithms are able to deal with several different data classes. We have used this toolbox in several applications from computational biology, some of them coming with no less than 50 million training examples and others with 7 billion test examples. With more than a thousand installations worldwide, SHOGUN is already widely adopted in the machine learning community and beyond. SHOGUN is implemented in C++ and interfaces to *MATLAB*™ *R*, *Octave*, *Python*, and has a stand-alone command line interface. The source code is freely available under the GNU General Public License, Version 3 at `http://www.shogun-toolbox.org`.

Keywords: support vector machines, kernels, large-scale learning, Python, Octave, R

1. Introduction

With the great advancements of machine learning in the past few years, many new learning algorithms have been proposed, implemented in C/C++, and made publicly available. Their availability

∗. Also at Friedrich Miescher Laboratory, Max Planck Society, Spemannstr. 39, 72076 Tübingen, Germany.
†. Current Address LIFE Biosystems GmbH, Poststr. 34, 69115 Heidelberg, Germany.
‡. Also at Trifense GmbH, Germendorfer Str. 79, 16727 Velten.

has enabled systematic comparisons between newly developed methods, leading to an increased visibility, and supporting their broad adoption in the community of machine learning as well as in many others (see discussion in Sonnenburg *et al*., 2007). However, for example, there currently exist more than 20 different publicly available implementations of Support Vector Machine (SVM) solvers. Each one comes with its own interface, a small set of available kernel functions, and unique benefits and drawbacks. There is no single unified way of interfacing with these implementations, even though they all are based on essentially the same methodology of supervised learning. This restraints users from fully taking advantage of the recent developments in machine learning algorithms.

This motivated us to develop a machine learning toolbox that provides an easy, unified way for solving certain types of machine learning problems. The result is a toolbox, called SHOGUN, with a focus on large-scale learning using kernel methods and SVMs. It provides a generic interface to 15 SVM implementations, among them SVMlight, LibSVM, GPDT, SVMLin, LibLinear, and OCAS. The SVMs can be easily combined with more than 35 different kernel functions.[1] The toolbox not only provides efficient implementations of the most common kernels, like the linear, polynomial, Gaussian, and sigmoid, but also comes with several recently developed kernels such as the locality improved, Fisher, TOP, spectrum, and the WD kernels for sequence analysis (see Sonnenburg et al., 2007; Ben-Hur et al., 2008; Schweikert *et al*., 2009, and references therein). Moreover, it offers options for using precomputed kernels and allows easy integration of new implementations of kernels. One of SHOGUN's key features is the *combined kernel* to construct weighted linear combinations of multiple kernels that may even be defined on different input domains. Also, several Multiple Kernel Learning algorithms based on different regularization strategies are available to optimize the weighting of the kernels (e.g., Sonnenburg et al., 2006a; Kloft et al., 2010).

Currently, two- and multiclass classification and regression are best supported. In addition to kernel and distance based methods, SHOGUN implements many linear methods and features algorithms to train hidden Markov models. Furthermore, it provides the basic functionality for solving label sequence learning problems (Hidden Semi-Markov SVMs; cf. Rätsch and Sonnenburg, 2007; Schweikert *et al*., 2009). The input feature objects can be dense or sparse vectors of strings, integers (8, 16, 32 or 64 bit; signed or unsigned), or floating point numbers (32 or 64 bit), and can be converted into different feature types. Chains of "pre-processors" (e.g., subtracting the mean) can be attached to each feature object allowing on-the-fly pre-processing. Finally, several commonly used performance measures, like accuracy and area under ROC or precision-recall curves, are implemented in SHOGUN.

An important aspect in the design of SHOGUN was to enable very large-scale learning. It is structured in a way that there is as little as possible overhead for storing the data and intermediate results. Moreover, whenever possible, we implemented auxiliary routines that allow faster computation of combinations of kernel elements that lead to significant speedup during training (for some SVM implementations, e.g., SVMlight and GPDT) and evaluation (Sonnenburg et al., 2007). Furthermore, linear SVMs can be efficiently trained using on the fly computed feature spaces, even mixing sparse, dense and other data types.

This allowed us to use SHOGUN for solving several large-scale learning problems in biological sequence analysis, for example, splice site recognition with up to 50 million example sequences for training (Sonnenburg et al., 2007; Franc and Sonnenburg, 2009) and transcription start site recog-

1. Complete lists of SVM and kernel implementations together with user and developer documentation is available at `http://www.shogun-toolbox.org/doc`.

nition with almost 7 billion test sequences (Sonnenburg et al., 2006b). SHOGUN's core functions are encapsulated in a library (`libshogun`) and are easily accessible and extendible by C++ application developers. What sets SHOGUN apart from many other machine learning toolboxes, is that it provides interactive user interfaces to most major scripting languages that are currently used in scientific computing, in particular *Python*, *MATLAB*, *Octave*, *R*, and a command-line version.

All classes and functions are documented and come with over 600 examples and a tutorial for new users and developers is part of the release.[2] Furthermore, there is an open access tutorial on SVMs (Ben-Hur et al., 2008) that provides a SHOGUN-based command-line tool for illustrative examples (available at `http://svmcompbio.tuebingen.mpg.de`). Maintaining high code quality is ensured by a test suite that supports running the algorithms for each interface on predefined inputs in order to detect breakage. SHOGUN runs on POSIX platforms, such as *Linux*, *BSD*, *Mac OS X*, *Cygwin*, and *Solaris*.

2. Modular, Extendible Object-Oriented Design

SHOGUN is implemented in an object-oriented way using C++ as the programming language. All objects inherit from `CSGObject`, which provides means for garbage collection via reference counting, serialization, and versioning of the object. The implementations of many classes employ *templates* enabling SHOGUN's support of many different data types without code duplication. As the source code and user documentation is automatically generated using `doxygen` and written in-place in the header files, it also drastically reduces the amount of work needed to maintain the documentation. As an example of SHOGUNs object-oriented design, consider the class `CClassifier`: From this class, `CKernelMachine`, for example, is derived and provides basic functions for applying a trained kernel classifier (computing $f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \mathrm{k}(\mathbf{x}, \mathbf{x}_i) + b$) thus enabling code re-use whenever possible. The same holds for `CDistanceMachine` and `CLinearClassifier` (which provides $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ etc.). Currently, SHOGUN implements 329 classes (see Figure 1 in supplementary material available at `http://www.shogun-toolbox.org/jmlr10` for a sketch of the main classes).[3]

3. Interfaces to Scripting Languages and Applications

Built around SHOGUN's core are two types of interfaces: A modular interface that makes use of the SWIG (`http://www.swig.org`), and a static interface. Thanks to SWIG, the modular interface provides the exact same objects in a modular object-oriented way that are available from C++ to other languages, such as *R*, *Python*, and *Octave*. Using so-called `typemaps`, it is convenient to provide type mappings from the native datatype used in the interface to SHOGUN. For example, a function `void set_features(double* features, int n, int m)` can be called directly from *Octave* with a single matrix argument, for example, `set_features(randn(3,4))`. The variables *n* and *m* are then automatically set to the matrix dimensions and together with a data pointer passed to the SHOGUN core.

SHOGUN also provides a static interface with the same structure for all supported plattforms , including the command-line interface where inputs are provided as either strings or files. It is

2. Examples are also accessible online at `http://www.shogun-toolbox.org/doc/examples.html`.
3. See `http://www.shogun-toolbox.org/doc/annotated.html` for the full annotated class listing.

implemented independent of the target language through the class `CSGInterface`, which provides abstract functions to deliver or obtain data from any particular plattform.

A community around SHOGUN is continuously developing, with a growing number of projects building on it (cf. `http://mloss.org/software/tags/shogun`) and a mailing list with more than 100 subscribed users.[4] By 10/2009, there had been at least 1,100 installations under the *Linux* distributions *Debian* and *Ubuntu*. We will continue to develop SHOGUN and are confident that it is and will continue to be useful, and will make an increasing impact beyond the machine learning community by benefiting diverse applications.

Acknowledgments

We acknowledge partial support by the EU under the PASCAL2 Network of Excellence (ICT-216886) as well as DFG Grants MU 987/6-1 and RA-1894/1-1. VF was supported by the EU Reintegration grant SEMISOL (PERG04-GA-2008-239455).

References

A Ben-Hur, CS Ong, S Sonnenburg, B Schölkopf, and G Rätsch. Support vector machines and kernels for computational biology. *PLoS Computat Biol*, 4(10):e1000173, 2008.

V Franc and S Sonnenburg. Optimized cutting plane algorithm for large-scale risk minimization. *Journal of Machine Learning Research*, 10:2157–2192, 2009.

M Kloft, U Brefeld, S Sonnenburg, P Laskov, K-R Müller, and A Zien. Efficient and accurate l_p-norm multiple kernel learning. In *Proc. NIPS 21*. MIT, Cambridge, MA, 2010.

G Rätsch and S Sonnenburg. Large-scale hidden semi-Markov SVMs. In *Proc. NIPS 19*, pages 1161–1168. MIT Press, Cambridge, MA, 2007.

G Schweikert *et al*. mGene: accurate SVM-based gene finding with an application to nematode genomes. *Genome Research*, 19(11):2133–43, Nov 2009.

S Sonnenburg, G Rätsch, C Schäfer, and B Schölkopf. Large-scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, 2006a.

S Sonnenburg, A Zien, and G Rätsch. ARTS: Accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–480, 2006b.

S Sonnenburg, G Rätsch, and K Rieck. Large-scale learning with string kernels. In L Bottou et al., editor, *Large-Scale Kernel Machines*, pages 73–103. MIT Press, 2007.

S Sonnenburg *et al*. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8:2443–2466, 2007.

4. See `http://www.shogun-toolbox.org` for instructions on how to subscribe and `http://news.gmane.org/gmane.comp.ai.machine-learning.shogun` for an archive

Journal of Machine Learning Research 11 (2010) 1803-1831 Submitted 4/09; Revised 11/09; Published 6/10

How to Explain Individual Classification Decisions

David Baehrens* BAEHRENS@CS.TU-BERLIN.DE
Timon Schroeter* TIMON@CS.TU-BERLIN.DE
Technische Universität Berlin
Franklinstr. 28/29, FR 6-9
10587 Berlin, Germany

Stefan Harmeling* STEFAN.HARMELING@TUEBINGEN.MPG.DE
MPI for Biological Cybernetics
Spemannstr. 38
72076 Tübingen, Germany

Motoaki Kawanabe† MOTOAKI.KAWANABE@FIRST.FRAUNHOFER.DE
Fraunhofer Institute FIRST.IDA
Kekulestr.7
12489 Berlin, Germany

Katja Hansen KHANSEN@CS.TU-BERLIN.DE
Klaus-Robert Müller KLAUS-ROBERT.MUELLER@TU-BERLIN.DE
Technische Universität Berlin
Franklinstr. 28/29, FR 6-9
10587 Berlin, Germany

Editor: Carl Edward Rasmussen

Abstract

After building a classifier with modern tools of machine learning we typically have a black box at hand that is able to predict well for unseen data. Thus, we get an answer to the question *what* is the most likely label of a given unseen data point. However, most methods will provide no answer *why* the model predicted a particular label for a single instance and what features were most influential for that particular instance. The only method that is currently able to provide such explanations are decision trees. This paper proposes a procedure which (based on a set of assumptions) allows to explain the decisions of *any* classification method.

Keywords: explaining, nonlinear, black box model, kernel methods, Ames mutagenicity

1. Introduction

Automatic nonlinear classification is a common and powerful tool in data analysis. Machine learning research has created methods that are practically useful and that can classify unseen data after being trained on a limited training set of labeled examples.

Nevertheless, most of the algorithms do not *explain* their decision. However in practical data analysis it is essential to obtain an instance-based explanation, that is, we would like to gain an

∗. The first three authors contributed equally.
†. Also at Technische Universität Berlin, Franklinstr. 28/29, FR 6-9, 10587 Berlin, Germany.

understanding what input features made the nonlinear machine give its answer for each individual data point.

Typically, explanations are provided jointly for all instances of the training set, for example feature selection methods (including Automatic Relevance Determination) find out which inputs are salient for a good generalization (see Guyon and Elisseeff, 2003, for a review). While this can give a coarse impression about the global usefulness of each input dimension, it is still an ensemble view and does not provide an answer on an instance basis.[1] In the neural network literature also solely an ensemble view was taken in algorithms like input pruning (e.g., Bishop, 1995; LeCun et al., 1998). The only classification that does provide individual explanations are decision trees (e.g., Hastie et al., 2001).

This paper proposes a simple framework that provides local explanation vectors applicable to *any* classification method in order to help understanding prediction results for single data instances. The local explanation yields the features relevant for the prediction at the very points of interest in the data space, and is able to spot local peculiarities that are neglected in the global view, for example, due to cancellation effects.

The paper is organized as follows: We define local explanation vectors as class probability gradients in Section 2 and give an illustration for Gaussian Process Classification (GPC). Some methods output a prediction without a direct probability interpretation. For these we propose in Section 3 a way to estimate local explanations. In Section 4 we apply our methodology to learn distinguishing properties of Iris flowers by estimating explanation vectors for a k-NN classifier applied to the classic Iris data set. In Section 5 we discuss how our approach applied to a SVM classifier allows us to explain how digit "2" is distinguished from digit "8" in the USPS data set. In Section 6 we focus on a more real-world application scenario where the proposed explanation capabilities prove useful in drug discovery: Human experts regularly decide how to modify existing lead compounds in order to obtain new compounds with improved properties. Models capable of explaining predictions can help in the process of choosing promising modifications. Our automatically generated explanations match with chemical domain knowledge about toxifying functional groups of the compounds in question. Section 7 contrasts our approach with related work and Section 8 discusses characteristic properties and limitations of our approach, before we conclude the paper in Section 9.

2. Definitions of Explanation Vectors

In this Section we will give definitions for our approach of local explanation vectors in the classification setting. We start with a theoretical definition for multi-class Bayes classification and then give a specialized definition being more practical for the binary case.

For the multi-class case, suppose we are given data points $x_1, \ldots, x_n \in \Re^d$ with labels $y_1, \ldots, y_n \in \{1, \ldots, C\}$ and we intend to learn a function that predicts the labels of unlabeled data points. Assuming that the data is IID-sampled from some unknown joint distribution $P(X, Y)$, we define the

1. This point is illustrated in Figure 1 (Section 2). Applying feature selection methods to the training set (a) will lead to the (correct) conclusion that both dimensions are equally important for accurate classification. As an alternative to this ensemble view, one may ask: Which features (or combinations thereof) are most influential in the vicinity of each particular instance. As can be seen in Figure 1 (c), the answer depends on where the respective instance is located. On the hypotenuse and at the corners of the triangle, both features contribute jointly, whereas along each of the remaining two edges the classification depends almost completely on just one of the features.

Bayes classifier,

$$g^*(x) = \arg\min_{c \in \{1,\ldots,C\}} P(Y \neq c \mid X = x)$$

which is optimal for the 0-1 loss function (see Devroye et al., 1996).

For the Bayes classifier we define the *explanation vector* of a data point x_0 to be the derivative with respect to x at $x = x_0$ of the conditional probability of $Y \neq g^*(x_0)$ given $X = x$, or formally,

Definition 1

$$\zeta(x_0) := \frac{\partial}{\partial x} P(Y \neq g^*(x) \mid X = x)\Big|_{x=x_0}.$$

Note that $\zeta(x_0)$ is a d-dimensional vector just like x_0 is. The classifier g^* partitions the data space $\Re^d$ into up to C parts on which g^* is constant. We assume that the conditional distribution $P(Y = c \mid X = x)$ is first-order differentiable w.r.t. x for all classes c and over the entire input space. For instance, this assumption holds if $P(X = x \mid Y = c)$ is for all c first-order differentiable in x and the supports of the class densities overlap around the border for all neighboring pairs in the partition by the Bayes classifier. The vector $\zeta(x_0)$ defines on each of those parts a vector field that characterizes the flow away from the corresponding class. Thus entries in $\zeta(x_0)$ with large absolute values highlight features that will influence the class label decision of x_0. A positive sign of such an entry implies that increasing that feature would lower the probability that x_0 is assigned to $g^*(x_0)$. Ignoring the orientations of the explanation vectors, ζ forms a continuously changing (orientation-less) vector field along which the class labels change. This vector field lets us *locally* understand the Bayes classifier.

We remark that $\zeta(x_0)$ becomes a zero vector, for example, when $P(Y \neq g^*(x) \mid X = x)|_{x=x_0}$ is equal to one in some neighborhood of x_0. Our explanation vector fits well to classifiers where the conditional distribution $P(Y = c \mid X = x)$ is usually not completely flat in some regions. In the case of deterministic classifiers, despite of this issue, Parzen window estimators with appropriate widths (Section 3) can provide meaningful explanation vectors for many samples in practice (see also Section 8).

In the case of binary classification we directly define local explanation vectors as local gradients of the probability function $p(x) = P(Y = 1 \mid X = x)$ of the learned model for the positive class.

For a probability function $p : \Re^d \to [0,1]$ of a classification model learned from examples $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in \Re^d \times \{-1, +1\}$ the explanation vector for a classified test point x_0 is the local gradient of p at x_0:

Definition 2

$$\eta_p(x_0) := \nabla p(x)|_{x=x_0}.$$

By this definition the explanation η is again a d-dimensional vector just like the test point x_0 is. The sign of each of its individual entries indicates whether the prediction would increase or decrease when the corresponding feature of x_0 is increased locally and each entry's absolute value gives the amount of influence in the change in prediction. The vector η gives the direction of the steepest ascent from the test point to higher probabilities for the positive class. For binary classification the negative version $-\eta_p(x_0)$ indicates the changes in features needed to increase the probability for the negative class which may be especially useful for x_0 predicted in the positive class.

For an example we apply Definition 2 to model predictions learned by Gaussian Process Classification (GPC), see Rasmussen and Williams (2006). GPC is used here for three reasons:
(i) In our real-world application we are interested in classifying data from drug discovery, which is an area where Gaussian processes have proven to show state-of-the-art performance, see, for example, Obrezanova and Segall (2010), Obrezanova et al., Schroeter et al. (2007c), Schroeter et al. (2007a), Schroeter et al. (2007b), Schwaighofer et al. (2007), Schwaighofer et al. (2008) and Obrezanova et al. (2008). It is natural to expect a model with high prediction accuracy on a complex problem to capture relevant structure of the data which is worth explaining and may give domain specific insights in addition to the values predicted. For an evaluation of the explaining capabilities of our approach on a complex problem from chemoinformatics see Section 6.
(ii) GPC does model the class probability function used in Definition 2 directly. For other classification methods such as Support Vector Machines that do not provide a probability function as its output we give an example for an estimation method starting from Definition 1 in Section 3.
(iii) The local gradients of the probability function can be calculated analytically for differentiable kernels as we discuss next.

Let $\overline{f}(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i)$ be a Gaussian Process (GP) model trained on sample points $x_1, \ldots, x_n \in \Re^d$ where k is a kernel function and α_i are the learned weights of each sample point. For a test point $x_0 \in \Re^d$ let $\mathrm{var}_f(x_0)$ be the variance of $f(x_0)$ under the GP posterior of f. Because the posterior cannot be calculated analytically for GP classification models, we used an approximation by expectation propagation (EP) (Kuss and Ramussen, 2005). In the case of the probit likelihood term defined by the error function, the probability for being of the positive class $p(x_0)$ can be computed easily from this approximated posterior as

$$p(x_0) = \frac{1}{2}\mathrm{erfc}\left(\frac{-\overline{f}(x_0)}{\sqrt{2}\cdot\sqrt{1+\mathrm{var}_f(x_0)}}\right),$$

where erfc denotes the complementary error function (see Equation 6 in Schwaighofer et al., 2008). Then the local gradient of $p(x_0)$ is given by[2]

$$\nabla p(x)|_{x=x_0} = \frac{\exp\left(\frac{-\overline{f}(x_0)^2}{2(1+\mathrm{var}_f(x_0))}\right)}{\sqrt{2\pi}}\left(\frac{\nabla\overline{f}(x)|_{x=x_0}}{\sqrt{1+\mathrm{var}_f(x_0)}} - \frac{1}{2}\frac{\overline{f}(x_0)}{(1+\mathrm{var}_f(x_0))^{\frac{3}{2}}}\nabla\mathrm{var}_f(x)|_{x=x_0}\right). \quad (1)$$

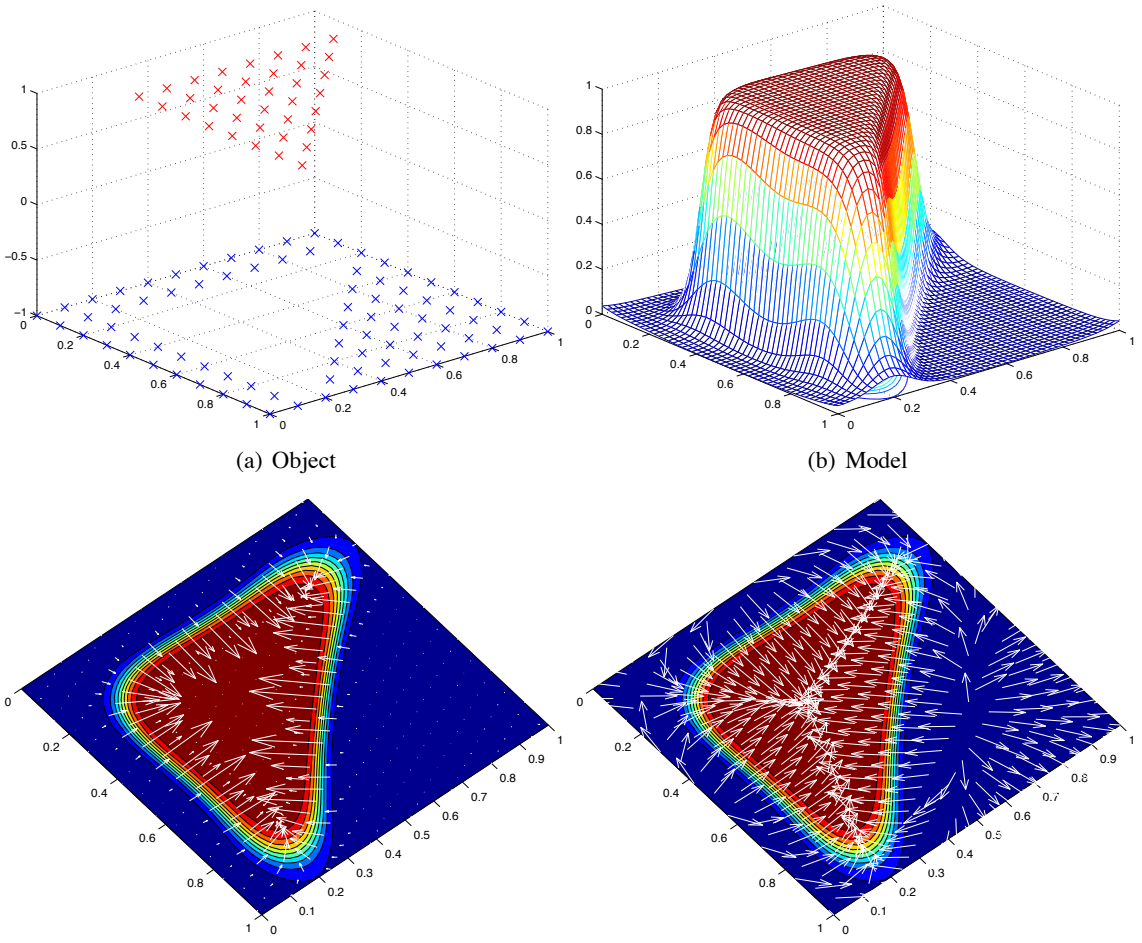
As a kernel function choose, for example, the RBF-kernel $k(x_0, x_1) = \exp(-w(x_0 - x_1)^2)$, which has the derivative $(\partial/\partial x_{0,j})k(x_0, x_1) = -2w\exp(-w(x_0 - x_1)^2)(x_{0,j} - x_{1,j})$ for $j \in \{1, \ldots, d\}$. Then the elements of the local gradient $\nabla\overline{f}(x)|_{x=x_0}$ are

$$\frac{\partial\overline{f}}{\partial x_{0,j}} = -2w\sum_{i=1}^{n}\alpha_i\exp(-w(x_0 - x_i)^2)(x_{0,j} - x_{i,j}) \quad \text{for } j \in \{1, \ldots, d\}.$$

2. For a detailed derivation, see Appendix A.1.

For $\text{var}_f(x_0) = k(x_0,x_0) - k_*^T(K+\Sigma)^{-1}k_*$ the derivative is given by[3]

$$\nabla \text{var}_f(x)|_{x=x_0} = \frac{\partial \text{var}_f}{\partial x_{0,j}} = \left(\frac{\partial}{\partial x_{0,j}}k(x_0,x_0)\right) - 2 * k_*^T(K+\Sigma)^{-1}\frac{\partial}{\partial x_{0,j}}k_* \quad \text{for } j \in \{1,\ldots,d\}.$$



(a) Object (b) Model

(c) Local explanation vectors (d) Direction of explanation vectors

Figure 1: Explaining simple object classification with Gaussian Processes

Panel (a) of Figure 1 shows the training data of a simple object classification task and panel (b) shows the model learned using GPC.[4] The data is labeled -1 for the blue points and $+1$ for the red points. As illustrated in panel (b) the model is a probability function for the positive class which gives every data point a probability of being in this class. Panel (c) shows the probability gradient of the model together with the local gradient explanation vectors. Along the hypotenuse and at the corners of the triangle explanations from both features interact towards the triangle class while along

3. Here $k_* = (k(x_0,x_1),\ldots,k(x_0,x_n))^T$ is the evaluation of the kernel function between the test point x_0 and every training point. Σ is the diagonal matrix of the variance parameter. For details see Rasmussen and Williams (2006, Chapter 3).
4. Hyperparameters were tuned by a gradient ascend on the evidence.

the edges the importance of one of the two feature dimensions dominates. At the transition from the negative to the positive class the length of the local gradient vectors represents the increased importance of the relevant features. In panel (d) we see that explanations close to the edges of the plot (especially in the right hand side corner) point away from the positive class. However, panel (c) shows that their magnitude is very small. For discussion of this issue see Section 8.

3. Estimating Explanation Vectors

Several classification methods directly estimate the decision rule, which often has no interpretation as the probability function in terms of Definition 2. For example Support Vector Machines estimate a decision function of the form

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) + b,$$

$\alpha_i, b \in \Re$. Suppose we have two classes (each with one cluster) in one dimension (see Figure 2) and train a SVM with RBF kernel. For points outside the data clusters $f(x)$ tends to zero. Thus, the derivative of $f(x)$ (shown as arrows above the curves) for points on the very left or on the very right side of the axis will point to the wrong side. In the following, we will explain how explanations can

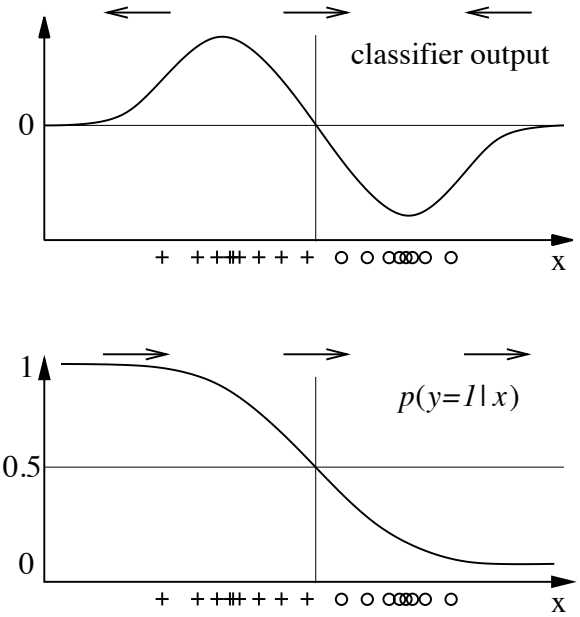


Figure 2: Classifier output of an SVM (top) compared to $p(y=1|x)$ (bottom).

be obtained for such classifiers.

In practice, we do not have access to the true underlying distribution $P(X,Y)$. Consequently, we have no access to the Bayes classifier as defined in Section 2. Instead, we can apply sophisticated learning machinery like Support Vector Machines (Vapnik, 1995; Schölkopf and Smola, 2002; Müller et al., 2001) that estimates some classifier g that tries to mimic g^*. For test data points $z_1, \ldots, z_m \in \Re^d$ which are assumed to be sampled from the same unknown distribution as the training data, g estimates labels $g(z_1), \ldots, g(z_m)$. Now, instead of trying to explain g^*, to which we have

no access, we will define explanation vectors that help us understand the classifier g on the test data points.

Since we do not assume that we have access to some intermediate real-valued classifier output here (of which g might be a thresholded version and which further might not be an estimate of $P(Y=c \mid X=x)$), we suggest to approximate g by another classifier $\hat{g}$, the actual form of which resembles the Bayes classifier. There are several choices for $\hat{g}$, for example, GPC, logistic regression, and Parzen windows.[5] In this paper we apply Parzen windows to the training points to estimate the weighted class densities $P(Y=c) \cdot P(X \mid Y=c)$, for the index set $I_c = \{i \mid g(x_i) = c\}$

$$\hat{p}_\sigma(x, y=c) = \frac{1}{n} \sum_{i \in I_c} k_\sigma(x - x_i) \tag{2}$$

and with $k_\sigma(z)$ being a Gaussian kernel $k_\sigma(z) = \exp(-0.5\, z^\top z/\sigma^2)/\sqrt{2\pi\sigma^2}$ (as always other kernels are also possible). This estimates $P(Y=c \mid X=x)$ for all c,

$$\hat{p}_\sigma(y=c|x) = \frac{\hat{p}_\sigma(x, y=c)}{\hat{p}_\sigma(x, y=c) + \hat{p}_\sigma(x, y\neq c)} \approx \frac{\sum_{i \in I_c} k_\sigma(x - x_i)}{\sum_i k_\sigma(x - x_i)} \tag{3}$$

and thus is an estimate of the Bayes classifier (that mimics g),

$$\hat{g}_\sigma(x) = \arg \min_{c \in \{1,\ldots,C\}} \hat{p}_\sigma(y \neq c \mid x).$$

This approach has the advantage that we can use our estimated classifier g to generate any amount of labeled data for constructing $\hat{g}$. The single hyper-parameter σ is chosen such that $\hat{g}$ approximates g (which we want to explain), that is,

$$\hat{\sigma} := \arg\min_\sigma \sum_{j=1}^{m} I\left\{ g(z_j) \neq \hat{g}_\sigma(z_j) \right\},$$

where $I\{\cdots\}$ is the indicator function. σ is assigned the constant value $\hat{\sigma}$ from here on and omitted as a subscript. For $\hat{g}$ it is straightforward to define explanation vectors:

Definition 3

$$\hat{\xi}(z) := \frac{\partial}{\partial x}\, \hat{p}(y \neq g(z) \mid x)\Big|_{x=z} = \frac{\left(\sum_{i \notin I_{g(z)}} k(z - x_i)\right)\left(\sum_{i \in I_{g(z)}} k(z - x_i)(z - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2} - \frac{\left(\sum_{i \notin I_{g(z)}} k(z - x_i)(z - x_i)\right)\left(\sum_{i \in I_{g(z)}} k(z - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2}.$$

This is easily derived using Equation (3) and the derivative of Equation (2), see Appendix A.3.1. Note that we use g instead of $\hat{g}$. This choice ensures that the orientation of $\hat{\xi}(z)$ fits to the labels assigned by g, which allows better interpretations.

In summary, we imitate the classifier g which we would like to explain locally by a Parzen window classifier $\hat{g}$ that has the same form as the Bayes estimator and for which we can estimate the

5. For Support Vector Machines Platt (1999) fits a sigmoid function to map the outputs to probabilities. In the following, we will present a more general method for estimating explanation vectors.

explanation vectors using Definition 3. Practically there are some caveats: The mimicking classifier $\hat{g}$ has to be estimated from g even in high dimensions; this needs to be done with care. However, in principle we have an arbitrary amount of training data available for constructing $\hat{g}$ since we may use our estimated classifier g to generate labeled data.

4. Explaining Iris Flower Classification by k-Nearest Neighbors

The Iris flower data set (introduced in Fisher, 1936) describes 150 flowers from the genus Iris by four features: sepal length, sepal width, petal length, and petal width, all of which are easily measured properties of certain leaves of the corolla of the flower. There are three clusters in this data set that correspond to three different species: *Iris setosa*, *Iris virginica*, and *Iris versicolor*.

Let us consider the problem of classifying the data points of *Iris versicolor* (class 0) against the other two species (class 1). We applied standard classification machinery to this problem as follows:

- Class 0 consists of all examples of *Iris versicolor*.
- Class 1 consists of all examples of *Iris setosa* and *Iris virginica*.
- Randomly split 150 data points into 100 training and 50 test examples.
- Normalize training and test set using the mean and variance of the training set.
- Apply k-nearest neighbor classification with $k = 4$ (chosen by leave-one-out cross-validation on the training data).
- Training error is 3% (i.e., 3 mistakes in 100).
- Test error is 8% (i.e., 4 mistakes in 50).

In order to estimate explanation vectors we mimic the classification results with a Parzen window classifier. The best fit (3% error) is obtained with a kernel width of $\sigma = 0.26$ (chosen by leave-one-out cross-validation on the training data).

Since the explanation vectors live in the input space we can visualize them with scatter plots of the initially measured features. The resulting *explanations* (i.e., vectors) for the test set are shown in Figure 3. The blue markers correspond to explanation vectors of *Iris setosa* and the red markers correspond to those of of *Iris virginica* (both class 1). Both groups of markers point to the green markers of *Iris versicolor*. The most important feature is the combination of petal length and petal width (see the corresponding panel), the product of which corresponds roughly to the area of the petals. However, the resulting explanations for the two species in class 1 are different:

- *Iris setosa* (class 1) is different from *Iris versicolor* (class 0) because its petal area is *smaller*.
- *Iris virginica* (class 1) is different from *Iris versicolor* (class 0) because its petal area is *larger*.

Also the dimensions of the sepal (another part of the blossom) are relevant, but not as distinguishing.

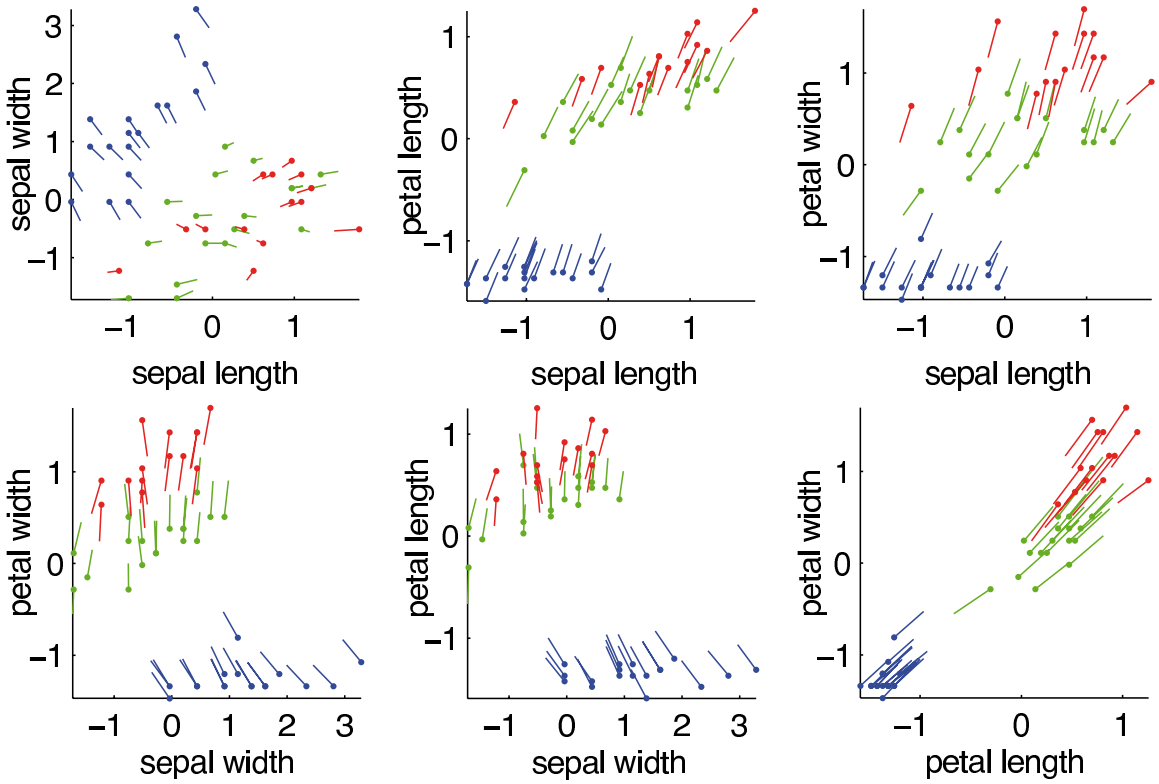


Figure 3: Scatter plots of the explanation vectors for the test data. Shown are all explanation vectors for both classes: class 1 containing *Iris setosa* (shown in blue) and *Iris virginica* (shown in red) versus class 0 containing only the species *Iris versicolor* (shown in green). Note that the explanations why an Iris flower is not an *Iris versicolor* is different for *Iris setosa* and *Iris virginica*.

5. Explaining USPS Digit Classification by Support Vector Machine

We now apply the framework of estimating explanation vectors to a high dimensional data set, the USPS digits. The classification problem that we designed for illustration purposes is detailed in the following list:

- digits: 16×16 images that are reshaped to 256×1 dimensional column vectors
- classifier: SVM from Schwaighofer (2002) with RBF kernel width $\sigma = 1$ and regularization constant $C = 10$ (chosen by grid search in cross-validation on the training data).
- training set: 47 "twos", 53 "eights"; training error 0.00
- test set: 48 "twos", 52 "eights"; test error 0.05

We approximated the estimated class labels obtained by the SVM with the Parzen window classifier (Parzen window size $\sigma = 10.2505$, chosen by grid search in cross-validation on the training data). The SVM and the Parzen window classifier only disagreed on 2% of the test examples, so a good

Figure 4: USPS digits (training set): "twos" (left) and "eights" (right) with correct classification. For each digit from left to right: (i) explanation vector (with black being negative, white being positive), (ii) the original digit, (iii-end) artificial digits along the explanation vector towards the other class.

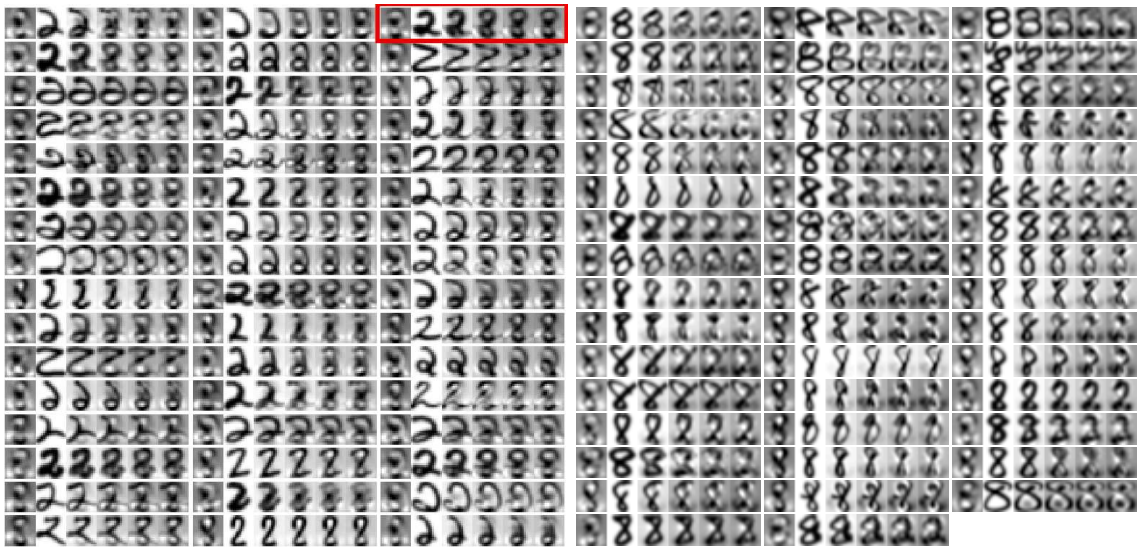


Figure 5: USPS digits (test set bottom part): "twos" (left) and "eights" (right) with correct classification. For each digit from left to right: (i) explanation vector (with black being negative, white being positive), (ii) the original digit, (iii-end) artificial digits along the explanation vector towards the other class.

fit was achieved. Figures 4 and 5 show our results. All parts show three examples per row. For each example we display from left to right: (i) the explanation vector, (ii) the original digit, (iii-end) artificial digits along the explanation vector towards the other class.[6] These artificial digits should help to understand and interpret the explanation vector. Let us first have a look at the results on the training set:

Figure 4 (left panel): Let us focus on the top example framed in red. The line that forms the "two" is part of some "eight" from the data set. Thus the parts of the lines that are missing show up in the explanation vector: if the dark parts (which correspond to the missing lines) are added to the "two" digit then it will be classified as an "eight". In other words, because of the lack of those parts the digit was classified as a "two" and not as an "eight". A similar explanation holds for the middle example framed in red in the same Figure. Not all examples transform easily to "eights": Besides adding parts of black lines, some existing black spots (that make the digit be a "two") must be removed. This is reflected in the explanation vector by white spots/lines. The bottom "two", framed in red, is actually a dash and is in the data set by mistake. However, its explanation vector shows nicely which parts have to be added and which have to be removed.

Figure 4 (right panel): We see similar results for the "eights" class. The explanation vectors again tell us how the "eights" have to change to become classified as "twos". However, sometimes the transformation does not reach the "twos". This is probably due to the fact that some of the "eights" are inside the cloud of "eights".

On the test set the explanation vectors are not as pronounced as on the training set. However, they show similar tendencies:

Figure 5 (left panel): We see the correctly classified "twos". Let's focus on the example framed in red. Again the explanation vector shows us how to edit the image of the "two" to transform it into an "eights", that is, exactly which parts of the digit were important for the classification result. For several other "twos" the explanation vectors do not directly lead to the "eights" but weight the different parts of the digits that were relevant for the classification.

Figure 5 (right panel): Similarly to the training data, we see that also these explanation vectors are not bringing all "eights" to "twos". Their explanation vectors mainly suggest to remove most of the "eights" (black pixels) and add some black in the lower part (the light parts, which look like a white shadow).

Overall, the explanation vectors tell us how to edit our example digits to change the assigned class label. Hereby, we get a better understanding of the reasons why the chosen classifier classified the way it did.

6. Explaining Mutagenicity Classification by Gaussian Processes

In the following Section we describe an application of our local gradient explanation methodology to a complex real world data set. Our aim is to find structure specific to the problem domain that has *not*

6. For the sake of simplicity, no intermediate updates were performed, that is, artificial digits were generated by taking equal-sized steps in the direction given by the original explanation vector calculated for the original digit.

been fed into training explicitly but is captured implicitly by the GPC model in the high-dimensional feature space used to determine its prediction. We investigate the task of predicting Ames mutagenic activity of chemical compounds. Not being mutagenic (i.e., not able to cause mutations in the DNA) is an important requirement for compounds under investigation in drug discovery and design. The Ames test (Ames et al., 1972) is a standard experimental setup for measuring mutagenicity. The following experiments are based on a set of Ames test results for 6512 chemical compounds that we published previously.[7]

GPC was applied as follows:

- Class 0 consists of non-mutagenic compounds.
- Class 1 consists of mutagenic compounds.
- Randomly split 6512 data points into 2000 training and 4512 test examples such that:
  - The training set consists of equally many class 0 and class 1 examples.
  - For the steroid compound class the balance in the training and test set is enforced.
- 10 additional random splits were investigated individually. This confirmed the results presented below.
- Each example (chemical compound) is represented by a vector of counts of 142 molecular substructures calculated using the DRAGON software (Todeschini et al., 2006).
- Normalize training and test set using the mean and variance of the training set.
- Apply GPC model with RBF kernel.
- Performance (84 % area under curve) confirms our previous results (Hansen et al., 2009). Error rates can be obtained from Figure 6.

Together with the prediction we calculated the explanation vector (as introduced in Definition 2) for each test point. The remainder of this Section is an evaluation of these local explanations.

In Figures 7 and 8 we show the distribution of the local importance of selected features across the test set: For each input feature we generate a histogram of local importance values, as indicated by its corresponding entry in the explanation vector of each of the 4512 test compounds. The features examined in Figure 7 are counts of substructures known to cause mutagenicity. We show all approved "specific toxicophores" introduced by Kazius et al. (2005) that are also represented in the DRAGON set of features. The features shown in Figure 8 are known to detoxify certain toxicophores (again see Kazius et al., 2005). With the exception of 7(e) the toxicophores also have a toxifying influence according to our GPC prediction model. Feature 7(e) seems to be mostly irrelevant for the prediction of the GPC model on the test points. In contrast the detoxicophores show overall negative influence on the prediction outcome of the GPC model. Modifying the test compounds by adding toxicophores will increase the probability of being mutagenic as predicted by the GPC model while adding detoxicophores will decrease this predicted probability.

7. See Hansen et al. (2009) for results of modeling this set using different machine learning methods. The data itself is available online at `http://ml.cs.tu-berlin.de/toxbenchmark`.

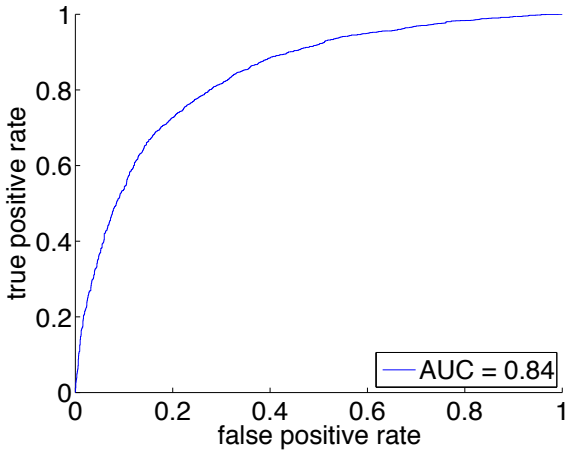


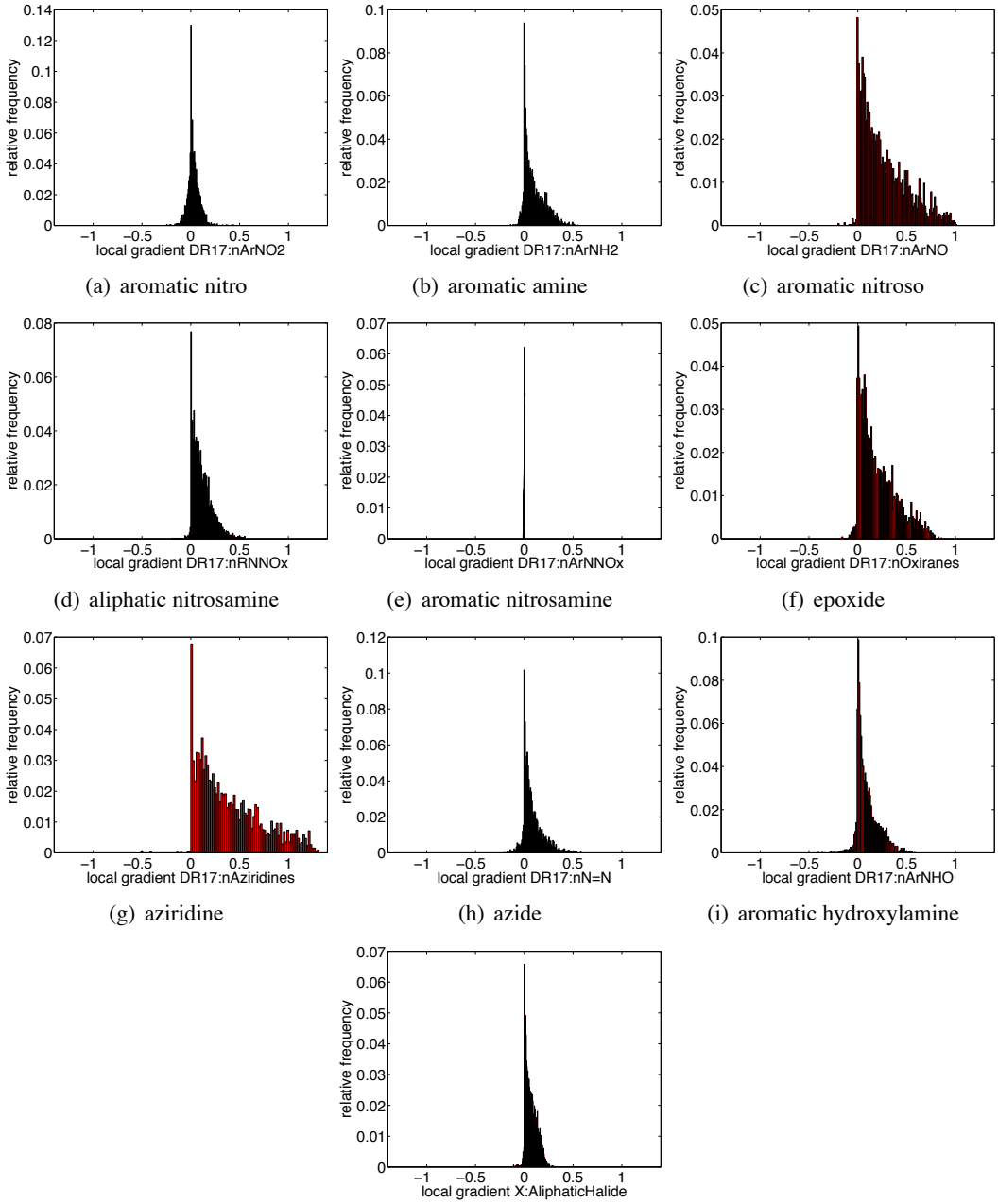
Figure 6: Receiver operating characteristic curve of GPC model for mutagenicity prediction

We have seen that the conclusions drawn from our explanation vectors agree with established knowledge about toxicophores and detoxicophores. While this is reassuring, such a sanity check required existing knowledge about which compounds are toxicophores and detoxicophores and which are not. Thus it is interesting to ask whether we also could have *discovered* that knowledge from the explanation vectors. To answer this question we ranked all 142 features by the means of their local gradients.[8] Clear trends result: 9 out of 10 known toxicophores can be found close to the top of the list (mean rank of 19). The only exception (rank 81) is the aromatic nitrosamine feature.[9] This trend is even stronger for the detoxicophores: The mean rank of these five features is 138 (out of 142), that is, they consistently exhibit the largest negative local gradients. Consequently, the established knowledge about toxicophores and detoxicophores could indeed have been *discovered* using our methodology.

In the following paragraph we will discuss steroids[10] as an example of an important compound class for which the meaning of features differs from this global trend, so that local explanation vectors are needed to correctly identify relevant features.

Figure 9 displays the difference in relevance of epoxide (a) and aliphatic nitrosamine (c) substructures for the predicted mutagenicity of steroids and non-steroid compounds. For comparison we also show the distributions for compounds chosen at random from the test set (b,d). Each subfigure contains two measures of (dis-)similarity for each pair of distributions. The p-value of the Kolmogorov-Smirnoff test (KS) gives the probability of error when rejecting the hypothesis that both relative frequencies are drawn from the same underlying distribution. The symmetrized

8. Tables resulting from this ranking are made available as a supplement to this paper and can be downloaded from the journals website.
9. This finding agrees with the result obtained by visually inspecting Figure 7(e). We found that only very few compounds with this feature are present in the data set. Consequently, detection of this feature is only possible if enough of these few compounds are included in the training data. This was not the case in the random split used to produce the results presented above.
10. Steroids are natural products and occur in humans, animals, and plants. They have a characteristic backbone containing four fused carbon-rings. Many hormones important to the development of the human body are steroids, including androgens, estrogens, progestagens, cholesterol and natural anabolics. These have been used as starting points for the development of many different drugs, including the most reliable contraceptives currently on the market.



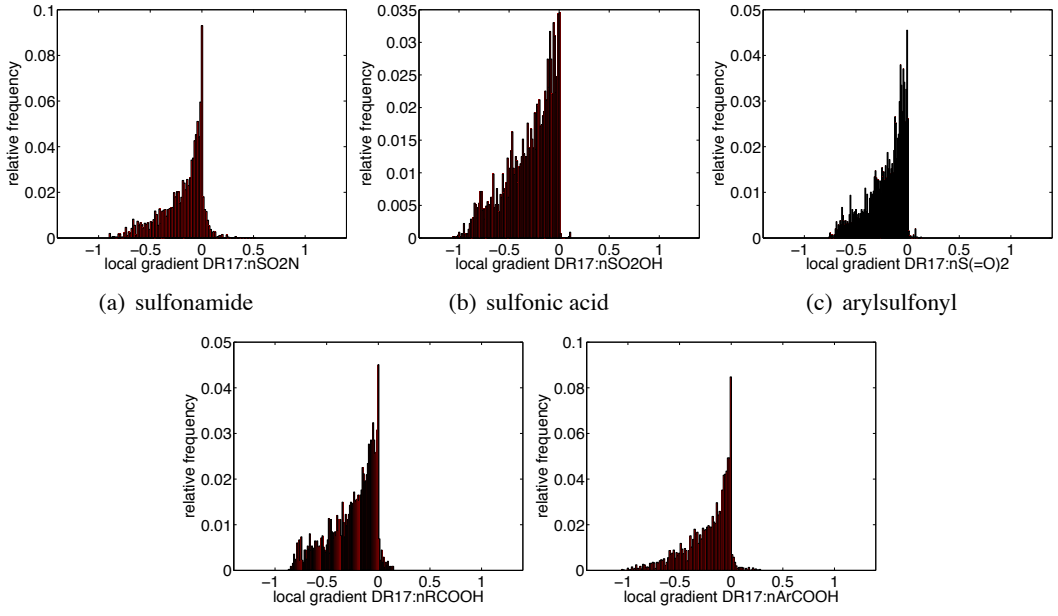
(a) aromatic nitro (b) aromatic amine (c) aromatic nitroso

(d) aliphatic nitrosamine (e) aromatic nitrosamine (f) epoxide

(g) aziridine (h) azide (i) aromatic hydroxylamine

(j) aliphatic halide

Figure 7: Distribution of local importance of selected features across the test set of 4512 compounds. Nine out of ten known toxicophores (Kazius et al., 2005) indeed exhibit positive local gradients.



(a) sulfonamide (b) sulfonic acid (c) arylsulfonyl

(d) aliphatic carboxylic acid (e) aromatic carboxylic acid

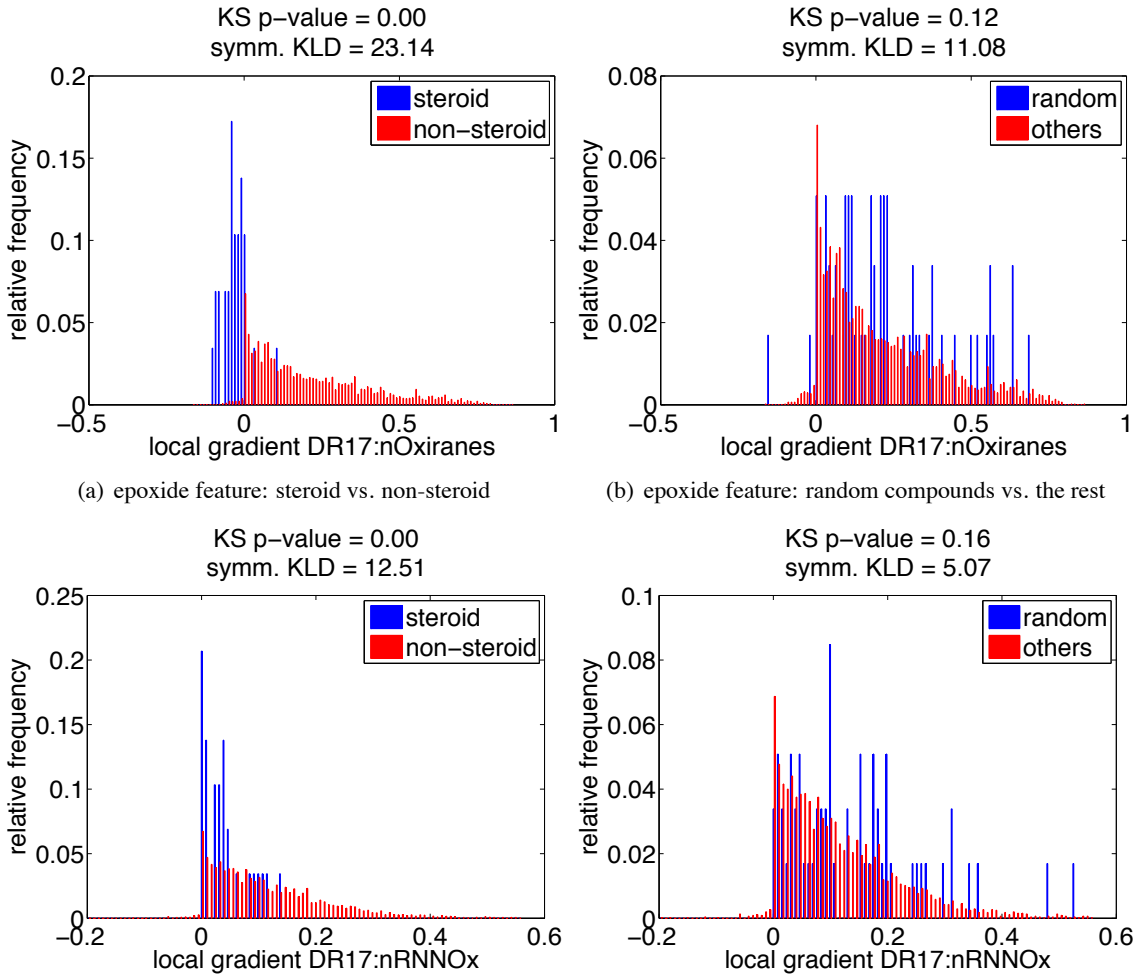
Figure 8: Distribution of local importance of selected features across the test set of 4512 compounds. All five known detoxicophores exhibit negative local gradients

Kullback-Leibler divergence (KLD) gives a metric of the distance between the two distributions.[11] While containing epoxides generally tends to make molecules mutagenic (see discussion above), we do not observe this effect for steroids: In Figure 9(a), almost all epoxide containing non-steroids exhibit positive gradients, thereby following the global distribution of epoxide containing compounds as shown in Figure 7(f). In contrast, almost all epoxide containing steroids exhibit gradients just below zero. "Immunity" of steroids to the epoxide toxicophore is an established fact and has first been discussed by Glatt et al. (1983). This peculiarity in chemical space is clearly exhibited by the local explanation given by our approach. For aliphatic nitrosamine, the situation in the GPC model is less clear but still the toxifying influence seems to be less in steroids than in many other compounds. To our knowledge, this phenomenon has not yet been discussed in the pharmaceutical literature.

In conclusion, we can learn from the explanation vectors that:

- Toxicophores tend to make compounds mutagenic (class 1).
- Detoxicophores tend to make compounds non-mutagenic (class 0).
- Steroids are immune to the presence of some toxicophores (epoxide, possibly also aliphatic nitrosamine).

11. Symmetry is achieved by averaging the two Kullback-Leibler divergences: $\frac{KL(P1,P2)+KL(P2,P1)}{2}$, compare to Johnson and Sinanovic (2000). To prevent zero-values in the histograms which would lead to infinite KL distances, an $\varepsilon > 0$ has been added to each bin count.



(a) epoxide feature: steroid vs. non-steroid

(b) epoxide feature: random compounds vs. the rest

(c) aliphatic nitrosamine feature: steroid vs. non-steroid

(d) aliphatic nitrosamine feature: random compounds vs. the rest

Figure 9: The local distribution of feature importance to steroids and random non-steroid compounds significantly differs for two known toxicophores. The small local gradients found for the steroids (shown in blue) indicate that the presence of each toxicophore is irrelevant to the molecules toxicity. For non-steroids (shown in red) the known toxicophores indeed exhibit positive local gradients.

7. Related Work

Assigning potentially different explanations to individual data points distinguishes our approach from conventional feature extraction methods that extract global features that are relevant for all data points, that is, those features that allow to achieve a small overall prediction error. Our notion of explanation is not related to the prediction error, but only to the label provided by the prediction algorithm. Even if the error is large, our framework is able to answer the question *why* the algorithm has decided on a data point the way it did.

The explanation vector proposed here is similar in spirit to sensitivity analysis which is common in various areas of information science. A classical example is outlier sensitivity in statistics (Hampel et al., 1986). In this case, the effects of removing single data points on estimated parameters are evaluated by an influence function. If the influence for a data point is significantly large, it is detected as an outlier and should be removed for the following analysis. In regression problems, leverage analysis is a procedure along similar lines. It detects leverage points which have potential to give large impact on the estimate of the regression function. In contrast to the influential points (outliers), removing a leverage sample may not actually change the regressor, if its response is very close to the predicted value. E.g., for linear regression the samples whose inputs are far from the mean are the leverage points. Our framework of explanation vectors considers a different view. It describes the influence of *moving* single data points locally and it thus answers the question which directions are locally most influential to the prediction. The explanation vectors are used to extract sensitive features that are relevant to the prediction results, rather than detecting/eliminating the influential samples.

In recent decades, explanation of results by expert systems has been an important topic in the Artificial Intelligence community. Especially for expert systems based on Bayesian belief networks, such explanation is crucial in practical use. In this context sensitivity analysis has also been used as a guiding principle (Horvitz et al., 1988). There the influence is evaluated by removing a set of variables (features) from the evidence and the explanation is constructed from those variables that affect inference (relevant variables). For example, Suermondt (1992) measures the cost of omitting a single feature E_i by the cross-entropy

$$H^{-}(E_i) = H(p(D|E); P(D|E\backslash E_i)) = \sum_{j=1}^{N} P(d_j|E) \log \frac{P(d_j|E)}{p(d_j|E\backslash E_i)},$$

where E denotes the evidence and $D = (d_1, \ldots, d_N)^T$ is the target variable. The cost of a subset $F \subset E$ can be defined similarly. This line of research is more connected to our work, because explanation can depend on the assigned values of the evidence E, and is thus local.

Similarly Robnik-Sikonja and Kononenko (2008) and Strumbelj and Kononenko (2008) try to explain the decision of trained kNN-, SVM-, and ANN-models for individual instances by measuring the difference in their prediction with sets of features omitted. The cost of omitting features is evaluated as the information difference, the log-odds ratio, or the difference of probabilities between the model with knowledge about all features and with omissions, respectively. To know what the prediction would be without the knowledge of a certain feature the model is retrained for every choice of features whose influence is to be explained. To save the time of combinatorial training Robnik-Sikonja and Kononenko (2008) propose to use neutral values which have to be estimated by a known prior distribution of all possible parameter values. As a theoretical framework for considering feature interactions, Strumbelj and Kononenko (2008) propose to calculate the differences between model predictions for every choice of feature subset.

For multi-layer perceptrons Fraud and Clrot (2002) measure the importance of individual input variables on clusters of test points. Therefore the change in the model output is evaluated for the change of a single input variable in a chosen interval while all other input variables are fixed. Lemaire and Feraud (2007) use a similar approach on an instance by instance basis. By considering each input variable in turn there is no way to measure input feature interactions on the model output (see LeCun et al., 1998).

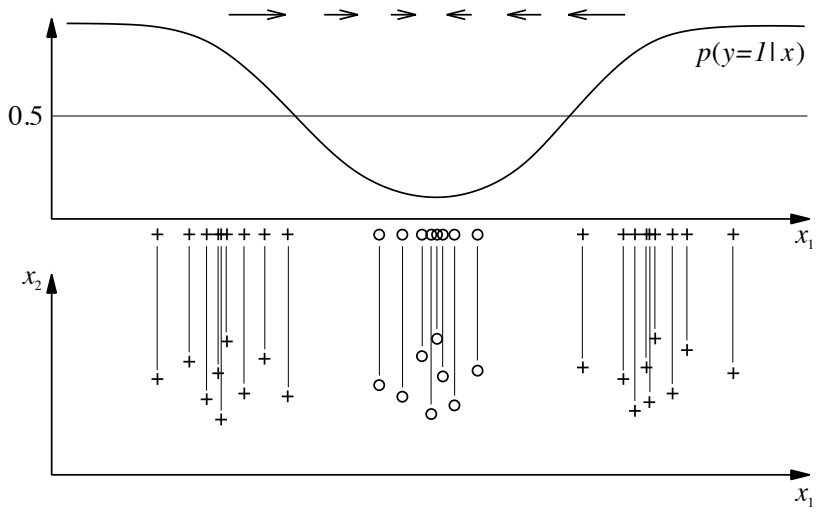


Figure 10: $\zeta(x)$ is the zero vector in the middle of the cluster in the middle.

The principal differences between our approach and these frameworks are: (i) We consider continuous features and no structure among them is required, while some other frameworks start from binary features and may require discretization steps with the need to estimate parameters for it. (ii) We allow changes in any direction, that is, any weighted combination of variables, while other approaches only consider one feature at a time or the omission of a set of variables.

8. Discussion

We have shown that our methods for calculating / estimating explanation vectors are useful in a variety of situations. In the following we discuss their limitations.

8.1 What Can We Do if the Derivative is Zero?

This situation is depicted in Figure 10. In the lower panel we see a two-dimensional data set consisting of three clusters. The middle cluster has a different class than the clusters on the left and on the right. Only the horizontal coordinate (i.e., x_1) is relevant for the classification. The upper panel shows the projected data and a representative slice of $\zeta(x)$. However, the explanation $\zeta(x)$ for the center point of the middle cluster is the zero vector, because at that point $p(Y=1|X=x)$ is maximal. What can we do in such situations? Actually, the (normalized) explanation vector is derived from the following optimization problem for finding the locally most influential direction: $\text{argmax}_{\|\varepsilon\|=1}\{p(Y\neq g^*(x_0)|X=x_0+\varepsilon)-p(Y\neq g^*(x_0)|X=x_0)\}$. In case that the first derivative of the above criterion is zero, its Taylor expansion starts from the second order term, which is a quadratic form of its Hessian matrix. In the example data set with three clusters, the explanation vector is constant along the second dimension. The most *interesting* direction is given by the eigenvector corresponding to the largest eigenvalue of the Hessian. This direction will be in our example along the first dimension. Thus, we can learn from the Hessian that the first coordinate is relevant for the classification, but we do not obtain an orientation for it. Instead it means that both directions (left and right) will influence the classification. However, if the conditional distri-

bution $P(Y = 1 \mid X = x)$ is flat in some regions, no meaningful explanation can be obtained by the gradient-based approach with the remedy mentioned above. Practically, by using Parzen window estimators with larger widths, the explanation vector can capture coarse structures of the classifier at the points that are not so far from the borders. In A.3.2 we give an illustration of this point. In the future, we would like to work on global approaches, for example, based on distances to the borders, or extensions of the approach by Robnik-Sikonja and Kononenko (2008). Since these procedures are expected to be computationally demanding, our proposal is useful in practice, in particular for probabilistic classifiers.

8.2 Does Our Framework Generate Different Explanations for Different Prediction Models?

When using the local gradient of the model prediction directly as in Definition 2 and Section 6, the explanation follows the given model precisely by definition. For the estimation framework this depends on whether the different classifiers classify the data differently. In that case the explanation vectors will be different, which makes sense, since they should explain the classifier at hand, even if its estimated labels were not all correct. On the other hand, if the different classifiers agree on all labels, the explanation will be exactly equal.

8.3 Which Implicit Limitations Do Analytical Gradients Inherit From Gaussian Process Models?

A particular phenomenon can be observed at the boundaries of the training data: Far from the training data, Gaussian Process Classification models predict a probability of 0.5 for the positive class. When querying the model in an area of the feature space where predictions are negative, and one approaches the boundaries of the space populated with training data, explanation vectors will point away from any training data and therefore also away from areas of positive prediction. This behavior can be observed in Figure 1(d), where unit length vectors indicate the direction of explanation vectors. In the right hand side corner, arrows point away from the triangle. However, we can see that the length of these vectors is so small that they are not even visible in Figure 1(c). Consequently, this property of GPC models does not pose a restriction for identifying the locally most influential features by investigating the features with the highest absolute values in the respective partial derivatives, as shown in Section 6.

8.4 Stationarity of the Data

Since explanation vectors are defined as local gradients of the model prediction (see Definition 2), no assumption on the data is made: The local gradients follow the predictive model in any case. If, however, the model to be explained assumes stationarity of the data, the explanation vectors will inherit this limitation and reflect any shortcomings of the model (e.g., when the model is applied to non-stationary data). Our method for estimating explanation vectors, on the other hand, assumes stationarity of the data.

When modeling data that is in fact non-stationary, appropriate measures to deal with such data sets should be taken. One option is to separate the feature space into stationary and non-stationary parts using Stationary Subspace Analysis as introduced by von Bünau et al. (2009). For further approaches to data set shift see Sugiyama et al. (2007b), Sugiyama et al. (2007a), and the book by Quionero-Candela et al. (2009).

9. Conclusion

This paper proposes a method that sheds light on the black boxes of nonlinear classifiers. In other words, we introduce a method that can explain the local decisions taken by arbitrary (possibly) nonlinear classification algorithms. In a nutshell, the estimated explanations are local gradients that characterize how a data point has to be moved to change its predicted label. For models where such gradient information cannot be calculated explicitly, we employ a probabilistic approximate mimic of the learning machine to be explained.

To validate our methodology we show how it can be used to draw new conclusions on how the various Iris flowers in Fisher's famous data set are different from each other and how to identify the features with which certain types of digits 2 and 8 in the USPS data set can be distinguished. Furthermore, we applied our method to a challenging drug discovery problem. The results on that data fully agree with existing domain knowledge, which was not available to our method. Even local peculiarities in chemical space (the extraordinary behavior of steroids) was discovered using the local explanations given by our approach.

Future directions are two-fold: First we believe that our method will find its way into the tool boxes of practitioners who not only want to automatically classify their data but who also would like to understand the learned classifier. Thus using our explanation framework in computational biology (see Sonnenburg et al., 2008) and in decision making experiments in psychophysics (e.g., Kienzle et al., 2009) seems most promising. The second direction is to generalize our approach to other prediction problems such as regression.

Acknowledgments

This work was supported in part by the FP7-ICT Programme of the European Community, under the PASCAL2 Network of Excellence, ICT-216886 and by DFG Grant MU 987/4-1. We would like to thank Andreas Sutter, Antonius Ter Laak, Thomas Steger-Hartmann and Nikolaus Heinrich for publishing the Ames mutagenicity data set (Hansen et al., 2009).

Appendix A.

In the following we present the derivation of direct local gradients and illustrate aspects like the effect of different kernel functions, outliers and local non-linearities. Furthermore we present the derivation of explanation vectors based on the parzen window estimation and illustrate how the quality of the fit of the Parzen window approximation affects the quality of the estimated explanation vectors.

A.1 Derivation of Direct Local Gradients

Equation (1) is derived by the following steps:

$$
\begin{aligned}
&\nabla p(x)|_{x=x_0} \\
=&\nabla \frac{1}{2}\operatorname{erfc}\left(\frac{-\overline{f}(x)}{\sqrt{2}*\sqrt{1+\operatorname{var}_f(x)}}\right)\Bigg|_{x=x_0} \\
=&\nabla \frac{1}{2}\left(1-\operatorname{erf}\left(\frac{-\overline{f}(x)}{\sqrt{2}*\sqrt{1+\operatorname{var}_f(x)}}\right)\right)\Bigg|_{x=x_0} \\
=&-\frac{1}{2}\nabla \operatorname{erf}\left(\frac{-\overline{f}(x)}{\sqrt{2}*\sqrt{1+\operatorname{var}_f(x)}}\right)\Bigg|_{x=x_0} \\
=&-\frac{\exp\left(\frac{-\overline{f}(x_0)^2}{2(1+\operatorname{var}_f(x_0))}\right)}{\sqrt{\pi}}\nabla\left(\frac{-\overline{f}(x)}{\sqrt{2}*\sqrt{1+\operatorname{var}_f(x)}}\right)\Bigg|_{x=x_0} \\
=&-\frac{\exp\left(\frac{-\overline{f}(x_0)^2}{2(1+\operatorname{var}_f(x_0))}\right)}{\sqrt{\pi}}\left(-\frac{1}{\sqrt{2}}\nabla\left(\frac{\overline{f}(x)}{\sqrt{1+\operatorname{var}_f(x)}}\right)\Bigg|_{x=x_0}\right) \\
=&\frac{\exp\left(\frac{-\overline{f}(x_0)^2}{2(1+\operatorname{var}_f(x_0))}\right)}{\sqrt{2\pi}}\left(\frac{\nabla\overline{f}(x)|_{x=x_0}}{\sqrt{1+\operatorname{var}_f(x_0)}}+\overline{f}(x_0)\left(\nabla\operatorname{var}_f(x)\big|_{x=x_0}*-\frac{1}{2}\left(1+\operatorname{var}_f(x_0)\right)^{-\frac{3}{2}}\right)\right) \\
=&\frac{\exp\left(\frac{-\overline{f}(x_0)^2}{2(1+\operatorname{var}_f(x_0))}\right)}{\sqrt{2\pi}}\left(\frac{\nabla\overline{f}(x)|_{x=x_0}}{\sqrt{1+\operatorname{var}_f(x_0)}}-\frac{1}{2}\frac{\overline{f}(x_0)}{\left(1+\operatorname{var}_f(x_0)\right)^{\frac{3}{2}}}\nabla\operatorname{var}_f(x)|_{x=x_0}\right).
\end{aligned}
$$

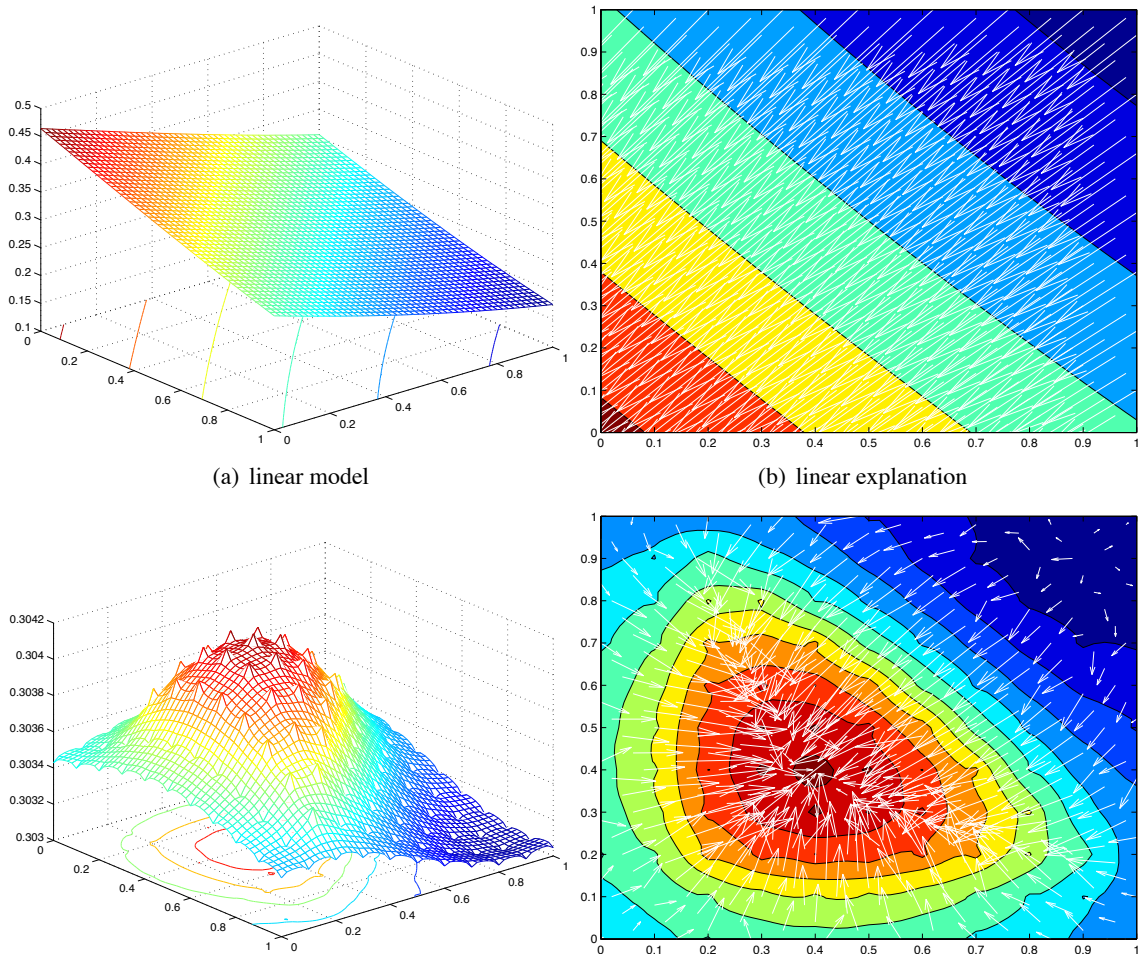
A.2 Illustration of Direct Local Gradients

In the following we give some illustrative examples of our method to explain models using local gradients. Since the explanation is derived directly from the respective model, it is interesting to investigate its acurateness depending on different model parameters and in instructive scenarios. We examine the effects that local gradients exhibit when choosing different kernel functions, when introducing outliers, and when the classes are not linearly separable locally.

A.2.1 CHOICE OF KERNEL FUNCTION

Figure 11 shows the effect of different kernel functions on the triangle toy data from Figure 1. The following observations can be made:

- In any case note that the local gradients explain the model, which in turn may or may not capture the true situation.
- In Subfigure 11(a) the linear kernel leads to a model which fails to capture the non-linear class separation. This model misspecification is reflected by the explanations given for this model in Subfigure 11(b).
- The rational quadratic kernel is able to more accurately model the non-linear separation. In Subfigure 11(c) a non-optimal degree parameter has been chosen for illustrative purposes.



(a) linear model

(b) linear explanation

(c) rational quadratic model

(d) rational quadratic explanation

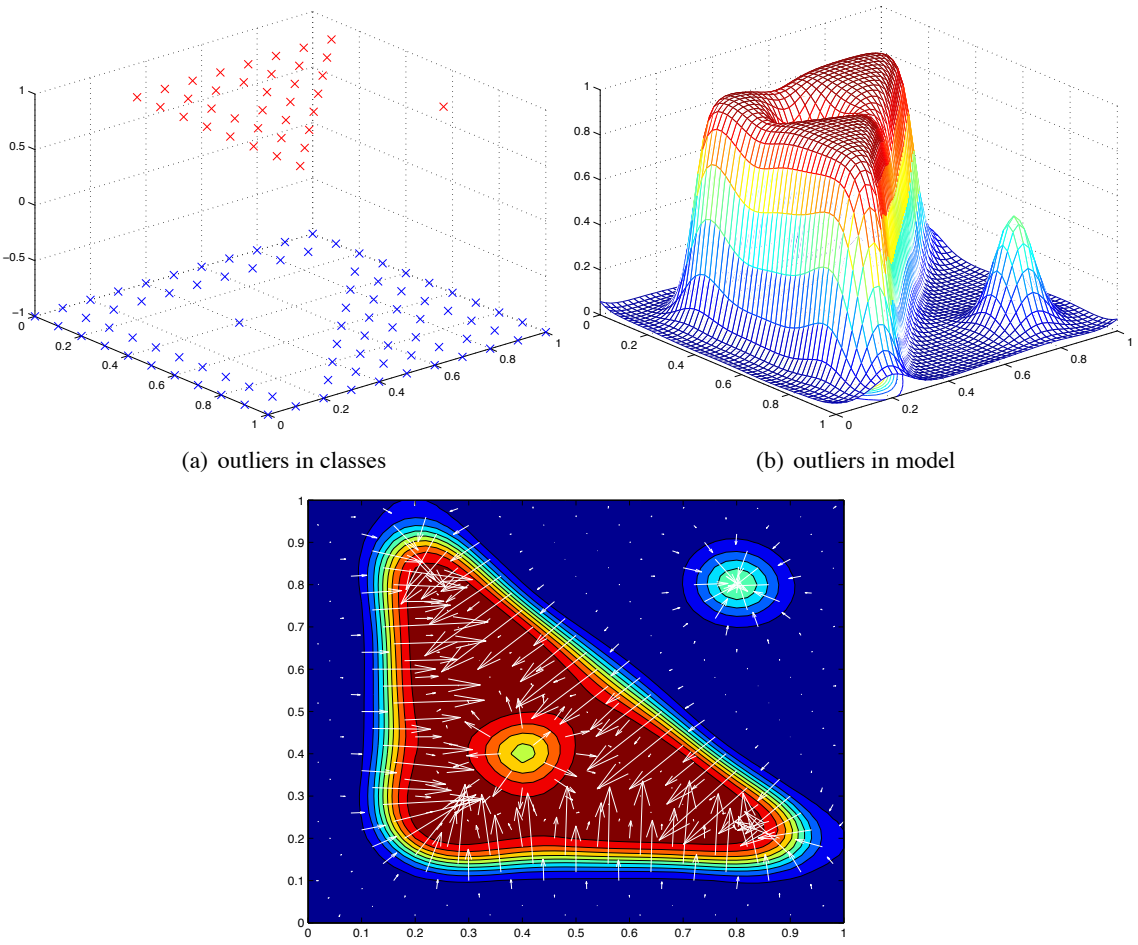
Figure 11: The effect of different kernel functions to the local gradient explanations

For other parameter values the rational quadratic kernel leads to similar results as the RBF kernel function used in Figure 1.

- The explanations in Subfigure 11(d) obtained for this model show local perturbations at the small "bumps" of the model but the trends towards the positive class are still clear. As previously observed in Figure 1, the explanations make clear that both features interact at the corners and on the hypotenuse of the triangle class.

A.2.2 OUTLIERS

In Figure 12 the effects of two outliers in the classification data to GPC with RBF kernel are shown. Once more, note that the local gradients explain the model, which in turn may or may not capture the true situation. The size of the region affected by the outliers depends on the kernel width parameter. We consider the following items:



(a) outliers in classes

(b) outliers in model

(c) outlier explanation

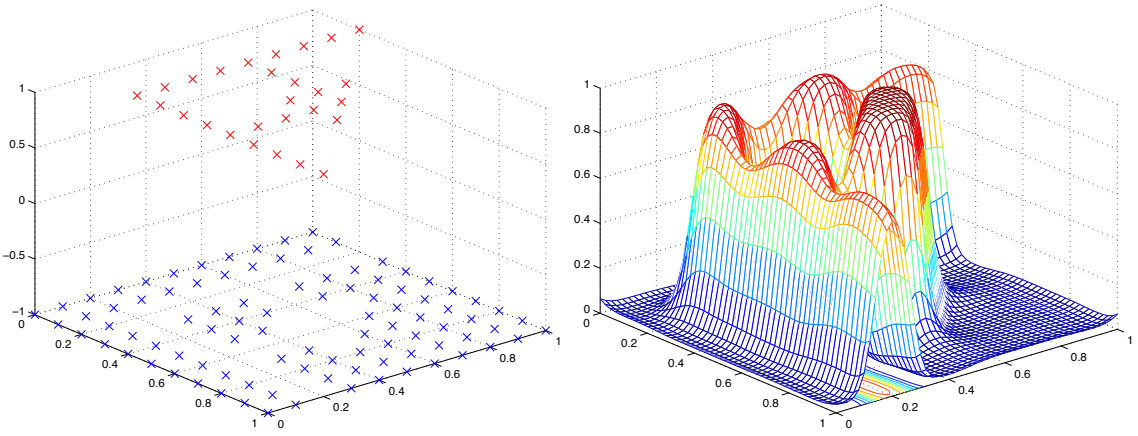
Figure 12: The effect of outliers to the local gradient explanations

- Local gradients are in the same way sensitive to outliers as the model which they try to explain. Here a single outlier deforms the model and with it the explanation which may be extracted from it.
- Being derivatives the sensitivity of local gradients to a nearby outlier is increased over the sensitivity of the model prediction itself.
- Thus the local gradient of a point near an outlier may not reflect a true explanation of the features important in reality. Nevertheless it is the model here which is wrong around an outlier in the first place.
- The histograms in the Figures 7, 8, and 9 in Section 6 show the trends of the respective features in the distribution of all test points and are thus not affected by single outliers.

To compensate for the effect of outliers to the local gradients of points in the affected region we propose to use a sliding window method to smooth the gradients around each point of interest.

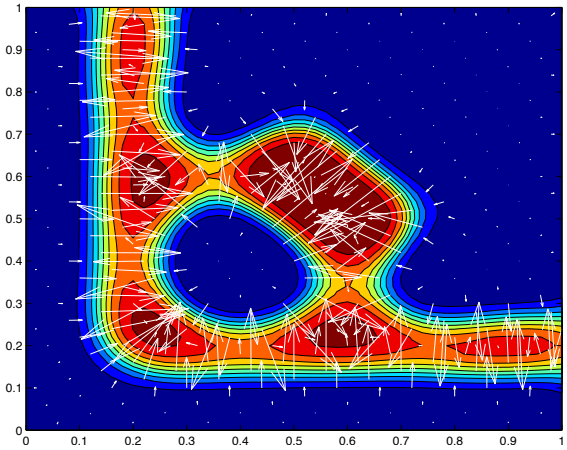
Thus for each point use the mean of all local gradients in the hypercube centered at this point and of appropriate size. This way the disrupting effect of an outlier is averaged out for an appropriately chosen window size.

A.2.3 LOCAL NON-LINEARITY



(a) locally non-linear object

(b) locally non-linear model



(c) locally non-linear explanation

Figure 13: The effect of local non-linearity to the local gradient explanations

The effect of locally non-linear class boundaries in the data is shown in Figure 13 again for GPC with an RBF kernel. The following points can be observed:

- All the non-linear class boundaries are accurately followed by the local gradients.
- The circle shaped region of negative examples surrounded by positive ones shows the full range of feature interactions towards the positive class.
- On the ridge of single positive instances the model introduces small valleys which are reflected by the local gradients.

A.3 Estimating by Parzen Window

Finally we elaborate on some details of our estimation approach of local gradients by Parzen window approximation. First we give the derivation to obtain the explanation vector and second we examine how the explanation varies with the goodness of fit of the Parzen window method.

A.3.1 DERIVATION OF EXPLANATION VECTORS

These are more details on the derivation of Definition 3. We use the index set $I_c = \{i \mid g(x_i) = c\}$:

$$\begin{aligned}
&\frac{\partial}{\partial x} k_\sigma(x) = -\frac{x}{\sigma^2} k_\sigma(x) \\
&\frac{\partial}{\partial x} \hat{p}_\sigma(x, y \neq c) = \frac{1}{n} \sum_{i \notin I_c} k_\sigma(x - x_i) \frac{-(x - x_i)}{\sigma^2} \\
&\frac{\partial}{\partial x} \hat{p}_\sigma(y \neq c | x) \\
&= \frac{\left(\sum_{i \notin I_c} k(x - x_i)\right)\left(\sum_{i=1}^n k(x - x_i)(x - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2} \\
&- \frac{\left(\sum_{i \notin I_c} k(x - x_i)(x - x_i)\right)\left(\sum_{i=1}^n k(x - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2} \\
&= \frac{\left(\sum_{i \notin I_c} k(x - x_i)\right)\left(\sum_{i \in I_c} k(x - x_i)(x - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2} \\
&- \frac{\left(\sum_{i \notin I_c} k(x - x_i)(x - x_i)\right)\left(\sum_{i \in I_c} k(x - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2}
\end{aligned}$$

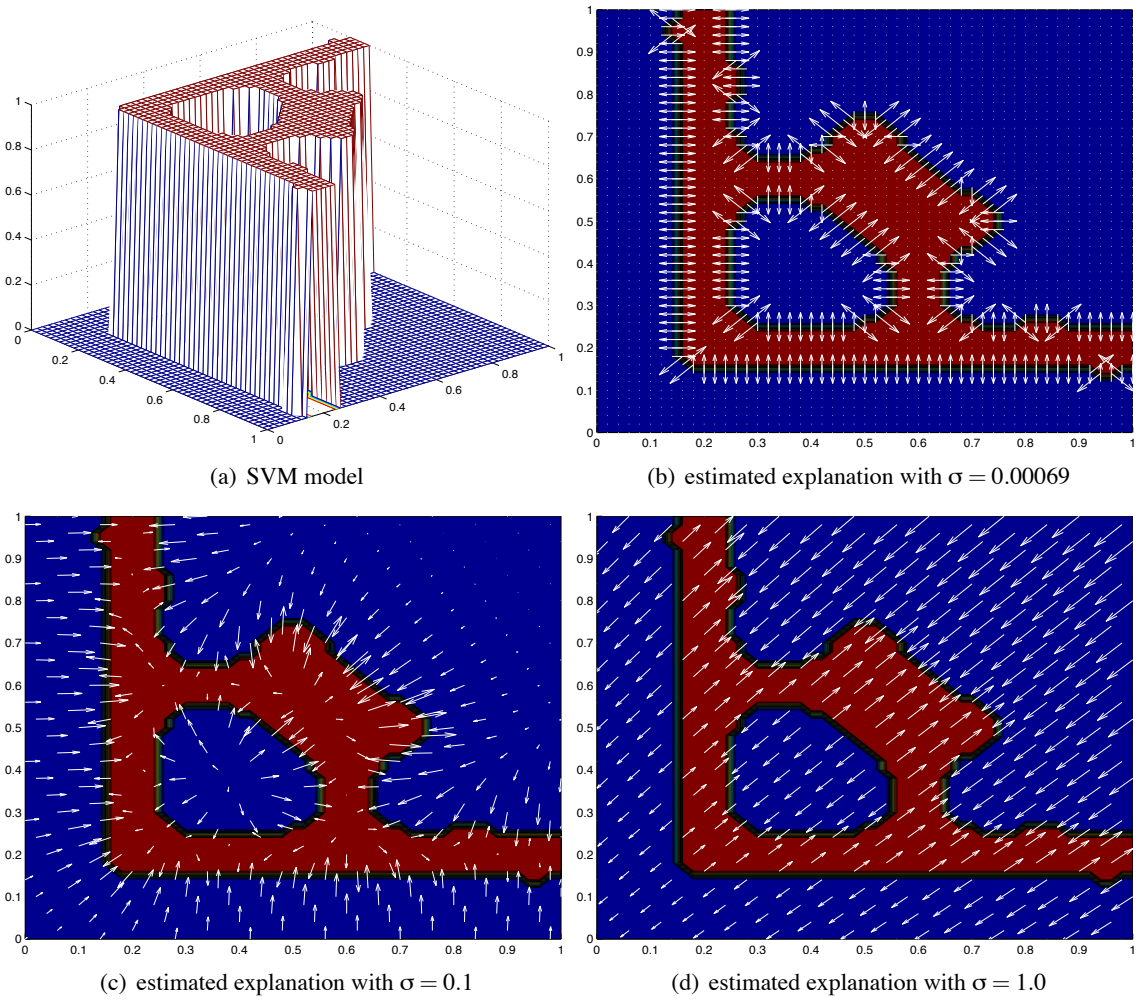
and thus for the index set $I_{g(z)} = \{i \mid g(x_i) = g(z)\}$

$$\begin{aligned}
\hat{\zeta}(z) &= \frac{\partial}{\partial x} \hat{p}(y \neq g(z) \mid x)\Big|_{x=z} \\
&= \frac{\left(\sum_{i \notin I_{g(z)}} k(z - x_i)\right)\left(\sum_{i \in I_{g(z)}} k(z - x_i)(z - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2} \\
&- \frac{\left(\sum_{i \notin I_{g(z)}} k(z - x_i)(z - x_i)\right)\left(\sum_{i \in I_{g(z)}} k(z - x_i)\right)}{\sigma^2 \left(\sum_{i=1}^n k(z - x_i)\right)^2}.
\end{aligned}$$

A.3.2 GOODNESS OF FIT BY PARZEN WINDOW

In our estimation framework the quality of the local gradients depends on the approximation of the classifier we want to explain by Parzen windows for which we can calculate the explanation vectors as given by Definition 3.

Figure 14(a) shows an SVM model trained on the classification data from Figure 13(a). The local gradients estimated for this model by different Parzen window approximations are depicted in Subfigures 14(b), 14(c), and 14(d). We observe the following points:



(a) SVM model

(b) estimated explanation with $\sigma = 0.00069$

(c) estimated explanation with $\sigma = 0.1$

(d) estimated explanation with $\sigma = 1.0$

Figure 14: Good fit of Parzen window approximation affects the quality of the estimated explanation vectors

- The SVM model was trained with $C = 10$ and using an RBF kernel of width $\sigma = 0.01$.

- In Subfigure 14(b) a small window width has been chosen by minimizing the mean absolute error over the validation set of labels predicted by the SVM classifier. Thus we obtain explaining local gradients on the class boundaries but zero vectors in the inner class regions. While this resembles the piecewise flat SVM model most accurately it may be more useful practically to choose a larger width to obtain non-zero gradients pointing to the borders in this regions as well. For a more detailed discussion of zero gradients see Section 8.

- A larger width practically useful in this example is shown in Subfigure 14(c). Here the local gradients in the inner class regions point to the other class as well.

- For a too large window width in Subfigure 14(d) the approximation fails to obtain local gradients which closely follow the model. Here only two directions are left and the gradients for the blue class on the left and on the bottom point in the wrong direction.

References

B. N. Ames, E. G. Gurney, J. A. Miller, and H. Bartsch. Carcinogens as frameshift mutagens: Metabolites and derivatives of 2-acetylaminofluorene and other aromatic amine carcinogens. *Proceedings of the National Academy of Sciences of the United States of America*, 69(11):3128–3132, 1972.

C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Number 31 in Applications of Mathematics. Springer, New York, 1996.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7: 179–188, 1936.

R. Fraud and F. Clrot. A methodology to explain neural network classification. *Neural Networks*, 15(2):237 – 246, 2002. doi: 10.1016/S0893-6080(01)00127-7.

H. Glatt, R. Jung, and F. Oesch. Bacterial mutagenicity investigation of epoxides: drugs, drug metabolites, steroids and pesticides. *Mutation Research/Fundamental and Molecular Mechanisms of Mutagenesis*, 111(2):99–118, 1983. doi: 10.1016/0027-5107(83)90056-8.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. Wiley, New York, 1986.

K. Hansen, S. Mika, T. Schroeter, A. Sutter, A. Ter Laak, T. Steger-Hartmann, N. Heinrich, and K.-R. Müller. A benchmark data set for in silico prediction of ames mutagenicity. *Journal of Chemical Information and Modelling*, 49(9):2077–2081, 2009.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

E. J. Horvitz, J. S. Breese, and M. Henrion. Decision theory in expert systems and artificial inteligence. *Journal of Approximation Reasoning*, 2:247–302, 1988. Special Issue on Uncertainty in Artificial Intelligence.

D. H. Johnson and S. Sinanovic. Symmetrizing the Kullback-Leibler distance. Technical report, IEEE Transactions on Information Theory, 2000.

J. Kazius, R. McGuire, and R. Bursi. Derivation and validation of toxicophores for mutagenicity prediction. *J. Med. Chem.*, 48:312–320, 2005.

W. Kienzle, M. O. Franz, B. Schölkopf, and F. A. Wichmann. Center-surround patterns emerge as optimal predictors for human saccade targets. *Journal of Vision*, 9(5):1–15, 2009.

M. Kuss and C. E. Ramussen. Assesing approximate inference for binary gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

Y. LeCun, L. Bottou, G.B. Orr, and K.-R. Müller. Efficient backprop. In G.B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–53. Springer, 1998.

V. Lemaire and R. Feraud. Une méthode d'interprétation de scores. In *EGC*, pages 191–192, 2007.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *Neural Networks, IEEE Transactions on*, 12(2):181–201, 2001.

Olga Obrezanova and Matthew D. Segall. Gaussian processes for classification: QSAR modeling of ADMET and target activity. *Journal of Chemical Information and Modeling*, April 2010. ISSN 1549-9596. doi: doi:10.1021/ci900406x. URL `http://dx.doi.org/10.1021/ci900406x`.

O. Obrezanova, G. Csányi, J. M. R. Gola, and M. D. Segall. Gaussian processes: A method for automatic QSAR modelling of adme properties. *J. Chem. Inf. Model.*

O. Obrezanova, J. M. R. Gola, E. J. Champness, and M. D. Segall. Automatic QSAR modeling of adme properties: blood-brain barrier penetration and aqueous solubility. *J. Comput.-Aided Mol. Des.*, 22:431–440, 2008.

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, 1999.

J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Springer, 2006.

M. Robnik-Sikonja and I. Kononenko. Explaining classifications for individual instances. *IEEE TKDE*, 20(5):589–600, 2008.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT, 2002.

T. Schroeter, A. Schwaighofer, S. Mika, A. Ter Laak, D. Suelzle, U. Ganzer, N. Heinrich, and K.-R. Müller. Estimating the domain of applicability for machine learning QSAR models: A study on aqueous solubility of drug discovery molecules. *Journal of Computer Aided Molecular Design*, 21(9):485–498, 2007a.

T. Schroeter, A. Schwaighofer, S. Mika, A. Ter Laak, D. Suelzle, U. Ganzer, N. Heinrich, and K.-R. Müller. Machine learning models for lipophilicity and their domain of applicability. *Mol. Pharm.*, 4(4):524–538, 2007b.

T. Schroeter, A. Schwaighofer, S. Mika, A. Ter Laak, D. Sülzle, U. Ganzer, N. Heinrich, and K.-R. Müller. Predicting lipophilicity of drug discovery molecules using gaussian process models. *ChemMedChem*, 2(9):1265–1267, 2007c.

A. Schwaighofer. SVM Toolbox for Matlab, Jan 2002. URL `http://ida.first.fraunhofer.de/~anton/software.html`.

A. Schwaighofer, T. Schroeter, S. Mika, J. Laub, A. Ter Laak, D. Sülzle, U. Ganzer, N. Heinrich, and K.-R. Müller. Accurate solubility prediction with error bars for electrolytes: A machine learning approach. *Journal of Chemical Information and Modelling*, 47(2):407–424, 2007.

A. Schwaighofer, T. Schroeter, S. Mika, K. Hansen, A. Ter Laak, P. Lienau, A. Reichel, N. Heinrich, and K.-R. Müller. A probabilistic approach to classifying metabolic stability. *Journal of Chemical Information and Modelling*, 48(4):785–796, 2008.

S. Sonnenburg, A. Zien, P. Philips, and G. Rätsch. POIMs: positional oligomer importance matrices — understanding support vector machine based signal detectors. *Bioinformatics*, 2008.

E. Strumbelj and I. Kononenko. Towards a model independent method for explaining classification for individual instances. In I.-Y. Song, J. Eder, and T.M. Nguyen, editors, *Data Warehousing and Knowledge Discovery*, volume 5182 of *Lecture Notes in Computer Science*, pages 273–282. Springer, 2008.

H. Suermondt. *Explanation in Bayesian Belief Networks*. PhD thesis, Department of Computer Science and Medicine, Stanford University, Stanford, CA, 1992.

M. Sugiyama, M. Krauledat, and K.-R. Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, May 2007a.

M. Sugiyama, S. Nakajima, H. Kashima, P. von Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems 20*. MIT Press, 2007b.

R. Todeschini, V. Consonni, A. Mauri, and M. Pavan. DRAGON for Windows and Linux 2006. `http://www.talete.mi.it/help/dragon_help/` (accessed 27 March 2009), 2006.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

P. von Bünau, F. C. Meinecke, F. J. Király, and K.-R. Müller. Finding stationary subspaces in multivariate time series. *Physical Review Letters*, 103(21):214101, 2009.